JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

ANANTAPUR

DIABETIC RETINOPATHY DETECTION USING ARTIFICIAL INTELLIGENCE

A project report Submitted to the

Jawaharlal Nehru Technological University ANANTAPUR.

In partial fulfilment for the Award of the degree of

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS AND COMMUNICATION ENGINEERING

By

G. PAVAN KUMAR
(H.T. No: 17JN1A0456)

G. RAKESH
(H.T. No: 17JN1A0443)

B. MADHUSUDHANA
(H.T. No: 17JN1A0415)

CH. PRATHYUSH
(H.T. No: 17JN1A0427)

M. BRAHMA SAI
(H.T. No: 16JN1A0464)

Under the Esteemed Guidance of

Mr. V. PRAVEEN KUMAR , M.Tech
Associate Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SREE VENKATESWARA COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi and Affiliated to Jawaharlal Nehru Technological University – Anantapur)
GOLDEN NAGAR, NH5 BYPASS ROAD, NORTH RAJUPALEM, KODAVALURU (V&M), SPSR NELLORE.

2017-2021

SREE VENKATESWARA COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi and Affiliated to Jawaharlal Nehru Technological University – Anantapur)
GOLDEN NAGAR, NH5 BYPASS ROAD, NORTH RAJUPALEM, KODAVALURU (V&M), SPSR NELLORE.

Department of
ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

Certified that the project entitled "**DIABETIC RETINOPATHY DETECTION USING ARTIFICIAL INTELLIGENCE**" submitted by **G. PAVAN KUMAR (Roll No. 17JN1A0456), G. RAKESH (Roll No. 17JN1A0443), B. MADHUSUDHANA (Roll No. 17JN1A0415), CH. PRATHYUSH (Roll No. 17JN1A0427) & M. BRAHMA SAI (Roll No. 16JN1A0464)** in partial fulfilment for theaward of degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING** by **Jawaharlal Nehru Technological University, Anantapur** during the academic year **2020-2021.** It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Signature of the guide

Mr. V. Praveen Kumar

Signature of Head of the Dept.

Mr. P. Rajesh

Principal

Dr. P. Kumar Babu

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

We are highly thankful to **Mr. V. PRAVEEN KUMAR, Associate Professor** in the **Department of Electronics and Communication Engineering** for his inspiring guidance and for providing the background knowledgeto deal with the problem at every phase of our project in a systematic manner.

We have immense pleasure in expressing our sincere thanks and deep sense of gratitude to **Mr.P. Rajesh,** Head of the Department of Electronics and Communication Engineering for extending necessary facilities for the completion of the project.

With immense respect, we express our sincere gratitude to **Dr.P. Babunaidu, Chairman,** and **Dr.P. Kumar Babu,** Principal for permittingus to take up our project work and to complete the project successfully.

We would like to express our sincere thanks to all faculty members of the Department for their continuous cooperation, which has given us the cogency to build-up adamant aspiration over the completion of our project.

Finally we thank one and all who directly and indirectly helped us to complete the project successfully.

G. Pavan Kumar (Roll No. 17JN1A0456)

G. Rakesh (Roll No. 17JN1A0443)

B. Madhusudhana (Roll No. 17JN1A0415)

CH. Prathyush (Roll No. 17JN1A0427)

M. Brahma Sai (Roll No. 16JN1A0464)

Dept. of Electronics and Communication Engineering
Sree Venkateswara College of Engineering,
NH5 Bypass Road, North Rajupalem, Nellore.

DECLARATION

We **G. PAVAN KUMAR (Roll No. 17JN1A0456), G. RAKESH (Roll No. 17JN1A0443), B. MADHUSUDHANA (Roll No. 17JN1A0415), CH. PRATHYUSH (Roll No. 17JN1A0427) & M. BRAHMA SAI (Roll No. 16JN1A0464)** do hear by declare that the project entitled "**DIABETIC RETINOPATHY DETECTION USING ARTIFICIAL INTELLIGENCE**" was carried out by our batch under the guidance of **Mr.V.PRAVEEN KUMAR.** This Project work submitted to Jawaharlal Nehru Technological University, Anantapur in fulfillment of requirement for the award of **BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING** during the academic year 2020-2021.

Project associates,

G. Pavan Kumar	(Roll No. 17JN1A0456)
G. Rakesh	(Roll No. 17JN1A0443)
B. Madhusudhana	(Roll No. 17JN1A0415)
CH. Prathyush	(Roll No. 17JN1A0427)
M. Brahma Sai	(Roll No. 16JN1A0464)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Diabetic Retinopathy (DR) is a common complication of diabetes mellitus that causes blood leaks in the retina, which causes lesions on the retina that affect vision. If it is not detected early, it can cause blindness. They are two main stages of diabetic eye diseases named, Non-Proliferative Diabetic Retinopathy (NPDR), which is an early stage of diabetic eye disease, and in this stage, the vision will be blurry. And the second one is Proliferative Diabetic Retinopathy (PDR), which is an advanced stage of diabetic eye disease and in this stage, the vision might see a few dark floaters. However, diabetic retinopathy has isolated symptoms, if it gets worse you can notice some of the symptoms like blurry vision, seeing blank or dark areas, and fringe-like patterns.

Diabetic Retinopathy (DR) early detection and treatment can significantly reduce the risk of vision loss. Haemorrhages, hard Exudates and Micro-aneurysms (HEM) that appear in the retina are the early sign of DR. The manual diagnosis process of DR retina fundus images by ophthalmologists is time, effort, and cost consuming process.

Computer-aided diagnosis results in much better accuracy and more economical. There are so many different computer-aided techniques available to detect the stage of DR based on different constraints. One method is Local Ternary Pattern (LTP) and Local Energy-based Shape Histogram (LESH) is used to detect the DR. This method uses Support Vector Machine (SVM) to classify the DR; it has very less accuracy and requires features like LTP and LESH. Another method is a quite recent one that is based on deep learning which is one of the most common techniques that has achieved better performance in many areas. Convolution Neural Network (CNN) are more widely used as a deep learning method in medical image analysis, and they are highly effective. Here we are going to study different deep learning models to detect the DR and analysing with a real-time environment.

1. INTRODUCTION

Diabetic retinopathy is one of the most predominant complications of diabetes and can develop in individuals who are living with either type 1 or type 2 diabetes, and it generally affects both eyes. The condition develops slowly throughout many years; therefore, it is essential to undergo regular eye tests when you have Diabetes. Prevention of retinopathy or slowing down of the progression can be established by keeping excellent control of blood sugar levels. Retinopathy is basically impaired blood vessels in the retina which is the thin inner light-sensitive layer situated in the back of the eyes.

There are millions of individuals worldwide living with diabetes, and about eighty percent of diabetic patients are likely to develop some stage of diabetic retinopathy. There are predictably no early symptoms associated with the early stages of the condition, allowing the disease to advance until the point where it affects vision.

In type 1 diabetes, the insulin production in the pancreas is permanently damaged, whereas in type 2 diabetes, the person is suffering from increased resistance to insulin. Type 2 diabetes is a familial disease but also related to limited physical activity and lifestyle. Diabetes may cause abnormalities in the retina (diabetic retinopathy), kidneys (diabetic nephropathy), and nervous system (diabetic neuropathy). Diabetes is also a major risk factor for cardiovascular diseases. Diabetic retinopathy is a microvascular complication of diabetes, causing abnormalities in the retina, and in the worst case, blindness. Typically, there are no salient symptoms in the early stages of diabetic retinopathy, but the number and severity predominantly increase during the time. Diabetic retinopathy typically begins as small changes in the retinal capillaries.

FIG 1.1: Fundus Image

As the retinopathy advances, the blood vessels become obstructed which causes microinfarcts in the retina. These microinfarcts are called soft exudates (Fig. 1.2(d)). When a significant number of intraretinal hemorrhages, soft exudates, or intraretinal microvascular abnormalities are encountered, the state of the retinopathy is defined as severe nonproliferative diabetic retinopathy. The severe nonproliferative diabetic retinopathy can quickly turn into proliferative diabetic retinopathy when an extensive lack of oxygen causes the development of new fragile vessels. This is called neovascularization (Fig. 1.2(e)) which is a serious eyesight-threatening state. The proliferative diabetic retinopathy may cause a sudden loss in visual acuity or even permanent blindness due to vitreous hemorrhage or tractional detachment of the central retina. After a diagnosis of diabetic retinopathy, regular monitoring is needed due to the progressive nature of the disease. However, broad screenings cannot be performed since the fundus image examination requires the attention of medical experts. For the screening, automatic image processing methods must be developed.

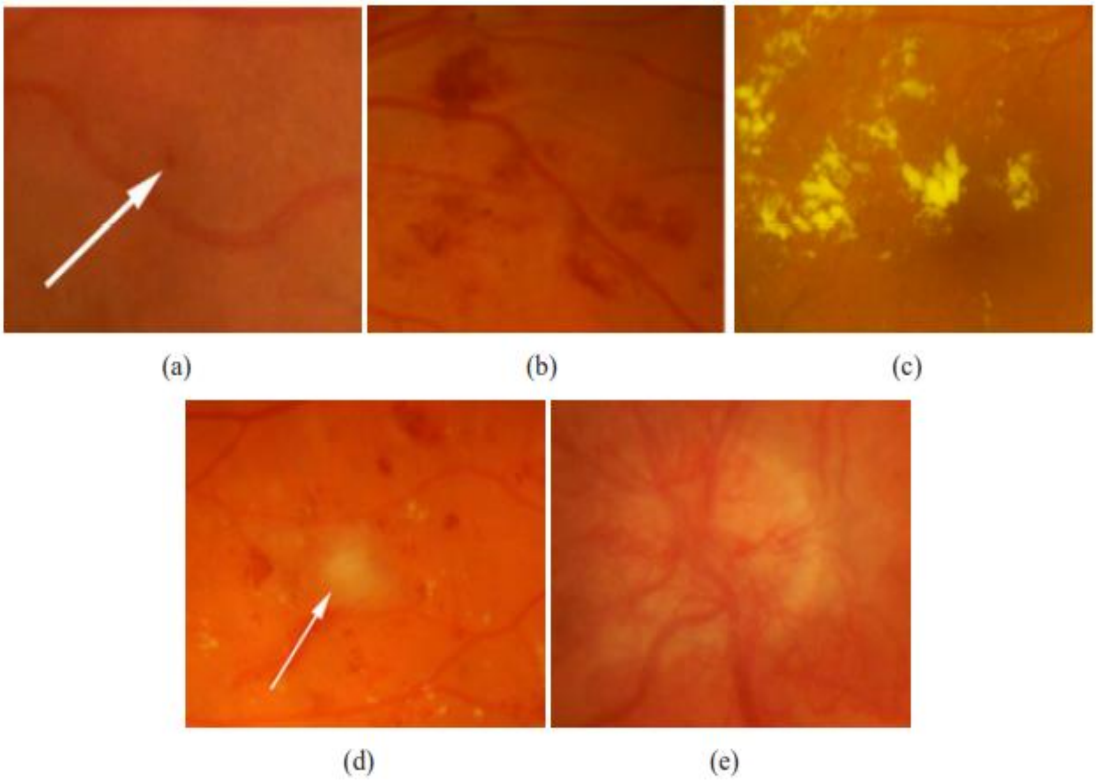

FIG 1.2: Abnormal findings in the eye fundus caused by the diabetic retinopathy: (a) microaneurysms (marked with an arrow), (b) hemorrhages, (c) hard exudates, (d) soft exudate (marked with an arrow), and (e) neovascularization.

Diabetes is a sickness or disease that effects insulin making procedure of the body and prompts demise. Different software analysis frameworks have been created in the course of the most recent decade. Since human specialists for the most part center around some normal sores related with DR, for example, microaneurysms, hemorrhages and hard exudates while assessing retina photos, numerous works focused on automated identify and section these sores or compute some numerical records. CNN calculation gives advantage over others by perceiving designs under outrageous inconstancy, for example, in the event of manually written characters. Diabetic Retinopathy (DR) is the analysis based retinal wound, which brought about by the rise of glucose levels in blood, this can be at last maliciously lead to vison impedance or permanent loss of sight. As it has been indicated by the standards of World Health Organization, it has been scientifically evaluated that over 75% of individuals who have diabetes for over 20 years will have some type of DR in the life stages.

Diabetic Retinopathy (DR) is an eye disease that damages the retina of patients with long-standing diabetes. This is an ocular complication of the eye that affects 75% of diabetic patients leading to blindness in the age group of 20-64. There are different ways to diagnose DR. The World Health Organization reports that about 347 million people in the world are affected by DR. About 366 million adults with diabetes is estimated by International Diabetes Federation. This figure is expected to rise to 552 million by 2030. Estimated occurrence of type 2 diabetes mellitus and diabetic retinopathy is quite high in India, according to the studies that have been conducted so far. Based on a survey in 2000, the top three countries with highest number of diabetes mellitus are India (31.7 million), China (20.8 million) and USA (17.7 million). Trained clinicians are required to examine the color fundus photographs of retina and detect DR. The process of identifying DR involves detection of lesions with vascular abnormalities. This is an effective way of detection but requires the service of experienced clinicians for analysis of the photographs manually, which is time-consuming. Rural areas, where the rate of diabetes is usually high, lack the expertise of well-trained clinicians and sophisticated equipment that are necessary for detection of DR. Better infrastructure with automated detection techniques are now required to tackle the growing number of individuals with diabetes. An early detection can help to avert or decrease the spread of DR which otherwise might cause blindness.

2. EFFECTS

2.1: Statistics about DR Patients in worldwide and Indiawide

2.1.1: World-wide

Diabetic retinopathy (DR), a major microvascular complication of diabetes, has a significant impact on the world's health systems. Globally, the number of people with DR will grow from 126.6 million in 2010 to 191.0 million by 2030, and we estimate that the number with vision-threatening diabetic retinopathy (VTDR) will increase from 37.3 million to 56.3 million, if prompt action is not taken. Despite growing evidence documenting the effectiveness of routine DR screening and early treatment, DR frequently leads to poor visual functioning and represents the leading cause of blindness in working-age populations. DR has been neglected in health-care research and planning in many low-income countries, where access to trained eye-care professionals and tertiary eye-care services may be inadequate. Demand for, as well as, supply of services may be a problem. Rates of compliance with diabetes medications and annual eye examinations may be low, the reasons for which are multifactorial. Innovative and comprehensive approaches are needed to reduce the risk of vision loss by prompt diagnosis and early treatment of VTDR.

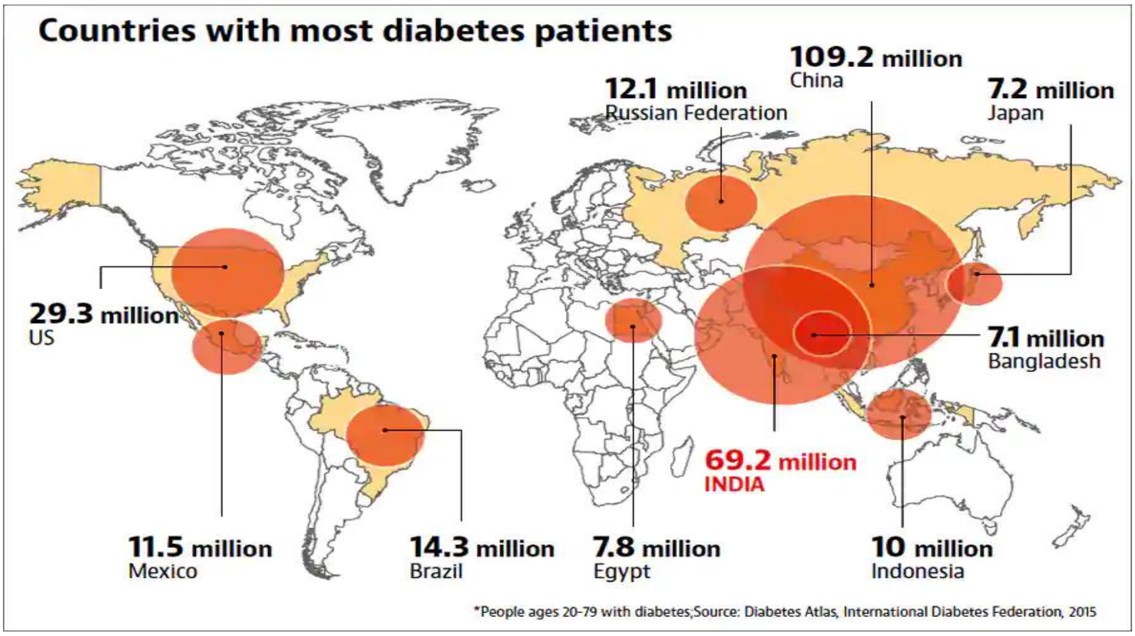

FIG 2.1: Countries with most diabetes patients

In the American National Health and Nutrition Examination Survey (NHNES, 2005–2008), 28.5% of diabetic patients had some degree of DR, 4.4% had VTDR. Similar prevalence estimates are seen in many other developed countries. In the not-so-distant past, DR was thought to be relatively uncommon in developing countries like China and India. It has now become apparent that many low- and middle-income countries are also confronting this challenge, and the prevalence is similar or even higher than that reported in developed countries. China is a good example of a country facing both, the epidemic of diabetes and DR. China is estimated to have 92.4 million adults with diabetes, and a recent report in rural China showed that 43% of the patients with diabetes already have retinopathy and 6.3% have VTDR.

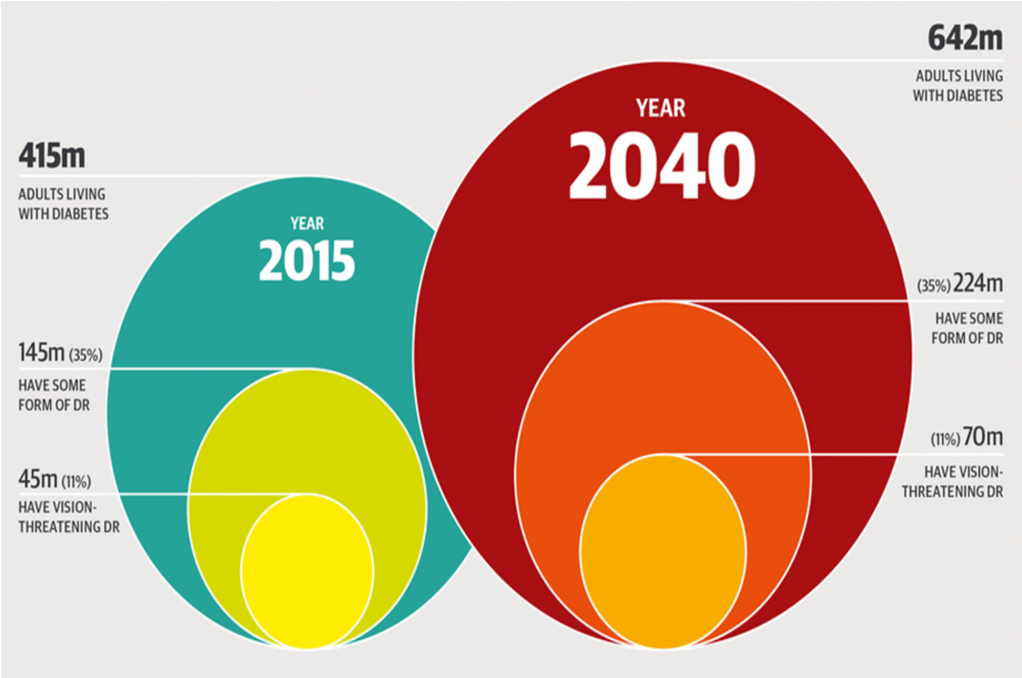

FIG 2.2: Worldwide statistics of DR

2.1.2: Incidence of DR

While accurate figures are difficult to obtain for the incidence of DR, the results of the Wisconsin Epidemiologic Study of Diabetic Retinopathy (WESDR) showed that the overall incidence of DR in a 10-year interval from 1980–1982 to 1990–1992 was 74%, and among those with DR at baseline, 64% had more severe retinopathy and 17% developed PDR. These figures were 89%, 76%, and 30%, respectively among the younger-onset group (diagnosed before age 30 years); and 67%, 53%, and 10%, respectively, among the older-onset group who did not use insulin. In the 25-year follow-up of the WESDR type-1 diabetes group, almost all patients (97%) developed DR, and among these, 42% progressed to PDR, 29% developed macular edema (ME) and 17% had clinically significant ME.

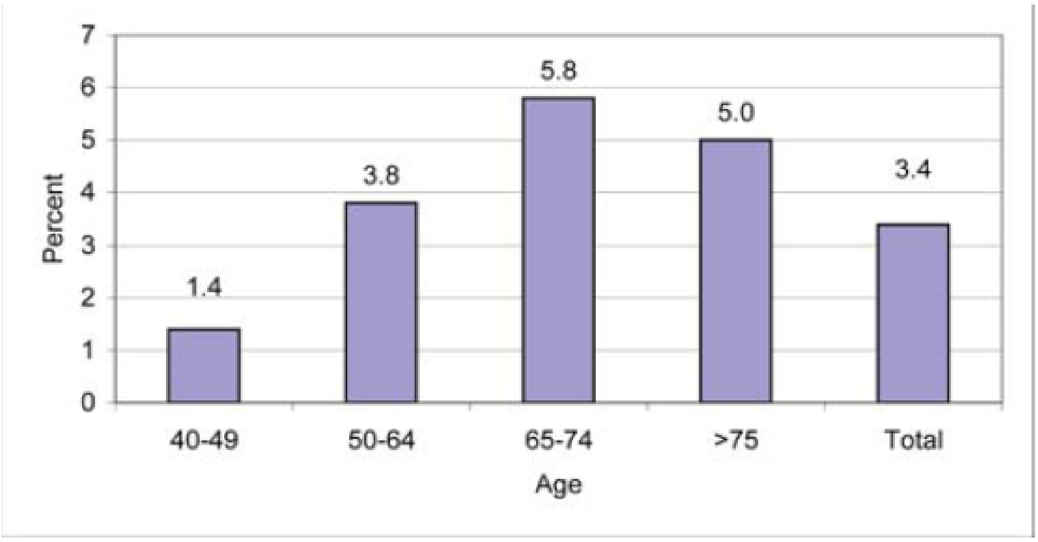

FIG 2.3: Prevalence of Diabetic Retinopathy among adults 40 years and older in 2000

2.1.3: Decline in the Prevalence/Incidence of Diabetic Retinopathy among those with diabetes

In the past three decades, the prevalence and incidence of DR among patients with type 1 diabetes have declined in the US, Australia, and other developed countries. A systemic review of 28 studies showed that participants reported on between1986 and 2008 had a lower incidence of PDR (2.6% vs. 19.5%) and severe visual loss (3.2% vs. 9.7%) at 4 years, compared with the 1975–1985 cohort, although the results do not differentiate type-1 from type-2 diabetes. This decline may be due to improved glycemic control in recent decades, but it is too early to know if the decrease is on-going. There is also a lack of data to compare the effects of different treatment regimens (e.g., multiple daily injections versus continuous subcutaneous insulin infusion) on the incidence and progression of DR. In the WESDR cohort, the annual incidence of PDR declined from 3.4% to 1.4% among the type-1 diabetes, and the incidence of clinically significant macular edema (CSME) from 1.0% to 0.4%. Nevertheless, this decline may not occur in low- or middle-income countries where the programs on early HbA1c screening and effective blood sugar and blood pressure control are unavailable. While studies have documented a decline in the incidence of DR among those with type-1 diabetes, the trend of DR among patients with type-2 diabetes remains uncertain.

2.1.4: Consequences of DR

DR is rapidly emerging as a global health issue that may threaten patients' visual acuity and visual functioning. Although treatment of established retinopathy can reduce the risk for visual loss by 60%, DR remains the leading cause of blindness among working-age adults in the world. The proportion of blindness attributable to DR ranges from 3–7% in much of South-East Asia and the Western Pacific region to 15–17% in the developed regions of the Americas and Europe. In addition to the direct consequences of visual impairment, DR, particularly in its vision-threatening stages, has a substantial and negative impact on patients' emotional well-being, although the exact mechanisms remain to be determined.

The financial costs of DR are mounting. Depending on the prevalence of diabetes and the organization of health systems, diabetes is estimated to account for 11.6% of the annual health-care budgets in most countries, and DR makes a big contribution to this figure. In the United States alone, the direct annual costs of DR were estimated to be USD$490 million in 2004. In Sweden, the annual average healthcare cost of any DR, PDR, and DME amounts to USD$93.6, USD$334.1, and USD$280.8, respectively, per patient. Health economic data on the cost of DR in low- and middle-income countries is currently not available.

2.1.5: INDIA-WISE

The Union Health Ministry's first National Diabetes and Diabetic Retinopathy Survey (2015-19) has revealed that the prevalence of Diabetic Retinopathy (DR) is 16.9 per cent while the prevalence of sight-threatening DR is 3.6 per cent.

It may be noted that diabetes and diabetic retinopathy are emerging as a significant non-communicable disease leading to ocular morbidity.

The survey has also revealed the prevalence of mild retinopathy at about 11.8 per cent. The prevalence of diabetes in the surveyed population has been recorded 11.8 per cent while the prevalence of Known Diabetes (KD) -- 8.0 per cent.

For this, the health ministry collaborated with the All India Institute of Medical Sciences (AIIMS) - RP Eye Centre to conduct this massive survey in which 21 districts were included. The total sample size for survey was 63,000 people aged 50 years and above.

"Our aim was to assess the prevalence of diabetic retinopathy and sight-threatening diabetic retinopathy (STDR) among people with diabetes and to evaluate the coverage of diabetic retinopathy examinations among people with known diabetes," (Prof) Dr Atul Kumar, Chief of RP Eye Centre at AIIMS told ANI.

The present survey was therefore planned to estimate the burden of diabetic retinopathy in the population aged >=50 years for assisting the planning and prioritization of diabetic eye services, said Dr Kumar.

The survey mentions that there are no recent studies on the prevalence of diabetic retinopathy (DR) in different parts of India. This used to make it difficult to identify where DR screening and treatment programs are most needed. Most available DR prevalence estimates are from diabetic clinics, which is subject to bias limiting their use in planning ophthalmic services for diabetics in the general population.

The WHO has estimated the global prevalence of diabetes among adults over 18 years of age as 8.5 per cent in 2014.

The International Diabetes Federation report 2017 estimated that there were 425 million diabetics in the world and this figure is expected to increase to 629 million by the year 2045.

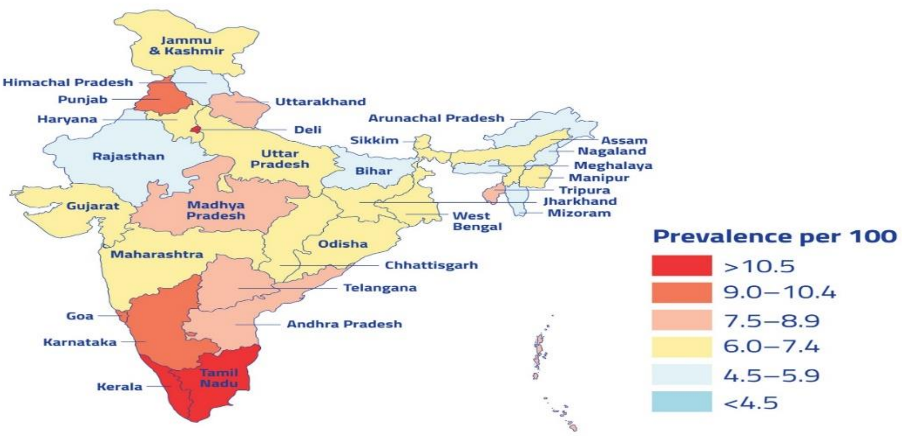

FIG 2.4: India wise statistics of DR

The increase is disproportionately high in developing countries. There are an estimated 72.96 million cases of diabetes in the adult population of India.

The urban prevalence ranges between 10.9 per cent-14.2 per cent and rural prevalence 3.0 per cent-7.8 per cent among the population aged 20 years above with a much higher prevalence among individuals aged over 50 years (INDIAB Study), states the survey report.

Union Health Minister, Dr Harsh Vardhan, today released the findings National Blindness and Visually Impaired Survey, 2015-19 at Prithvi Bhawan in observance of the World Sight Day. Vardhan also released the results of National Diabetes and Diabetic Retinopathy Survey, 2015-19.

2.2: SYMPTOMS AND EFFECTS

2.2.1: SYMPTOMS

You might not have symptoms in the early stages of diabetic retinopathy. As the condition progresses, diabetic retinopathy symptoms may include:

- Spots or dark strings floating in your vision (floaters)
- Blurred vision
- Fluctuating vision
- Impaired color vision
- Dark or empty areas in your vision
- Vision loss

Diabetic retinopathy usually affects both eyes.

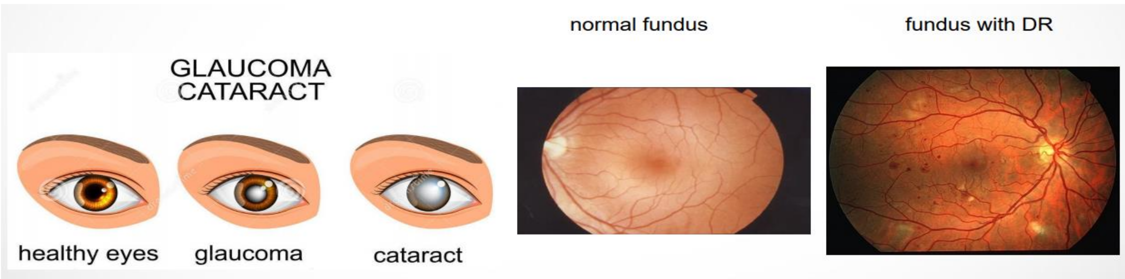

FIG 2.5: Different kinds of Eyes

2.2.2: EFFECTS

Over time, too much sugar in your blood can lead to the blockage of the tiny blood vessels that nourish the retina, cutting off its blood supply. As a result, the eye attempts to grow new blood vessels. But these new blood vessels do not develop properly and can leak easily.

There are two types of diabetic retinopathy:

Early diabetic retinopathy: In this more common form — called nonproliferative diabetic retinopathy (NPDR) — new blood vessels are not growing (proliferating).

When you have NPDR, the walls of the blood vessels in your retina weaken. Tiny bulges (microaneurysms) protrude from the vessel walls of the smaller vessels, sometimes leaking fluid and blood into the retina. Larger retinal vessels can begin to dilate and become irregular in diameter, as well. NPDR can progress from mild to severe, as more blood vessels become blocked.

Nerve fibers in the retina may begin to swell. Sometimes the central part of the retina (macula) begins to swell (macular edema), a condition that requires treatment.

Advanced diabetic retinopathy: Diabetic retinopathy can progress to this more severe type, known as proliferative diabetic retinopathy. In this type, damaged blood vessels close off, causing the growth of new, abnormal blood vessels in the retina, and can leak into the clear, jelly-like substance that fills the center of your eye (vitreous).

Eventually, scar tissue stimulated by the growth of new blood vessels may cause the retina to detach from the back of your eye. If the new blood vessels interfere with the normal flow of fluid out of the eye, pressure may build up in the eyeball. This can damage the nerve that carries images from your eye to your brain (optic nerve), resulting in glaucoma.

2.2.3: RISK FACTORS

Anyone who has diabetes can develop diabetic retinopathy. Risk of developing the eye condition can increase because of:

- Duration of diabetes — the longer you have diabetes, the greater your risk of developing diabetic retinopathy

- Poor control of your blood sugar level
- High blood pressure
- High cholesterol
- Pregnancy
- Tobacco use
- Being African American, Hispanic, or Native American

2.2.4: COMPLICATIONS

Diabetic retinopathy involves the abnormal growth of blood vessels in the retina. Complications can lead to serious vision problems:

- **Vitreous hemorrhage.** The new blood vessels may bleed into the clear, jelly-like substance that fills the center of your eye. If the amount of bleeding is small, you might see only a few dark spots (floaters). In more-severe cases, blood can fill the vitreous cavity and completely block your vision.
- Vitreous hemorrhage by itself usually doesn't cause permanent vision loss. The blood often clears from the eye within a few weeks or months. Unless your retina is damaged, your vision may return to its previous clarity.
- **Retinal detachment.** The abnormal blood vessels associated with diabetic retinopathy stimulate the growth of scar tissue, which can pull the retina away from the back of the eye. This may cause spots floating in your vision, flashes of light or severe vision loss.
- **Glaucoma.** New blood vessels may grow in the front part of your eye and interfere with the normal flow of fluid out of the eye, causing pressure in the eye to build up (glaucoma). This pressure can damage the nerve that carries images from your eye to your brain (optic nerve).
- **Blindness.** Eventually, diabetic retinopathy, glaucoma or both can lead to complete vision loss.

FIG 2.6: Normal vision V/S DR Vision

2.3: CLASSIFICATION

Classification is a large domain in the field of statistics and machine learning. Generally, classification can be broken down into two areas:

1. Binary classification, where we wish to group an outcome into one of two groups.
2. Multi-class classification, where we wish to group an outcome into one of multiple (more than two) groups.

In this post, the focus will be on using a variety of classification algorithms across both domains, less emphasis will be placed on the theory behind them.

We can use libraries in Python such as scikit-learn for machine learning models, and Pandas to import data as data frames.

2.3.1: Binary Classification

For binary classification, we are interested in classifying data into one of two *binary* groups - these are usually represented as 0's and 1's in our data.

We will look at data regarding coronary heart disease (CHD) in South Africa. The goal is to use different variables such as *tobacco usage*, *family history*, *ldl cholesterol levels*, *alcohol usage*, *obesity* and more.

We began by exploring the simplest form of classification: binary. This helped us to model data where our response could take one of two states.

We then moved further into multi-class classification, when the response variable can take any number of states.

We also saw how to fit and evaluate models with training and test sets. Furthermore, we could explore additional ways to refine model fitting among various algorithms.

Non-proliferative Diabetic Retinopathy (NPDR):

- No DR
- Very Mild NPDR
- Mild NPDR
- Moderate NPDR
- Severe NPDR
- Very Severe NPDR

Proliferative Diabetic Retinopathy (PDR):

- Mild to Moderate PDR
- High Risk PDR

FIG 2.7: Classification of Diabetic Retinopathy

2.4: PREVENTION AND TREATMENT

2.4.1: Prevention

You cannot always prevent diabetic retinopathy. However, regular eye exams, good control of your blood sugar and blood pressure, and early intervention for vision problems can help prevent severe vision loss.

If you have diabetes, reduce your risk of getting diabetic retinopathy by doing the following:

- **Manage your diabetes.** Make healthy eating and physical activity part of your daily routine.

- **Monitor your blood sugar level.** You may need to check and record your blood sugar level several times a day — more-frequent measurements may be required if you are ill or under stress.
- **Ask your doctor about a glycosylated haemoglobin test.** The glycosylated haemoglobin test, or haemoglobin A1C test, reflects your average blood sugar level for the two- to three-month period before the test. For most people, the A1C goal is to be under 7 percent.
- **Keep your blood pressure and cholesterol under control**. Eating healthy foods, exercising regularly, and losing excess weight can help. Sometimes medication is needed, too.
- **If you smoke or use other types of tobacco, ask your doctor to help you quit.** Smoking increases your risk of various diabetes complications, including diabetic retinopathy.
- **Pay attention to vision changes.** Contact your eye doctor right away if you experience sudden vision changes or your vision becomes blurry, spotty, or hazy.

Remember, diabetes does not necessarily lead to vision loss. Taking an active role in diabetes management can go a long way toward preventing complications.

Managing your diabetes is the best way to lower your risk of diabetic retinopathy. That means keeping your blood sugar levels as close to normal as possible. You can do this by getting regular physical activity, eating healthy, and carefully following your doctor’s instructions for your insulin or other diabetes medicines.

To help control your blood sugar, you will need a special test called an A1c test. This test shows your average blood sugar level over a 3-month period. Talk with your doctor about lowering your A1c level to help prevent or manage diabetic retinopathy.

The risk of developing diabetic retinopathy can be lessened by taking the following precautions:

- Taking a dilated eye examination once a year
- Managing diabetes strictly through medicine, insulin, diet, and exercise
- Test blood sugar levels regularly
- Test urine for ketone levels regularly

Although there is no absolute cure to Diabetic Retinopathy, diagnosis and treatment during the initial stages help maintain vision.

2.4.2: Treatment

Treating diabetic retinopathy depends on several factors, including the severity of the condition and how it has responded to previous treatments.

In the early stages, a doctor may decide to monitor the person's eyes closely without intervening. This approach is known as watchful waiting.

In some cases, a person may need a comprehensive dilated eye exam as often as every 2–4 months.

Individuals will need to work with their doctor to control diabetes. Good blood sugar control can significantly slow the development of diabetic retinopathy.

In most cases of advanced diabetic retinopathy, the person will require surgical treatment.

The following options are available:

2.4.2.1: Laser treatment

Scatter laser surgery, or panretinal photocoagulation, takes place in a doctor's office or an eye clinic. A doctor uses targeted lasers to shrink blood vessels in the eye and seal the leaks from abnormal blood vessels.

This treatment can either stop or slow down the leakage of blood and the buildup of fluid in the eye. People may need more than one session.

The procedure involves the doctor placing numbing medicine in the eye and then aiming a strong beam of light into the eye using a special lens.

Laser treatment comes with certain risks, such as a loss of peripheral vision, color vision, and night vision. A person can talk to their doctor about the relative benefits and risks of this treatment.

2.4.2.2: Injections

Certain medicines can reduce swelling and minimize leakage from blood vessels in the eyes. Medicines may include anti-VEGF drugs and corticosteroids.

Eye injections involve the doctor taking the following steps:

- Placing numbing medicine on the eye.
- Cleaning the eye to help prevent infections.
- Placing the medicine in the eye using a very small needle.

People may need to get regular injections, but over time, they usually require injections less frequently.

2.4.2.3: Eye surgery

If a person has problems with the retina or vitreous, they may benefit from a vitrectomy. This procedure is the removal of some of the vitreous from the eye.

A surgeon will perform this procedure in a hospital under general or monitored anesthesia.

The aim is to replace cloudy vitreous or blood to improve vision and to help the doctor find and repair any sources of retinal bleeding.

After removing the cloudy or bloody vitreous, the surgeon will insert a clear liquid or gas in its place. The body will absorb the liquid or gas over time and create new vitreous in its place.

After the surgery, the person will usually need to wear an eye patch for about a day and use eye drops to reduce swelling and prevent infections.

If the doctor puts a gas bubble in the eye, the person will need to hold their head in a certain position for a few days or weeks to make sure that the bubble stays in the right place. They will also need to avoid flying and visiting places at high altitudes until the bubble goes away.

Surgery is not a cure for diabetic retinopathy, but it may stop or slow the progression of symptoms. Diabetes is a long-term condition, and subsequent retinal damage and vision loss may still occur despite treatment.

3. LITERATURE SURVEY

3.1: EXISTING SYSTEM

Numerous techniques are tested by researchers in the area for DR classification with encouraging results. Recent work for addressing blood vessel segmentation includes the application of CNN (LeNet-5 architecture) as feature extractor. Three heads are used in this model at different layers of the convnent which are then fed into three random forests. The final classifier achieved an accuracy of 0.97 and 0.98 on the DRIVE and STARE dataset. An automatic segmentation of blood vessels in color fundus images is implemented by M. Melinscak et al using deep max-pooling convent to separate the blood vessels. The model contains a deep max-pooling convolutional neural networks to segment blood vessels. It deployed 10- layer architecture for achieving a maximum accuracy of around 0.94. It was carried around 4-convolutional and 4-max pooling layer with 2 additional fully connected layers for vessel segmentation. Automated analysis of DR using images processing techniques are introduced by Adarsh et al. In this approach, extraction of retinal blood vessels, exudate, micro-aneurysms, haemorrhages, and texture features takes place, followed by construction of Multiclass SVM using area of lesions and texture features. Impressive results are reported using the publicly available datasets DIARETDB0 and DIARETDB1 with accuracy of 0.96 and 0.946, respectively.

3.2: ALEXNET

3.2.1: History

AlexNet was primarily designed by Alex Krizhevsky. It was published with Ilya Sutskever and Krizhevsky’s doctoral advisor Geoffrey Hinton and is a Convolutional Neural Network or CNN.

After competing in ImageNet Large Scale Visual Recognition Challenge, AlexNet shot to fame. It achieved a top-5 error of 15.3%. This was 10.8% lower than that of runner up.

The primary result of the original paper was that the depth of the model was absolutely required for its high performance. This was quite expensive computationally but was made feasible due to GPUs or Graphical Processing Units, during training.

3.2.2: ALEXNET ARCHITECTURE

AlexNet was the first convolutional network which used GPU to boost performance.

1. AlexNet architecture consists of 5 convolutional layers, 3 max-pooling layers, 2 normalization layers, 2 fully connected layers, and 1 softmax layer.
2. Each convolutional layer consists of convolutional filters and a nonlinear activation function ReLU.
3. The pooling layers are used to perform max pooling.
4. Input size is fixed due to the presence of fully connected layers.
5. The input size is mentioned at most of the places as 224x224x3 but due to some padding which happens it works out to be 227x227x3.
6. AlexNet overall has 60 million parameters.

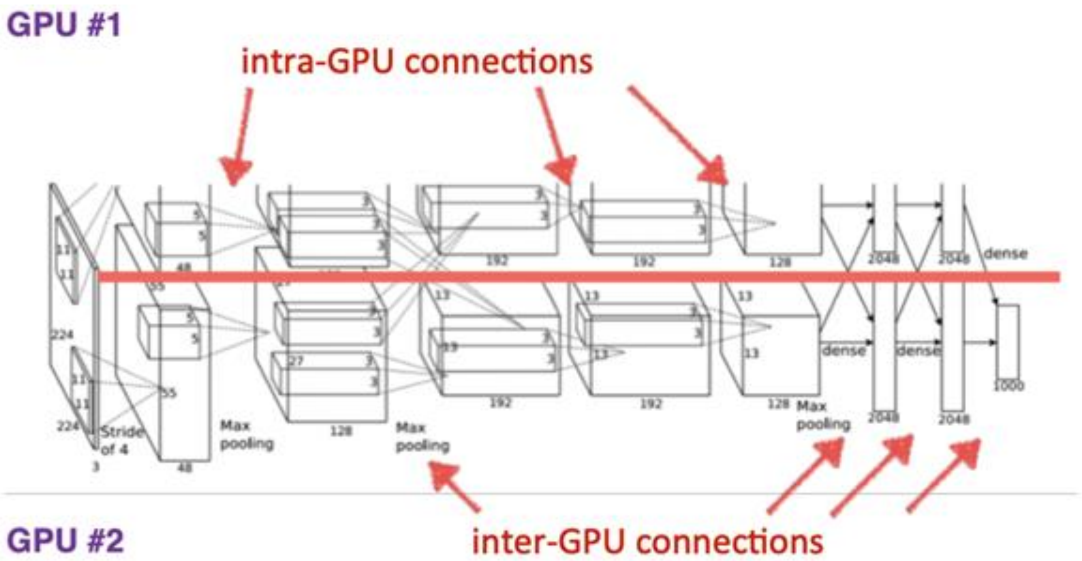

FIG 3.1: ALEXNET GPU connections

3.2.3: Model Details

The model which won the competition was tuned with specific details-

1. ReLU is an activation function.
2. Used Normalization layers which are not common anymore.
3. Batch size of 128.
4. SGD Momentum as learning algorithm.
5. Heavy Data Augmentation with things like flipping, jittering, cropping, color normalization, etc.

6. Assembling of models to get the best results.

AlexNet was trained on a GTX 580 GPU with only 3 GB of memory which could not fit the entire network. So, the network was split across 2 GPUs, with half of the neurons (feature maps) on each GPU.

Let us explore the architecture of this convolutional neural network.

Conv1

First, we will apply convolutional layer: filter size is f=11, and several filters is 96. In this convolutional layer we will also use a stride of 4. This stride of 4 will decrease dimensions of an input volume by a factor of 4, so after this first convolutional layer we will get 55×55×96 volume.

Maxpool1

The next layer is Maxpooling layer. In this layer we will use a 3×3 filter and a stride of 2. This will reduce the dimensions of 55×55×96 volume to 27×27×256 because we are using a stride of 2.

Conv2

Next layer is a convolutional layer with a filter size f=5 so we are also using a same convolution, so we will get the same dimensions 27×27×256.

Maxpool2

After this same convolution, we will apply Maxpooling with a 3×3 filter and a stride of 2. This will reduce the height and width to 13.

Conv3, Conv4, Conv5

Next, we will apply 3 3×3 same convolution layers with padding = 1 and a stride = 1. In the first two convolutional layers we will use 384 filters and in the third (in Conv5 layer) we will use 256 filters.

Maxpool3

Next, we will apply the third Maxpool layer with a stride of 2, so we have the volume with dimensions 6×6×256. If we multiply out these numbers 6×6×256=9216. We are going to unroll this into 9216 nodes.

FC6, FC7, FC8

Finally, AlexNet has tree fully connected layers. The first two layers have 4096 nodes, whereas the third fully connected layer has 1000 units. Finally, we have the softmax to output which one of 1000 classes the object could be.

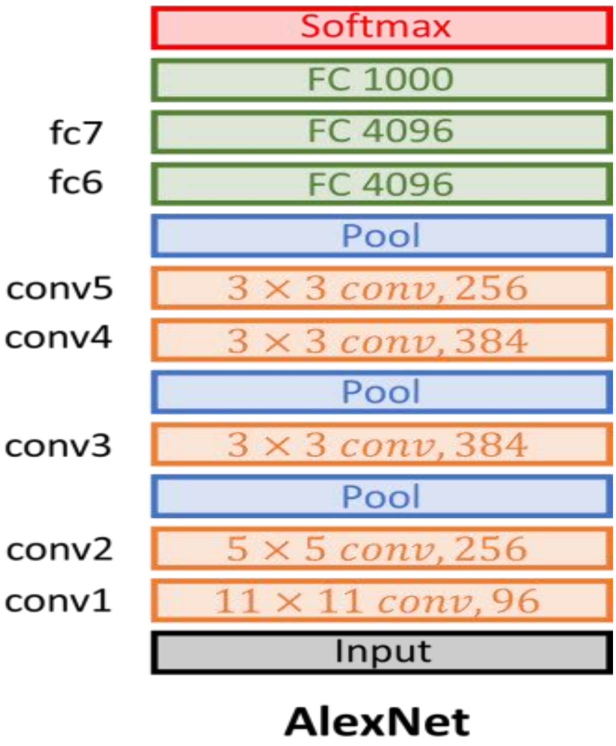

FIG 3.2: AlexNet architecture

AlexNet is like LeNet, but much larger. We have stated that LeNet−5 has about 60000 parameters. On the other hand, Alexnet has about 60 million parameters which are a big number of parameters to be learned. Splitting these layers across two (or more) GPUs may help to speed up the process of training. Notice also that here we have a lot of hyperparameters that authors of AlexNet had to come up with. Another aspect of this architecture they made it much better than the LeNet was using the ReLU activation function.

3.3: VGG 16 and 19

In this network smaller filters are used, but the network was built to be deeper than convolutional neural networks we have seen in the previous posts.

Remarkable thing about the VGG−16 is that instead of having so many hyper parameters we will use a much simpler network. We will focus on just having conv layers that are just 3×3 filters with a stride of 1, and with the same padding.

In all Maxpooling layers we will use 2×2 filters with a stride of 2.

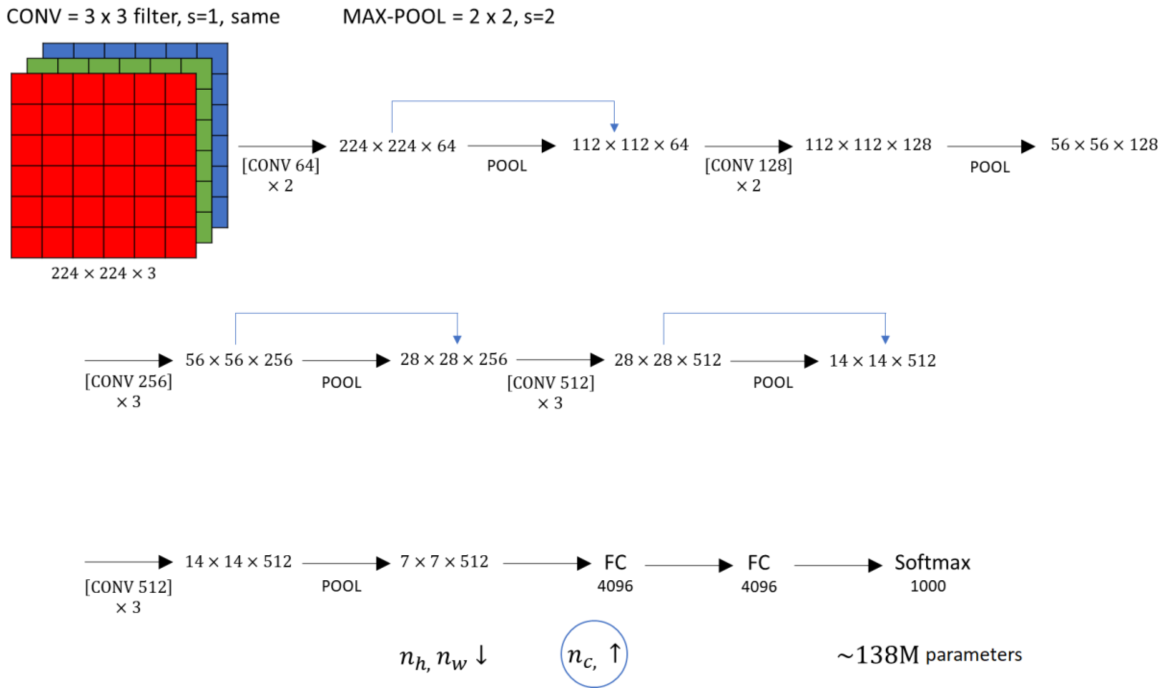

FIG 3.3: Architecture of VGG−16

Let us go through the architecture.

- The first two layers are convolutional layers with 3×3 filters, and in the first two layers we use 64 filters so we end up with a 224×224×64 volume because we're using same convolutions (height and width are the same). So, this (CONV64) ×2 represents that we have 2conv layers with 64 filters. The filters are always 3×3 with stride of 1 and they're always implemented with the same convolutions.
- Then, we use a pooling layer which will reduce height and width of a volume: it goes from 224×224×64 down to 112×112×64.
- Then we have a couple more conv layers. Here we use 128 filters and because we use the same convolutions, a new dimension will be 112×112×128.
- Then, a pooling layer is added so new dimension will be 56×56×128.
- 2conv layers with 256 filters.
- The pooling layer.
- A few more conv layers with 512 filters.
- A pooling layer.
- A few more conv layers with 512 filters.

- A pooling layer.
- At the end we have final 7×7×512 into Fully connected layer (FC) with 4096 units, and in a SoftMax output one of a 1000 classes.

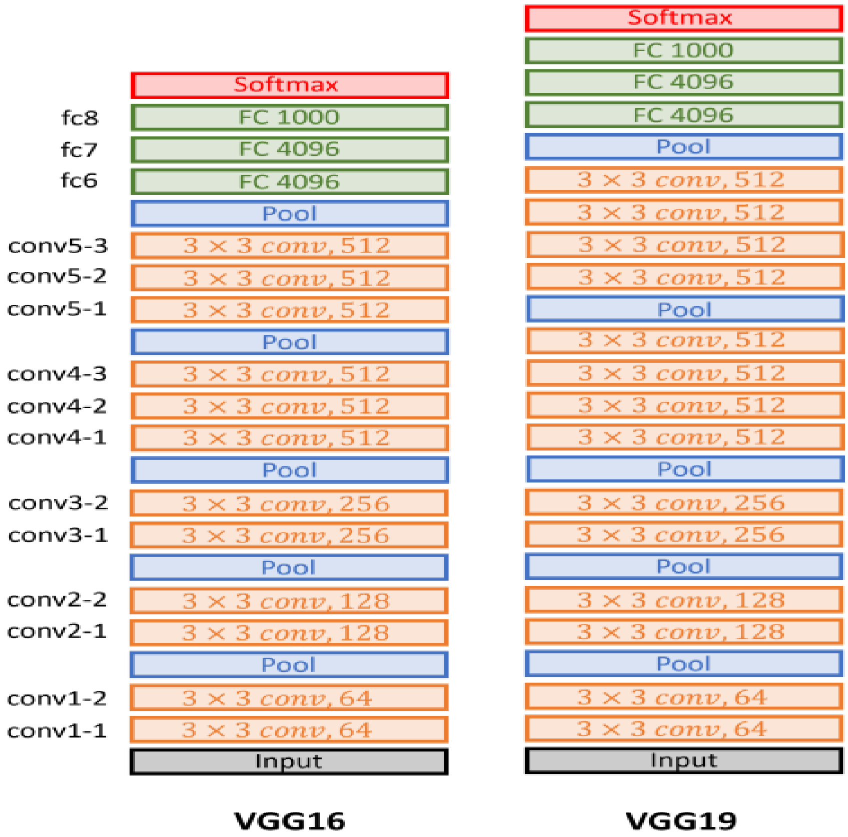

FIG 3.4: Layers of VGG−16 and VGG−19

Number 16 in the name VGG−16 refers to the fact that this has 16 layers that have some weights. This is a large network and has a total of about 138 million parameters. That is large even by modern standards. However, the simplicity of the VGG−16 architecture made it quite appealing. We can tell that this architecture is quite uniform. There are a few conv layers followed by a pooling layer which reduces the height and width of a volume. If we look at several filters, we use we can see that we have 64 filters and then we double it to 128 and then to 256 and in the last layers we use 512 layers. The number of filters we use is roughly doubling on every step or doubling through every stack of conv layer and that is another simple principle used to design the architecture of this network. The main downside was that it was a large network in terms of the number of parameters to be trained. VGG−19 neural network which is bigger than VGG−16, but because VGG−16 does almost as well as the VGG−19 a lot of people will use VGG−16.

3.4 GOOGLE NET

Google Net (or Inception V1) was proposed by research at Google (with the collaboration of various universities) in 2014 in the research paper titled "Going Deeper with Convolutions". This architecture was the winner at the ILSVRC 2014 image classification challenge. It has provided a significant decrease in error rate as compared to previous winners AlexNet (Winner of ILSVRC 2012) and ZF-Net (Winner of ILSVRC 2013) and significantly less error rate than VGG (2014 runner up). This architecture uses techniques such as 1×1 convolutions in the middle of the architecture and global average pooling.

3.4.1: Features of GoogleNet

The GoogLeNet architecture is very different from previous state-of-the-art architectures such as AlexNet and ZF-Net. It uses many kinds of methods such as 1×1 convolution and global average pooling that enables it to create deeper architecture. In the architecture, we will discuss some of these methods:

- ***1×1* convolution:** The inception architecture uses 1×1 convolution in its architecture. These convolutions used to decrease the number of parameters (weights and biases) of the architecture. By reducing the parameters, we also increase the depth of the architecture. Let us look at an example of a 1×1 convolution below:
  - For Example, if we want to perform 5×5 convolution having 48 filters without using 1×1 convolution as intermediate:

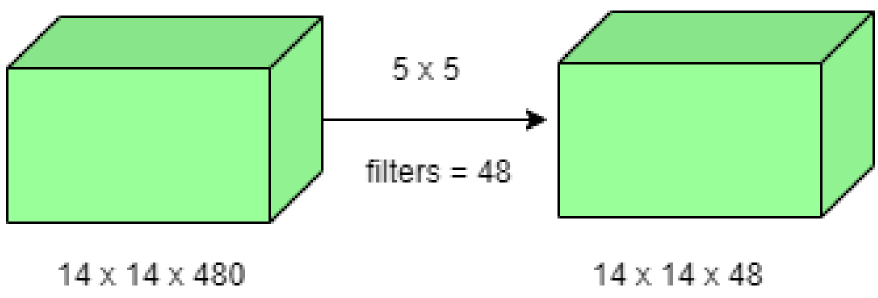

FIG 3.5: Convolution filter = 48

- Total Number of operations: *(14 x 14 x 48) x (5 x 5 x 480) = 112.9 M*
- With 1×1 convolution :

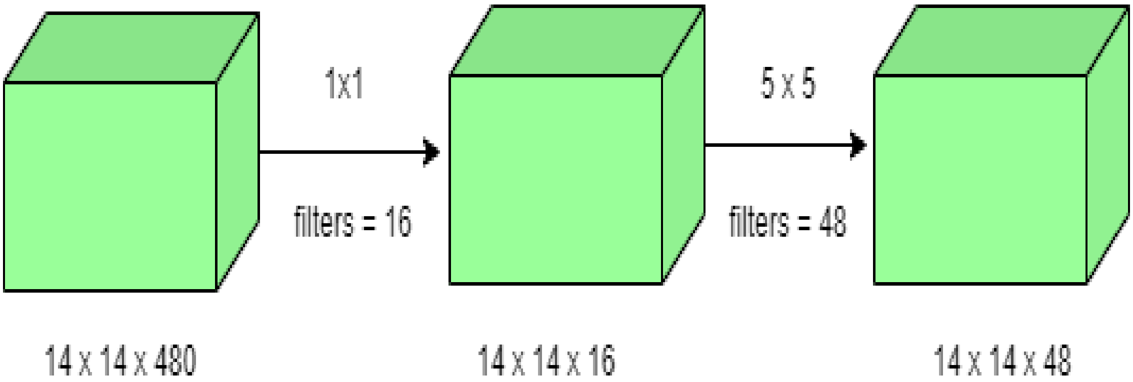

FIG 3.6: Convolution filter stack with 16 and 48

- *(14 x 14 x 16) x (1 x 1 x 480) + (14 x 14 x 48) x (5 x 5 x 16) = 1.5M + 3.8M = 5.3M* which is much smaller than 112.9M.
- **Global Average Pooling:**
  In the previous architecture such as AlexNet, the fully connected layers are used at the end of the network. These fully connected layers contain most parameters of many architectures that causes an increase in computation cost.
  In GoogLeNet architecture, there is a method called global average pooling is used at the end of the network. This layer takes a feature map of 7×7 and averages it to 1×1. This also decreases the number of trainable parameters to 0 and improves the top-1 accuracy by 0.6%
- **Inception Module:**
  The inception module is different from previous architectures such as AlexNet, ZF-Net. In this architecture, there is a fixed convolution size for each layer.
  In the Inception module 1×1, 3×3, 5×5 convolution and 3×3 max pooling performed in a parallel way at the input and the output of these are stacked together to generated final output. The idea behind that convolution filters of different sizes will handle objects at multiple scale better.

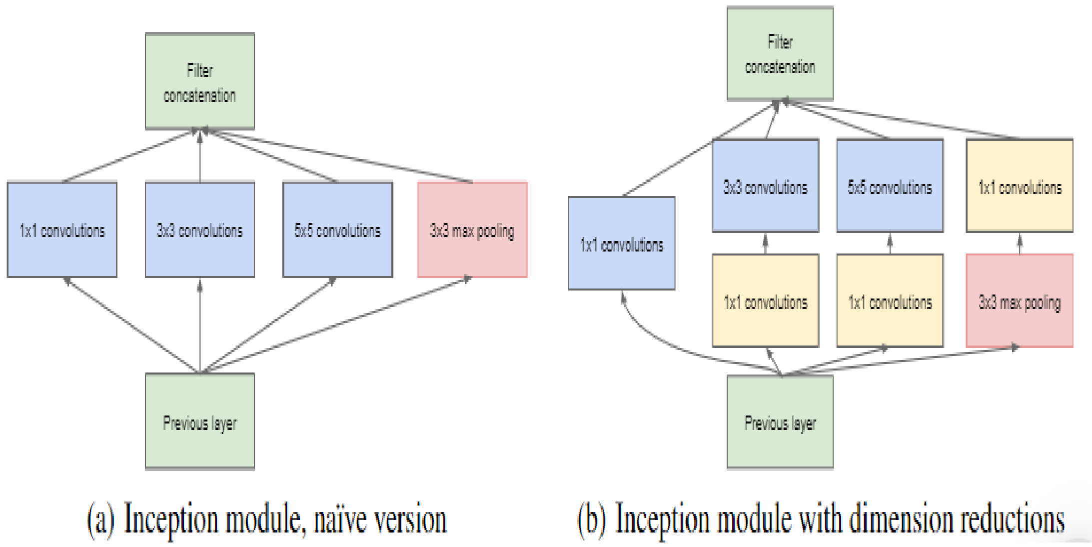

FIG 3.7: Inception Module

- **Auxiliary Classifier for Training:**
  Inception architecture used some intermediate classifier branches in the middle of the architecture, these branches are used during training only. These branches consist of a 5×5 average pooling layer with a stride of 3, a *1×1* convolutions with *128* filters, two fully connected layers of 1024 outputs and 1000 outputs and a softmax classification layer. The generated loss of these layers added to total loss with a weight of 0.3. These layers help in combating gradient vanishing problem and provide regularization.

3.4.2: Model Architecture

The overall architecture is 22 layers deep. The architecture was designed to keep computational efficiency in mind. The idea behind that the architecture can be run on individual devices even with low computational resources. The architecture also contains two auxiliary classifier layers connected to the output of Inception (4a) and Inception (4d) layers.

- Below is Layer by Layer architectural details of GoogLeNet.

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

TABLE 3.1: Model Architecture of GoogleNet

The architectural details of auxiliary classifiers as follows:

- An average pooling layer of filter size 5×5 and stride 3.
- A 1×1 convolution with 128 filters for dimension reduction and ReLU activation.
- A fully connected layer with 1025 outputs and ReLU activation
- Dropout Regularization with dropout ratio = 0.7
- A softmax classifier with 1000 classes output like the main softmax classifier.

This architecture takes image of size *224 x 224* with RGB colour channels. All the convolutions inside this architecture uses Rectified Linear Units (ReLU) as their activation functions.

GoogLeNet was the winner at ILSRVRC 2014 taking 1st place in both classification an detection task. It has top-5 error rate of 6.67% in classification task. An ensemble of 6 GoogLeNets gives 43.9 % mAP on ImageNet test set.

FIG 3.8: Flowchart of GoogleNet

4. DEEP LEARNING MODEL

4.1: INTRO TO DEEP LEARNING

Deep learning is a branch of machine learning which is completely based on artificial neural networks, as neural network is going to mimic the human brain so deep learning is also a kind of mimic of human brain. In deep learning, we don't need to explicitly program everything. The concept of deep learning is not new. It has been around for a couple of years now. It's on hype nowadays because earlier we did not have that much processing power and a lot of data.

4.1.1: Architectures

1. **Deep Neural Network** – It is a neural network with a certain level of complexity (having multiple hidden layers in between input and output layers). They are capable of modelling and processing non-linear relationships.
2. **Deep Belief Network (DBN)** – It is a class of Deep Neural Network. It is multi-layer belief networks.

   Steps for performing DBN:

   a. Learn a layer of features from visible units using Contrastive Divergence algorithm.

   b. Treat activations of previously trained features as visible units and then learn features of features.

   c. Finally, the whole DBN is trained when the learning for the final hidden layer is achieved.
3. **Recurrent** (perform same task for every element of a sequence) **Neural Network** – Allows for parallel and sequential computation. Similar to the human brain (large feedback network of connected neurons). They are able to remember important things about the input they received and hence enables them to be more precise.

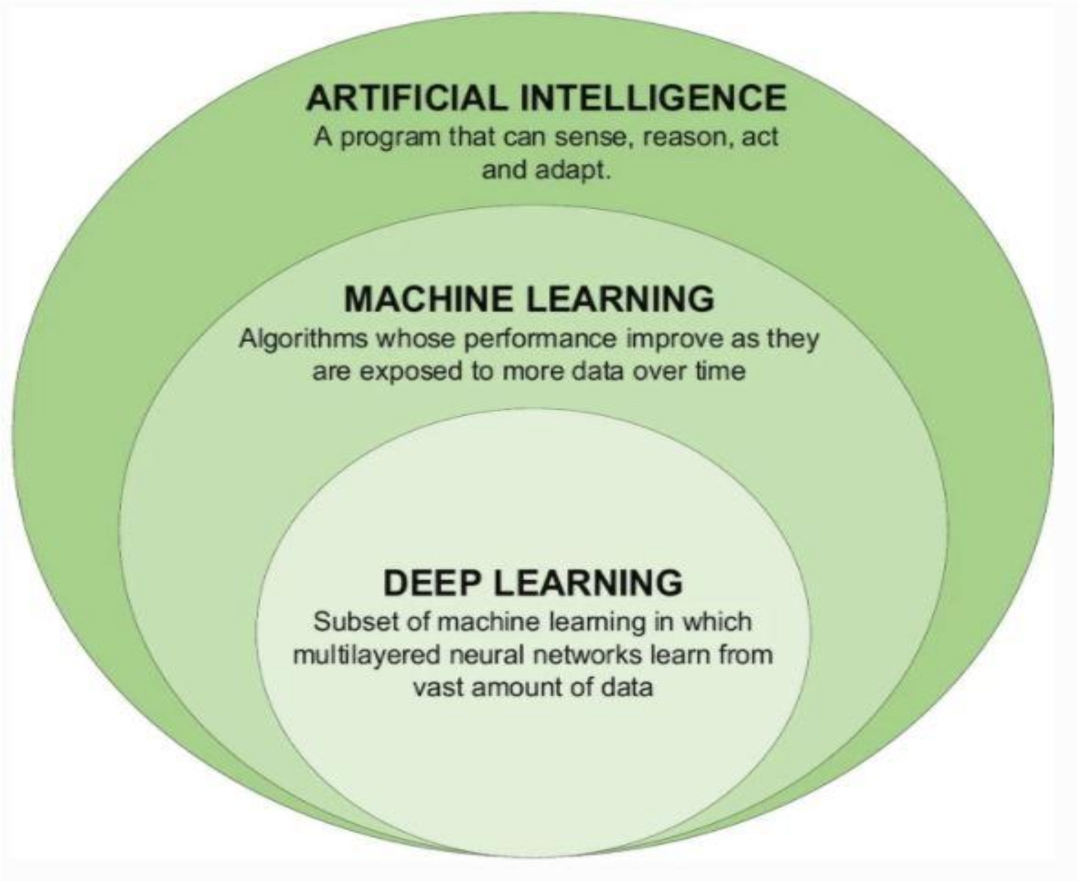

FIG 4.1: Deep Learning comparison

4.1.2: Working

First, we need to identify the actual problem to get the right solution and it should be understood, the feasibility of the Deep Learning should also be checked (whether it should fit Deep Learning or not). Second, we need to identify the relevant data which should correspond to the actual problem and should be prepared accordingly. Third, Choose the Deep Learning Algorithm appropriately. Fourth, Algorithm should be used while training the dataset. Fifth, Final testing should be done on the dataset.

4.2: ABOUT DATASET

4.2.1: Overview

Data is collected from the dataset provide by the Kaggle coding website and maintained by EyePacs. The dataset consists of colour fundus photographs collected from various sources. The images are classified based on the severity of DR, where each image was assigned to a class by a trained clinician1. The figure below shows the various stages of diabetic retinopathy (DR).

4.2.2: Class Imbalance

The class labels of the dataset are highly imbalanced i.e. more than 73% of the class are negative, which makes our model difficult to train. Table I below shows the class proportion statistics, where PDR and NPDR refers to proliferative and Non-proliferative DR respectively.

4.3 DATA AUGUMNTATION AND ABOUT DATASET

4.3.1: Data Augmentation

When you show a Neural Net different variation of the same image, it helps prevent overfitting. It also forces the Neural Net to memorize the key features and helps in generating additional data. Augmented images were created to increase the class size as there were limited number of training samples for some of the classes. Brightness of each of the images created after pre-processing were adjusted by converting the RGB image to float representation followed by converting into the original data type. This is done by adding a delta value to all the components of the image. The images are scaled appropriately, and both the image and delta are converted to float prior to addition. As the addition to the image is performed in floating point representation, the delta must be in the range [0, 1) whereas the pixel values are in [0, 1). The original and the brightness adjusted images are then rotated by 90 and 180 degree which inherently increase the class size 6 times. This makes our model immune to different orientations and lighting conditions.

4.3.2: Data Augmentation by Mirroring

Let us say we have an image of a cat in our training set. The mirror image is also a valid image of a cat. This mean that we can double the size of the training datasets by simply flipping the image above the vertical axis.

4.3.3: Data Augmentation by Random Crops

Also, cropping the original image randomly will lead to additional data that is just a shifted version of the original data.

The authors of AlexNet extracted random crops sized 227×227 from inside the 256×256 image boundary and used this as the network's inputs. Using this method, they increased the size of the data by a factor of 2048.

4.4: CONVOLUTIONAL NEURAL NETWORK

In deep learning, a **convolutional neural network** (**CNN** or **ConvNet**) is a class of deep neural network, most commonly applied to analyse visual imagery. They are also known as **shift invariant** or **space invariant artificial neural networks** (**SIANN**), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps. Counter-intuitively, most convolutional neural networks are only equivariant, as opposed to invariant, to translation.They have applications in image and video recognition, recommender systems, image classification, image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks makes them prone to overfitting data. Typical ways of regularization, or preventing overfitting, include penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in their filters. Therefore, on a scale of connectivity and complexity, CNNs are on the lower extreme.

Convolutional networks were inspired by biological processes 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 to optimize the filters (or kernels) through automated learning, whereas in traditional algorithms these filters are hand-engineered. This independence from prior knowledge and human intervention in feature extraction is a major advantage.

4.4.1: ARCHITECTURE

A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution. In a convolutional neural network, the hidden layers include layers that perform convolutions. Typically, this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix. This product is usually the Frobenius inner product, and its activation function is commonly ReLU. As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which in turn contributes to the input of the next layer. This is followed by other layers such as pooling layers, fully connected layers, and normalization layers.

4.4.2: CONVOLUTIONAL LAYERS

In a CNN, the input is a tensor with a shape: (number of inputs) x (input height) x (input width) x (input channels). After passing through a convolutional layer, the image becomes abstracted to a feature map, also called an activation map, with shape: (number of inputs) x (feature map height) x (feature map width) x (feature map channels). A convolutional layer within a CNN generally has the following attributes:

- Convolutional filters/kernels defined by a width and height (hyper-parameters).
- The number of input channels and output channels (hyper-parameters). One layer's input channel must equal the number of output channels (also called depth) of its input.

- Additional hyperparameters of the convolution operation, such as: padding, stride, and dilation.

Convolutional layers convolve the input and pass its result to the next layer. This is like the response of a neuron in the visual cortex to a specific stimulus. Each convolutional neuron processes data only for its receptive field. Although fully connected feed forward neural networks can be used to learn features and classify data, this architecture is generally impractical for larger inputs such as high resolution images. It would require a very high number of neurons, even in a shallow architecture, due to the large input size of images, where each pixel is a relevant input feature. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10,000 weights for *each* neuron in the second layer. Instead, convolution reduces the number of free parameters, allowing the network to be deeper. For example, regardless of image size, using a 5 x 5 tiling region, each with the same shared weights, requires only 25 learnable parameters. Using regularized weights over fewer parameters avoids the vanishing gradients and exploding gradients problems seen during back propagation in traditional neural networks. Furthermore, convolutional neural networks are ideal for data with a grid-like topology (such as images) as spatial relations between separate features are considered during convolution and/or pooling.

4.4.3: Pooling layers

Convolutional networks may include local and/or global pooling layers along with traditional convolutional layers. Pooling layers reduce the dimensions of data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling combines small clusters, tiling sizes such as 2 x 2 are commonly used. Global pooling acts on all the neurons of the feature map. There are two common types of pooling in popular use: max and average. *Max pooling* uses the maximum value of each local cluster of neurons in the feature map, while *average pooling* takes the average value.

4.4.4: Fully connected layers

Fully connected layers connect every neuron in one layer to every neuron in another layer. It is the same as a traditional multi-layer perceptron neural network (MLP). The flattened matrix goes through a fully connected layer to classify the images.

4.4.5: Receptive field

In neural networks, each neuron receives input from some number of locations in the previous layer. In a convolutional layer, each neuron receives input from only a restricted area of the previous layer called the neuron's *receptive field*. Typically, the area is a square (e.g. 5 by 5 neurons). Whereas, in a fully connected layer, the receptive field is the *entire previous layer*. Thus, in each convolutional layer, each neuron takes input from a larger area in the input than previous layers. This is due to applying the convolution over and over, which considers the value of a pixel, as well as its surrounding pixels. When using dilated layers, the number of pixels in the receptive field remains constant, but the field is more sparsely populated as its dimensions grow when combining the effect of several layers.

4.4.6: Weights

Each neuron in a neural network computes an output value by applying a specific function to the input values received from the receptive field in the previous layer. The function that is applied to the input values is determined by a vector of weights and a bias (typically real numbers). Learning consists of iteratively adjusting these biases and weights.

The vector of weights and the bias are called *filters* and represent particular features of the input (e.g., a particular shape). A distinguishing feature of CNNs is that many neurons can share the same filter. This reduces the memory footprint because a single bias and a single vector of weights are used across all receptive fields that share that filter, as opposed to each receptive field having its own bias and vector weighting.

Convolutional neural networks are variants of multilayer perceptrons, designed to emulate the behavior of a visual cortex. These models mitigate the challenges posed by the MLP architecture by exploiting the strong spatially local correlation present in natural images. As opposed to MLPs, CNNs have the following distinguishing features:

- 3D volumes of neurons. The layers of a CNN have neurons arranged in 3 dimensions: width, height and depth. Where each neuron inside a convolutional layer is connected to only a small region of the layer before it, called a receptive field. Distinct types of layers, both locally and completely connected, are stacked to form CNN architecture.

- Local connectivity: following the concept of receptive fields, CNNs exploit spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers. The architecture thus ensures that the learned "filters" produce the strongest response to a spatially local input pattern. Stacking many such layers leads to non-linear filters that become increasingly global (i.e. responsive to a larger region of pixel space) so that the network first creates representations of small parts of the input, then from them assembles representations of larger areas.
- Shared weights: In CNNs, each filter is replicated across the entire visual field. These replicated units share the same parameterization (weight vector and bias) and form a feature map. This means that all the neurons in each convolutional layer respond to the same feature within their specific response field. Replicating units in this way allows for the resulting activation map to be equivariant under shifts of the locations of input features in the visual field, i.e. they grant translational equivariance - given that the layer has a stride of one.
- Pooling: In a CNN's pooling layers, feature maps are divided into rectangular sub-regions, and the features in each rectangle are independently down sampled to a single value, commonly by taking their average or maximum value. In addition to reducing the sizes of feature maps, the pooling operation grants a degree of local translational invariance to the features contained therein, allowing the CNN to be more robust to variations in their positions.

Weight sharing dramatically reduces the number of free parameters learned, thus lowering the memory requirements for running the network and allowing the training of larger, more powerful networks.

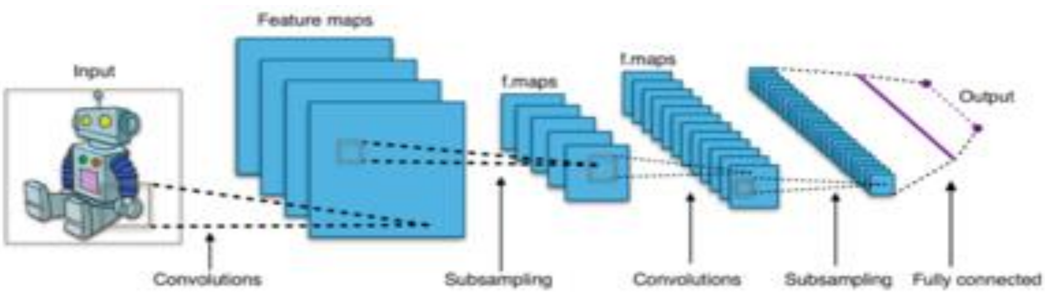

FIG 4.2: Convolutional Network

4.4.7: Parameter sharing

A parameter sharing scheme is used in convolutional layers to control the number of free parameters. It relies on the assumption that if a patch feature is useful to compute at some spatial position, then it should also be useful to compute at other positions. Denoting a single 2-dimensional slice of depth as a *depth slice*, the neurons in each depth slice are constrained to use the same weights and bias.

Since all neurons in a single depth slice share the same parameters, the forward pass in each depth slice of the convolutional layer can be computed as a convolution of the neuron's weights with the input volume. Therefore, it is common to refer to the sets of weights as a filter (or a kernel), which is convolved with the input. The result of this convolution is an activation map, and the set of activation maps for each different filter are stacked together along the depth dimension to produce the output volume. Parameter sharing contributes to the translation invariance of the CNN architecture.

Sometimes, the parameter sharing assumption may not make sense. This is especially the case when the input images to a CNN have some specific centered structure; for which we expect completely different features to be learned on different spatial locations. One practical example is when the inputs are faces that have been centered in the image: we might expect different eye-specific or hair-specific features to be learned in different parts of the image. In that case it is common to relax the parameter sharing scheme, and instead simply call the layer a "locally connected layer".

4.4.8: Pooling layer

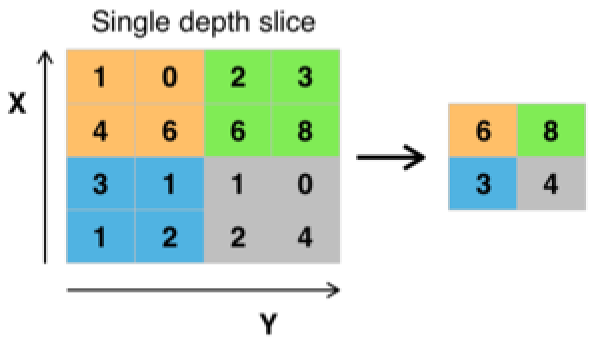

FIG 4.3: Max pooling filter 2x2

Max pooling with a 2x2 filter and stride = 2

Another important concept of CNNs is pooling, which is a form of non-linear down-sampling. There are several non-linear functions to implement pooling, where *max pooling* is the most common. It partitions the input image into a set of rectangles and, for each such sub-region, outputs the maximum.

Intuitively, the exact location of a feature is less important than its rough location relative to other features. This is the idea behind the use of pooling in convolutional neural networks. This is known as down-sampling It is common to periodically insert a pooling layer between successive convolutional layers (each one typically followed by an activation function, such as a ReLU layer) in a CNN architecture.[62]:460–461 While pooling layers contribute to local translation invariance, they do not provide global translation invariance in a CNN, unless a form of global pooling is used.[4][61] The pooling layer commonly operates independently on every depth, or slice, of the input and resizes it spatially. A very common form of max pooling is a layer with filters of size 2×2, applied with a stride of 2, which subsamples every depth slice in the input by 2 along both width and height, discarding 75% of the activations:

In this case, every max operation is over 4 numbers. The depth dimension remains unchanged (this is true for other forms of pooling as well).

In addition to max pooling, pooling units can use other functions, such as average pooling or ℓ_2-norm pooling. Average pooling was often used historically but has recently fallen out of favor compared to max pooling, which generally performs better in practice.

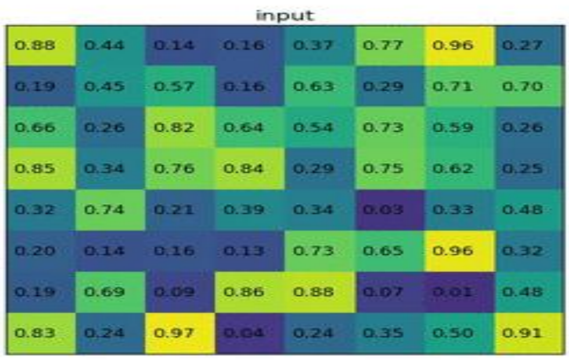

FIG 4.4: Pooling

RoI pooling to size 2x2. In this example region proposal (an input parameter) has size 7x5.

"Region of Interest" pooling (also known as RoI pooling) is a variant of max pooling, in which output size is fixed and input rectangle is a parameter.

Pooling is an important component of convolutional neural networks for object detection based on the Fast R-CNN architecture.

4.4.9: ReLU layer

ReLU is the abbreviation of rectified linear unit, which applies the non-saturating activation function .It effectively removes negative values from an activation map by setting them to zero. It introduces nonlinearities to the decision function and in the overall network without affecting the receptive fields of the convolution layers.

Other functions can also be used to increase nonlinearity, for example the saturating hyperbolic tangent and the sigmoid function ReLU is often preferred to other functions because it trains the neural network several times faster without a significant penalty to generalization accuracy.

4.4.10: Fully connected layer

After several convolutional and max pooling layers, the final classification is done via fully connected layers. Neurons in a fully connected layer have connections to all activations in the previous layer, as seen in regular (non-convolutional) artificial neural networks. Their activations can thus be computed as an affine transformation, with matrix multiplication followed by a bias offset (vector addition of a learned or fixed bias term).

4.4.11: Image recognition

CNNs are often used in image recognition systems. In 2012 an error rate of 0.23% on the MNIST database was reported. Another paper on using CNN for image classification reported that the learning process was "surprisingly fast"; in the same paper, the best published results as of 2011 were achieved in the MNIST database and the NORB database. Subsequently, a similar CNN called AlexNet won the ImageNet Large Scale Visual Recognition Challenge 2012.

When applied to facial recognition, CNNs achieved a large decrease in error rate. Another paper reported a 97.6% recognition rate on "5,600 still images of more than 10 subjects". CNNs were used to assess video quality in an objective way after manual training; the resulting system had a very low root mean square error.

The ImageNet Large Scale Visual Recognition Challenge is a benchmark in object classification and detection, with millions of images and hundreds of object classes. In the ILSVRC 2014, a large-scale visual recognition challenge, almost every highly ranked team used CNN as their basic framework. The winner GoogLeNet (the foundation of Deep Dream) increased the mean average precision of object detection to 0.439329, and reduced classification error to 0.06656, the best result to date. Its network applied more than 30 layers. That performance of convolutional neural networks on the ImageNet tests was close to that of humans.

In 2015 a many-layered CNN demonstrated the ability to spot faces from a wide range of angles, including upside down, even when partially occluded, with competitive performance. The network was trained on a database of 200,000 images that included faces at various angles and orientations and a further 20 million images without faces. They used batches of 128 images over 50,000 iterations.

4.5 RESNET – 18 Models

ResNet-18 is a convolutional neural network that is 18 layers deep. You can load a pretrained version of the network trained on more than a million images from the ImageNet database. The pretrained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. You can use classify to classify new images using the ResNet-18 model. Follow the steps of Classify Image Using GoogLeNet and replace GoogLeNet with ResNet-18.

To retrain the network on a new classification task, follow the steps of Train Deep Learning Network to Classify New Images and load ResNet-18 instead of GoogLeNet.

Resnets are built out of a **residual block**. Let us first describe what this is!

It consists of two layers of a neural network where we start off with some activation $a^{[l]}$, then we are passing it through a residual block and we will finally get $a^{[l+2]}$, as shown in the picture below.

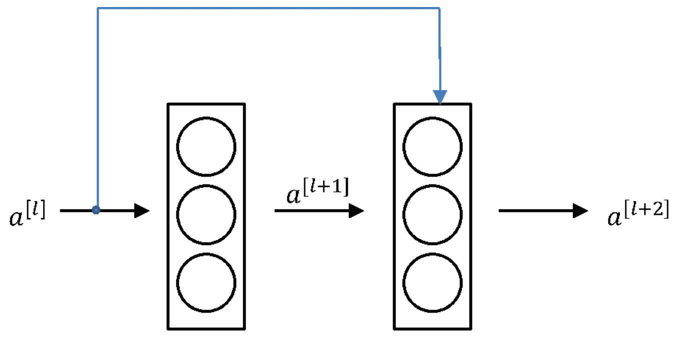

FIG 4.5: A Residual block

Comparison of Plain networks and Residual networks

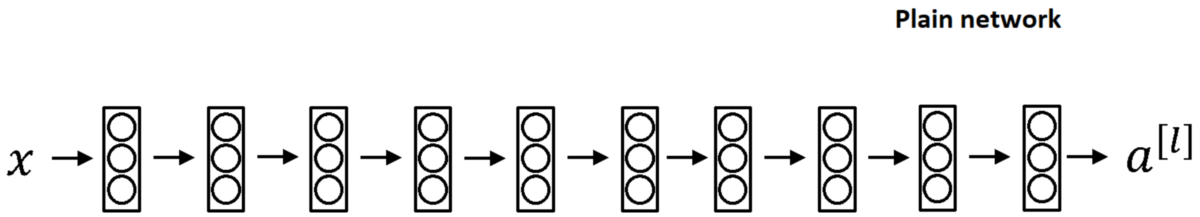

FIG 4.6: Plain network

In the above picture a “Plain network” is shown. It is a terminology of the Resnet paper. To turn this into a Resnet, we will add skip connections (or short connections). The picture below shows residual blocks stacked together and they represent a Residual network.

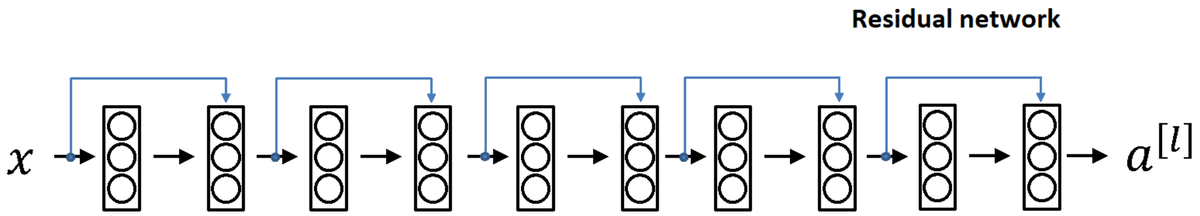

FIG 4.7: Residual network

It turns out that if we use a standard optimization algorithm such as gradient descents or one another algorithm to train a plain network we find that as we increase the number of layers, the training error will tend to decrease after a while but then it’ll tend to increase. In theory, as we make a neural network deeper, it should only do better and better on the training set. So, in theory, having a deeper network should only hope, however, in practice having a very deep plain network means that our optimization algorithm would have much harder time in training. Training error gets worse if we pick a network that is too deep.

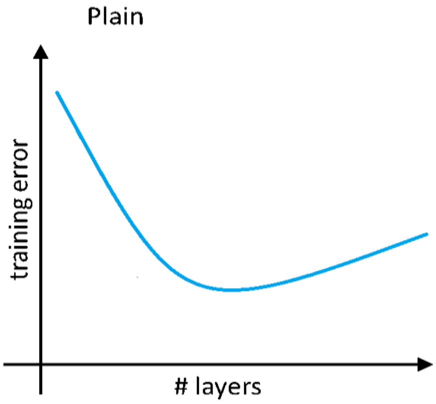

FIG 4.8: Training error for Plain networks

On the other hand, Resnets tend to have a constantly decreasing training error. This happens even if we train a network with over 100 layers. ResNets allow us to train much deeper neural networks without a loss in performance. Maybe at some point this graphic bellow will plateau and will flatten out and it doesn't help to increase a number of layers.

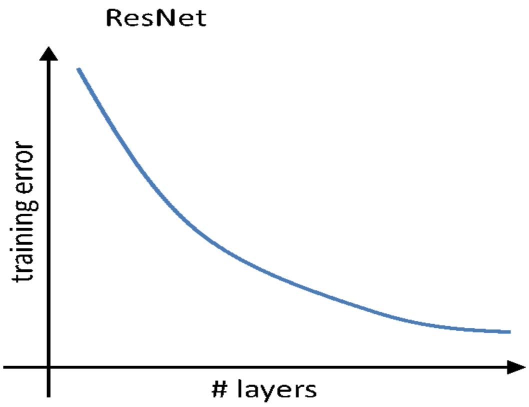

FIG 4.9: Training error of Residual network

4.6 DENSE LAYERS

Dense implements the operation: output = activation (dot(input, kernel) + bias) where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer (only applicable if use bias is True). These are all attributes of Dense.

Note: If the input to the layer has a rank greater than 2, then Dense computes the dot product between the inputs and the kernel along the last axis of the inputs and axis 1 of the kernel (using tf.tensordot).

Besides, layer attributes cannot be modified after the layer has been called once (except the trainable attribute). When a popular kwarg input_shape is passed, then keras will create an input layer to insert before the current layer. This can be treated equivalent to explicitly defining an InputLayer.

4.6.1: Arguments

- **units**: Positive integer, dimensionality of the output space.
- **activation**: Activation function to use. If you do not specify anything, no activation is applied (i.e. "linear" activation: a(x) = x).
- **use_bias**: Boolean, whether the layer uses a bias vector.
- **kernel_initializer**: Initializer for the kernel weights matrix.
- **bias_initializer**: Initializer for the bias vector.
- **kernel_regularizer**: Regularizer function applied to the kernel weights matrix.
- **bias_regularizer**: Regularizer function applied to the bias vector.
- **activity_regularizer**: Regularizer function applied to the output of the layer (its "activation").
- **kernel_constraint**: Constraint function applied to the kernel weights matrix.
- **bias_constraint**: Constraint function applied to the bias vector.

4.6.2: Input shape

N-D tensor with shape: (batch_size, ..., input_dim). The most common situation would be a 2D input with shape (batch_size, input_dim).

4.6.3: Output shape

N-D tensor with shape: (batch_size, ..., units). For instance, for a 2D input with shape (batch_size, input_dim), the output would have shape (batch_size, units).

5. SYSTEM REQUIREMENTS

SOFTWARE REQUIREMENTS

Operating System	:	Windows 7 (or more versions), Linux, MacOS.
Language	:	Python (Version 3 or more)
IDE	:	Anaconda – Spyder IDE or Google Colab

HARDWARE REQUIREMENTS

Processor	:	Intel Pentium.
RAM	:	4GB (min)

5.1: SOFTWARE DESCRIPTION

5.1.1: Python

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language. Python is a popular programming language. It was created in 1991 by Guido van Rossum.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- System scripting.

Python can be used on a server to create web applications. Python can be used alongside software to create workflows. Python can connect to database systems. It can also read and modify files. Python can be used to handle big data and perform complex mathematics. Python can be used for rapid prototyping, or for production-ready software development. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc). Python has a simple syntax like the English language. Python has syntax that allows developers to write programs with fewer lines than some other programming languages.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is like PERL and PHP.
- **Python is Interactive** − you can sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

5.1.2: History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

5.1.3: Python Features

Python's features include −

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.

- **Easy-to-maintain** − Python's source code is fairly easy-to-maintaining.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to bytecode for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X.

5.1.4: Python Syntax compared to other programming languages

- Python was designed to for readability and has some similarities to the English language with influence from mathematics.

- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope, such as the scope of loops, functions, and classes. Other programming languages often use curly brackets for this purpose.

5.1.5: Anaconda

Anaconda is the most popular python data science platform.

Anaconda Distribution

With over 6 million users, the open source Anaconda Distribution is the fastest and easiest way to do Python and R data science and machine learning on Linux, Windows, and Mac OS X. It's the industry standard for developing, testing, and training on a single machine.

Anaconda Enterprise

Anaconda Enterprise is an AI/ML enablement platform that empowers organizations to develop, govern, and automate AI/ML and data science from laptop through training to production. It lets organizations scale from individual data scientists to collaborative teams of thousands, and to go from a single server to thousands of nodes for model training and deployment.

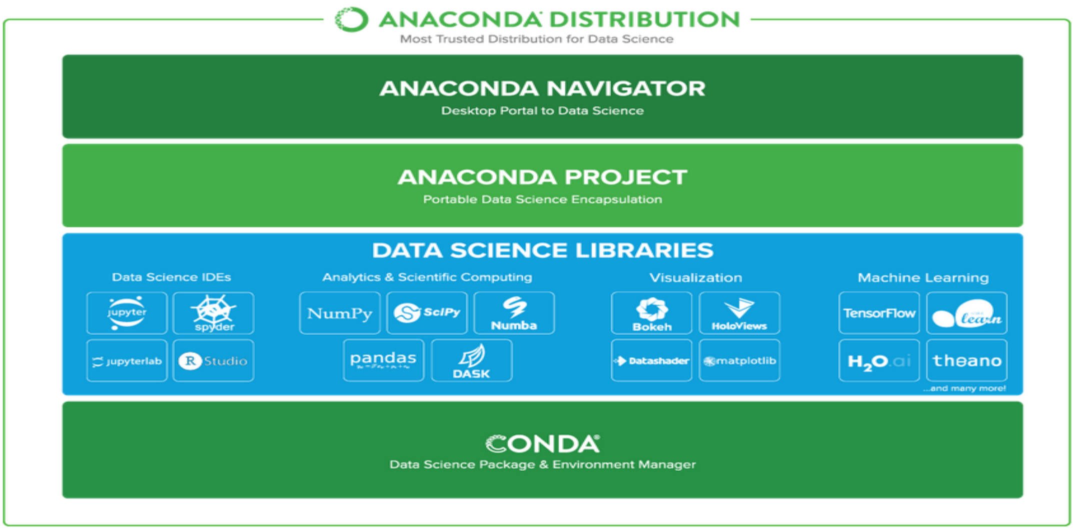

FIG 5.1: Anaconda Distributions

Anaconda Data Science Libraries

- Over 1,400 Anaconda-curated and community data science packages
- Develop data science projects using your favourite IDEs, including Jupyter, JupyterLab, Spyder, and RStudio
- Analyse data with scalability and performance with Dask, numpy, pandas, and Numba
- Visualize your data with Matplotlib, Bokeh, Datashader, and Holoviews
- Create machine learning and deep learning models with Scikit-learn, Tensorflow, h20, and Theano.

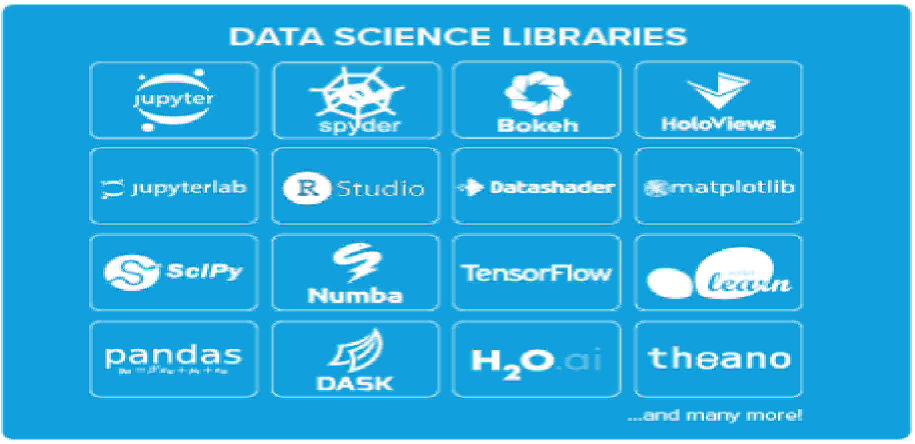

FIG 5.2: Anaconda Libraries for Data Science

Conda, the Data Science Package & Environment Manager

- Automatically manages all packages, including cross-language dependencies.
- Works across all platforms: Linux, macOS, Windows.
- Create virtual environments.
- Download conda packages from Anaconda, Anaconda Enterprise, Conda Forge, and Anaconda Cloud

Anaconda Navigator, the Desktop Portal to Data Science

- Install and launch applications and editors including Jupyter, RStudio, Visual Studio Code, and Spyder.
- Manage your local environments and data science projects from a graphical interface.

- Connect to Anaconda Cloud or Anaconda Enterprise.
- Access the latest learning and community resources.

Spyder

Spyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community. Strongly recommend the free, open-source Spyder Integrated Development Environment (IDE) for scientific and engineering programming, due to its integrated editor, interpreter console, and debugging tools. Spyder is included in Anaconda and other distributions.

5.2: FEASIBILITY STUDY

The feasibility study is carried out to test whether the proposed system is worth being implemented. The proposed system will be selected if it is best enough in meeting the performance requirements.

The feasibility carried out mainly in three sections namely.

• Economic Feasibility

• Technical Feasibility

• Behavioral Feasibility

5.2.1: Economic Feasibility

Economic analysis is the most frequently used method for evaluating effectiveness of the proposed system. More commonly known as cost benefit analysis. This procedure determines the benefits and saving that are expected from the system of the proposed system. The hardware in system department if sufficient for system development.

5.2.2: Technical Feasibility

This study centre around the system's department hardware, software and to what extend it can support the proposed system department is having the required hardware and software there is no question of increasing the cost of implementing the proposed system. The criteria, the proposed system is technically feasible, and the proposed system can be developed with the existing facility.

5.2.3: Behavioral Feasibility

People are inherently resistant to change and need enough training, which would result in lot of expenditure for the organization. The proposed system can generate reports with day-to-day information immediately at the user's request, instead of getting a report, which does not contain much detail.

5.3: TESTING OF PRODUCT

Testing of Product

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. The candidate system is subject to variety of tests-on-line response, Volume Street, recovery and security and usability test. A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that "al gears mesh", that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

Unit Testing

Unit testing is the testing of each module, and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as 'module testing'. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity.

Integration Testing

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing.

ii) Bottom-up integration testing.

White Box Testing

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we derived test cases that guarantee that all independent paths within a module have been exercised at least once.

Black Box Testing

- ✓ Black box testing is done to find incorrect or missing function
- ✓ Interface error
- ✓ Errors in external database access
- ✓ Performance errors
- ✓ Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called 'black box testing'. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

Validation Testing

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer.

User Acceptance Testing

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

Output Testing

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

6. PROPOSED MODEL

6.1: THEORY OF RESNETS

Deep convolutional neural networks have led to a series of breakthroughs for image classification. Deep networks naturally integrate low/mid/high level features and classifiers in an end-to-end multilayer fashion, and the “levels” of features can be enriched by the number of stacked layers (depth). Recent evidence reveals that network depth is of crucial importance, and the leading results on the challenging ImageNet dataset all exploit “very deep” models, with a depth of sixteen to thirty. Many other nontrivial visual recognition tasks have also greatly benefited from very deep models.

Driven by the significance of depth, a question arises: Is learning better networks as easy as stacking more layers? An obstacle to answering this question was the notorious problem of vanishing/exploding gradients, which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization and intermediate normalization layers, which enable networks with tens of layers to start converging for stochastic gradient descent (SGD) with backpropagation.

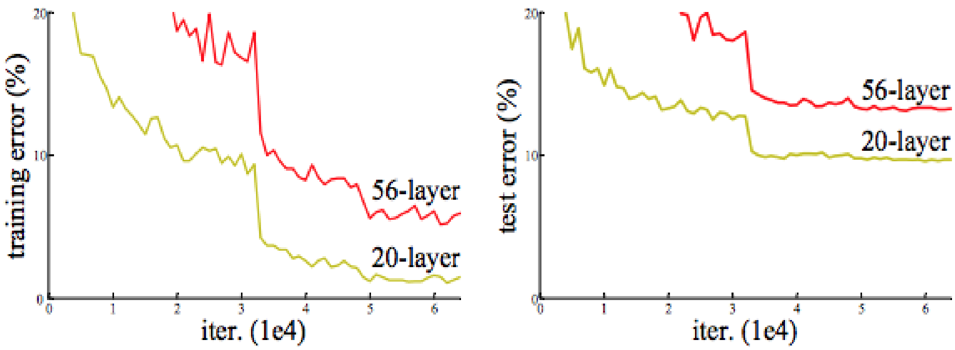

FIG 6.1: Training error (left) and test error (right) on CIFAR-10 with 20-layers and 56-layers “plain” networks.

When deeper networks can start converging, a degradation problem has been exposed: with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error, as reported in and thoroughly verified by our experiments. FIG 6.1 shows a typical example.

The degradation (of training accuracy) indicates that not all systems are similarly easy to optimize. Let us consider a shallower architecture and its deeper counterpart that adds more layers onto it. There exists a solution by construction to the deeper model: the added layers are identity mapping, and the other layers are copied from the learned shallower model. The existence of this constructed solution indicates that a deeper model should produce no higher training error than its shallower counterpart. But experiments show that our current solvers on hand are unable to find solutions that are comparably good or better than the constructed solution (or unable to do so in feasible time) framework. Instead of hoping each few stacked layers directly fit a desired underlying mapping, explicitly let these layers fit a residual mapping. Formally, denoting the desired underlying mapping as H(x), let the stacked nonlinear layers fit another mapping of F(x) := H(x)−x. The original mapping is recast into F(x)+x hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

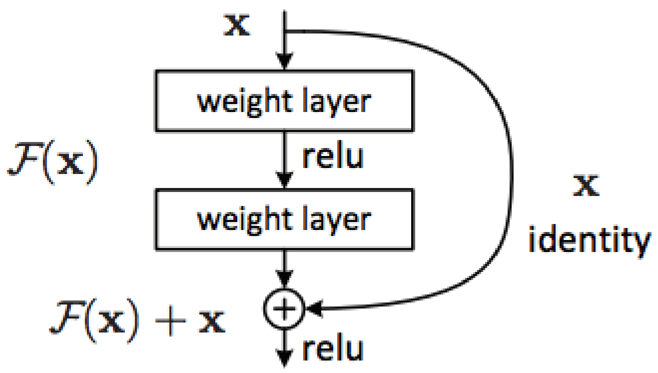

FIG 6.2: Residual learning building block

The formulation of F(x) +x can be realized by feedforward neural networks with “shortcut connections” (FIG 6.2). Shortcut connections are those skipping one or more layers. In our case, the shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers (FIG 6.2). Identity shortcut connections add neither extra parameter nor computational complexity. The entire network can still be trained end-to-end by SGD with backpropagation and can be easily implemented using common libraries without modifying the solvers.

Let present comprehensive experiments on ImageNet to show the degradation problem and evaluate our method. It shows that: 1) Our extremely deep residual nets are easy to optimize, but the counterpart “plain” nets (that simply stack layers) exhibit higher training error when the depth increases; 2) Our deep residual nets can easily enjoy accuracy gains from greatly increased depth, producing results substantially better than previous networks.

Similar phenomena are also shown on the CIFAR-10 set, suggesting that the optimization difficulties and the effects of our method are not just akin to a particular dataset. We present successfully trained models on this dataset with over 100 layers and explore models with over 1000 layers.

On the ImageNet classification dataset, it obtains excellent results by extremely deep residual nets. Our 152- layer residual net is the deepest network ever presented on ImageNet, while still having lower complexity than VGG nets. Our ensemble has 3.57% top-5 error on the ImageNet test set and won the 1st place in the ILSVRC 2015 classification competition. The extremely deep representations also have excellent generalization performance on other recognition tasks and lead us to further win the 1st places on: ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation in ILSVRC & COCO 2015 competitions. This strong evidence shows that the residual learning principle is generic, and we expect that it is applicable in other vision and non-vision problems.

6.2: BUILDING BLOCKS OF MODEL

The input data consists of five categories of images, and these are augmented before applying to the ResNET-18 block. Before that the image is upscaled or downscaled by 256x256 fixed size for the purpose of normalization and ease of operations.

The below FIG 6.3 shows that the brief exposure of ResNET-18 model consisting of Convolutional Neural Network’s (CNN’s).

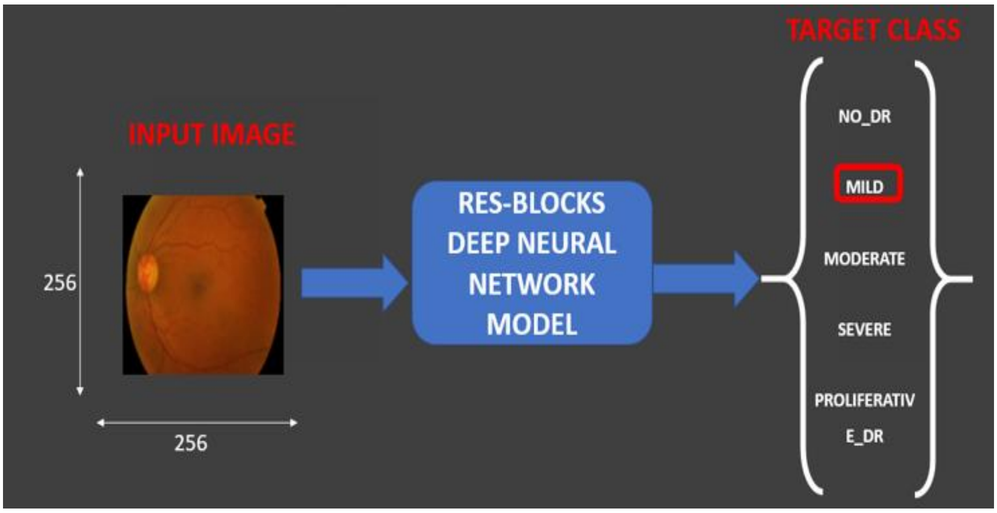

FIG 6.3: General block diagram to classify diabetic retinopathy

Inside these Res block deep neural network model there are several layers present in between the input image and target class those are:

- Convolutional Layers
- Pooling Layers (Down-Sampling)
- Flattening
- Deep Neural Network

These are shown in below FIG 6.4.

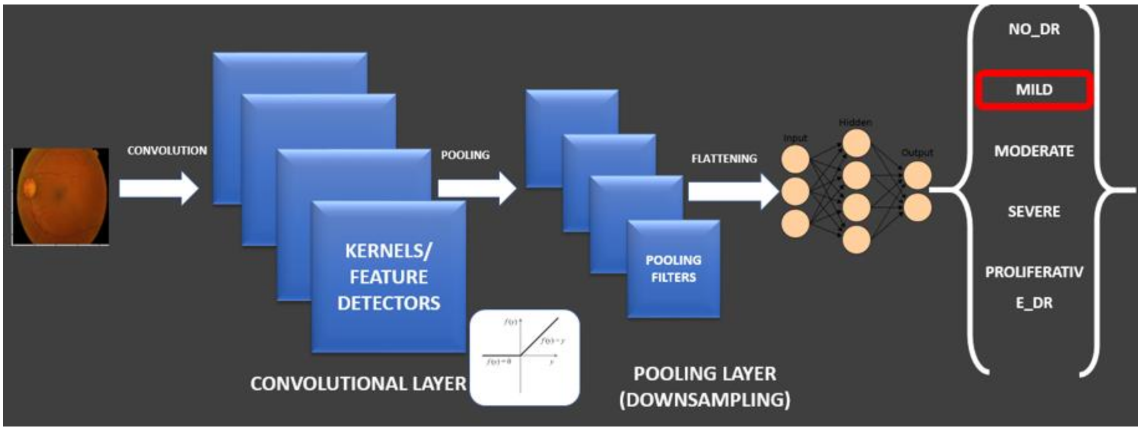

FIG 6.4: Model block diagram

These layers are used to reduce the dimensions of the image to 1-dimesional array for the further processing and analysis.

6.3: ResNET-18 MODEL

This model consists of mainly RES-BLOCK and followed by above stated layers. The RES-BLOCK are stacked around 4 or 5 layers that are present after the pooling layer and before the flattening layer this illustration is shown in below FIG 6.5.

FIG 6.4: Architecture of ResNET-18

Zero Padding:

Zero padding is a technique that allows us to preserve the original input size. This is something that we specify on a per-convolutional layer basis. With each convolutional layer, just as we define how many filters to have and the size of the filters, we can also specify whether to use padding.

Zero padding occurs when we add a border of pixels all with value zero around the edges of the input images. This adds kind of a padding of zeros around the outside of the image, hence the name zero padding.

Conv2D:

The most common type of convolution that is used is the 2D convolution layer and is usually abbreviated as conv2D. A filter or a kernel in a conv2D layer has a height and a width. They are generally smaller than the input image and so we move them across the whole image. The area where the filter is on the image is called the receptive field.

Conv2D filters extend through the three channels in an image (Red, Green, and Blue). The filters may be different for each channel too. After the convolutions are performed individually for each channels, they are added up to get the final convoluted image. The output of a filter after a convolution operation is called a feature map.

BatchNorm:

Before entering Batch normalization let's understand the term "Normalization". Normalization is a data pre-processing tool used to bring the numerical data to a common scale without distorting its shape. Generally, when we input the data to a machine or deep learning algorithm, we tend to change the values to a balanced scale. The reason we normalize is partly to ensure that our model can generalize appropriately.

Now coming back to Batch normalization, it is a process to make neural networks faster and more stable through adding extra layers in a deep neural network. The new layer performs the standardizing and normalizing operations on the input of a layer coming from a previous layer.

But what is the reason behind the term "Batch" in batch normalization? A typical neural network is trained using a collected set of input data called **batch**. Similarly, the normalizing process in batch normalization takes place in batches, not as a single input.

ReLU (Rectified Linear Unit):

ReLU stands for rectified linear activation unit and is considered one of the few milestones in the deep learning revolution. It is simple yet better than its predecessor activation functions such as sigmoid or tanh. ReLU activation function formula. Now how does ReLU transform its input? It uses this simple formula:

$$f(x)=max(0,x)$$

ReLU function is its derivative both are monotonic. The function returns 0 if it receives any negative input, but for any positive value x, it returns that value back. Thus, it gives an output that has a range from 0 to infinity.

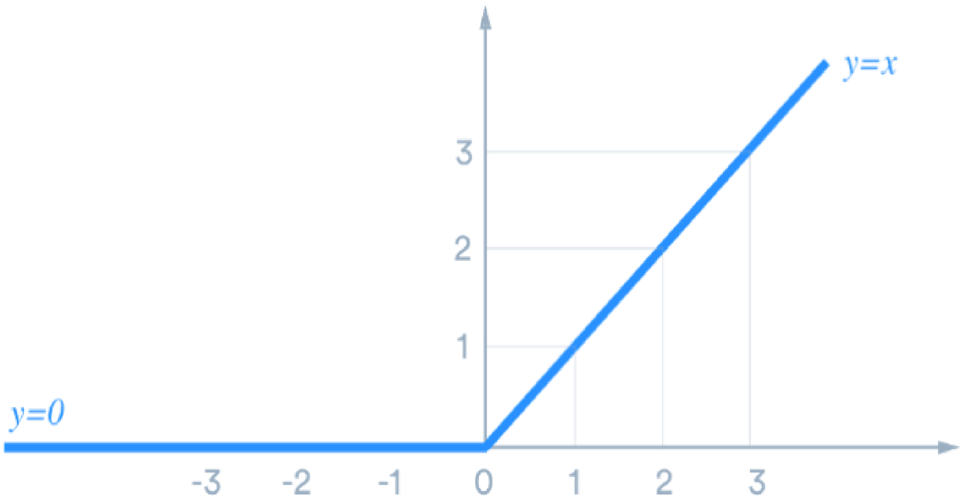

FIG 6.5: ReLU function

MaxPool2D:

Maximum pooling, or max pooling, is a pooling operation that calculates the maximum, or largest, value in each patch of each feature map. ... The maximum pooling operation can be added to the worked example by adding the MaxPooling2D layer provided by the Keras API.

RES-BLOCK:

The ResBlock is constructed out of normal network layers connected with rectified linear units (ReLUs) and a pass-through below that feeds through the information from previous layers unchanged. The network part of the ResBlock can consist of an arbitrary number of layers, but the simplest is two.

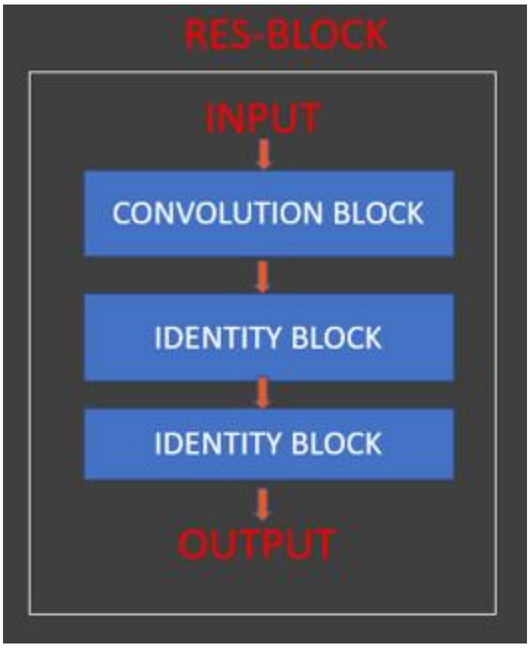

FIG 6.6: Architecture of RES-BLOCK

The RES-BLOCK consists of

1. Convolutional Block
2. Identity Block

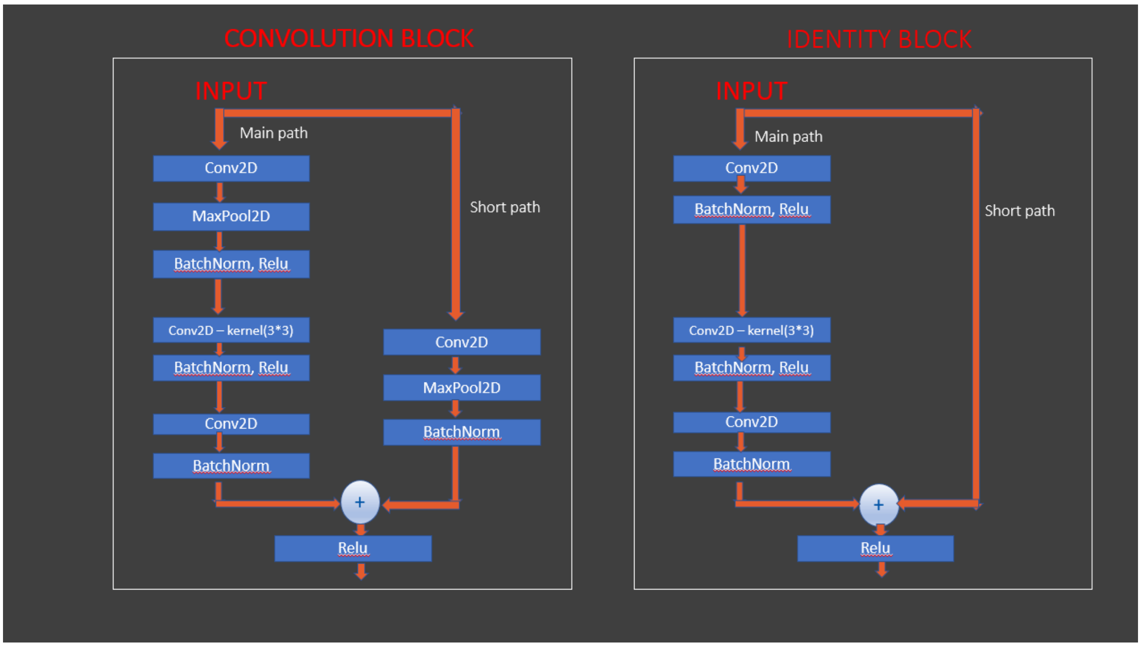

FIG 6.7: Architecture of Convolutional Block and Identity Block

AveragePooling2D:

Average pooling operation for spatial data. Downsamples the input along its spatial dimensions (height and width) by taking the average value over an input window (of size defined by pool_size) for each channel of the input. The window is shifted by strides along each dimension.

Flattening:

Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. We flatten the output of the convolutional layers to create a single long feature vector. And it is connected to the final classification model, which is called a fully-connected layer.

Dense Layer:

The dense layer is a neural network layer that is connected deeply, which means each neuron in the dense layer receives input from all neurons of its previous layer. The dense layer is found to be the most used layer in the models.

In the background, the dense layer performs a matrix-vector multiplication. The values used in the matrix are parameters that can be trained and updated with the help of backpropagation.

The output generated by the dense layer is an 'm' dimensional vector. Thus, dense layer is basically used for changing the dimensions of the vector. Dense layers also apply operations like rotation, scaling, translation on the vector.

SoftMax:

The softmax function is a function that turns a vector of K real values into a vector of K real values that sum to 1. The input values can be positive, negative, zero, or greater than one, but the softmax transforms them into values between 0 and 1, so that they can be interpreted as probabilities. If one of the inputs is small or negative, the softmax turns it into a small probability, and if an input is large, then it turns it into a large probability, but it will always remain between 0 and 1.

7. RESULTS

The proposed architecture was developed using a software package (Anaconda or Google Colab). The implementation was central processing unit (CPU) specific. All experiments were performed on a computer server with an Intel Xeon E5-2620 processor (2 GHz), 8 GB of RAM.

Testing accuracy metric

Testing accuracy is an estimation that demonstrates the precision and accuracy of any of the proposed models. Additionally, the confusion matrix is an accurate measurement that provides more insight regarding the achieved testing accuracy. FIG 7.1 presents the confusion matrices for the distinctive CNN models used in this research.

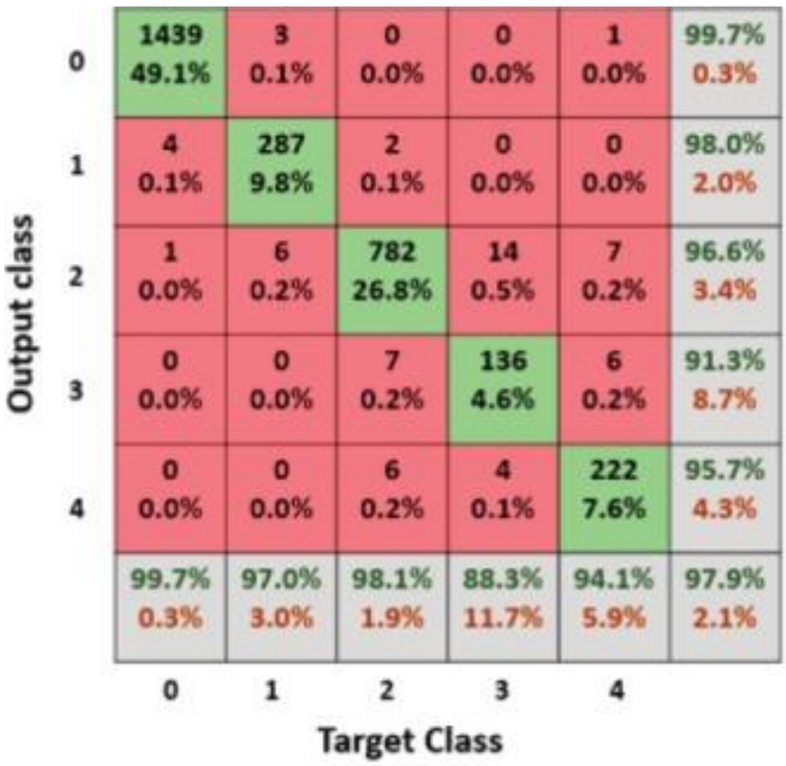

FIG 7.1: Confusion matrix of the model

Table 7.1 presents the class and total testing accuracy for different CNN models. The table illustrates that AlexNet, VGG16, and VGG19 achieved a very close total testing accuracy to each other at 97.9%, 97.7%, and 97.4%, respectively. The three models also achieved top-class testing detection accuracy, except for class 3, which achieved the best accuracy using GoogleNet. According to total testing accuracy, the

AlexNet model achieved the highest accuracy at 97.9%. Additionally, AlexNet had the least number of layers with only 8 layers and, as the result, had fewer calculations and less complexity.

Classes and total testing accuracy for the different CNN models

Accuracy/Model	AlexNet	VGG16	ResNet18	SqueezeNet	VGG19	GoogleNet
Class 0	99.7%	99.8%	99.5%	97.8%	99.6%	99.7%
Class 1	98.0%	96.3%	90.7%	80.0%	98.6%	96.8%
Class 2	96.6%	98.1%	97.3%	87.5%	97.6%	91.4%
Class 3	91.3%	89.1%	91.4%	67.8%	88.8%	92.3%
Class 4	95.8%	92.7%	89.8%	80.9%	88.7%	94.4%
Total Accuracy	97.9%	97.8%	96.8%	90.3%	97.4%	96.3%

Table 7.1: Total testing accuracy of different CNN models

Performance evaluation and discussion

To evaluate the performance of the proposed models, more performance matrices need to be investigated through this research. The most common performance measures in the field of DL are precision, recall, and F1 score, which are presented from equation (1) to equation (3), respectively.

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{2}$$

$$\text{F1 Score} = 2.\frac{\text{Precision} \,.\, \text{Recall}}{(\text{Precision} + \text{Recall})} \tag{3}$$

where TP is the count of true positive samples, TN is the count of true negative samples, FP is the count of false positive samples, and FN is the count of false negative samples from a confusion matrix.

Table 4 presents the performance metrics for the different proposed CNN models. The table illustrates that the AlexNet model achieved the highest percentage for the precision and recall metrics, while VGG16 achieved the highest percentage for the recall metric.

Performance metrices for the different CNN models

Metric/Model	Alex Net	VGG16	Res Net 18	Squeeze Net	VGG19	Google Net
Precision	96.23%	95.19%	93.75%	82.80%	94.64%	94.92%
Recall	95.42%	96.02%	94.57%	82.16%	95.76%	90.63%
F1 Score	95.82%	95.60%	94.16%	82.48%	95.20%	92.73%

Table 7.2: Performance metrics for the different CNN models

According to the achieved results for both overall testing accuracy and the performance metrics, AlexNet is the most appropriate CNN model for the APTOS 2019 dataset for medical DR detection with a testing accuracy of 97.9%. Moreover, Res Net 18 and VGG19 also achieved competitive results, as illustrated in Table 7.1 and Table 7.2.

8. CONCLUSION

DR is a diabetes complication that affects the eyes. This disease may cause no symptoms or only mild vision problems but eventually, can cause blindness. In Egypt, more than 6 million people (7.2% of the population) suffer from DR. With advances in computers algorithms, such as AI and DL models, the opportunities for the detection of DR at the early stages increases. Early detection will increase the chances of recovery and reduce the possibility of vision loss in patients. In this paper, deep transfer learning models for medical DR detection were investigated on the APTOS 2019 dataset. According to literature surveys, this research is considered one the first studies that used the APTOS 2019 dataset, as it was released in the second quarter of 2019. Augmentation techniques were used to overcome the overfitting problem and increased the dataset images to be 4 times larger than the original dataset. The deep transfer learning models selected in this paper are AlexNet, ResNet18, SqueezeNet, GoogleNet, VGG16, and VGG19. The overall testing accuracy and performance metrics (precision, recall, and F1 score) showed that the AlexNet model achieved the highest testing accuracy (97.9%), precision, and F1 score percentage. Moreover, this model utilized a minimum number of layers, which decreased the training time and the computational complexity.

9. REFERENCES

[1] V. Raman, P. Then and P. Sumari. "Proposed retinal abnormality detection and classification approach: Computer-aided detection for diabetic retinopathy by machine learning approaches", Communication Software and Networks (ICCSN), 2016 8th IEEE International Conference on. IEEE, (2016).

[2] D. K. Prasad, L. Vibha and K. R. Venugopal, "Early detection of diabetic retinopathy from digital retinal fundus images", Intelligent Computational Systems (RAICS), 2015 IEEE Recent Advances in. IEEE, (2015).

[3] Retina in systemic disease : a color manual of ophthalmoscopy / Homayoun Tabandeh, Morton F. Goldberg 2009

[4] Goyal S, Laavalley M, Subramanian ML, Meta analysis and review on the effect on bevacizumab in diabetic macular edema, Graefes Arch Clin Exp Ophthalmol(2011) 249:15-27

[5] V. Gulshan, L. Peng, M. Coram, M. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, R. Kim, R. Raman, P. Nelson, J. Mega and D. Webster, "Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs," JAMA, vol. 316, no. 22, pp. 2402–2410, 2016.

[6] D. Ting, C. Cheung, G. Lim, G. Tan, N. Quang, A. Gan, H. Hamzah, R. Garcia-Franco, I. San Yeo, S. Lee, E. Wong, C. Sabanayagam, M. Baskaran, F. Ibrahim, N. Tan, E. Finkelstein, E. Lamoureux, I. Wong, N. Bressler, S. Sivaprasad, R. Varma, J. Jonas, M. He, C. Cheng, G. Cheung, T. Aung, W. Hsu, M. Lee and T. Wong, "Development and Validation of a Deep Learning System for Diabetic Retinopathy and Related Eye Diseases Using Retinal Images From Multiethnic Populations With Diabetes," JAMA, vol. 318, no. 22, pp. 2211–2223, 2017.

[7] C. P. Wilkinson, MD,1 Frederick L. Ferris, III, MD,2 Ronald E. Klein, MD, MPH,3 Paul P. Lee, MD, JD,4 Carl David Agardh, MD,5 Matthew Davis, MD,3 Diana Dills, MD,6 Anselm Kampik, MD,7 R. Pararajasegaram, MD,8 Juan T. Verdaguer, MD,9 representing the Global Diabetic Retinopathy Project Group, Proposed International Clinical Diabetic, Retinopathy and Diabetic Macular Edema Disease Severity Scales Ophthalmology Volume 110, Number 9,

September 2003 Proposed international clinical diabetic retinopathy and diabetic macular edema disease severity scales

[8] S. Roychowdhury, D. Koozekanani and K. Parhi, “Dream: Diabetic retinopathy analysis using machine learning”, IEEE Journal of Biomedical and Health Informatics, vol. 18, no. 5, (2014), pp. 1717–1728.

[9] T. Vatanen, T. Raiko, H. Valpola, and Y. LeCun. Pushing stochastic gradient towards second-order methods–backpropagation learning with transformations in nonlinearities. In Neural Information Processing, 2013.

[10] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2):157–166, 1994.

[11] Preferred Practice Patterns, Diabetic retinopathy, America Academy of Ophthalmology 2008. http://one.aao.org/CE/PracticeGuidelines/PPP_Content.aspx?cid=d0c853d3-219f-487b-a524-326ab3cecd9a

[12] Brett J. Rosenblatt and William E. Benson Diabetic Retinopathy Yanoff & Duker: Ophthalmology, 3rd ed.

[13] “Diabetes”, World Health Organization. November 1, 2014. Accessed December 26, (2014). http://www.who.int/mediacentre/factsheets/fs312/en/.

[14] Resnikoff S, Pascolini D, Etya'ale D, Kocur I, Pararajasegaram R, Pokharel GP, Mariotti SP. Global data on visual impairment in the year 2002. Bull World Health Organ. 2004 Nov;82(11):844-51. Epub 2004 Dec 14.

[15] S. Mohammadian, A. Karsaz and Y. M. Roshan, "A comparative analysis of classification algorithms in diabetic retinopathy screening," 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, pp. 84-89, 2017.

[16] S.Roychowdhury, D.D.Koozekanani, Keshab ok.Parhi, "DREAM: Diabetic Retinopathy assessment the usage of machine analyzing", IEEE diary of BioMedical and well being Informatics, Vol.18, No five, September (2014).

[17] R. Owens, S. Venkatesh, and J. Ross, "Edge detection is a projection," Pattern Recognition Letters, vol. 9, no. 4, pp. 233–244, 1989.

[18] J. A. Hanley and B. J. McNeil, ‘The meaning and use of the area under a receiver operating characteristic (ROC) curve’, Radiology, vol. 143, no. 1, pp. 29–36, 1982.

[19] Basic and Clinical Science Course, Section 12: Retina and Vitreous AAO, 2011-2012.

[20] The Effect of Intensive Diabetes Treatment On the Progression of Diabetic Retinopathy In Insulin-Dependent Diabetes Mellitus, The Diabetes Control and Complications Trial Research Group, Arch Ophthalmol. 1995; 113:36-51

[21] Soomro TA, Afifi AJ, Zheng L, Soomro S, Gao J, Hellwich O, et al. Deep Learning Models for Retinal Blood Vessels Segmentation: A Review. IEEE Access. 2019;7:71696–71717.

[22] Sun Y, Zhang D. Diagnosis and Analysis of Diabetic Retinopathy Based on Electronic Health Records. IEEE Access. 2019;7:86115–86120.

[23] Guariguata L, Whiting DR, Hambleton I, Beagley J, Linnenkamp U, Shaw JE. Global estimates of diabetes prevalence for 2013 and projections for 2035. Diabetes Research and Clinical Practice. 2014;103(2):137–149.

[24] Shahin EM, Taha TE, Al-Nuaimy W, Rabaie S El, Zahran OF, El-Samie FEA. Automated detection of diabetic retinopathy in blurred digital fundus images; 2012 8th International Computer Engineering Conference (ICENCO); 2012; pp. 20–25.

[25] Gao Z, Li J, Guo J, Chen Y, Yi Z, Zhong J. Diagnosis of Diabetic Retinopathy Using Deep Neural Networks. IEEE Access. 2019;7:3360–3370.

[26] Poostchi M, Silamut K, Maude RJ, Jaeger S, Thoma G. Image analysis and machine learning for detecting malaria. Translational Research. 2018;194:36–55.

[27] Khalifa NEM, Taha MHN, Hassanien AE. Aquarium Family Fish Species Identification System Using Deep Neural Networks. In: Hassanien AE, Tolba MF, Shaalan K, Azar AT, editors. Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2018; 2019; Cham. Springer International Publishing; pp. 347–356.

[28] Khalifa NE, Hamed Taha M, Hassanien AE, Selim I. Deep Galaxy V2: Robust Deep Convolutional Neural Networks for Galaxy Morphology Classifications; 2018 International Conference on Computing Sciences and Engineering (ICCSE); 2018; IEEE; pp. 1–6.

[29] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014. Sep 4, [cited 2017 Aug 4]; Available from: http://arxiv.org/abs/1409.1556.

[30] Wong SC, Gatt A, Stamatescu V, McDonnell MD. Understanding Data Augmentation for Classification: When to Warp?; 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA); 2016; IEEE; pp. 1–6.

[31] Goutte C, Gaussier E. A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation.

[32] Huang G, Liu Z, Maaten LVD, Weinberger KQ. Densely Connected Convolutional Networks; 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017; pp. 2261–2269.

[33] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision; Proceedings of the IEEE conference on computer vision and pattern recognition; 2016; pp. 2818–2826.

[34] Chollet F. Xception: Deep Learning with Depthwise Separable Convolutions; 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017; pp. 1800–1807.

[35] Liu S, Deng W. Very deep convolutional neural network based image classification using small training sample size; 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR); 2015; pp. 730–734.

[36] Deng J, Dong W, Socher R, Li L, Kai L, Li FF. ImageNet: A largescale hierarchical image database; 2009 IEEE Conference on Computer Vision and Pattern Recognition; 2009; pp. 248–255.

NAME	: **G. PAVAN KUMAR**
ROLL NO	: **17JN1A0456**
PH.NO	: **9441267425**
EMAIL ID	: **pavankumar42sn@gmail.com**

NAME	: **G. RAKESH**
ROLL NO	: **17JN1A0443**
PH.NO	: **7893286147**
EMAIL ID	: **gadhirajurakesh2233@gmail.com**

NAME	: **B. MADHUSUDHANA**
ROLL NO	: **17JN1A0415**
PH.NO	: **9182452504**
EMAIL ID	: **sunnymadhu37@gmail.com**

NAME	: **CH. PRATHYUSH**
ROLL NO	: **17JN1A0427**
PH.NO	: **6300138321**
EMAIL ID	: **prathyushkmr427@gmail.com**

NAME	: **M. BRAHMASAI**
ROLL NO	: **16JN1A0464**
PH.NO	: **9032597771**
EMAIL ID	: **brahmasai06@gmail.com**