Summer Training Report/ Synopsis/ Minor Project

On

Plant Seedling Classification Using Tensorflow

Industrial Training in Machine Learning

A Project Report/Synopsis submitted in partial fulfillment of the requirements for the award of

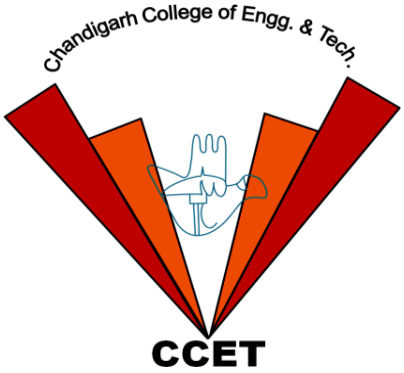
Bachelor of Engineering
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by

Nikhil Anand

(Roll no: C017342)

Under the supervision of
(Dr. Marco Gillies, University of London, Coursera)



CHANDIGARH COLLEGE OF ENGINEERING AND TECHNOLOGY
(DEGREE WING)

Government Institute under Chandigarh (UT) Administration, Affiliated to Panjab University , Chandigarh

Sector-26, Chandigarh. PIN-160019

July, 2020

CHANDIGARH COLLEGE OF ENGINEERING AND TECHNOLOGY (DEGREE WING)
Government Institute under Chandigarh (UT) Administration | Affiliated to Panjab University , Chandigarh
Sector-26, Chandigarh. PIN-160019 | Tel. No. 0172-2750947, 2750943
Website: www.ccet.ac.in | Email: principal@ccet.ac.in | Fax. No. :0172-2750872

Department of Computer Sc. & Engineering

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report entitled "Industrial Training in Machine Learning", in fulfillment of the requirement for the award of the degree Bachelor of Engineering in Computer Science & Engineering, submitted in CSE Department, Chandigarh College of Engineering & Technology(Degree wing) affiliated to Punjab University, Chandigarh, is an authentic record of my/our own work carried out during my degree under the guidance of < Dr. Ankit Gupta >. The work reported in this has not been submitted by me for award of any other degree or diploma.

Date:30/07/20 <Nikhil Anand>

Place: Chandigarh <Roll no. C017342>

CHANDIGARH COLLEGE OF ENGINEERING AND TECHNOLOGY (DEGREE WING)
Government Institute under Chandigarh (UT) Administration | Affiliated to Panjab University , Chandigarh
Sector-26, Chandigarh. PIN-160019 | Tel. No. 0172-2750947, 2750943
Website: www.ccet.ac.in | Email: principal@ccet.ac.in | Fax. No. :0172-2750872
Department of Computer Sc. & Engineering

CERTIFICATE

This is to certify that the Project work entitled "Calculator app." submitted by <Nikhil Anand and roll no. C017342 > fulfillment for the requirements of the award of Bachelor of Engineering Degree in Computer Science & Engineering at Chandigarh College of Engineering and Technology (Degree Wing), Chandigarh is an authentic work carried out by him/her under my supervision and guidance.

To the best of my knowledge, the matter embodied in the project has not been submitted to any other University / Institute for the award of any Degree .

Date :30/07/20 <AP- Dr. Ankit Gupta>

Place :Chandigarh <Deptt. of CSE>
<CCET(Degree Wing)>
< Chandigarh>

CHANDIGARH COLLEGE OF ENGINEERING AND TECHNOLOGY (DEGREE WING)
Government Institute under Chandigarh (UT) Administration | Affiliated to Panjab University , Chandigarh
Sector-26, Chandigarh. PIN-160019 | Tel. No. 0172-2750947, 2750943
Website: www.ccet.ac.in | Email: principal@ccet.ac.in | Fax. No. :0172-2750872
Department of Computer Sc. & Engineering

ACKNOWLEDGEMENT

The completion of this project could have been possible with continued and dedicated efforts and guidance of faculty and staff members of our institute . I acknowledge my gratitude to all of them . The acknowledgement would however be incomplete without specific mention as follows.

I wish my deep gratitude to Dr. Ankit Gupta , faculty member of our institute for his cooperation and guidance and providing me with necessary advices throughout the training period without which completion of this training could not have been possible.I choose this moment to acknowledge his contribution gratefully. Furthermore , I would also like to acknowledge with much appreciation the crucial role of our HOD. Prof. Dr. Sunil K Singh for this encouragement and providing all these facilities in the department .

This report has been prepared for the internship that has been done on Coursera , an online portal for industrial training in order to study pratical aspect of the course and implement the theory in real life . The aim of this internship is to be familiar to pratical aspect and uses of theoretical knowledge and clarifying the career goals, so I have successfully completed the internship and compiled this report as the summary and the conclusion that have been drawn from the internship experience.

Finally, I would like to say that I am indebted to my parents and teachers for everything they have done for me and without their guidance and their support I have been able to complete the training . And also I would like to thank God for being kind to me and driving me through this journey.

Nikhil Anand

CHANDIGARH COLLEGE OF ENGINEERING AND TECHNOLOGY (DEGREE WIN
Government Institute under Chandigarh (UT) Administration | Affiliated to Panjab University , Chandigarh
Sector-26, Chandigarh. PIN-160019 | Tel. No. 0172-2750947, 2750943 Website: www.ccet.ac.in | Email: principal@ccet.ac.in | Fax. No. :0172-2750872

Department of Computer Sc. & Engineering

ABSTRACT

Industrial training is an important phase of a student life. A well planned, properly executed and evaluated industrial training helps a lot in developing a professional attitude. It develop an awareness of industrial approach to problem solving, based on a broad understanding of process and mode of operation of organization. The aim and motivation of this industrial training is to receive discipline, skills, teamwork and technical knowledge through a proper training environment, which will help me, as a student in the field of Computer Science, to develop a responsiveness of the self-disciplinary nature of problems in information and communication technology. During a period of three months training at coursera , I was assigned to creating an efficient Calculator app in java which can help us to perform basic arithmetic operation such as addition ,multiplication, subtraction and division even in decimal numbers that can save our calculation error and provide us with precise answers and also save our time. As a result I vital to achieve the minimum requirement of the company, it will help me to boost my career as developer in java programming language.This report also contains basic theory of Machine Learning and also its history and some basic knowledge about the software which was used to design the project i.e Jupyter . and futher all this information is provided in form of chapters .

Throughout this industrial training, I have been learned new programming language that required for the creation of web application,embedded system, desktop GUI(graphical user interface), the process of the production lines and able to implement what I have learnt for the four weeks that can to put my contribution towards our technological growth of our society.

CERTIFICATE OF COMPLETION

LIST OF FIGURES:

CONTENT

Chapter-1
Introduction to Machine Learning

Today's Artificial Intelligence (AI) has far surpassed the hype of blockchain and quantum computing. This is due to the fact that huge computing resources are easily available to the common man. The developers now take advantage of this in creating new Machine Learning models and to re-train the existing models for better performance and results. The easy availability of High Performance Computing (HPC) has resulted in a sudden increased demand for IT professionals having Machine Learning skills.

1.1 What today's AI can do ?

When you tag a face in a Facebook photo, it is AI that is running behind the scenes and identifying faces in a picture. Face tagging is now omnipresent in several applications that display pictures with human faces. Why just human faces? There are several applications that detect objects such as cats, dogs, bottles, cars, etc. We have autonomous cars running on our roads that detect objects in real time to steer the car. When you travel, you use Google Directions to learn the real-time traffic situations and follow the best path suggested by Google at that point of time. This is yet another implementation of object detection technique in real time.

Let us consider the example of Google Translate application that we typically use while visiting foreign countries. Google's online translator app on your mobile helps you communicate with the local people speaking a language that is foreign to you.

There are several applications of AI that we use practically today. In fact, each one of us use AI in many parts of our lives, even without our knowledge. Today's AI can perform extremely complex jobs with a great accuracy and speed. Let us discuss an example of complex task to understand what capabilities are expected in an AI application that you would be developing today for your clients.

EXAMPLE

We all use Google Directions during our trip anywhere in the city for a daily commute or even for inter-city travels. Google Directions application suggests the fastest path to our destination at that time instance. When we follow this path, we have observed that Google is almost 100% right in its suggestions and we save our valuable time on the trip.

You can imagine the complexity involved in developing this kind of application considering that there are multiple paths to your destination and the application has to judge the traffic situation in every possible path to give you a travel time estimate for each such path. Besides, consider the fact that Google Directions covers the entire globe. Undoubtedly, lots of AI and Machine Learning techniques are in-use under the hoods of such applications.

Considering the continuous demand for the development of such applications, you will now appreciate why there is a sudden demand for IT professionals with AI skills.

1.2 MACHINE LEARNING -TRADITIONAL AI

The journey of AI began in the 1950's when the computing power was a fraction of what it is today. AI started out with the predictions made by the machine in a fashion a statistician does predictions using his calculator. Thus, the initial entire AI development was based mainly on statistical techniques.

STATISTICAL TECHNIQUE

The development of today's AI applications started with using the age-old traditional statistical techniques. You must have used straight-line interpolation in schools to predict a future value. There are several other such statistical techniques which are successfully applied in developing so-called AI programs. We say "so-called" because the AI programs that we have today are much more complex and use techniques far beyond the statistical techniques used by the early AI programs.

Some of the examples of statistical techniques that are used for developing AI applications in those days and are still in practice are listed here −

- Regression
- Classification
- Clustering
- Probability Theories
- Decision Trees

Here we have listed only some primary techniques that are enough to get you started on AI without scaring you of the vastness that AI demands. If you are developing AI applications based on limited data, you would be using these statistical techniques.

However, today the data is abundant. To analyze the kind of huge data that we possess statistical techniques are of not much help as they have some limitations of their own. More advanced methods such as deep learning are hence developed to solve many complex problems.

Chapter 2

What is Machine Learning and types of Machine Learning ?

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

But, using the classic algorithms of machine learning, text is considered as a sequence of keywords; instead, an approach based on semantic analysis mimics the human ability to understand the meaning of a text

Machine Learning is broadly categorized under the following headings −

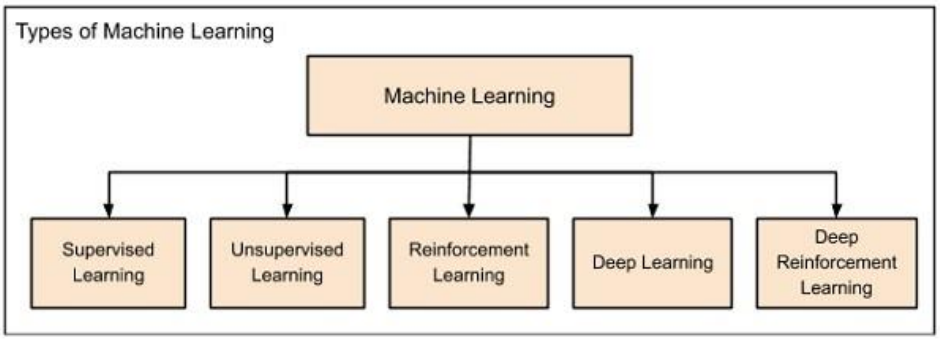


FIG 1

Machine learning evolved from left to right as shown in the above diagram.

- Initially, researchers started out with Supervised Learning. This is the case of housing price prediction discussed earlier.
- This was followed by unsupervised learning, where the machine is made to learn on its own without any supervision.
- Scientists discovered further that it may be a good idea to reward the machine when it does the job the expected way and there came the Reinforcement Learning.
- Very soon, the data that is available these days has become so humongous that the conventional techniques developed so far failed to analyze the big data and provide us the predictions.
- Thus, came the deep learning where the human brain is simulated in the Artificial Neural Networks (ANN) created in our binary computers.

- The machine now learns on its own using the high computing power and huge memory resources that are available today.
- It is now observed that Deep Learning has solved many of the previously unsolvable problems.
- The technique is now further advanced by giving incentives to Deep Learning networks as awards and there finally comes Deep Reinforcement Learning.

2.1 **Supervised Learning**

Supervised learning is analogous to training a child to walk. You will hold the child's hand, show him how to take his foot forward, walk yourself for a demonstration and so on, until the child learns to walk on his own.

Regression

Similarly, in the case of supervised learning, you give concrete known examples to the computer. You say that for given feature value x1 the output is y1, for x2 it is y2, for x3 it is y3, and so on. Based on this data, you let the computer figure out an empirical relationship between x and y.

Once the machine is trained in this way with a sufficient number of data points, now you would ask the machine to predict Y for a given X. Assuming that you know the real value of Y for this given X, you will be able to deduce whether the machine's prediction is correct.

Thus, you will test whether the machine has learned by using the known test data. Once you are satisfied that the machine is able to do the predictions with a desired level of accuracy (say 80 to 90%) you can stop further training the machine.

Now, you can safely use the machine to do the predictions on unknown data points, or ask the machine to predict Y for a given X for which you do not know the real value of Y. This training comes under the regression that we talked about earlier.

Classification

You may also use machine learning techniques for classification problems. In classification problems, you classify objects of similar nature into a single group. For example, in a set of 100 students say, you may like to group them into three groups based on their heights - short, medium and long. Measuring the height of each student, you will place them in a proper group.

Now, when a new student comes in, you will put him in an appropriate group by measuring his height. By following the principles in regression training, you will train the machine to classify a student based on his feature – the height. When the machine learns how the groups are formed, it will be able to classify any unknown new student correctly. Once again, you would use the test data to verify that the machine has learned your technique of classification before putting the developed model in production.

Supervised Learning is where the AI really began its journey. This technique was applied successfully in several cases. You have used this model while doing the hand-written recognition on your machine. Several algorithms have been developed for supervised learning. You will learn about them in the following chapters.

2.2 Unsupervised Learning

In unsupervised learning, we do not specify a target variable to the machine, rather we ask machine "What can you tell me about X?". More specifically, we may ask questions such as given a huge data set X, "What are the five best groups we can make out of X?" or "What features occur together most frequently in X?". To arrive at the answers to such questions, you can understand that the number of data points that the machine would require to deduce a strategy would be very large. In case of supervised learning, the machine can be trained with even about few thousands of data points. However, in case of unsupervised learning, the number of data points that is reasonably accepted for learning starts in a few millions. These days, the data is generally abundantly available. The data ideally requires curating. However, the amount of data that is continuously flowing in a social area network, in most cases data curation is an impossible task.The following figure shows the boundary between the yellow and red dots as determined by unsupervised machine learning. You can see it clearly that the machine would be able to determine the class of each of the black dots with a fairly good accuracy.

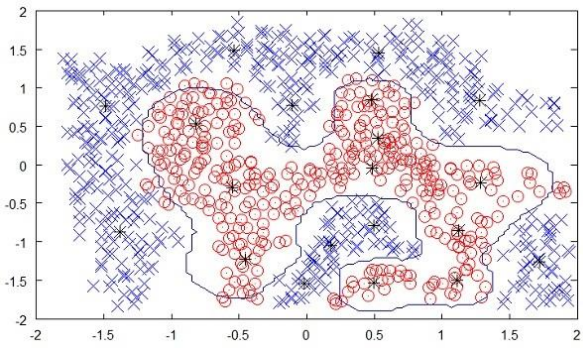


FIG 2

The unsupervised learning has shown a great success in many modern AI applications, such as face detection, object detection, and so on.

Algorithms for Unsupervised Learning

Let us now discuss one of the widely used algorithms for classification in unsupervised machine learning.

k-means clustering

The 2000 and 2004 Presidential elections in the United States were close — very close. The largest percentage of the popular vote that any candidate received was 50.7% and the lowest was 47.9%. If a percentage of the voters were to have switched sides, the outcome of the election would have been different. There are small groups of voters who, when properly appealed to, will switch sides. These groups may not be huge, but with such close races, they may be big enough to change the outcome of the election. How do you find these groups of people? How do you appeal to them with a limited budget? The answer is clustering.

Let us understand how it is done.

- First, you collect information on people either with or without their consent: any sort of information that might give some clue about what is important to them and what will influence how they vote.
- Then you put this information into some sort of clustering algorithm.
- Next, for each cluster (it would be smart to choose the largest one first) you craft a message that will appeal to these voters.
- Finally, you deliver the campaign and measure to see if it's working.

Clustering is a type of unsupervised learning that automatically forms clusters of similar things. It is like automatic classification. You can cluster almost anything, and the more similar the items are in the cluster, the better the clusters are. In this chapter, we are going to study one type of clustering algorithm called k-means. It is called k-means because it finds 'k' unique clusters, and the center of each cluster is the mean of the values in that cluster.

Cluster Identification

Cluster identification tells an algorithm, "Here's some data. Now group similar things together and tell me about those groups." The key difference from classification is that in classification you know what you are looking for. While that is not the case in clustering.

Clustering is sometimes called unsupervised classification because it produces the same result as classification does but without having predefined classes.

2.3 Reinforcement Learning

Consider training a pet dog, we train our pet to bring a ball to us. We throw the ball at a certain distance and ask the dog to fetch it back to us. Every time the dog does this right, we reward the dog. Slowly, the dog learns that doing the job rightly gives him a reward and then the dog starts doing the job right way every time in future. Exactly, this concept is applied in "Reinforcement" type of learning. The technique was initially developed for machines to play games. The machine is given an algorithm to analyze all possible moves at each stage of the game. The machine may select one of the moves at random. If the move is right, the machine is rewarded, otherwise it may be penalized. Slowly, the machine will start differentiating between right and wrong moves and after several iterations would learn to solve the game puzzle with a better accuracy. The accuracy of winning the game would improve as the machine plays more and more games.

The entire process may be depicted in the following diagram −

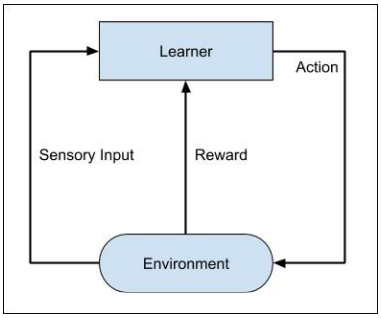


FIG 3

This technique of machine learning differs from the supervised learning in that you need not supply the labelled input/output pairs. The focus is on finding the balance between exploring the new solutions versus exploiting the learned solutions.

2.4 Deep Learning

The deep learning is a model based on Artificial Neural Networks (ANN), more specifically Convolutional Neural Networks (CNN)s. There are several architectures used in deep learning such as deep neural networks, deep belief networks, recurrent neural networks, and convolutional neural networks.

These networks have been successfully applied in solving the problems of computer vision, speech recognition, natural language processing, bioinformatics, drug design, medical image analysis, and games. There are several other fields in which deep learning is proactively applied. The deep learning requires huge processing power and humongous data, which is generally easily available these days.

2.5 Deep Reinforcement Learning

The Deep Reinforcement Learning (DRL) combines the techniques of both deep and reinforcement learning. The reinforcement learning algorithms like Q-learning are now combined with deep learning to create a powerful DRL model. The technique has been with a great success in the fields of robotics, video games, finance and healthcare. Many previously unsolvable problems are now solved by creating DRL models. There is lots of research going on in this area and this is very actively pursued by the industries.

So far, you have got a brief introduction to various machine learning models, now let us explore slightly deeper into various algorithms that are available under these models.

Chapter 3

Artificial Neural networks and Deep Learning

The idea of artificial neural networks was derived from the neural networks in the human brain. The human brain is really complex. Carefully studying the brain, the scientists and engineers came up with an architecture that could fit in our digital world of binary computers.

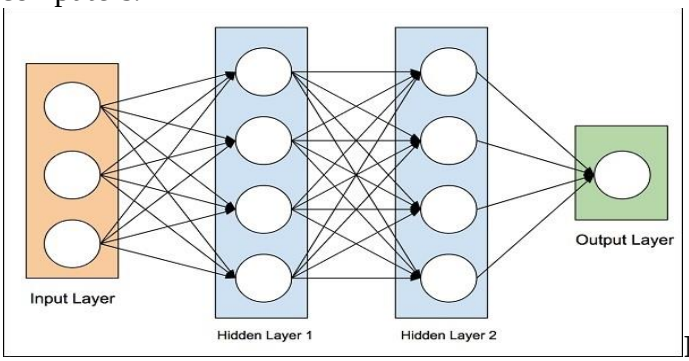


FIG 4

An artificial neural network (ANN) is the component of artificial intelligence that is meant to simulate the functioning of a human brain. Processing units make up ANNs, which in turn consist of inputs and outputs. The inputs are what the ANN learns from to produce the desired output. Backpropagation is the set of learning rules used to guide artificial neural networks. The practical applications for ANNs are far and wide, encompassing finance, personal communication, industry, education, and so on.

Deep Learning uses ANN. First we will look at a few deep learning applications that will give you an idea of its power.

Applications

Deep Learning has shown a lot of success in several areas of machine learning applications.

Self-driving Cars − The autonomous self-driving cars use deep learning techniques. They generally adapt to the ever changing traffic situations and get better and better at driving over a period of time.

Speech Recognition − Another interesting application of Deep Learning is speech recognition. All of us use several mobile apps today that are capable of recognizing our speech. Apple's Siri, Amazon's Alexa, Microsoft's Cortena and Google's Assistant – all these use deep learning techniques.

Mobile Apps − We use several web-based and mobile apps for organizing our photos. Face detection, face ID, face tagging, identifying objects in an image – all these use deep learning.

Untapped Opportunities of Deep Learning

After looking at the great success deep learning applications have achieved in many domains, people started exploring other domains where machine learning was not so far

applied. There are several domains in which deep learning techniques are successfully applied and there are many other domains which can be exploited. Some of these are discussed here.

- Agriculture is one such industry where people can apply deep learning techniques to improve the crop yield.
- Consumer finance is another area where machine learning can greatly help in providing early detection on frauds and analyzing customer's ability to pay.
- Deep learning techniques are also applied to the field of medicine to create new drugs and provide a personalized prescription to a patient.

The possibilities are endless and one has to keep watching as the new ideas and developments pop up frequently.

What is Required for Achieving More Using Deep Learning ?

To use deep learning, supercomputing power is a mandatory requirement. You need both memory as well as the CPU to develop deep learning models. Fortunately, today we have an easy availability of HPC – High Performance Computing. Due to this, the development of the deep learning applications that we mentioned above became a reality today and in the future too we can see the applications in those untapped areas that we discussed earlier.

Now, we will look at some of the limitations of deep learning that we must consider before using it in our machine learning application.

Deep Learning Disadvantages

Some of the important points that you need to consider before using deep learning are listed below −

- Black Box approach
- Duration of Development
- Amount of Data
- Computationally Expensive

We will now study each one of these limitations in detail.

Black Box approach

An ANN is like a blackbox. You give it a certain input and it will provide you a specific output. The following diagram shows you one such application where you feed an animal image to a neural network and it tells you that the image is of a dog.

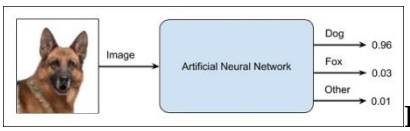


FIG 5

Why this is called a black-box approach is that you do not know why the network came up with a certain result. You do not know how the network concluded that it is a dog? Now consider a banking application where the bank wants to decide the creditworthiness of a client. The network will definitely provide you an answer to this question. However, will

you be able to justify it to a client? Banks need to explain it to their customers why the loan is not sanctioned?

Duration of Development

The process of training a neural network is depicted in the diagram below –

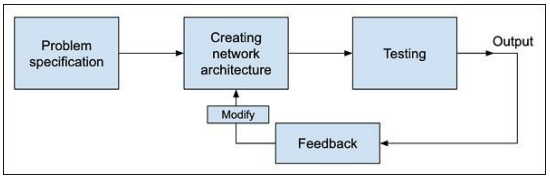


FIG 6

You first define the problem that you want to solve, create a specification for it, decide on the input features, design a network, deploy it and test the output. If the output is not as expected, take this as a feedback to restructure your network. This is an iterative process and may require several iterations until the time network is fully trained to produce desired outputs.

Amount of Data

The deep learning networks usually require a huge amount of data for training, while the traditional machine learning algorithms can be used with a great success even with just a few thousands of data points. Fortunately, the data abundance is growing at 40% per year and CPU processing power is growing at 20% per year as seen in the diagram given below –

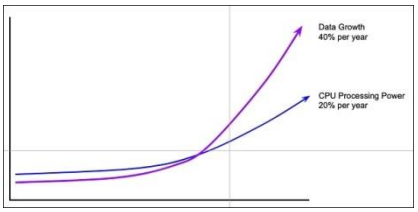


FIG 7

Computationally Expensive

Training a neural network requires several times more computational power than the one required in running traditional algorithms. Successful training of deep Neural Networks may require several weeks of training time.

In contrast to this, traditional machine learning algorithms take only a few minutes/hours to train. Also, the amount of computational power needed for training deep neural network heavily depends on the size of your data and how deep and complex the network is?

After having an overview of what Machine Learning is, its capabilities, limitations, and applications, let us now dive into learning “Machine Learning”.

3.1 Various other skills required for Learning

Machine Learning has a very large width and requires skills across several domains. The skills that you need to acquire for becoming an expert in Machine Learning are listed below −

- Statistics
- Probability Theories
- Calculus

- Optimization techniques
- Visualization

Necessity of Various Skills of Machine Learning

To give you a brief idea of what skills you need to acquire, let us discuss some examples –

Mathematical Notation

Most of the machine learning algorithms are heavily based on mathematics. The level of mathematics that you need to know is probably just a beginner level. What is important is that you should be able to read the notation that mathematicians use in their equations. For example - if you are able to read the notation and comprehend what it means, you are ready for learning machine learning. If not, you may need to brush up your mathematics knowledge.

$$fAN(net-\theta)=\begin{cases}\gamma net-\theta-\gamma if net-\theta\geq\epsilon if-\epsilon<net-\theta<\epsilon if net-\theta\leq-\epsilon fAN(net-\theta)=\{\gamma if net-\theta\geq\epsilon\\ net-\theta if-\epsilon<net-\theta<\epsilon-\gamma if net-\theta\leq-\epsilon\end{cases}$$

$$max\alpha[\sum i=1m\alpha-12\sum i,j=1mlabel(i)\cdot label(j)\cdot ai\cdot aj\langle x(i),x(j)\rangle]max\alpha[\sum i=1m\alpha-12\sum i,j=1mlabel(i)\cdot label(j)\cdot ai\cdot aj\langle x(i),x(j)\rangle]$$

$$fAN(net-\theta)=(e\lambda(net-\theta)-e-\lambda(net-\theta)e\lambda(net-\theta)+e-\lambda(net-\theta))fAN(net-\theta)=(e\lambda(net-\theta)-e-\lambda(net-\theta)e\lambda(net-\theta)+e-\lambda(net-\theta))$$

Probability Theory

Here is an example to test your current knowledge of probability theory: Classifying with conditional probabilities.

$$p(ci|x,y)=p(x,y|ci)p(ci)p(x,y)p(ci|x,y)=p(x,y|ci)p(ci)p(x,y)$$

With these definitions, we can define the Bayesian classification rule −

- If P(c1|x, y) > P(c2|x, y) , the class is c1 .
- If P(c1|x, y) < P(c2|x, y) , the class is c2 .

Optimization Problem

Here is an optimization function

$$max\alpha[\sum i=1m\alpha-12\sum i,j=1mlabel(i)\cdot label(j)\cdot ai\cdot aj\langle x(i),x(j)\rangle]max\alpha[\sum i=1m\alpha-12\sum i,j=1mlabel(i)\cdot label(j)\cdot ai\cdot aj\langle x(i),x(j)\rangle]$$

Subject to the following constraints −

$$\alpha\geq0,and\sum i-1m\alpha i\cdot label(i)=0\alpha\geq0,and\sum i-1m\alpha i\cdot label(i)=0$$

If you can read and understand the above, you are all set.

Visualization

In many cases, you will need to understand the various types of visualization plots to understand your data distribution and interpret the results of the algorithm’s output.

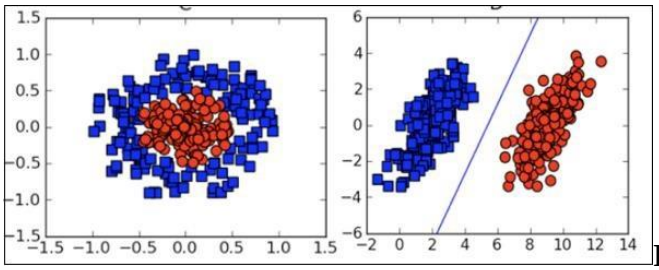


FIG 8

Besides the above theoretical aspects of machine learning, we need good programming skills to code those algorithms.

3.2 Applications of Machine Learning

1. Image Recognition:

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, Automatic friend tagging suggestion:

Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's face detection and recognition algorithm.

It is based on the Facebook project named "Deep Face," which is responsible for face recognition and person identification in the picture.

2. Speech Recognition

While using Google, we get an option of "Search by voice," it comes under speech recognition, and it's a popular application of machine learning.

Speech recognition is a process of converting voice instructions into text, and it is also known as "Speech to text", or "Computer speech recognition." At present, machine learning algorithms are widely used by various applications of speech recognition. Google assistant, Siri, Cortana, and Alexa are using speech recognition technology to follow the voice instructions.

3. Traffic prediction:

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

- Real Time location of the vehicle form Google Map app and sensors
- Average time has taken on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

4. Product recommendations:

Machine learning is widely used by various e-commerce and entertainment companies such as Amazon**,** Netflix**,** etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.

Google understands the user interest using various machine learning algorithms and suggests the product as per customer interest.

As similar, when we use Netflix, we find some recommendations for entertainment series, movies, etc., and this is also done with the help of machine learning.

5. Self-driving cars:

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

6. Email Spam and Malware Filtering:

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

- Content Filter
- Header filter
- General blacklists filter
- Rules-based filters
- Permission filters

Some machine learning algorithms such as **Multi**-Layer Perceptron**,** Decision tree**,** and Naïve Bayes classifier are used for email spam filtering and malware detection.

7. Virtual Personal Assistant:

We have various virtual personal assistants such as Google assistant**,** Alexa**,** Cortana**,** Sir**i**. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc.These virtual assistants use machine learning algorithms as an important part.These assistant record our voice instructions, send it over the server on a cloud, and decode it using ML algorithms and act accordingly.

8. Online Fraud Detection:

Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as fake accounts**,** fake ids, and steal money in the middle of a transaction. So to detect this, Feed Forward Neural network helps us by checking whether it is a genuine transaction or a fraud transaction.For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

9. Stock Market trading:

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's long short term memory neural network is used for the prediction of stock market trends.

10. Medical Diagnosis:

In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict the exact position of lesions in the brain.

It helps in finding brain tumors and other brain-related diseases easily.

11. Automatic Language Translation:

Nowadays, if we visit a new place and we are not aware of the language then it is not a problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.The technology behind the automatic translation is a sequence to sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

Chapter - 4

About the project

Plant-Seedling Classification

Problem statement

The target of this project is to distinguish between the different weed seeding and crop seeding of 12 different plant species. Hence its multiclassification problem.As we take .png image of a weed or a crop seedling and output the correspondent specie from our 12 classes. We are using CNN model to classify the plants. CNN is widely used in the field of computer vision for doing complicated task.

Seedling Classification Using Tensorflow

Abstract: Agriculture is very important to human continued existence and remains a key driver of many economies worldwide, especially in underdeveloped and developing economies. There is an increasing demand for food and cash crops, due to the increasing in world population and the challenges enforced by climate modifications, there is an urgent need to increase plant production while reducing costs. Preceding instrument vision methods established for selective weeding have confronted with major challenges for trustworthy and precise weed recognition. In this plant seedlings classification approach is presented with a dataset that contains images of approximately 960 unique plants belonging to 12 species at several growth stages It comprises annotated RGB images with a physical resolution of roughly 10 pixels per mm. Image classification has become one of the most important problems that Machine learning and deep learning can solve.

The problem here is the weed seedling is much like crop seedling and our goal is to be able to differentiate between them using Machine learning and deep learning techniques.

DATASET DESCRIPTION

This project we will use one of Kaggle Competition's dataset, this dataset contains images of approximately 960 unique plants belonging to 12 species at several growth stages It comprises annotated RGB images with a physical resolution of roughly 10 pixels per mm.

The problem here is the weed seedling is much like crop seedling and our goal is to be able to differentiate between them using Machine learning and deep learning techniques.

1. Data exploration

● Visualize the distribution of data

2. Data preprocessing

● Check for null and missing values

● resize images

● apply segmentation and sharpening for images

● 5.1 Label encoding

● 6.1 Split training and validation set

2. CNN

● 2.1 Define the model

● 2.2 Set the optimizer and annealer

● 2.3 Data augmentation

4. Evaluate the model

● 3.1 Training and validation curves

● 3.2 Confusion matrix

The final model is expected to be useful for classify the 12 different image species**.**

Traditional image classification approach

Traditional image processing approach involves interconnected steps such as segmentation, feature extraction and classification.... Many methods have been proposed for classification of WBCs using neural networks. It is a supervised machine learning algorithm which consists of input layer, hidden layer and output layer.

4.1 Source code

```
import numpy as np #dealing with arrays as model requires array to be paseed
```

```
import os #read or write a file within directory
import cv2  #dealing with images extracting data from images
import pandas as pd #data manipulation and analysis
from tqdm import tqdm # for  well-established ProgressBar
from random import shuffle #only shuffles the array along the first axis of a multi-dimensional array. The order of sub-arrays is changed but their contents remains the same.

LR = 1e-3  #0.001

MODEL_NAME = 'plantclassfication-{}-{}.model'.format(LR, '2conv-basic') # just so we remember which save

data_dir = ''
train_dir = os.path.join(data_dir, 'train')
test_dir = os.path.join(data_dir, 'test')
IMG_SIZE = 128

#list of categories in array format
CATEGORIES = ['Black-grass', 'Charlock', 'Cleavers', 'Common Chickweed', 'Common wheat', 'Fat Hen', 'Loose Silky-bent',
              'Maize', 'Scentless Mayweed', 'Shepherds Purse', 'Small-flowered Cranesbill', 'Sugar beet']
NUM_CATEGORIES = len(CATEGORIES)
print (NUM_CATEGORIES)
''' function that accept plant category and return array format of the vlaue , one-hot array
 am sure there's better way to do this .....'''

def label_img(word_label):
    if word_label == 'Black-grass': return [1,0,0,0,0,0,0,0,0,0,0,0]
    elif word_label == 'Charlock': return [0,1,0,0,0,0,0,0,0,0,0,0]
    elif word_label == 'Cleavers': return [0,0,1,0,0,0,0,0,0,0,0,0]
    elif word_label == 'Common Chickweed': return [0,0,0,1,0,0,0,0,0,0,0,0]
    elif word_label == 'Common wheat': return [0,0,0,0,1,0,0,0,0,0,0,0]
    elif word_label == 'Fat Hen': return [0,0,0,0,0,1,0,0,0,0,0,0]
    elif word_label == 'Loose Silky-bent': return [0,0,0,0,0,0,1,0,0,0,0,0]
    elif word_label == 'Maize': return [0,0,0,0,0,0,0,1,0,0,0,0]
    elif word_label == 'Scentless Mayweed': return [0,0,0,0,0,0,0,0,1,0,0,0]
    elif word_label == 'Shepherds Purse': return [0,0,0,0,0,0,0,0,0,1,0,0]
    elif word_label == 'Small-flowered Cranesbill': return [0,0,0,0,0,0,0,0,0,0,1,0]
    elif word_label == 'Sugar beet': return [0,0,0,0,0,0,0,0,0,0,0,1]
'''function that will create train data , will go thought all the file do this
----read the image in  grayscale mode ,resize it
---change it to numpy arrays and  append it to dataframe train with it`s associated category
'''

def create_train_data():
    train = []
    for category_id, category in enumerate(CATEGORIES):
        for img in tqdm(os.listdir(os.path.join(train_dir, category))):
            label=label_img(category)
```

```
            path=os.path.join(train_dir,category,img)
            img=cv2.imread(path,cv2.IMREAD_GRAYSCALE)
            img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
            train.append([np.array(img),np.array(label)])
    shuffle(train)
    return train
train_data = create_train_data()              #creating training data
'''function that will create test data , will go thought  file do this
----read the image in  grayscale mode ,resize it
---change it to numpy arrays and  append it to dataframe test but no category here of course
'''

def create_test_data():
    test = []
    for img in tqdm(os.listdir(test_dir)):
        path = os.path.join(test_dir,img)
        img_num = img
        img = cv2.imread(path,cv2.IMREAD_GRAYSCALE)
        img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
        test.append([np.array(img), img_num])

    shuffle(test)
    return test
test_data = create_test_data()          #creating test data

import tensorflow as tf #used for machine learning applications such as neural networks
import tflearn #modular and transparent deep learning library built on top of Tensorflow
from tflearn.layers.conv import conv_2d, max_pool_2d
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.estimator import regression
tf.reset_default_graph()

convnet = input_data(shape=[None, IMG_SIZE, IMG_SIZE, 1], name='input')

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
```

```
convnet = max_pool_2d(convnet, 5)

convnet = fully_connected(convnet, 1024, activation='relu')
convnet = dropout(convnet, 0.8)

convnet = fully_connected(convnet, 12, activation='softmax')
convnet = regression(convnet, optimizer='adam', learning_rate=LR,
loss='categorical_crossentropy', name='targets')

model = tflearn.DNN(convnet, tensorboard_dir='log')



if os.path.exists('{}.meta'.format(MODEL_NAME)):
    model.load(MODEL_NAME)
    print('model loaded!')

train = train_data
test = train_data

X = np.array([i[0] for i in train]).reshape(-1,IMG_SIZE,IMG_SIZE,1)
Y = [i[1] for i in train]

test_x = np.array([i[0] for i in test]).reshape(-1,IMG_SIZE,IMG_SIZE,1)
test_y = [i[1] for i in test]
model.fit({'input': X}, {'targets': Y}, n_epoch=3, validation_set=({'input': test_x},
{'targets': test_y}),
    snapshot_step=500, show_metric=True, run_id=MODEL_NAME)
model.save(MODEL_NAME)

#return Indexes of the maximal elements of a array
def label_return (model_out):
    if np.argmax(model_out) == 0: return  'Black-grass'
    elif np.argmax(model_out) == 1: return 'Charlock'
    elif np.argmax(model_out) == 2: return 'Cleavers'
    elif np.argmax(model_out) == 3: return 'Common Chickweed'
    elif np.argmax(model_out) == 4: return 'Common wheat'
    elif np.argmax(model_out) == 5: return 'Fat Hen'
    elif np.argmax(model_out) == 6: return 'Loose Silky-bent'
    elif np.argmax(model_out) == 7: return 'Maize'
    elif np.argmax(model_out) == 8: return 'Scentless Mayweed'
    elif np.argmax(model_out) == 9: return 'Shepherds Purse'
    elif np.argmax(model_out) == 10: return 'Small-flowered Cranesbill'
    elif np.argmax(model_out) == 11: return 'Sugar beet'

    import matplotlib.pyplot as plt

test_data = create_test_data()
fig=plt.figure(figsize = (18,10))
for num,data in enumerate(test_data[:12]):
```

```
    img_num = data[1]
    img_data = data[0]
    y = fig.add_subplot(3,4,num+1)
    orig = img_data
    data = img_data.reshape(IMG_SIZE,IMG_SIZE,1)
    model_out = model.predict([data])[0]
    str_label=label_return (model_out)
    y.imshow(orig,cmap='gray',interpolation='nearest')
    plt.title(str_label)
    y.axes.get_xaxis().set_visible(False)
    y.axes.get_yaxis().set_visible(False)
plt.show()
import pandas as pd
sample_submission = pd.read_csv('sample_submission.csv')
sample_submission.head(2)
test_data = create_test_data()
with open('sample_submission.csv','w') as f:
    f.write('file,species\n')
    for data in test_data:
        img_num = data[1]
        img_data = data[0]
        orig = img_data
        data = img_data.reshape(IMG_SIZE,IMG_SIZE,1)
        model_out = model.predict([data])[0]
        str_label=label_return (model_out)
        file = img_num
        species = str_label
        row = file + "," + species + "\n"
        f.write(row)
import pandas as pd
sample_submission = pd.read_csv('sample_submission.csv')
sample_submission.head(10)
```

4.2 Conclusion/ Results

To show the model quality we get the images which classified incorrectly in the test set this means the images that uncropped correctly can be misclassified.This model can help farmers to automate the task of classifying seedling plants and weed plants.We plotted the confusion matrix to observe which category is poorly classified by the classifier and observe its performance visually. We can see in the below confusion matrix plot, that the major misclassification happened between Loose Silky-bent and Black-grass. It looks like the classifier is having difficulty classifying these two categories. We plotted the confusion matrix to observe which category is poorly classified by the classifier and observe its performance visually. We can see in the below confusion matrix plot, that the major misclassification happened between

'Common wheat', 'Fat Hen' It looks like the classifier is having difficulty classifying these two categories. Hence, this is where the classifier needs improvement as these misclassification amount to 50% of the misclassification overall. We

believe overcoming this challenge will boost the F1-score significantly.

Justification

The results of the final classifier are much better than that of the benchmark model. It has score of 89.7% is a decent score when compared to 60.8% (benchmark model's performance). We believe the final solution will definitely contribute significantly towards solving the current problem and also with more training data and more preprocessing stages, there are possibilities of improving the model further.

Results

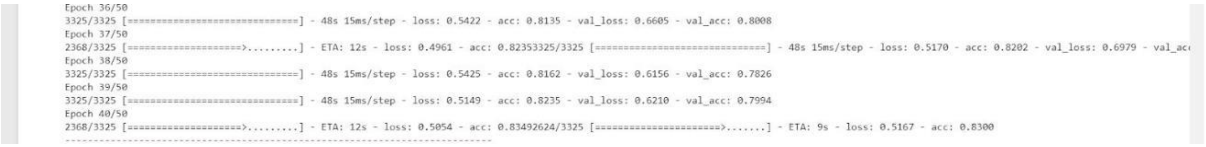
Model Evaluation and Validation:

During development, a validation set was used to evaluate the model. The final architecture and hyperparameters were chosen because they performed the best among the tried combinations.

complete description of the final model and the training process:

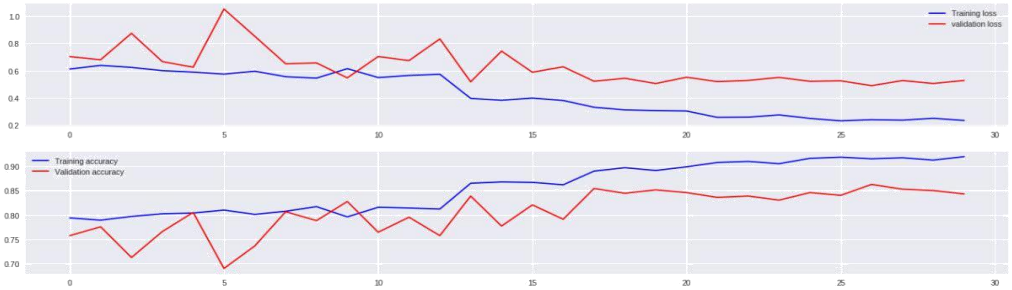
- The shape of the filters of the 1st and 2nd convolutional layers is 5*5 and 3*3 for the rest of convolutional layers.
- False positives are rare but present

After testing on validation set it give the score in the image:



With Test loss: 0.35367 and Test accuracy: 108. 915s

And after applying data augmentation. It gives a test score of .0.8973




```
model.fit({'input': X}, {'targets': Y}, n_epoch=3, validation_set=({'input': test
    snapshot_step=500, show_metric=True, run_id=MODEL_NAME)
```

```
Training Step: 2774  | total loss: 0.35367 | time: 108.915s
| Adam | epoch: 037 | loss: 0.35367 - acc: 0.8761 -- iter: 4736/4750
Training Step: 2775  | total loss: 0.35427 | time: 153.942s
| Adam | epoch: 037 | loss: 0.35427 - acc: 0.8791 | val_loss: 0.28483 - val_ac
c: 0.8973 -- iter: 4750/4750
--
```

As for the evaluation and validation, I have used 2 main values to checkAccuracy and validation loss.Before tuning some parameters (before reducing learning rate) From the figures we just shared, we can see that with 128 & 64 filters, we have reached a better score at 25 epochs and the validation accuracy is not dropping down

Having gone through many trials, coding and fixing bugs, and coding again, the current accuracy is more than enough to be trusted. Of course, there'll be an error ratio, but the accuracy is very acceptable and robust enough to be depended on in real life applications

.

Reflection:

We wanted to pick a problem that would give us much more clarity and exposure on how to deal with image classification problems, and we believe that choice of this problem has justified that need.

For this problem, after downloading the dataset from Kaggle, we loaded the dataset and converted the categorical file names into array of 1s and 0s using the labelbinarizer . We explored the data by visualizing the categorical distribution of the dataset by plotting a graph (using matplotlib) to check if the dataset is well balanced or not. After splitting the dataset into train, validation sets, the images were converted into 4D tensors for further processing. Once we built CNN models, we fed these tensors and evaluated the model's performance using confusion metrics. we watched, over multiple iterations, how does the number of layers in the convolutional network have a significant effect on the performance of the classifier, how does adding dropout layers reduce potential overfitting.

This project gave me a good insight on how to deal with future image classification problems and encouraged me to work on further improving my current model.

Improvement

As an improvement and future work, we would like to try data masking on the training set. Noise from the background of the images can be cancelled by masking images. We believe that without the background noise and restricting the visibility to the green leaves, the model can be trained better, and we may notice significant improvement in the performance.

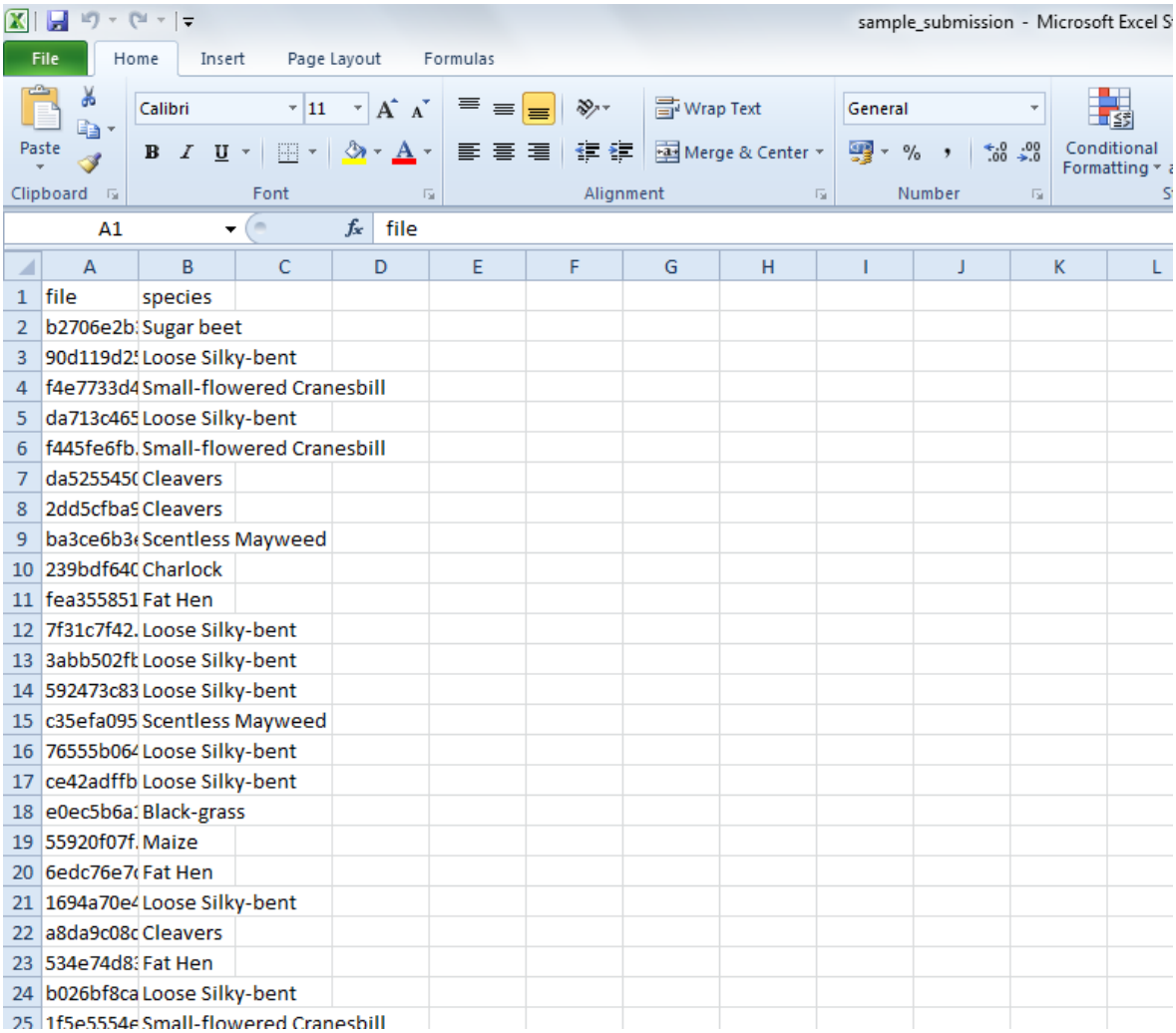
Another implementation that can be tried is data augmentation. As the dataset is highly unbalanced, augmenting data to the under-represented classes might give a good boost to the total number of training images yielding a well-balanced dataset. Training the model on such dataset may give us significant improvement in the performance.

References:

- https://en.wikipedia.org/wiki/HSL_and_HSV
- https://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm
- https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html /topic4.htm
- https://machinelearningmastery.com/evaluate-performance-deep-learning-models-kera s/
- https://www.kaggle.com/c/plant-seedlings-classification/data

OUTPUT:

Sample submission excel file example as



	A	B	C	D	E	F	G	H	I	J	K	L
1	file	species										
2	b2706e2b	Sugar beet										
3	90d119d2	Loose Silky-bent										
4	f4e7733d4	Small-flowered Cranesbill										
5	da713c465	Loose Silky-bent										
6	f445fe6fb	Small-flowered Cranesbill										
7	da525545(	Cleavers										
8	2dd5cfba9	Cleavers										
9	ba3ce6b3	Scentless Mayweed										
10	239bdf64(	Charlock										
11	fea355851	Fat Hen										
12	7f31c7f42	Loose Silky-bent										
13	3abb502f	Loose Silky-bent										
14	592473c83	Loose Silky-bent										
15	c35efa095	Scentless Mayweed										
16	76555b064	Loose Silky-bent										
17	ce42adffb	Loose Silky-bent										
18	e0ec5b6a	Black-grass										
19	55920f07f	Maize										
20	6edc76e7(	Fat Hen										
21	1694a70e4	Loose Silky-bent										
22	a8da9c08c	Cleavers										
23	534e74d8	Fat Hen										
24	b026bf8ca	Loose Silky-bent										
25	1f5e5554e	Small-flowered Cranesbill										

In [1]:
```
import numpy as np #dealing with arrays as model requires array to be paseed
import os #read or write a file within directory
import cv2  #dealing with images extracting data from images
import pandas as pd #data manipulation and analysis
from tqdm import tqdm # for  well-established ProgressBar
from random import shuffle #only shuffles the array along the first axis of a mul
LR = 1e-3
MODEL_NAME = 'plantclassfication-{}-{}.model'.format(LR, '2conv-basic') # just sc
```

In [2]:
```
data_dir = ''
train_dir = os.path.join(data_dir, 'train')
test_dir = os.path.join(data_dir, 'test')
IMG_SIZE = 128
```

In [3]:
```
#list of categories in array format
CATEGORIES = ['Black-grass', 'Charlock', 'Cleavers', 'Common Chickweed', 'Common
              'Maize', 'Scentless Mayweed', 'Shepherds Purse', 'Small-flowered Cr
NUM_CATEGORIES = len(CATEGORIES)
print (NUM_CATEGORIES)
```

```
12
```

In [4]:
```
''' function that accept plant category and return array format of the vlaue , or
 am sure there's better way to do this .......'''

def label_img(word_label):
    if word_label == 'Black-grass': return [1,0,0,0,0,0,0,0,0,0,0,0]
    elif word_label == 'Charlock': return [0,1,0,0,0,0,0,0,0,0,0,0]
    elif word_label == 'Cleavers': return [0,0,1,0,0,0,0,0,0,0,0,0]
    elif word_label == 'Common Chickweed': return [0,0,0,1,0,0,0,0,0,0,0,0]
    elif word_label == 'Common wheat': return [0,0,0,0,1,0,0,0,0,0,0,0]
    elif word_label == 'Fat Hen': return [0,0,0,0,0,1,0,0,0,0,0,0]
    elif word_label == 'Loose Silky-bent': return [0,0,0,0,0,0,1,0,0,0,0,0]
    elif word_label == 'Maize': return [0,0,0,0,0,0,0,1,0,0,0,0]
    elif word_label == 'Scentless Mayweed': return [0,0,0,0,0,0,0,0,1,0,0,0]
    elif word_label == 'Shepherds Purse': return [0,0,0,0,0,0,0,0,0,1,0,0]
    elif word_label == 'Small-flowered Cranesbill': return [0,0,0,0,0,0,0,0,0,0,1
    elif word_label == 'Sugar beet': return [0,0,0,0,0,0,0,0,0,0,0,1]
```

In [5]:
```
'''function that will create train data , will go thought all the file do this
----read the image in  grayscale mode ,resize it
---change it to numpy arrays and  append it to dataframe train with it`s associat

def create_train_data():
    train = []
    for category_id, category in enumerate(CATEGORIES):
        for img in tqdm(os.listdir(os.path.join(train_dir, category))):
            label=label_img(category)
            path=os.path.join(train_dir,category,img)
            img=cv2.imread(path,cv2.IMREAD_GRAYSCALE)
            img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
            train.append([np.array(img),np.array(label)])
    shuffle(train)
    return train
```

In [6]:
```
train_data = create_train_data()                #creating training data
```

```
100%|████████████████████████████████████| 263/263 [00:12<00:00, 21.35it/s]
100%|████████████████████████████████████| 390/390 [00:08<00:00, 48.02it/s]
100%|████████████████████████████████████| 287/287 [00:02<00:00, 97.12it/s]
100%|████████████████████████████████████| 611/611 [00:05<00:00, 106.33it/s]
100%|████████████████████████████████████| 221/221 [00:04<00:00, 44.26it/s]
100%|████████████████████████████████████| 475/475 [00:05<00:00, 84.02it/s]
100%|████████████████████████████████████| 654/654 [00:21<00:00, 30.64it/s]
100%|████████████████████████████████████| 221/221 [00:06<00:00, 32.89it/s]
100%|████████████████████████████████████| 516/516 [00:06<00:00, 83.34it/s]
100%|████████████████████████████████████| 231/231 [00:02<00:00, 86.12it/s]
100%|████████████████████████████████████| 496/496 [00:05<00:00, 88.68it/s]
100%|████████████████████████████████████| 385/385 [00:12<00:00, 30.85it/s]
```

In [7]:
```
'''function that will create test data , will go thought  file do this
----read the image in  grayscale mode ,resize it
---change it to numpy arrays and  append it to dataframe test but no category her

def create_test_data():
    test = []
    for img in tqdm(os.listdir(test_dir)):
        path = os.path.join(test_dir,img)
        img_num = img
        img = cv2.imread(path,cv2.IMREAD_GRAYSCALE)
        img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
        test.append([np.array(img), img_num])

    shuffle(test)
    return test
```

In [ ]:
```

```

In [8]:
```
test_data = create_test_data()          #creating test data
```

```
100%|████████████████████████████████████| 794/794 [00:05<00:00, 134.20it/s]
```

In [9]:
```
import tensorflow as tf #used for machine learning applications such as neural ne
import tflearn #modular and transparent deep learning library built on top of Ter
from tflearn.layers.conv import conv_2d, max_pool_2d
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.estimator import regression
tf.reset_default_graph()

convnet = input_data(shape=[None, IMG_SIZE, IMG_SIZE, 1], name='input')

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = fully_connected(convnet, 1024, activation='relu')
convnet = dropout(convnet, 0.8)

convnet = fully_connected(convnet, 12, activation='softmax')
convnet = regression(convnet, optimizer='adam', learning_rate=LR, loss='categoric

model = tflearn.DNN(convnet, tensorboard_dir='log')



if os.path.exists('{}.meta'.format(MODEL_NAME)):
    model.load(MODEL_NAME)
    print('model loaded!')

train = train_data
test = train_data

X = np.array([i[0] for i in train]).reshape(-1,IMG_SIZE,IMG_SIZE,1)
Y = [i[1] for i in train]

test_x = np.array([i[0] for i in test]).reshape(-1,IMG_SIZE,IMG_SIZE,1)
test_y = [i[1] for i in test]
```

```
curses is not supported on this machine (please install/reinstall curses for
an optimal experience)
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-packages\tensorflow
\python\framework\op_def_library.py:263: colocate_with (from tensorflow.pytho
```

```
n.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-packages\tflearn\ini
tializations.py:119: UniformUnitScaling.__init__ (from tensorflow.python.ops.
init_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.initializers.variance_scaling instead with distribution=uniform to get
equivalent behavior.
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-packages\tflearn\lay
ers\core.py:239: calling dropout (from tensorflow.python.ops.nn_ops) with kee
p_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - k
eep_prob`.
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-packages\tflearn\obj
ectives.py:66: calling reduce_sum_v1 (from tensorflow.python.ops.math_ops) wi
th keep_dims is deprecated and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-packages\tensorflow
\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops)
is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
```

In [15]:
```
model.fit({'input': X}, {'targets': Y}, n_epoch=3, validation_set=({'input': test
    snapshot_step=500, show_metric=True, run_id=MODEL_NAME)
```

```
Training Step: 2774  | total loss: 0.35367 | time: 108.915s
| Adam | epoch: 037 | loss: 0.35367 - acc: 0.8761 -- iter: 4736/4750
Training Step: 2775  | total loss: 0.35427 | time: 153.942s
| Adam | epoch: 037 | loss: 0.35427 - acc: 0.8791 | val_loss: 0.28483 - val_ac
c: 0.8973 -- iter: 4750/4750
--
```

In [16]:
```
model.save(MODEL_NAME)
```

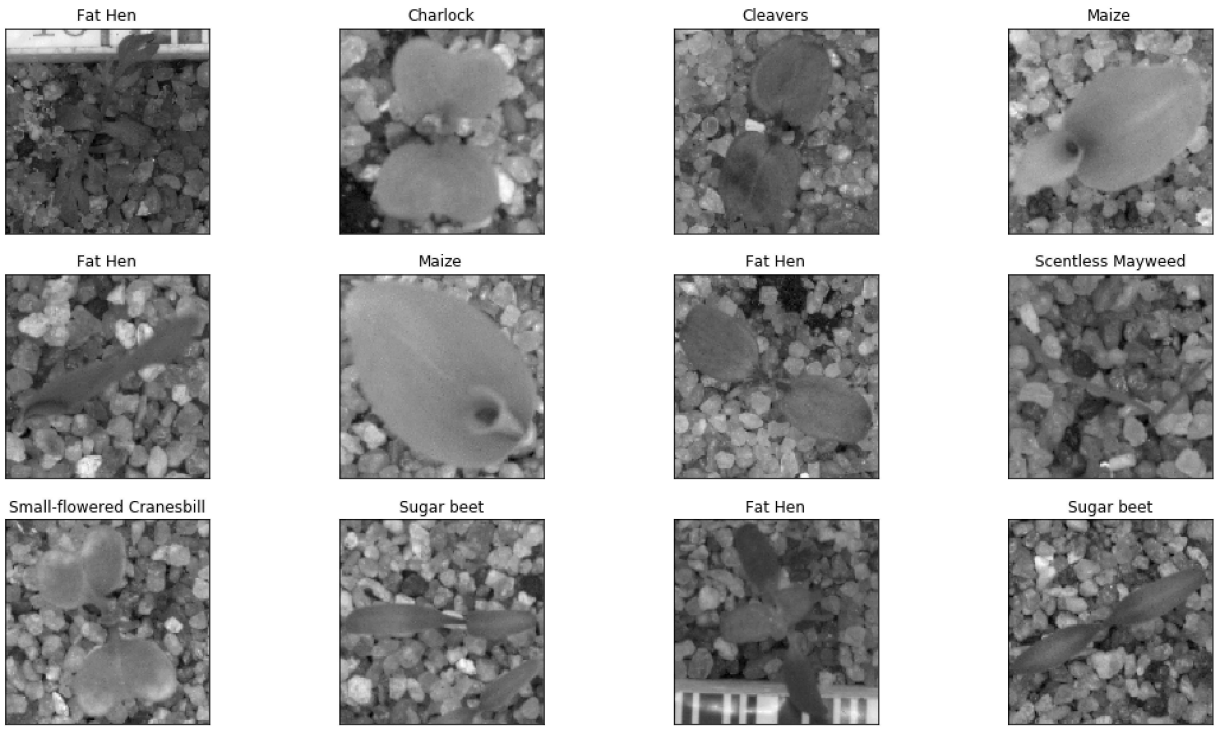
```
INFO:tensorflow:C:\Users\DELL\Desktop\nikhil project\plantclassfication-0.001-2
conv-basic.model is not in all_model_checkpoint_paths. Manually adding it.
```

In [18]:
```
#return Indexes of the maximal elements of a array
def label_return (model_out):
    if np.argmax(model_out) == 0: return  'Black-grass'
    elif np.argmax(model_out) == 1: return 'Charlock'
    elif np.argmax(model_out) == 2: return 'Cleavers'
    elif np.argmax(model_out) == 3: return 'Common Chickweed'
    elif np.argmax(model_out) == 4: return 'Common wheat'
    elif np.argmax(model_out) == 5: return 'Fat Hen'
    elif np.argmax(model_out) == 6: return 'Loose Silky-bent'
    elif np.argmax(model_out) == 7: return 'Maize'
    elif np.argmax(model_out) == 8: return 'Scentless Mayweed'
    elif np.argmax(model_out) == 9: return 'Shepherds Purse'
    elif np.argmax(model_out) == 10: return 'Small-flowered Cranesbill'
    elif np.argmax(model_out) == 11: return 'Sugar beet'
```

In [20]:
```
import matplotlib.pyplot as plt

test_data = create_test_data()
fig=plt.figure(figsize = (18,10))
for num,data in enumerate(test_data[:12]):
    img_num = data[1]
    img_data = data[0]
    y = fig.add_subplot(3,4,num+1)
    orig = img_data
    data = img_data.reshape(IMG_SIZE,IMG_SIZE,1)
    model_out = model.predict([data])[0]
    str_label=label_return (model_out)
    y.imshow(orig,cmap='gray',interpolation='nearest')
    plt.title(str_label)
    y.axes.get_xaxis().set_visible(False)
    y.axes.get_yaxis().set_visible(False)
plt.show()
```

```
100%|██████████| 794/794 [00:03<00:00, 252.61it/s]
```



In [21]:
```
import pandas as pd
sample_submission = pd.read_csv('sample_submission.csv')
sample_submission.head(2)
```

Out[21]:

	file	species
0	0021e90e4.png	Sugar beet
1	003d61042.png	Sugar beet

In [22]:
```
test_data = create_test_data()
with open('sample_submission.csv','w') as f:
    f.write('file,species\n')
    for data in test_data:
        img_num = data[1]
        img_data = data[0]
        orig = img_data
        data = img_data.reshape(IMG_SIZE,IMG_SIZE,1)
        model_out = model.predict([data])[0]
        str_label=label_return (model_out)
        file = img_num
        species = str_label
        row = file + "," + species + "\n"
        f.write(row)
```

```
100%|██████████| 794/794 [00:03<00:00, 250.06it/s]
```

In [23]:
```
import pandas as pd
sample_submission = pd.read_csv('sample_submission.csv')
sample_submission.head(10)
```

Out[23]:

	file	species
0	177d7e2a4.png	Fat Hen
1	8cfd98117.png	Loose Silky-bent
2	a2b703e21.png	Charlock
3	a8b431a3e.png	Fat Hen
4	b7ad92859.png	Common Chickweed
5	99569b224.png	Fat Hen
6	3b73c3b61.png	Black-grass
7	a1da8be3c.png	Maize
8	d350a25fa.png	Fat Hen
9	9643fc5f4.png	Charlock

In [ ]:
```
#work done output written in CSV File
```

In [28]:
```
!jupyter nbconvert --to html Project
```

```
[NbConvertApp] Converting notebook Project.ipynb to html
[NbConvertApp] Writing 746008 bytes to Project.html
```

In [ ]:
```

```