KING'S College LONDON

University of London

6CCS3PRJ Final Year Project

Deep Learning Approaches to Image Classification for Identifying Type of Food and its Calories

Final Project Report

Author: Mohammad Ashraful Islam Sadi

Supervisor: Mischa Dohler

Student ID: 1473013

April 22, 2019

Abstract

This paper describes an effective method to classify the type of fast food from popular franchises like KFC, McDonalds, Subway etc. The purpose of this study is to evaluate Transfer Learning with two custom deep convolutional neural networks for the classification of fast food images. The idea is to test the robustness of the technique by applying it in two different algorithms using the same data.Hand clicked images of 8 different categories of fast food were collected and then later rescaled to (255, 255) resolution. A total of 4152 images are used for training and 1032 images are used for validation. At the end of the study, both the algorithms are evaluated based on performance, training time and size of the output deep learning model. The two different algorithms used in the project are Xception and Apple VisionFeaturePrint.Scene. Both the algorithms are currently very popular due to their high performance and eifficient optimization process [1–4]. The project tests the robustness of both algorithms when used with Transfer Learning. The study also demonstrates a way by which image classification can be done from the phone, rather than getting the results from web servers. First the existing market leading ways of tracking calories in food are explored along with their pitfalls. Then a specific generic image recognition system is explored with advantages and disadvantages of it, which is recently gaining popularity in the market. Later, different Machine Learning to the problem domain are explored and considered based on their pros and cons. After extensive research and implementation testing, the proposed design is a custom Deep Convolutional Neural Network which uses Transfer Learning. The most significant part of the paper comes after that which is the architecture chosen for the Deep ConvNet and the type of fine tuning applied to it during training phase, followed by a programmatic implementation. The paper also dives into some caveats that should be taken into account on how to train the food classification models and common pitfalls to avoid. The results of the implementation are evaluated in terms of performance of the final deep learning model and possible future work are proposed. At the end the conclusions were being made on the research findings discovered, followed by the legal, social, ethical and professional issues, if exists are explored.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Mohammad Ashraful Islam Sadi

April 22, 2019

Acknowledgements

I would like to express my deepest gratitude to my supervisor Professor Mischa Dohler for his continuous encouragement, advice and support throughout the entire project. He was very kind to provide me the chance to explore the exciting field of Computer Vision and gain knowledge on state of the art image recognition models and Deep Learning Algorithms. Through his guided steps I was able to learn a lot on Deep Learning and its use cases. I would also like to thank the open source communities and online MOOCs e.g. UDEMY and LYNDA for providing excellent tutorials and software libraries which contributed to the project. I would like to thank Apple, Keras and TensorFlow for making their documentations publicly available for research purposes.

Contents

Chapter 1

Introduction

1.1 Project Objectives

The objective of this project is to leverage the image inference capabilities of Neural Networks through Transfer Learning, to detect the type of food and calories in it. Tracking calories in consumed food can be useful for anyone who is very conscious about what he/she eats on a daily basis. With the increasing growth in technology, it is possible now to track the amount of calories in a piece of food using deep learning. The implementation is also useful for people who does not always have an active internet on their devices as the application would work fine even if it is in offline mode. This means that users do not have to exchange their personal daily food information in the cloud to get a prediction of the type of food. It is first important to acknowledge and analyse the advancements made in the field of Artificial Intelligence, Machine Learning and Computer Vision to complete the project object. After the completion of this analysis, based on the options explored, two model architecture will be chosen for the project and explored through implementation. After that the results of both the approaches will be evaluated and future work will be proposed.

Computer Vision softwares are transforming the way people approach to image recognition problems, by not only making it easier to access, but also easier to use during execution. According to a research published by InDataLabs in 2018, these softwares are making lives easier and interesting for the common people. USA and China are currently heavily invested in this technology with investment figures leading up to 120 million and 3.9 billion USD respectively [5]. Traditional Machine Learning algorithms, although very complex, still struggle to generate good results when presented with a vision problem such as classification. According to Yann

LeCun, Yoshua Bengio and Geoffrey Hinton, useful machine learning techniques are limited in their ability to process raw natural data [6]. Deep Learning algorithms have been observed to outperform traditional machine learning approaches and generate much better benchmarks in image classification. The algorithms become more efficient when presented with more relevant data to learn, which is not true in machine learning algorithms. The main aim through out this project is to explore the approaches in existing market leading calorie tracking softwares and identify the short comings, eventually implementing a software able to address those issues.

Over the years, calorie tracking applications gained popularity as more people are getting conscious to the idea of healthy eating. Such applications enables the users to manually input the name of their meals through text and get a prediction of the amount of calories the food contains. The process is very inconvenient to use due to the manual approach of searching the food from the database and entering the name. Thus it is a good idea to handle the data input system in the form of an image of the food consumed. This also enable the users to interact with the food through their smartphone cameras making the overall process very user-friendly. As mentioned earlier, recent advancements made in the field of Computer Vision have made this approach to food and calorie detection very useful.

1.2 Report Structure

The report is divided into specific sections with the first section being the Introduction which discusses the project objectives, structure and thesis contribution.

The second section of the report discusses the motivation for the research project, current industry leading ways of tracking calories in food, important background information on Artificial Intelligence and Deep Learning. It is very important to acknowledge the depth of effort and research put into the field of Computer Vision so far.

The third and fourth section describes the main contributions of the project outlining the type of solutions used for the problem identified in Section 2 and how to implement it using the best Software Development Methodologies.

The fifth section reinforces the third and fourth by providing a practical way of implementation the deep learning approaches decided for the project. Evaluation is the sixth section of the report where the project evaluates the simulation results from Section 5 in tabular and graphs form.

The seventh and eight sections of the project discuss possible improvements to improve the deep learning models to be more efficient for future use and overall conclusion for the

research. It identifies the problems in each approach and enforces the completion of requirement specification.

The last section of the project explores various sectors of issues linked to the research which are Legal, Social, Ethical and Professional.

1.3 Thesis contribution

The thesis set out to test the effectiveness of the hypothetical technique Transfer Learning and came to a certain conclusions at the end which are:

- The project explore a novel technique in Deep Learning for analyzing fast food images which is Transfer Learning. Transfer learning leverages the recent advancements in deep learning to use pre trained ImageNet weights.
- The thesis solidies the hypothesis that weights initialised with ImageNet values perform better than random initialisation in terms of computation resources. Better ImageNet architectures leads to better classification results.
- Transfer learning works well when presented with a relatively small data set. The technique successfully managed to achieve good accuracy values on two different approaches with same data.
- The project experimentally shows the implementation of two Transfer learning approaches when used as a feature extractor.
- The first Transfer learning approach with custom algorithm Xception achieved more than 90% accuracy (91%). The approach allowed the user to use Xception as a feature extractor.
- The second Transfer learning approach with custom algorithm VisionFeaturePrint.Scene algorithm also achieved more than 90% accuracy (99%). The algorithm only takes the training data and allows the user to manually set up number of iterations and few types of augmentations. Later the deep learning CNN model is wrapped up with an iOS application.
- This is the first time VisionFeaturePrint.Scene has been implemented to detect type of fast food. There has no previous research related to this domain with this specific algorithm.

- The iOS mobile application 'CalorieTracker.ai' managed to perform offline image classifications.
- The project introduces a new way to automate the classification of fast food images which is better than the current market leading ones.
- The project also emphasizes on the importance of adding more data set for retraining to improve the performance of both the custom Deep CNNs in future work.

Chapter 2

Background

2.1 Motivation for Food and Calorie Detection

Do you like exploring new places without thinking about the food offerings of the location? Of course, you do, who does not. Imagine being a tourist in a new place you have never visited before and the only source of information you have with you is your mobile phone. Now when exploring new places, the need for maintaining healthy diet is a very important aspect to keep your body active with the right ingredients that would supplement you during your stay. The current state of the art calorie measurement relies on the back of the packet food with nutrition values.

Figure 2.1: Values of calories in a packed food

But what about unpacked food which are freshly prepared, hot and ready to eat? Ideally you will take out your phone and perform a google search which requires the presence of a decent internet connection. Lets even complicate the situation a bit more and say that your phone does not even possess a network sim yet for providing internet. You just came to the

new country and have not bought a mobile sim yet. The project aims to address this issue and motivate a first step towards the implementation of a mobile application that takes a picture of the food and within microseconds provides a measurement of the number of calories in that particular meal. The application proves its robustness by giving you accurate predictions in real time, without the presence of a network connection or internet.

The need for automatic food and calorie detection systems from images has increased substantially as people are becoming more aware of the benefits of a healthy diet. The applications not only will provide the user a fast classification of the type of fast food but also the calories in it. This can be very helpful for medical patients with certain calorie restrictions in their diets, ultimately benefiting them to providing a safer way of consuming food [**?**].

2.1.1 The Fitness Mobile Application Market

According to Statista, the volume of mobile fitness app market which was 1.77 billion dollars in 2016, is expected to rise to 4.1 billion by 2021 [7]. As more and more people are warming up to the idea of seeking a healthy lifestyle, the fitness app market is evolving at a high pace. A study done by Global-Go company in 2015 showed that healthy and fitness apps came in second in terms of daily active users [7]. The figure below demonstrates the growth of the fitness app market from 2015 and the predictions generated for 2021.

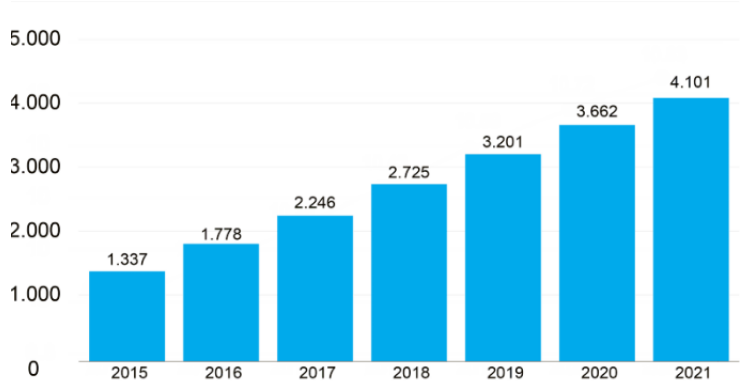


Figure 2.2: Rise of Fitness application market - Source: https://traffichabits.com/the-top-ten-most-profitable-fitness-apps-markets-448dbbdded6c

2.2 Scope of the Research Project

The scope of the project is constrained by an allocated time of five months and being only one person working on the project. It is also constrained by the data that will be used for the

implementation which should include:

1. A fair amount of both training and validation data for each desired class of food.
2. Eight different desired classes of food.

Due to a fixed amount of time being available, time boxing will enforce the adjusting of the scope to the time available. The project will be broken down into multiple stages, in order to check the validity of the implementation and whether or not it meets the desired end product. In increasing order, the stages are as follows:

- Data Preparation Stage:
  - Gather data of the different types of food items from global high street franchises e.g. KFC, McDonald's etc.
  - Classify the visual data and label them into their appropriate categories with calorie amount.
- Image Recognition Model Building Stage:

  Here two different image recognition models will be implemented using Transfer Learning, one by building a custom deep learning CNN and the other one, by using Apple's machine learning framework called Create ML released in WWDC 2018.

  The steps for the first approach are:

  1. Implement a custom Deep Convolutional Neural Network(CNN) using Transfer Learning.
  2. Supply the custom deep CNN with the acquired training data to learn specific features about the food objects.
  3. Observe the accuracy results on the validation images and decide on the acceptance level for the model to be extended.

  Second approach includes:

  1. Passing the training data set into the Create ML framework and let it automatically use Transfer Learning for training.
  2. Supply the trained model with test data set for validation.
  3. Similar to the previous approach 'Observe the accuracy results on the validation images and decide on the acceptance level for the model to be extended'.

- Wrap up the model in an iOS mobile application:
  - Convert the image recognition model into core ml format and use it inside an iOS application for image prediction through the built in camera application.

2.3 Current industry leading ways of identifying food and calories

Several calorie tracking applications already exist currently in the market which allow users to track their intake of food and the amount of calories in it. These applications have been dominating the industry for many years now serving millions of customers. The sub section will now explore the leading mobile applications for calorie tracking in 2019, starting from the most popular one.

2.3.1 Case Study 1: MyFitnessPal

The most popular and used mobile application in the calorie tracking industry is MyFitnessPal. According to Lifewire, the mobile application boasts a database of more than six million food information [7]. The application uses the power of bar code detection to track any food from its packet or tin. It uses the bar code to look up in its database table for the type of food manually registered. The figure below demonstrates how the application works in real time.

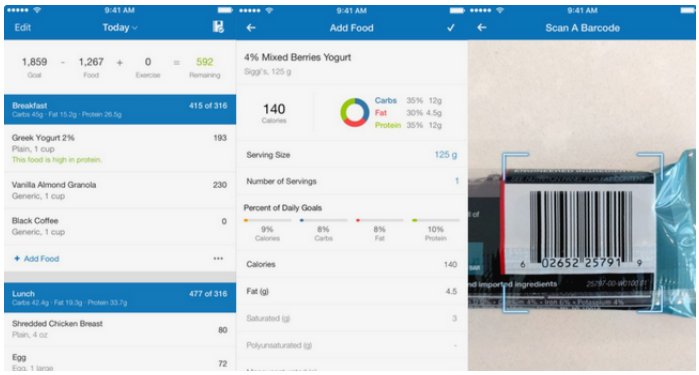


Figure 2.3: My FitnessPal bar code detection - Source: TheGuardian, 2016

The application works only by either the user manually logging the food he/she consumed or by scanning the bar code through the camera application (which does not cover un-packed food). That is the biggest limitation of this popular application with more than 50 million customer database. Although the bar code scanning is efficient, the process still requires the

presence of the packet. Without the packet, the user can still search for the food from the cloud database of the application which requires the presence of WIFI or network provided internet, but that will be a random guess from the system, hence losing accuracy. Currently the application has a yearly fee of 39 pounds.

2.3.2 Case Study 2: Lose It!

The second mobile application is the popular application Lose It!. According to thedroidguy.com, the app also boasts a large number of active users and is one of the most popular calorie tracking application of 2019 [8]. The application has popular food franchises like KFC, Nando's etc.

Figure 2.4: Mobile Application LoseIt - Source: The Guardian, 2016

You can search for the food or browse from the restaurant database menu and it will tell you the amount of calories that meal has. The application also has the option to take a picture of the food and let the system provide suggestions of the type of food it think it is. The application has a similar disadvantage to that of MyFitnessPal, it requires a constant presence of either WIFI or network provided internet connection as all of its database are in the cloud. When the user requests the data for any specific food item, it grabs the calorie information of that food from the cloud.

There are more calorie tracking mobile applications on the market such as Calorie Counter, SparkPeople CalorieTracker etc which use the same tracking methodology described in the previous two applications. As a result, none of the method of tracking calories in the market is currently efficient. The figure below shows the statistics of the market leading applications for calorie tracking.

Figure 2.5: Comparison of different mobile dietary applications. Total number of downloads information is taken from Android App store as Apple does not provide that information. Source: researchgate

App Name	Record by Photo	Food Photo Recognition	Record by Search	Record by barcode scan	Record by free text	Rating out of 5	# Rating	Launch time	# Download
MyFitnessPal	N	NA	Y	Y	Y	4.6	1,000,000	2005	50,000,000
FatSecret	Y	N	Y	Y	Y	4.4	193,000	2007	10,000,000
Noom Coach	N	NA	Y	Y	Y	4.3	166,000	2010	10,000,000
Lose It!	Y	Y	Y	Y	Y	4.4	55,000	2011	5,000,000
Sparkpeople	N	N A	Y	Y	Y	4.4	22,000	2012	1,000,000
MyNetDiary	N	NA	Y	Y	Y	4.5	18,000	2010	1,000,000
MyPlate	N	NA	Y	Y	Y	4.6	16,000	2010	1,000,000

2.3.3 Case Study 3: Calorie Mama AI

Out of all the available mobile applications, Calorie Mama AI is the most interesting and efficient application. The application is rapidly gaining popularity as Samsung has already integrated the software's API (Application Programming Interface) in its native Bixby application [9]. The platform allows the user to take a picture and it automatically identifies the type of food and the amount of calories it contains. It is currently not state of the art as it generates more than 5 suggestions of different food from each picture. As a result, it asks the user to verify its prediction, eventually learning gradually from user feedback.

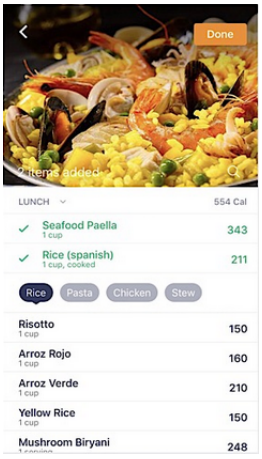


Figure 2.6: Calorie Mama AI application - Source: Google

Although the application uses machine learning for its image recognition technology, it still requires the presence of internet to do the task. Without the internet the classifier wont function and will not be able to generate any predictions. As a result, the application will struggle to function when the user goes to a new place and has not set up the internet connection yet.

As seen many times, that the lack of internet highly affects the performance of all the mobile applications. The project aims to implement a mobile application which will contain the image classifier inside the app so that it does not require internet connection to generate food and calorie predictions from images. But before that, lets look at some of the major advancements in the field of Artificial Intelligence for the past few years and how it can help in planning the implementation of the project.

2.4 Artificial Intelligence

Due to the recent advancements in cognitive technologies, the growth of Artificial Intelligence is promising efficient solutions to problems like Image recognition, which were thought to be unsolvable by any system just 10 years before. According to Dan Sommers, Senior Director, Market Intelligence Lead at Qlik, the rise of AI does not necessarily mean the replacement of human in the workplace, but day by day, it is hard to ignore the benefits or using the techniques provided by Artificial Intelligence systems [10].

"In 2019 we will see artificial intelligence evolve to be designed around humans. AI will likely add more jobs than it takes away, eliminating manual processes and leaving workers with the time needed to be empowered by AI" - Sommers told 'The Next Web'.

According to Wikipedia, AI can be stated as the cognitive expertise demonstrated by intelligently trained machines, in contrast to natural intelligence observed in human and animals over the decades [11]. A recent survey by Forbes in 2019, revealed some remarkable statistics:

- About 47 percent of business executives said their companies have deployed at least one AI capabilities to make their business processes more efficient and 21 percent said that they have multiple AI deployments in their business processes [12].
- This year 70 percent of AI service based companies claimed that they are going to use AI from the cloud as it is more convenient in terms of portability and ease of access [12].

One of the pioneers of 5g technology, Dr Mischa Dohler have recently given a talk on Artificial Intelligence at the AI and Robotics Summit 'The Future is Now', Dec 2018. He put forward some ground breaking concepts of developing machine learning models that can co exist with human to drive businesses forward.

"AI will change business significantly in the future but i think the biggest add on is that AI will co exist with human decision making" - Dr Mischa Dohler, Dec 2018.

From the context of the project, AI can be thought of as the machine capable of learning and applying state of the art visual recognition algorithms to image based problems without being explicitly programmed for them. As well as making a big impact in the medical and finance sector, AI is also causing big waves in the employment sector as well. A report published by LinkedIn in December 2018, states the growth of AI skills based jobs as one of the fastest growing in the industry with a global increase of 190 percent, from 2015 to 2017. The need for a machine learning engineer has grown by 12 times in the past year [13].

One of the pioneer of Artificial Intelligence in modern times, Andrew Ng claims that in order to build better business models we need AI integrated into the core components.

"AI is the new electricity"- by Andrew Ng Just like the rise of electricity transformed every major industry in the world, with the rise of AI there is a huge possibility of a big transformation in all major industries soon [14].

2.5 Machine Learning

Machine learning is a use of man-made reasoning (AI) that gives frameworks the capacity to consequently take in and enhance as a matter of fact without being expressly modified. Machine learning circles around the improvement of PC programs that can get to information and use it learn for themselves. The way toward learning starts with perceptions or information, for example, models, direct understanding, or guidance, so as to search for examples in information and settle on better choices later on dependent on the precedents that we give. The essential point is to permit the PCs adapt naturally without human mediation or help and modify activities as needs be [15].

According to Wikipedia, Machine learning (ML) is the logical investigation of calculations and measurable models that PC frameworks use to adequately play out a particular undertaking without utilizing unequivocal directions, depending on examples and deduction. It is viewed as a subset of man-made reasoning. Machine learning calculations construct a numerical model of test information, known as "preparing information", so as to settle on forecasts or choices without being expressly customized to play out the task. Machine learning calculations are utilized in the utilizations of email separating, location of system interlopers, and PC vision, where it is infeasible to build up a calculation of explicit directions for playing out the assignment. Machine learning is firmly identified with computational measurements, which focuses around making forecasts utilizing PCs [16].

In 2019, Machine Learning is being used in numerous places ranging from predicting the

repairing period of combat vehicles in the U.S Army to predicting the likelihood of a patient's death in Google, which surprisingly 95 percent accurate. Stock markets are also currently re-evaluated using ML frameworks and the algorithms under the hood are making the trading process more efficient. By the end of last year (2018), it was very clear that AI assistants are already taking over the consumer and call center industry. Popular voice based AI Alexa, can already do a lot of cloud based functions for you such as calling an UBER, ordering food from your favourite restaurants and help in many house chores, by just using your voice to control it [87]. Most big telecom companies in the US and UK have already shifted to the use of AI assistants to connect customer queries to the right available person of the company for reducing overheads and providing a better user experience [17].

Machine Learning is divided into two main components of learning: Supervised and Unsupervised.

2.5.1 Supervised Learning

Supervised learning, with regards to man-made consciousness (AI) and machine learning, is a sort of framework in which both info and wanted yield information are given. Info and yield information are marked for order to give a learning premise to future information preparing. Supervised machine learning frameworks furnish the learning calculations with realized amounts to help future decisions. Chat bots, self-driving autos, facial acknowledgment programs, master frameworks and robots are among the frameworks that may utilize either directed or unsupervised learning. Supervised learning frameworks are for the most part connected with recovery based AI however they may likewise be fit for utilizing a generative learning model.

Preparing information for directed learning incorporates a lot of precedents with combined info subjects and wanted yield (which is additionally alluded to as the supervisory flag). In supervised learning for image processing, for instance, an AI framework may be furnished with named pictures of vehicles in classes, for example, autos and trucks. After an adequate measure of perception, the framework ought to almost certainly recognize and order unlabelled pictures, at which time preparing can be said to be finished [18].

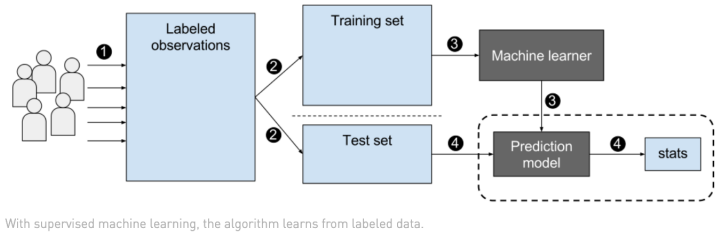


Figure 2.7: Visualization of Supervised Machine Learning Process - Source: NVIDIA

The majority of sensible machine learning uses supervised learning. Supervised learning is wherever you've got input variables (x) associated an output variable (Y) and you employ an algorithmic rule to be told the mapping perform from the input to the output.

$$Y = f(X)$$

Figure 2.8: The base equation for supervised machine learning - Source: MachineLearningMastery

The goal is to approximate the mapping perform thus well that after you have new input file (x) that you just will predict the output variables (Y) for that data. It is referred to as supervised learning as a result of the method of associate algorithmic rule learning from the training data set is thought of as a lecturer direction the educational process. we all know the right answers, the algorithmic rule iteratively makes predictions on the coaching knowledge and is corrected by the teacher. Learning stops once the algorithmic rule achieves a suitable level of performance [19].Two main areas where supervised learning is most useful are classification and regression problems.

Classification is the process of categorizing a group of objects, while only using some basic data features that describe them. There are lots of classifiers out there today like Logistic Regression, Support Vector Machines, Naive Bayes and of Neural Networks (described a bit later in the paper). The firing of a classifier or activation as it is commonly called generates a score. A high score means that the prediction is accurate and the classifier is successful in classifying the desired output from the set of visual images [20].

Classification issues raise the algorithmic rule to predict a distinct price, characteristic the input file because the member of a selected category, or group. During a coaching data set of animal pictures, that will mean every photograph was pre-labeled as cat, native bear or turtle. The algorithmic rule is then evaluated by however accurately it will properly classify

new pictures of different koalas and turtles [21]. From the project perspective, classification of a fast food and predicting the calories it contains can be one of the use cases.

On the opposite hand, regression issues examine continuous knowledge. One use case, statistical regression, ought to sound acquainted from pure mathematics class: given a selected x price, what is the expectation of the y variable?

In a regression problem, the system tries to predict a value for an input based on previous information. Now if you know a bit about supervised machine learning, you would not be wrong in assuming that classification and regression were similar to each other. In one way, regression is almost similar to the classification problem. An input is being mapped to an output based on earlier observations. But in the case of regression, you are trying to estimate a value and not just a class of an observation. Two important characteristics of regression are that the responses that can be expected from the model are always quantitative in nature. Also, the model can only be created by taking into consideration past data. A popular algorithm that performs regression is the Linear regression algorithm [22].

An additional realistic machine learning example is one involving many variables, like Associate in Nursing algorithmic rule that predicts the worth of a flat in San Francisco supported square footage, location and proximity to transport. Supervised learning is, thus, best suited to issues wherever there's a group of obtainable reference points or a ground truth with that to coach the algorithmic rule [21].

2.5.2 Unsupervised Machine Learning

It is the situation where the machine is trained using unclassified and unlabelled data, allowing the algorithm to inference a pattern without the instruction of a desired output. To group unsorted data sets according to similarities, patterns and differences without no prior knowledge is the main challenge for the system here. The lack of instructions to learn ensures that no form of prior training is provided to the system. Thus the machine is restricted to seek out any hidden structure in an un-tagged sea of data set by itself [23]. Essentially the AI goes into the inconvenience dazzle with just its optimal intelligent operations to manual it. Striking since it appears, Unsupervised framework learning is the ability to determine complex issues the use of simply the information data, and the paired on/off practical insight components that every one PC frameworks are developed on. No reference statistics is required [19].

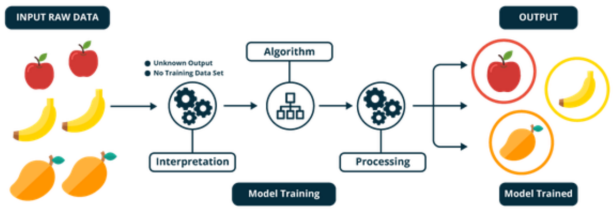


Figure 2.9: Unsupervised Machine Learning - Source:https://medium.com/@gowthamy/machine-learning-supervised-learning-vs-unsupervised-learning-f1658e12a780

2.6 Introduction to Deep Learning: Artificial Neural Networks

Deep Learning is a specialised form of supervised learning which is more powerful than usual Machine Learning methodology. In the past, several studies has been successfully carried which proves the efficiency of deep learning features in different applications [24–28]. Artificial Neural Networks (ANNs) are programming executions of the neuronal structure of our minds. Considering the complex structure of a human brain, it is easier to start off with the brain containing neurons which are somewhat similar to natural switches. These can change their yield state contingent upon the quality of their electrical or compound information. The neural system in an individual's cerebrum is a colossally interconnected system of neurons, where the yield of some random neuron might be the contribution to a large number of different neurons. The connections are reinforced as learning happens by more than once enacting certain neural associations over others. The chances of producing an idea result given a predefined input becomes more likely. This learning includes input when the ideal result happens, the neural associations causing that result winds up reinforced [29].

A neural network is highly structured and comes in layers. The first layer is the input layer, the final layer is the output layer and all the other layers in between are referred to as hidden layer. A neural network can also be thought as the result of spinning classifiers together in the web of nodes. This is because each node in the hidden and output layers has its own classifier. A set of inputs is passed to the first hidden layer, the activations from that layer are passed to the next layer and so on, until the process reaches the output layer, where the results of the classification are determined by the scores at each node. This happens for each set of inputs [20]. The figure demonstrates the above concept visually.

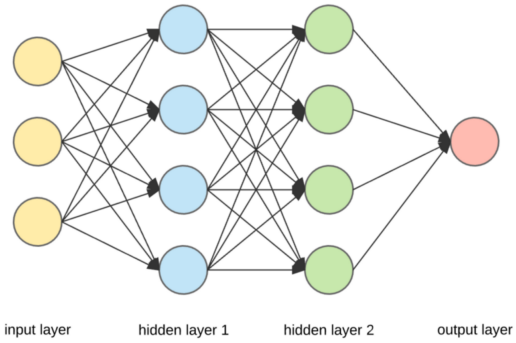


Figure 2.10: Artificial Neural Networks - Source: Towards Data Science, Medium

Artificial neural networks endeavour to disentangle and imitate this mind conduct. They can be prepared in a regulated or unsupervised way. In a regulated ANN, the system is prepared by giving coordinated information and yield information tests, with the goal of inspiring the ANN to give an ideal yield to a given info. A model is an email spam channel the info preparing information could be the include of different words the body of the email, and the yield preparing information would be an arrangement of whether the email was really spam or not. In the event that numerous instances of messages are gone through the neural system this enables the system to realize what input information makes it likely that an email is spam or not [29].

2.6.1 Neurons

The fundamental unit of calculation in a neural system is the neuron, frequently called a hub or unit. It gets contribution from some different hubs, or from an outer source and processes a yield. Each info has a related weight (w), which is appointed based on its relative significance to different information sources. The hub applies a capacity to the weighted entirety of its information sources [30].

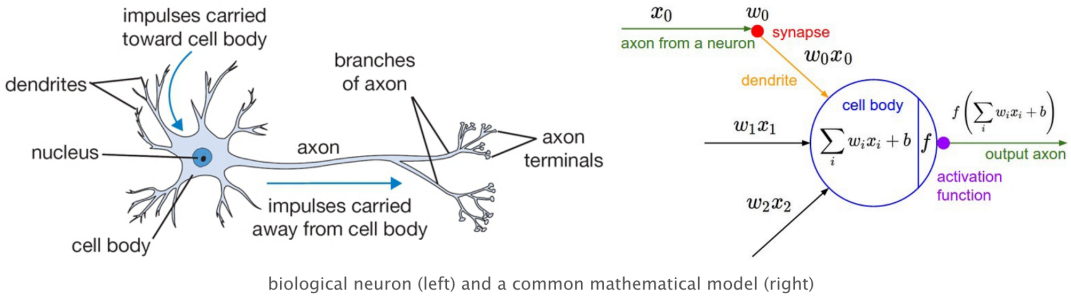


Figure 2.11: Artificial Neuron - Source: https://towardsdatascience.com/a-gentle-introduction-to-neural-networks-series-part-1-2b90b87795bc

2.6.2 Why use Deep Learning when dealing with classification problems

We are currently living in an era where Neural Nets have the possibility to revolutionise the field of Artificial Intelligence. From the medieval age, we observed sophisticated systems being very good at repetitive calculations and detailed instructions. But they were not recognised to be good at understanding patterns and solving related complex classification problems. This is all about to change, thanks to deep learning. A basic architecture tool like Support Vector Machine (SVM) or Logistic Regression is good enough to help a system in analysing simple patterns. Neural nets start to outperform competition when the data contains multiple different inputs. In 1995, Bishop [31] showed that Artificial Neural Networks fits the data with the highest measure of success when dealing with pattern recognition tasks. But a neural net with a small number of layers lacks effectiveness at predicting when the patterns start to get complex, because the number of nodes required in each layer grows exponentially with the number of possible patterns in the data. As a result the accuracy suffer highly as the training becomes way too computationally expensive. Basic classification engines and shallow neural nets start becoming useless when dealing with a more intricate pattern like image of a burger. Hence the effective and practical choice in this situation is a deep net [14].

Deep net models have shown better performances than shallow models [32,33]. Such models are now being used to develop even more efficient algorithms for vision problems. How does a deep net successfully recognise these complex patterns? The mystery is that deep nets can separate the unpredictable examples into a progression of less complex examples. For example, lets say that the net had to decide whether or not an image contained a burger and what type of burger it is. A deep net would first use edges to detect different parts of the burger like type of bun, type of fillet inside (some burgers are distinguished by two fillets), presence of cheese and so on. The net would then combine the results together to form the whole burger. The core of the strength of deep nets comes from the significant feature of simple patterns as building blocks to detect complex patterns. Over the years the prediction accuracy of the nets enjoyed an impressive growth. In 2015, a deep net from Google beat human in recognising patterns and the model was pretty accurate in identifying the dominant objects in an image [34].

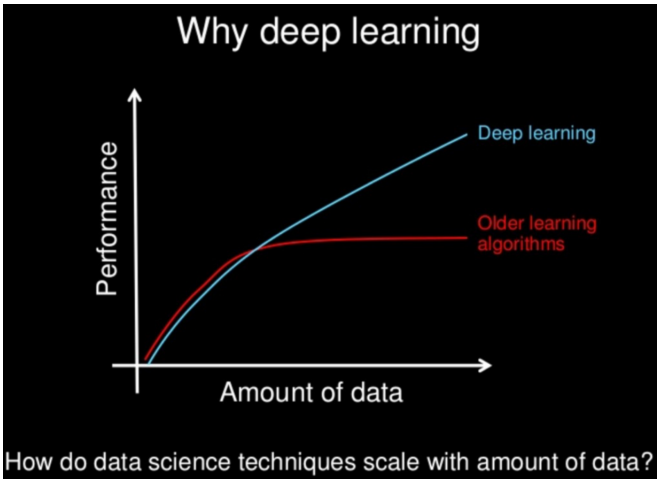


Figure 2.12: Performance of Machine Learning algorithms with increasing data - Source: Analyticsvidhya

Its not staggering that deep nets were aroused by the structure of our own human minds. Indeed, even in the early time frame of neural systems, analysts wished to connect a curiously large scope of perceptrons along in a layered with web, a hypothesis that improved their exactness in prediction. It is trusted that our minds have a dreadfully profound structure which we will in general decode designs rather like a profound web. By identifying and combining simple patterns, we recognise sophisticated patterns. However there is one disadvantage to the deep net process. Deep nets take much larger time to train and require vast amounts of data as the prediction can only be accurate as more layers in the network are labelled with the sets of data provided. The good news is that the ongoing advances in computing power have amazingly reduced the quantity of time needed to train the net [14]. Currently, a high performance CPU with 100K images can finish training a complex net within roughly 6 hours [35].

2.6.3 A peek at Convolutional Neural Networks

Over the last few years, Convolutional Neural Nets undergone a series of game changing innovations which resulted them being one of the most popular methods of image classification in the field of Computer Vision [48]. Just like how kids learn how to recognise objects, the algorithm for a CNN (short for convolutional neural net) need to be fed with enormous visual datasets for it to make predictions on pictures it has never seen before [36]. Computers as opposed to human beings have a very different observation approach to images, which only consists of numbers. According to freecodecamp.org, every image is treated as a two dimensional array of numbers by a computer and a CNN is very effective when the task is to teach an algorithm how

to recognise objects in images [36]. deep and complex CNNs have been used over the years to advance the state of the art image classification research [37].

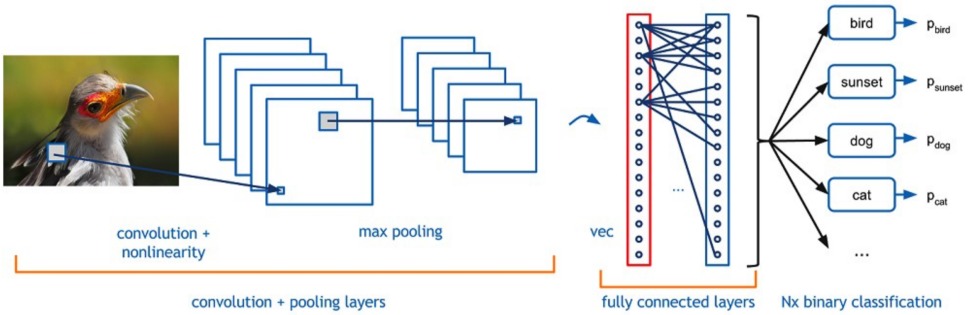


Figure 2.13: Typical Convolutional Neural Network - Source: Deshpande, Github

Just like neural networks, CNNs consists of neurons with learnable weights and biases, where each neuron responds with an output after receiving multiple inputs, taking a weighted sum and passing it through an activation function. The algorithm has a multi layer architecture which takes in and outputs feature maps at both stages [38]. One of the biggest differentiating feature of CNNs from neural networks is that they operate by volumes as the input is a multi-channeled image instead of a vector in neural nets [39].

The paper will dive into convolutional neural nets later in the research illustrating the concept more deeply.

- Brief Summary of Deep Learning
  - In short, deep Learning can be perceived as a Machine Learning framework injected with steroids. Each layer in the ANN (Artificial Neural Network) extracts different, yet important information from the visual datasets. In the context of the project, one layer of the net could be detecting the edges of a burger or sandwich and another layer could be detecting the contents of the food. According to the co-founder and leader of Google Brain, Andrew Ng, deep learning benefits will unlock many hidden potentials within the industry.
  - "one reason why deep learning has taken off like crazy is because it is fantastic at supervised learning" - Ng, 2014 [39].
  - Even with unstructured, interconnected and diverse datasets, deep learning enables computer to solve sophisticated queries. The algorithms under the hood boost up performance when they learn more. It would be quite interesting to observe in the

future how deep learning solves more complex problems as the technology starts to mature [40].

2.7 Transfer Learning

One of the most powerful concepts of Deep learning is Transfer Learning. In the previous section it is established that deep nets are the most viable option for accurately solving classification problems. In lots of real world applications, deep learning solutions are often challenged by limited data set problems. Training a deep CNN with a small dataset from scratch leads to Overfitting problem and the quality of model is very low [41]. Overfitting usually results when the model performs really well on the training data but opposite on the validation set.

Transfer Learning is a technique through which the knowledge obtained in a source task is transferred to a target task [41]. Through Transfer Learning, it is possible to apply the knowledge, a neural net gained from a separate task. So for example, a neural neural which has been previously trained to recognise objects like cats, can be trained to read x-ray scans. The pre-trained image recognition neural net is trained on (X, Y) pairs, where X is an image and Y is an object. An image of any object like cats or dogs or anything else. The next step is to delete the last output layer and the weights feeding into it, of the net and create a set of randomly initialized weights just for the last layer. The next step is to train the pre-trained model on radiology images with small data sets and eventually build a system that can generate image recognition predictions [42].

According to one of main pioneers in AI, Andrew Ng, Transfer is going to change the way developers interact with machine learning models as not only the training time is much faster but one fraction of the data is required for the specialized classification.

"Transfer Learning will the next driver of ML success" - Andrew Ng, NIPS 2016 tutorial

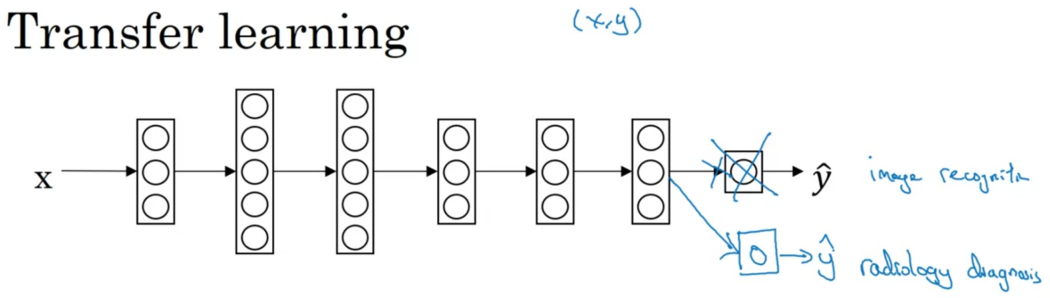


Figure 2.14: Transfer Learning - Source: Coursera.org

The research project will employ the concept of Transfer Learning into core areas of generating the deep learning model. It will build its customized model on top of VisionFeaturePrint.Scene that backs up Apple Vision Framework [43]. The deep CNN has been pretrained with a huge amount of data set containing a lot of classes, which will later be trained on food images like burgers with different labelled classifications.

The powerful concept of borrowing prior learning from a deep net, Transfer Learning is independent of any popular framework. It is currently advertised in all of the leading cloud service providers like Google Tensor Flow, Amazon AWS Machine learning, Azure Machine Learning, IBM Watson Studio etc and also in mobile devices through TensorFlow Lite (Android) and CoreML (iOS). Desktop software MATLAB can also be a good tool to use train models through transfer learning. Now that the popularity of the concept is established, the research paper will take a mobile development approach to the concept by building towards an iOS application.

Chapter 3

Design and Specification

3.1 Project Specification

The purpose of this section is to demystify the relevant functional and non functional requirements of the project. It will provide a clear outlining of the type of features to be implemented later in the project through continuous development. To achieve this, first a deep learning algorithm must be chosen for the purpose which will provide the project with the best results. The best architecture for food and calorie detection will be examined after that so that any errors can be detected before starting implementation.

3.1.1 Functional Requirements

Functional requirements refer to the function or list of functions that are mandatory for the system to perform to accommodate the needs of the stakeholders. In other words, it describes the set of functions that the system do not how it performs those functions [44]. The functional requirements for this deep custom transfer learning model are:

1. The model should be a state of the art Deep custom Convolutional Neural Network. The chosen pre trained algorithm for Transfer Learning must be able to assist generating good validation accuracy and a small loss value suggesting minor Back-Propagation for updating weights.

2. The model must be able to train on a small data set as there are not much franchise based food classification data available online.

3. The model should be able to apply Augmentation to the training data sets which will yield better results.

4. The training process has to stop automatically once the validation accuracy starts to decrease.

5. Any iOS application built from the deep learning model must be able to generate predictions without the need of any external network connections.

3.1.2 Non Functional Requirements

Any requirement which does not classify as functional requirement is non functional [45]. In other words presence of non functional requirements is essential for the functional requirements to perform accurately. It can be divided into two categories:

- Execution Qualities - Examples include security and usability, observable at run time [45].
- Evolution Qualities - These are qualities related to the static architecture of the algorithm such as maintainability, test-ability, ability to scale etc [45].

The non functional requirement for the project is:

- During training, the software must be able to demonstrate its progress at each stage to keep the user updated on the training and validation accuracy and loss.

3.2 Requirement of Accuracy

Since the research boasts on the idea of the Transfer Learning technique, the custom CNN model built from the project must be able to demonstrate robust validation accuracy. Without the validation accuracy going over 90%, it cannot serve as a replacement to the existing systems. The target of the research is to create a model which can learn from a small data set with the help of some rich deep learning algorithms. Splitting of total data sets into training and validation sections is a common practice in machine learning during training process. The training set contains the labelled samples from which the algorithm can extract high level features and pass them onto the classifier. In other words, the model learns from the training images and use these learning for predictions when presented with validation samples. The validation set also contains labelled samples but they are only introduced to the model during the last stage of each iteration so that the model knows how much to back-propagate after calculating the loss value. Back-propagating will update the weight values and will tested in next iteration on the effectiveness of the values. If validation accuracy improves then the weight values are kept unchanged until a better accuracy value is generated.

3.3 Implementation Design

The first implementation of Transfer Learning approach will be a deep Convolutional Neural Network (CNN) will be implemented in this project. Over the years deep Convolutional Neural Networks have outperformed many machine learning models in the field of computer vision. It is currently one of the most efficient algorithm excelling in Computer Vision, achieving state of the art results in various benchmarks. Deep CNNs even outperform human level consciousness and response times in many domains these days. A suitable pre-trained deep CNN algorithm will be selected in Chapter 4.2 and the a custom classifier will be built on top of it which will be able to classify eight classes of fast food.

The second approach to Transfer Learning will be using Apple's Machine Learning framework Create ML is to build a deep learning model which can then be later used inside Core ML 2 to make predictions. Later down the report, in Sector 3.5 both the frameworks will be described.

The research aims to closely observe and compare the outcomes of both the approaches in terms of training time and performance. The below two charts provides the flow of the project in both approaches. All the components of both the charts will be discussed in details throughout the report.

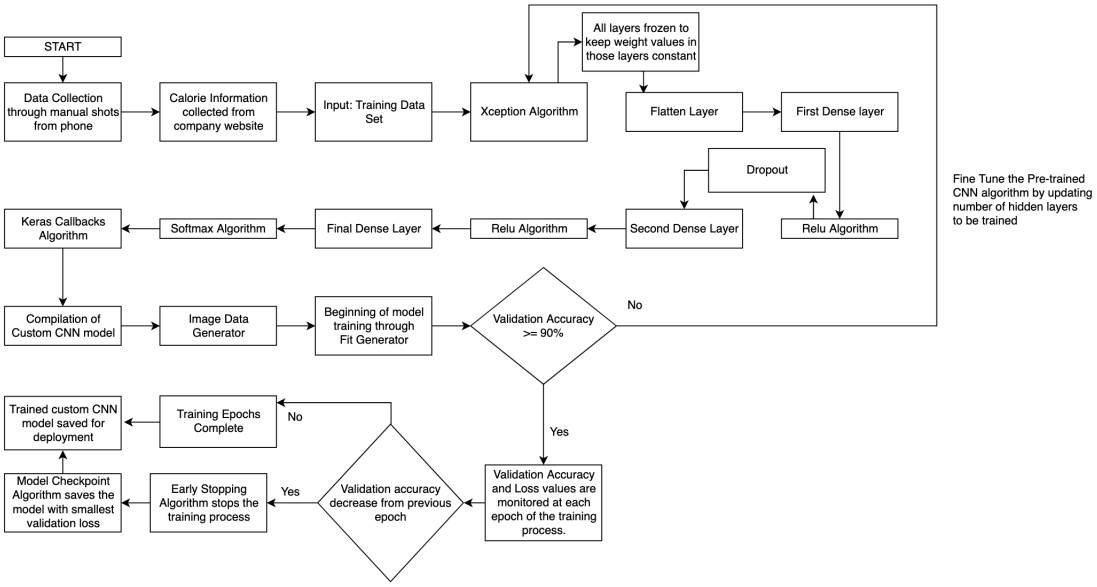


Figure 3.1: Flow of the project for the first Transfer Learning approach using Xception

MLFlow1.png MLFlow1.png Chapters/Images/3.Design/Create MLFlow1.png

Figure 3.2: Flow of the project for the second Transfer Learning approach using XceptionFeaturePrint.Scene

3.4 Convolutional Neural Networks

In order to understand the contribution of the research paper for a transfer learning approach, it is very important for the reader to understand an effective algorithm, Convolutional Neural Network in fair amount of depth. The paper will go deeper into describing CNNs and then build up on that towards Transfer Learning. This involves taking a popular pre-trained model and then replacing the head(neuron) of the model with a fully connected CNN which will be the act as the image classifier. This would save the algorithm from being too computationally expensive and unstable.

During the 90's, CNNs were the benchmarks in recognizing hand written digits and face detection [46]. In recent years the algorithm has demonstrated good accuracy in handling complex image classification tasks. The reasons for this exceptional growth of popularity of the algorithm are availability of large training data sets and large computational power to facilitate the learning process [46].

3.4.1 Abstract

Please take 5-10 seconds looking at the image below.

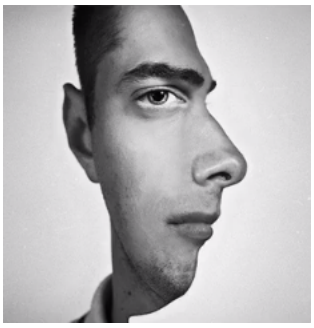


Figure 3.3: Source: https://www.udemy.com/machinelearning/learn/v4/t/lecture/6761138?start=750

Do you think its half the face of a man of a photo taken from sideways of a face (man or woman). Upon looking at the right corner of the image, its feels likely to be a face from

sideways and inspecting the left corner makes it equally likely to be half image of a frame of a man viewed from above. What our brain looks for is features from the image and comes to a conclusion depending on those features. Our brain performs these processes due to the presence of a convolutional structure of neurons.

3.4.2 Architecture of a typical CNN

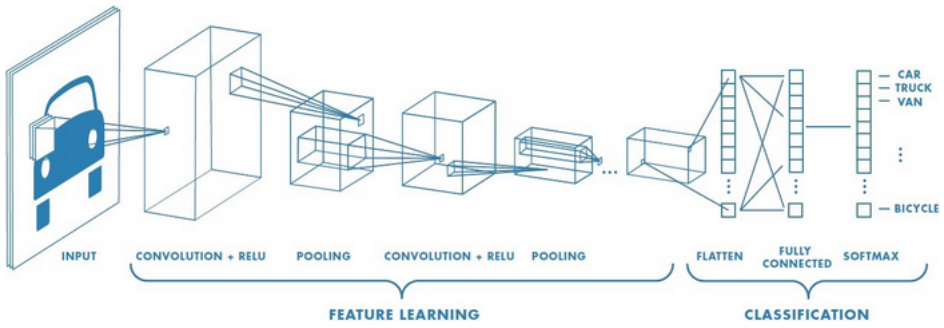


Figure 3.4: CNN architecture from high level - Source: Prabhu, Medium

A typical CNN consists of a convolution, pooling and a fully connected layer. To increase accuracy the number of convolution with filters and pooling layers are increased to extract and store more features from the input image. After connecting with the fully connected layer, a function named 'Softmax'(a wonderful activation function that turns numbers aka logits into probabilities that sum to one [47]) is applied which makes a classification on the type of object in the image with a probability of between 0 and 1 [48].

3.4.3 How do Convolutional Neural Network work

- Step 1.1: Convolution Operation

  The process acts as the image feature extraction part of the classification [49]. From a mathematical perspective, convolution is a derived function which shows how the shape of one function is modified by another function through integration [50].

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

Figure 3.5: The math behind Convolution Operation - Source: SuperDataSceince

Coming back to the neural network perspective, it is an operation which takes an input image and uses multiple feature extractors to extract relevant features from the image. After extraction, the features are stored in feature maps which is passed on to the next layer.

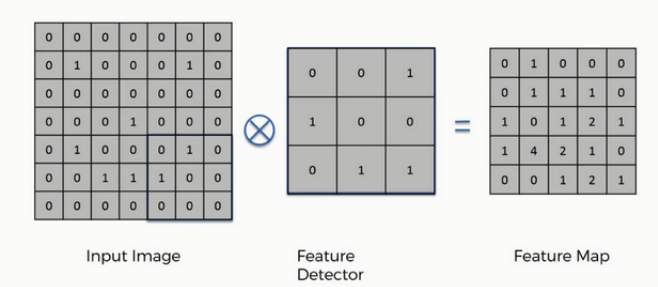


Figure 3.6: Convolution Operation - Source: SuperDataScience

The main idea behind the operation is to reduce the size of the image therefore reducing computational complexity for the process and also preserve the important features of the image. It is not wrong to say that the operation does lose information from the input as the feature maps generated possess fewer cells. But as mentioned earlier, the purpose of convolution is to extract the most desirable features and exclude the rest [50].

- Step 1.1.1: Padding

  In the event of the feature detector not fitting the input image perfectly:

  - The image is padded with zeros so that it fits the filtering, or
  - The part of the image is dropped where the feature detector failed to fit and the process is known as valid padding as the purpose of this is to preserve only the relevant part of the image.

Activation Algorithms in Deep Learning:

Each neuron in a CNN has an activation algorithm which fires up when needed to introduce non linearity in data [51]. The weights and biases of the neuron is used to decide whether it can be fired or not [52]. Various activation algorithms are used in controlling the output of the neurons in a network in domains such as Image classification, Natural Language Processing etc. Usually for a linear model, the output is also linear in nature and hence an activation algorithm saves the model from decreasing the linearity [52]. In the past, the algorithms 'Sigmoid' and 'Tan-h' have been very popular in being used as

the activation function of the neurons in multi-layered Perceptrons. But both of them run into a common problem in Deep Learning which is the Vanising Gradient where the updates by the stochastic gradient descent becomes very small. The reason for that is because the algorithms are contractive almost everywhere and the large gradient values becomes almost zero [53]. To avoid running into the problem Nair et al introduced the Rectifier Linear Unit(ReLU) algorithm in 2010 [54]

- Step 1.2: ReLU Activation

  The Rectifier Linear Unit or ReLU, is not considered as a separate different step from convolution. The process acts as a required step to increase the amount of non-linearity in the input image because images are usually non-linear when considered features like colours, borders, etc).

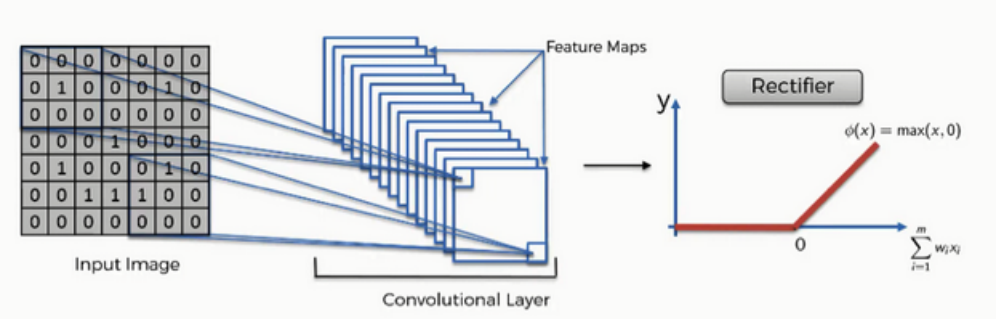


Figure 3.7: ReLU Operation - Source: SuperDataScience

The rectifier algorithm directly outputs the input value if positive or otherwise outputs zero. It overcomes the vanishing gradient problem and is currently the default activation function when buidling CNNs. As a result the network is easier to train and often achieve better performance [55].

- Step 2: Pooling The operation aims to decrease the parameters of the input image when it is considered too large to pass through the fully connected CNN layer. It uses spatial pooling or down sampling of the feature maps by reducing the dimensions and retaining the important features. There are three types of spatial pooling:
  - Max Pooling
  - Average Pooling Pooling

  The most commonly used one is the max pooling process which only extracts the largest element from the rectified feature map [48].

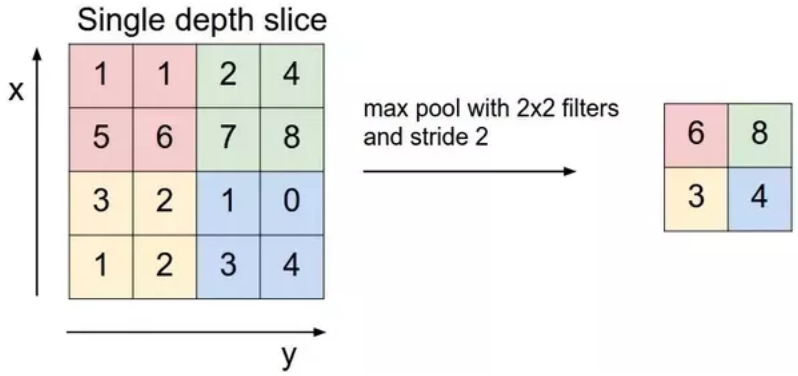


Figure 3.8: Max Pooling - Source: Prabhu, Medium

When used, max pooling also acts as a Noise Suppressant. The operation performs denoising in parallel with dimension reduction and excludes the noisy activations altogether. Since average pooling only performs dimension reduction, max pooling is much more efficient in certain times [56].

- Step 3: Flattening

  The completion of pooling operation is a signal of the model successfully understanding the features of the input image. The final output from the former layer is then flattened and fed into a regular neural network for classification.

- Step 4: Full Connection

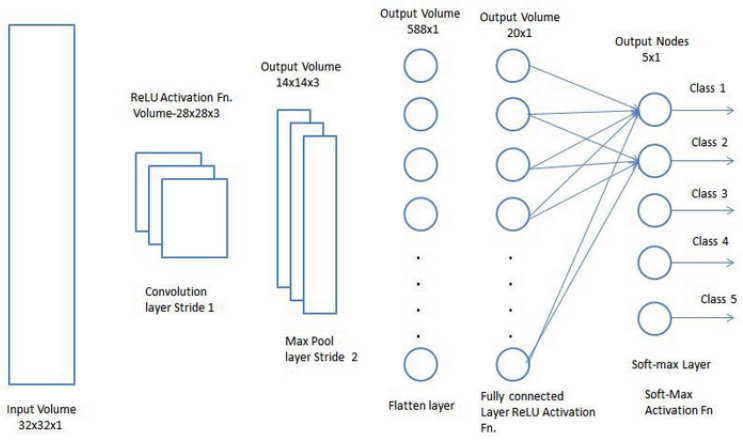


Figure 3.9: Implementation of fully connected layer - Source: TowardsDataScience, Medium

In order to understand the next part of the network its important to first understand some commonly used outstanding algorithms Softmax, Categorical Crossentropy and Adam.

The knowledge of these algorithms will help to understand the implementation of the project better as well.

3.4.4 Softmax

The algorithm is one of the core element in deep learning classification by turning an arrays of logits into array containing probabilities, which sum to 1 [47]. It is used at the last dense layer of the CNN, where the output values from the previous layer is fed directly into the last layer. The reason for the approach is due to the last layer of the network being the classifier which will be displaying the user the end predicted value. The algorithm cannot be applied to all the output scores 'S' independently as it depends on all elements of 'S' [57]. The Softmax function for a given class Si can be computed as:

$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}}$$

Figure 3.10: Softmax Algorithm mathematical notation - Source: Gombru, Github

For each class in 'C', the scores generated by the network are sj.

3.4.5 Categorical_Crossentropy

When compiling a CNN model the loss algorithm (one of the three parameters used for compilation), Categorical_Crossentropy is used if the classification class is more than two in value. Since the project aims to train eight different classes of food, the algorithm fits well into the use case.

3.4.6 Adam

Its an adaptive algorithm specifically tailored for deep learning networks training by computing individual learning rates for different parameters [58]. Adam requires low memory to run and works well despite little tuning [59]. It is very straight forward to implement and and is computationally very efficient [60]. Over the years, the algorithm has been seen to outperform its rival algorithms like RMSProp, SGD, Stochastic Gradient Descent and more.

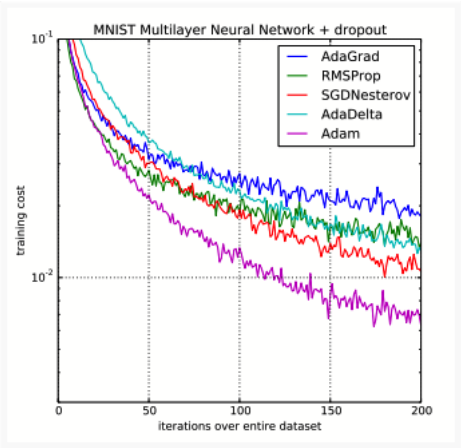


Figure 3.11: Performance of Adam compared to other optimizing algorithms - Source: https://arxiv.org/pdf/1412.6980.pdf

The last step is to add a fully connected layer which is a less computationally expensive way of learning non- linear combinations of the high level features extracted from the convolution layer. The flattened output is fed into the feed-forward neural network and in every instance of the training process, Backpropagation (a supervised learning algorithm which uses gradient descent method) technique is applied. After a reasonable number of epochs, the model uses 'Softmax Classification' algorithm to identify and classify the dominating objects in the image [56].

To summarize in a brief way, a typical CNN starts with the input image where the image is first passed into one or series of convolution layers and relevant features are extracted generating feature maps. After that ReLU algorithm is applied to all the feature maps increasing the non linearity in the data as all negative values are converted to zero. The output of this layer is passed to the Max Pooling Layer where only the dominant features of the maps are collected and passed to one or more dense layers, with the last one implementing the Softmax algorithm. The training process is optimized by the Adam algorithm. The last layer acts as the classifier for the model and generates the predictions. Softmax takes all the predictions and converts the numbers into probabilities adding up to 1.

3.5 Apple Machine Learning Framework under the hood

3.5.1 Core ML 2

In 2017, Apple first introduced its very first machine learning framework called Core ML which was a huge step towards integrating AI features into native iOS mobile applications for Apple developers. Apple was using machine learning technologies before that in Sir, predictive keyboard etc, but Core ML provided the opportunity to build custom trained models in to the hands of developers as well as the consumers they served. The framework utilizes the underlying low level technologies in Apple platforms to provide efficient machine learning tools to implement in native apps. It takes full benefit of the CPU and GPU capabilities and use Metal and Accelerate to provide fast speed for operations, which in turn saves the need to access the Web for each API request [61].

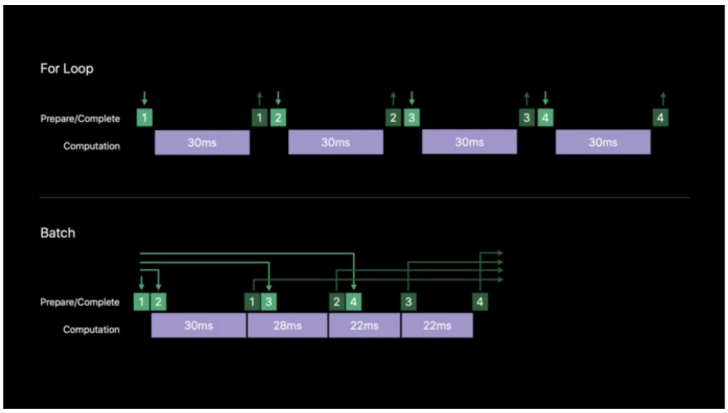


Figure 3.12: Batch Prediction in Core ML 2 - Source: Apple Documentation

With Core ML 2 Apple has introduced a new process called Batch Prediction which allows the user to run his/her personalised ML model on a set of data and receive a set of outputs. The biggest advantage over the first iteration (Core ML) is that it allows the developers to generate multiple predictions on a set of data without the need of continuous for-loops as the process used to slow down the overall performance. [**?**]

"Predicts output features values from the given a batch of input feature values"- Apple Documentation

3.5.2 Introducing Quantization

Another hot feature of Core ML 2 is Quantization, which allows developers to significantly reduce the size of their machine learning models by sometimes 4x times. The efficient feature

targets the previous problem of having big size ML models which in turn makes the overall mobile application too big to upload and download from the App Store. According to popular online blog website Medium, Quantization works by reducing the number of bits used to identify bits in ML models [62].

Models used 32 bit floats to describe weights in iOS 11 which was later improved to 16 bits with the introduction of half- precision floats for the same accuracy. Now with iOS 12, the models usage went down to 1 bit as the weights can be encoded using any number of bits. As a result, the user can have a discreet subset instead of repetitive representation of values weights would have with floats [62].

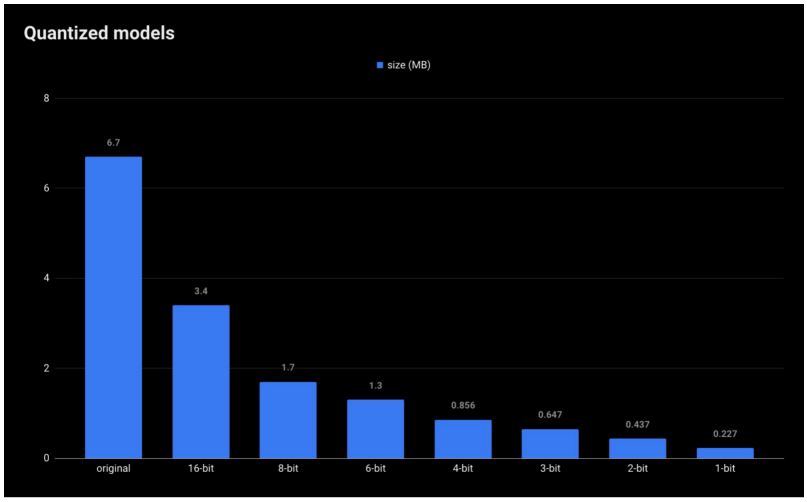


Figure 3.13: Quantized Models in iOS 12 - Source: Apple Documentation

3.5.3 Performance

Batch Prediction process allows Core ML 2 to be 30 percent faster from its predecessor Core ML, with much less lines of code which reduces the pressure on the developers as the system is taking care of it under the hood. Previously the code snippet for looping through all data would look something like this:

```
// Loop over inputs
for i in 0..< modelInputs.count {
    modelOutputs[i] = model.prediction(from: modelInputs[i], options: options)
}
```

Figure 3.14: Source: Appcoda

In the above piece of code, for each input the user is requesting the model to generate a prediction and yield an output based on some options which however, took a long time going through every input.

With the BatchAPI Apple has successfully introduced a more efficient way where all the inputs are provided to the model and an accurate prediction is its result, which in turn takes a significant shorter time [19].

```
modelOutputs = model.prediction(from: modelInputs, options: options)
```

Figure 3.15: Source: Appcoda

3.5.4 Create ML

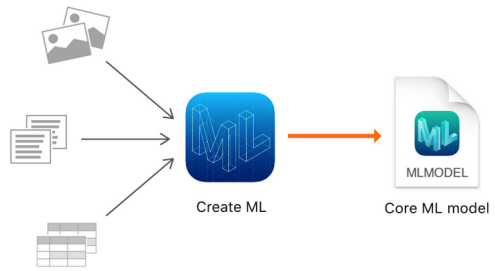


Figure 3.16: Apple Create ML Framework - Source: Apple documentation

It has been known for a long time that training deep learning algorithm is a tedious and time consuming task. With the introduction of Create ML framework in WWDC 2018, it is now much easier to build deep learning models on the go for Apple devices. The best part being that it does not require the user to be an expert in Machine Learning to build his/her own deep learning image based classifier.

According to learnopencv.com, some key advantages of using Create ML in visual recognition are:

1. Utilization is easier: It is much easier for developers without Machine Learning knowledge to build deep learning models [63]. If the user got access to a big dataset of either text or images, then more than half the work is already done and with just a few lines of execution code, the system will be able to train and start making inferences. In comparison to popular tools like Keras, Tensorflow and Caffe, which usually requires extensive amount of code and lacks a friendly visual interface, this is much simpler, yet powerful approach [64].

2. Speed: The training and inference duration are significantly reduced due to the presence of hardware acceleration. The user can use couple of hundreds of pictures to create a sophisticated accurate training dataset, which usually takes about couple of seconds without Augmentation. [63]

3. Size: Traditionally training a deep learning model on a GPU, requires the utilization of systems like Mobilenet or bigger and make the size smaller by pruning and quantization. The models can range from couple of megabytes to hundred, which is not that reliable when deploying into a portable device. The solution offered by Create ML was converting the model into just a few kilobytes, immensely reducing the space taken in the device. [63]

4. Train once, use on all Apple gadgets: The model prepared utilizing Create ML can be coordinated into iOS, macOS, tvOS and watchOS utilizing CoreML. According to Appcoda, Create ML leverages the machine learning architecture built into the apple framework. When the user downloads iOS 12 or macOS Mojave, the system automatically downloads the machine learning infrastructure along with it. That way, when the user creates his/her own ML model, it can consume up less room since a large portion of the information is now on the client's gadget [64].

With the dispatch of Create ML, Apple has made machine learning a local innovation inside Apple biological system. There won't be an easier method to make and prepare machine learning model other than Create ML. With Create ML, we can utilize center Apple advancements like Swift and Xcode to make completely prepared machine learning models that can be provided to the apps. AI/ML wave is all over and this is an enormous declaration that opens ways to many machine learning devices in Apple biological system [65].

Transfer Learning - You may have auricularly discerned that a Neural Network is information hungry - it takes double digit(more than 10) thousand samples, if not a sizeably voluminous number of information focuses to prepare one. Why Create ML requires less than half of that? The short answer is Transfer Learning.

In Transfer Learning, we utilize the engineering and loads of a pre-trained algorithm which is mundanely prepared on an expansive dataset for an alternate assignment. With this algorithm as the base, in Transfer Learning fine tuning is performed on the last yield layers to suit the job that requires to be done. It is not required to retrain the whole system for custom classification. This incomprehensibly expedites the preparation procedure and furthermore requires an a lot littler dataset. For instance, we could utilize a pre-trained algorithm, ResNet that has been prepared on an astronomical dataset like ImageNet for 1000 relegations utilizing a dataset of

in excess of a million pictures.

According to Learn OpenCV, Apple utilizes this powerful concept of Transfer Learning to enable clients build powerful image classification models [63]. The site also claims that the accuracy achieved by Create ML is better than human level.

Apple utilizes the expression 'highlight extractor' rather than the more typical 'pre-prepared model'. This is on the grounds that the rudimentary layers of the pre-prepared model proselyte a picture into an element vector that is subsidiary for arrangement assignments. Apple CoreML system gives the clients a chance to do on- contrivance derivation utilizing the models made utilizing Create ML. This holds client security and does not require any web sodality as required by the applications utilizing electronic induction [63]. The underlying Vision framework beneath Create ML is optimized highly to perform best on Apple hardware by using GPU acceleration.

With the addition of both Core ML 2 and Create ML, Apple has significantly reduced the time for training datasets of 20,000, without Augmentation, from 24 hours to a whooping 48 minutes in MacBook Pro and only 18 minutes in iMac Pro. The size of the model is also reduced from 90mb to only 3 mb [66].

3.6 Other Competitive Machine Learning Frameworks

3.6.1 Google Machine Learning Kit, ML Kit

Introduced in May 2018 at Google I/O, it bridges the gap between developers and the deep learning universe. Google has initiated the challenge of deploying deep learning frameworks into mobile applications and comes with some pre trained deep learning APIs which are: Image labelling, Text Recognition, Face Detection, Barcode Scanning and Landmark Recognition.

"Whether you're new or experienced in machine learning, you can implement the functionality you need in just a few lines of code. There's no need to have deep knowledge of neural networks or model optimisation to get started" - said Google [67]

The platform is optimised for supporting both iOS and Android applications as it aims for the mass market developer pool. It also provides the option to send request to the APIs from either within the device locally or from the cloud (which is more accurate but at the same time much larger in size). Example, The on device ML model provides 400 labels for object detection where the cloud one features about 10000 [68].

Figure 3.17: Comparison between Google ML Kit Image recognition service on the device and the cloud - Source: Goodaudience.com

3.6.2 Tensor Flow Lite

If developers are looking for more customizable options for their personalized machine learning models for small devices, Google aims to solve the problem using TensorFlow Lite which is the successor of the popular TensorFlow cloud platform. Once the TensorFlow Lite model are uploaded into Firebase console, it acts as an additional API Layer to the custom model [68].

As machine learning chips are becoming more available in mobile applications, TensorFlow Lite aims to provide lower latency inference performance. According to Google, it will recognised as the industry standard platform in the future for deploying models on mobile and embedded devices. The platform is designed to improve machine learning model loading times and support hardware acceleration due to it being lightweight, fast and optimised for mobile devices. Like Core ML 2, TensorFlow Lite also have some out of the box pre trained models for quick use, which can later be tweaked and retrained to accompany specific needs. [69]

3.6.3 Why use Create ML instead of Google ML Kit/Tensor Flow Lite/Keras/TensorFlow

The research project is aimed at individuals who wants to have machine learning capabilities on the go without requiring extensive knowledge in Artificial Intelligence especially Neural Networks. For a developer who has minimum knowledge on Convolutional Neural Nets, TensorFlow can be a challenging interface to start with as the platform is aimed towards data scientists [70]. According to popular website Quora, TensorFlow is a very low level platform with a steep learning curve. Due to the odd structure of the platform, it can be hard to debug

errors sometimes [71]. In comparison, Apple's Create ML provides a much more friendly interface with easy access for anyone who wants to try out and explore the breadth of ML models already provided in their platforms or build custom ones.

One of the biggest advantage of Apple Create ML is that the whole model training process is done on the device (Macbook), without requiring any contact to any external server. Upon deployment to an iOS mobile application, the model is immediately ready for making predictions. This in fact solidifies user data protection as all the other platforms require making API calls to their cloud servers either to train the model or get the predictions straight from the cloud.

"User privacy is fully respected" - Francesco Rossi from Apple told developers in a recent conference session . "By running on machine learning models on device, we guarantee that the data never leaves the device of the users." - Source CNBC [84]. As a result, the device does not require internet connection to serve the needs of its customers in making accurate predictions. As a result, the user does not need to purchase a subscription from an external server like Microsoft, AWS, Google and other companies. Its totally free, thanks to the latest framework 'Create ML' [72].

In addition to being secure, the second biggest advantage of Create ML compared to state of the art deep learning frameworks is the model size when ready to be deployed. Every deep framework which has been proven to be effective weights more than 100 mb (some more than 1 gb)even for a simple classification model. In comparison to that, a VisionFeaturePrint.Scene model will most of the time weight less than a megabyte. As a result the inference process is significantly faster and optimized as smaller size depicts lighter models.

Since the idea of the project was to bring the power of Transfer Learning in iOS devices, using the official native framework, Xcode provided by Apple is a great advantage. The framework has been built and tested to perform very efficiently and optimize the training.

3.7 Softwares used for the project

Although established in the previous section, the justified choice of using Create ML, the project aims to implement two separate approaches to achieve more concrete evidence which reinforces the hypothesis. The detailed description of both the approaches are provided in Chapter 5. The section evaluates the different types of frameworks used for the implementation below.

3.7.1 Swift

Swift 5 is used to build both the front end and backend design of the project for the second implementation of Transfer Learning on VisionFeaturePrint.Scene mentioned in section 3.3. Introduced in 2014 at WWDC, Swift is one of the fastest growing popular language in the Apple Machine learning community with features being modern, fast and interactive, officially stated by Apple. In July 2018 Swift raised to the eleventh position in Tiobe Index top 50 programming dialects. In the recently referenced PYPL rating, Swift is in the ninth place. To the extent Swift was engendered by such tech goliath as Apple, it is rudimentally used to make applications for iOS, programming for MacOS, watchOS and even tvOS. Among iOS applications indited in Swift, there are Hubspot application, Lyft, Pandora, OpenTable, Airbnb, LinkedIn, Yahoo Weather and Khan Academy. At WWDC 2017 Apple's group presented Swift 4. A year ago the organization presented Swift 4.2 which incorporate a few upgrades [73]. According to theswiftdev.com, Apple likewise reported Swift 5 to be accessible in early 2019, which set its primary goal to achieve ABI (Application Binary Interface) stability [14]. StackOverFlow survey for 2017 showed that Swift was the among the top five most popular programming languages for the third year in a row [74]. According to hackr-blog, in terms of job opportunities, it is also equally popular as one of the most prominent ones [75].

The main advantages of Swift include:

1. Easy and Safe - Swift posses a simple syntax and it is very easy to read and code. Being a statically typed language has the added benefit of discovering bugs and fixing them before compilation.

2. It is very efficient in terms of handling memory management on its own with the help of sophisticated tool called ARC (Automatic Reference Counting).

3. The language is open source from 2015 and being elected as the second most loved language in Github in 2016. Since then, Apple pushed the language to be the company's brain-child cross platform. Currently its also available in Linux operating system and soon to be available in Windows as well, as stated by Apple. Tech mammoths like IBM contributed to Swift by building the popular tool sandbox which made Swift as a fully functioning cloud player. Thus the server side of Swift can interact with a huge number of backend tools. Compared to the most popular language in the world for 2019, Python, Apple claims Swift to be 8.4x faster [73]. According to stackshare, Swift has less disadvantages of either using proficiency or easy of accessing than Python and enterprise language Java [76].

3.7.2 Paper Onboarding

The tutorial of the mobile application will be build using an online framework called Paper Onboarding. This is an additional tool for enhancing the user experience factor and not one of the core requirements of the project. The framework details will be referenced in the code section. The tool will be added to the project by using Cocoa Pods in iOS.

3.7.3 Python

The custom Deep CNN algorithm will be developed using Python 3.7. Application of Transfer Learning technique will be also done in the same language. According to recent article in Sayone, by the end of 2019 Python will be the most widely accepted programming language due to the business advantages the framework provides [77]. The figure below depicts the growth Python over the past few years and its advancements.

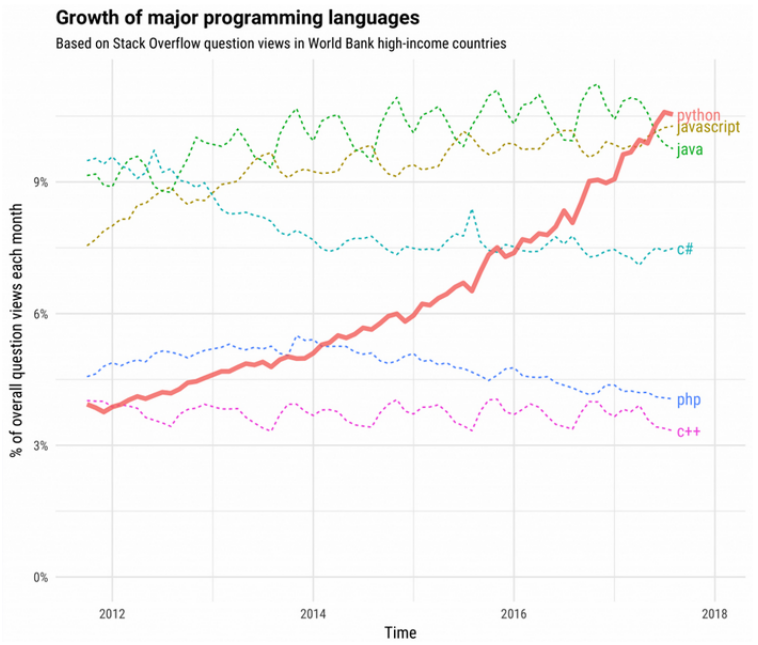


Figure 3.18: Number of questions asked on different programming languages in StackOverFlow - Source: Sayonetech

Python is currently the hottest language on the market to build deep learning softwares. It is also the first choice for data scientists for building machine learning models. The language provides extensive libraries which facilitates the objective of the project such Numpy, Scikit Learn, Glob, Matplotlib, PyTorch, PyBrain etc. It is one of the most simplest and easy to pick up languages in the programming ecosystem. The community of Python is also very vibrant

and always ready to help any new comer struggling with any implementation.

3.7.4 TensorFlow

TensorFlow is a popular free open source library introduced by the researchers at Google used for deep learning applications. TensorFlow is better than its rival deep learning frameworks like Theano and Torch. According to StackOverFlow, Theano and Torch provides deep learning research and development frameworks, but in addition to those, TensorFlow provides frameworks for deployment as well [78].

TensorFlow provides great support for portable training as previously training deep learning algorithms were very expensive and only possible with the help of a GPU. But thanks to tensorflow, now it is possible to implement a deep learning and train according to the desired specification of the user [79]. The project will use TensorFlow in the backend along with a deep learning framework called Keras, which is described in details in the next sub-section.

3.7.5 Keras

Although TensorFlow provides a collection of advanced deep learning algorithms, it is not one of the most user friendly framework to work with. In order to accommodate the limitation, the project was implemented the high level library Keras. According to visualstudiomagazine.com, Keras is a code library with a user friendly Python language interface [80]. Writing code for deep learning algorithms can be difficult not to mention time consuming when dealing with low level parameters. With the help of Keras, it is much easier to implement a very powerful deep learning framework by letting Keras take care of the low level stuffs under the hood. The main limitation of the framework is being available only for Python language [81]. Since the limitation does not effect the implementation of the project anyway, the Keras API was used mainly to code the core deep learning model. Like TensorFlow Keras is also free and has got all popular deep learning pre-trained models such as VGG16, ResNet50, Xception, Inception etc. Initializing a pre-trained model in Keras is just one line of code with appropriate arguments passed to the algorithm. The loss function in the custom deep CNN algorithm for the project will be implemented using the in-built loss algorithm in Keras. It saves the project from writing a loss algorithm from scratch and take advantage of the supplied one. Keras also provides an efficient way of data handling by implementing training and validation set queues automatically with every iteration [81].

In addition to the ability of training customized models, Keras provides two very special

algorithms called 'EarlyStopping' and 'ModelCheckpoint'. 'EarlyStopping' keeps tracks of the validation accuracy and stops the training when the accuracy starts to decrease. In the event of validation accuracy staying the same but validation loss value increasing, the algorithm stops the training process. 'ModelCheckpoint' keeps track of the best validation loss value. The project will describe the algorithm more descriptively during Section 5 'Implementation'.

3.8 Hardware

The device used for building the platform is Apple MacBook Pro 2017 model. The RAM is 16GB 2133MHz LPDDR3 specification. The processor is 2.5GHz dual- core 7th generation Intel Core i7 processor with Turbo Boost up to 4.0Ghz. The software of the machine is MacOS High Sierra version 12. For the mobile implementation, the device used is Apple iPhone 8 plus which powers a 3 GB RAM and a 12MP dual lens camera.

3.9 Data Sets and Input Images

The datasets for the research project will be gathered organically by manually clicking photos of the desired fast food. The paper mostly focusses on tried and tested products which would give a more concrete view of the calorie in a food as the recipe is continued for decades. The model will be provided a minimum of 650 images per category of the classification. So altogether 5200 images are used in the research with an 80:20 splitting between train and test data.

3.10 Methodologies used in Software Development Process

3.10.1 Waterfall

First introduced by Dr. Winston W.Royce back in 1970, waterfall model is one of the oldest software development process [82]. The model is considered to be very rigid and follows a logical progression of steps in the software development life cycle. Implementing waterfall approach in software building is a fairly straight forward process, although there are sometimes minor differences in the way different developers use it.

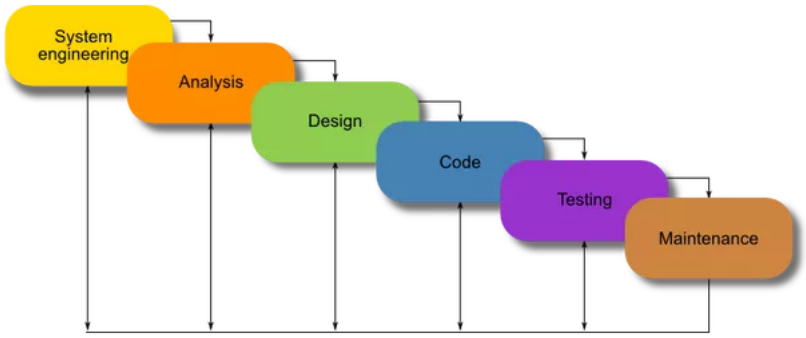


Figure 3.19: Steps in a Waterfall approach for Software Development - Source: Airbrake.io

The six main stages of the waterfall process are:

- Requirements This is the initial stage where all the requirements are established which serves as the basis for the all future development of the software building process. The result of the process is a document containing the requirements approved by all the stakeholders of the software. The document states what the software should do rather than how it should do that action [82].

- Analysis Product models and business logics for guiding production, are generated in this stage from an in depth analysis of the system specifications. The team also discusses the feasibility of the specifications in terms of financial and technical costs.

- Design A formal design is proposed in this stages from the output of the first two stages. The resulting document outlines the technical investments such as hardware, data sources, architecture, services, programming languages and external libraries [83]

- Implementation This is the stage where the source code is programmed for the software using the design specifications from the previous stage. Usually the system is designed in smaller components before being assembled together at the end.

- Testing A very important stage in the waterfall methodology is the testing stage when the software built needs to be tested with various benchmarks to determine performance and accuracy. Several tests such quality assurance(QA), unit testing, integration testing and beta testing take place to underline any issues encountered. If everything works fine until here, the waterfall process continues to move forward.

- Operations/Development The software built is fully functional at this stage and is deployed either in the Web or mobile platform depending on the specification agreed previ-

ously. The product is released to the stakeholders for real time use.

- Maintenance The product is monitored for its performance statistics and dealt with, if any issues arise on the client side. Over time, more versions of the product are released fixing bugs which may cause disruption between the client and the server.

3.10.2 Advantages and Disadvantages of the Waterfall Process

Although the waterfall process is very simple and easy to understand, it has shown to be less effective for projects where requirements carry a high risk of changing. It always uses a rigid style through out development which is hard to alter if anything goes wrong later in the process. Example - If an issue is discovered during the testing stage, it would very difficult to debug and expensive as the process assumed everything staying the same at the upper stages.

3.10.3 Agile Methodology

Founded by a small group of four people, an agile is an empowering process letting companies design and build the right product [84]. Instead of focusing on tools and processes, the focus is geared towards individuals and interactions who share a joint theory of the working software being more important than detailed documentations. Spending time on contract negotiations becomes less relevant than customer collaboration. Most importantly in the event of a change, the strategy should be to respond to that rather than following a plan. The process started with only the above principles and took the software industry by storm in the recent few years as iterative development started getting appreciations. Overall the methodology promotes a culture where inspection and adaptation occurs frequently based on the change.

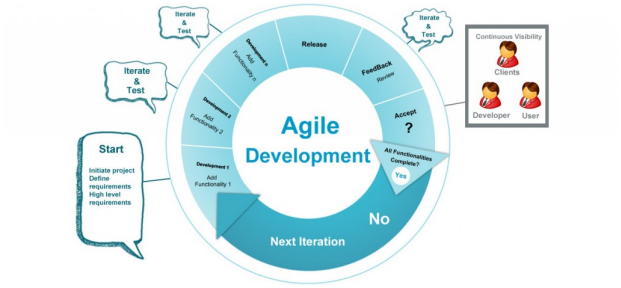


Figure 3.20: Agile Development in Practice - Source: Asahitechnologies.com

As compared to Waterfall approach, Agile process have short planning cycles which accommodate changes at any phase depending on the variations of requirements. Hence the cost of

rework is significantly reduced. The figure below shows the Agile process with the flow direction and involvement of stakeholders.

3.10.4 Methodology used for the Project

The project leverages the power of Agile methodologies throughout the software development cycle. As soon as it was established, IBM Watson not being suitable to use, the project approached the way of building its own Custom Transfer Learning Model. In the beginning the intention was to use the CNN algorithm of IBM Watson for Transfer Learning to detect calories. Due to Watson Studio not being a free platform, the problem was addressed and a different direction for solving the problem was explored. Before coming to deep learning, different machine learning algorithms were explored such as K-NN and SVM. After that it was established that implementing a deep CNN model will the best option for the project. During the data set gathering stage, the project came into a road block as there are insufficient data on the web. The methodology instantly addressed the issue and later images of desired classes were obtained manually. Following the Agile system, the first iteration was build a very general Artificial Neural Network for testing. The second iteration of the project was to code a simple Convolutional Neural Network from scratch in Keras to gain some hands on experience. After the third data gathering stage of building a custom deep CNN, the fourth iteration was to fine tune the custom pre-trained CNN algorithm. The final iteration is to collect the results of both the Transfer Learning approaches and evaluate keeping important factors into consideration.

As seen earlier in the report, Transfer learning required presence of a pre trained algorithm which will work as a convolution layer to the custom model. Different pre trained algorithms are explored and compared among each other for their Size and Top -1 accuracy values before making a selection. After that the project moved on to the next iteration which is implementation.

Chapter 4

Transfer Learning with Xception and VisionFeaturePrint.Scene

4.1 Transfer Learning

As mentioned earlier in the paper, a lot of training data is required in order to build a good sophisticated image classifier. The CNN can start learning from scratch the low level features and then reach to high level after some layers and operation. The whole process also is computationally very expensive as training an entire network from scratch takes time and resources. Transfer learning helps in over coming the mentioned problems above by requiring less data and being computationally affordable, while still maintaining accuracy.

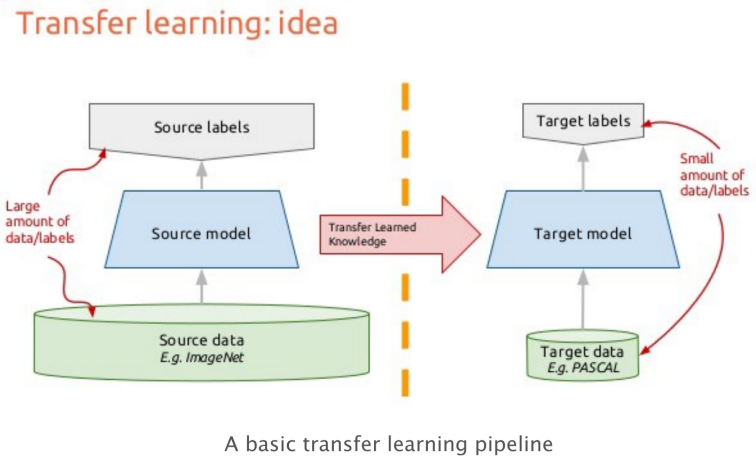


Figure 4.1: Transfer Learning from high level - Source: TowardsDataScience

In transfer learning the process takes advantages of acquiring feature detection knowledge from pre trained deep learning models. Deep CNNs have been shown good accuracy when extracting information from a specific domain and then applying that knowledge in a separate task [85]. A pre-trained deep learning model is been trained and tested on millions of images to recognise objects. It leverages the knowledge of recognising high level features to train on the custom new layer of the network at the top.

Many pre-trained models used in Transfer learning consist of large convolutional neural networks. Usually CNNs has two parts:

- A convolutional base which is a stack of convolution and pooling layers and the objective of these layers is to extract features from the input image [86].
- A classifier which consists of one or more fully connected layers and its goal is to classify images based on the features learned from the previous part with the help of the activation function [86].

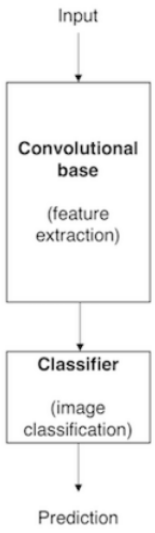


Figure 4.2: Conventional CNN - Source: TowardsDataScience

Using pre-trained models for transfer learning starts of by replacing the original classifier with a custom one specifically designed to solve any specific problem. There are three strategies to this process:

- **Training the entire model**

  The pre-trained network is trained from scratch and only the architecture is used. It is very important to use a large data set as training from scratch can be very expensive to

teach the network and requires a lot of computational power [87].

- **Training few layers and leave the rest frozen**

  The higher layers of a CNN refer to the problem dependent features while the lower layers focuses on more general features which are independent of the problem. If the data set for training is large, parameters are small in number and the new data set is different then training more layers of the pre-trained algorithm to the new classification is a good option. Vice versa, if the data set is small and large number of parameters, then most layers are kept frozen to avoid over fitting problem [87].

- **Freezing the convolutional base**

  This is only applicable when the data set present for training is small and the parameters are also a small number. The approach is to freeze the convolutional base and keep it in its previous form when it was trained before with big data set. The output from this base is passed straight into the classifier. So the pre-trained network act as a feature extracting mechanism which comes handy when high computational power is not available [88]. The use of the pre-trained network as a feature extractor also significantly reduces the chances of the Overfitting problem [89].

The research will use strategy three to approach the project as it satisfies the requirements of the presence of small different data set and computational power. The question remains now is which pre-trained model/algorithm to use for the project. A novel architecture Xception is explored in the next section. To test the theoretical depth of fine tuning, extensive training iteration has been done with different number of frozen layers. The screenshots of the terminal outputs are in the Appendix section ...

4.2 Xception

Xception is a pretrained model present in Keras API. It is the successor of 2015 ILSVRC 1st Runner Up, Inception V3 from Google [90]. The model stands for Xtreme version of Inception and is better in performance than its predecessors. It uses an architecture quite different from the traditional CNNs as it improved massively in the convolution process which causes the most overhead during training process. The authors of the model introduced a new concept called Depthwise Separable Convolution. Depthwise Separable Convolution has existed in Inception models before but Xception changed the architecture to achieve better results. The architecture

consists of 36 Convolutional layers which forms the feature extraction zone of the algorithm. The layers have been structured into 14 modules, all of which have residual connections around them except the first and last layer [91]

The figure below shows the original depthwise convolution in InceptionV3.

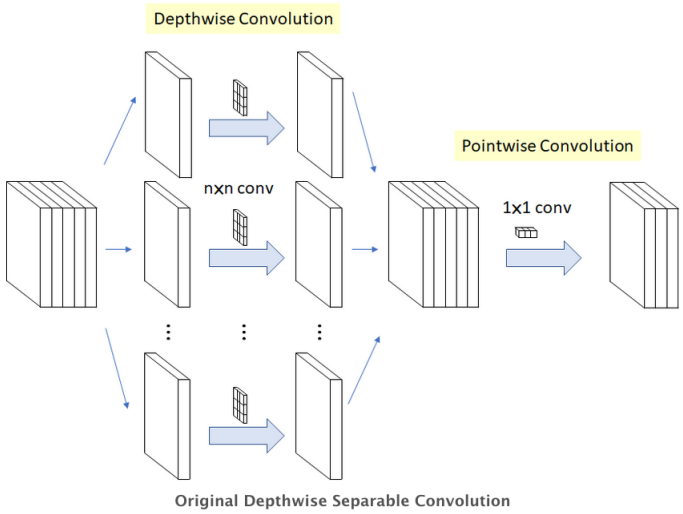


Figure 4.3: Original depthwise separable convolution - Source: TowardsDataScience

The process involves having two convolutions:

- Depthwise convolution: It is the channel-wise n x n spatial convolution [90]. Example in the figure above there are five channels, so there will be 5 n x n convolutions
- Pointwise convolution: It is the 1 x 1 convolution to change the dimension [90].

4.2.1 Modified Depthwise Separable Convolution in Xception

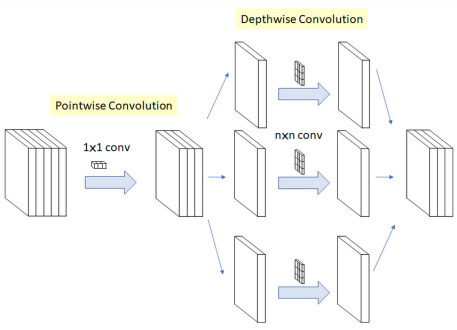


Figure 4.4: Modified Depthwise Convolution

The process is reversed in Xception as point-wise convolution is carried out before depth-wise convolution. The outcome depicts the presence of no intermediate ReLU non-linearity and greater accuracy [90]. The figure below shows the overall architecture of Xception.

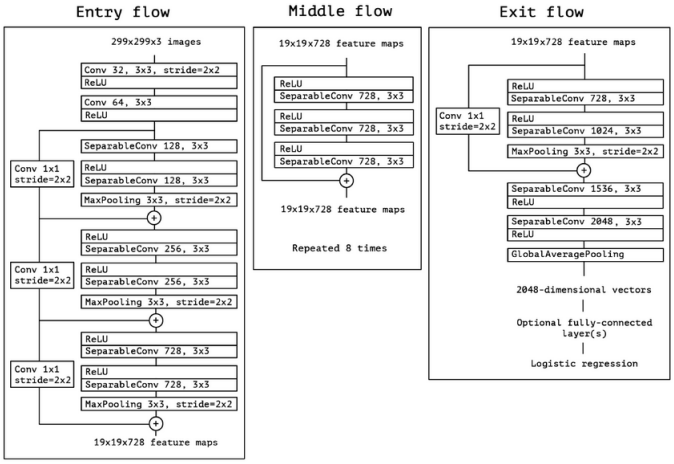


Figure 4.5: Architecture of Xception - Source :Chollet, Xception:Deep Learning with Depthwise Separable Convolutions

The architecture also uses residual connections which helps achieve better results than its competitors. In short it is a linear stack of modified depthwise separable convolutions with residual connections [91].

Figure 4.6: Performance of state of the art pre-trained models in Keras - Source: Keras Documentation

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
ResNeXt50	96 MB	0.777	0.938	25,097,128	-
ResNeXt101	170 MB	0.787	0.943	44,315,560	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159
InceptionResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-

As seen in the figure in the previous page, Xception is the smallest pre-trained algorithm in the chart which posses the third highest accuracy of 79 percent. The second highest accurate

model is more than two times and the most accurate model is three times larger in size than Xception. The algorithm was also tested not long ago, for classification task like age prediction, where it demonstrated robust performance over other famous pre-trained models [92]. In a transfer learning study last year, the algorithm was tested on how well it can transfer the features learned from previous training [31]. The implementation proved to be very effective as the accuracy value was more than 85%. In 2018 three stanford researchers showed that Xception performs well than other algorithms when tested with food images using Transfer Learning [93].This justifies the choice of using Xception for the image classification project.

4.3 VisionFeaturePrint.Scene

This section describes the pre-trained algorithm underneath Create ML, providing powerful features for Create ML to utilize for Transfer Learning. According to raywenderlich.com, the algorithm backs the vision framework in Apple devices and has been trained on a very large visual data set to classify a lot of different classes [94]. VisionFeaturePrint.Scene acts as the convolution layer during training and most time is spent in extracting 2048 features from the image data set which includes low level shapes and textures. After that it spends a small amount of time training a logistic regression model to separate the data set into 8 classes.

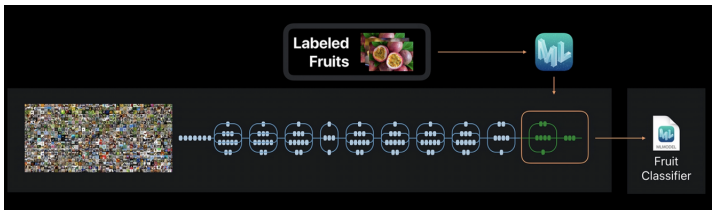


Figure 4.7: Transfer Learning with VisionFeaturePrint.Scene - Source: Apple WWDC 2018

The above figure depicts the Transfer Learning process with VisionFeaturePrint.Scene. In the context of the project, the 'Fruit Classifier' can be thought of as the fast food classifier. The algorithm performed really well previously when taught to classify lung cancer infected cells [4].

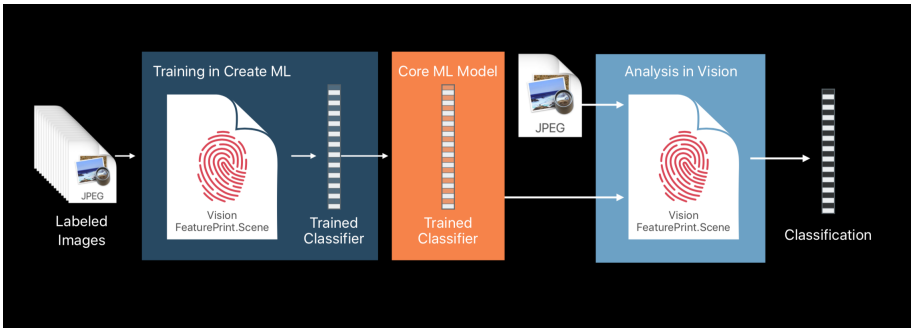


Figure 4.8: How Create ML integrates with VisionFeaturePrint.Scene through Transfer Learning - Source: Apple WWDC 2018

The above figure provides a visual way of understanding the Apple Transfer Learning process in depth. According to Frank Doepke, Senior Engineer at Apple, VisionFeaturePrint.Scene is a good choice for developers because it is already optimized to perform well on Apple devices.

"The other thing of course why we believe that this is a good choice for use, it's already optimized. And we know a few things about our hardware or GPU and CPUS and we really optimized a lot on that model so that it performs best on our devices"- Doepke, WWDC 2018 [95].

One of the most important point he mentioned during the presentation, was to run the classifier only when the user wants to as classifiers are deep Convolutional Neural Networks that are fairly computationally expensive [95].

Chapter 5

Implementation

The implementation steps and strategies are described in this section in a concise way to keep the objective on the architecture choices instead of in depth code discussion. The architectural choices made are the most important parts of the section as most Keras models follow a similar architecture, which use a TensorFlow backend.

- The project will dive into two specific Transfer Learning approaches:

1. Transfer Learning on custom data using Keras pre-trained algorithm Xception.
2. Transfer Learning on custom data using Apple pre-trained algorithm VisionFeaturePrint.Scene.

As a result the project is demonstrating two different ways of transfer learning on the same data set. The first way requires access to the cloud as Keras is a high level cloud API and it uses TensorFlow as its backend which is also located in the cloud. The second approach keeps all the computation and data on the device, more specifically in Apple Xcode.

5.1 Environment Set Up

The process begins with installing Python 3.7, Keras 2.2.4, TensorFlow 1.13 and Xcode 10.2 which comes with pre built Swift 5. The installation did not take much time for the first three packages and took roughly 1 hour to install the latest Xcode. Before installing Keras, TensorFlow needed to be installed first due to the reason mentioned earlier (Keras uses TensorFlow backend). Installing TensorFlow installed its dependencies Numpy and Scipy automatically. After that Keras was installed as well and the whole process was smooth.

5.2 Image Data set Preparation

The image data sets were prepared and kept in the same directory labelled as 'Training' and 'Testing'. The 'Training' directory contain eight image folders of the classes the project want to identify which are:

Figure 5.1: Number of fast food categories for classification

Brand	**Fast Food Name**	**Amount of Calories Present**
KFC	Fillet Rice Box	448 kcal
KFC	Fries	250 kcal
KFC	Chicken(Thigh)	285 kcal
KFC	Zinger Stacker Burger	780 kcal
KFC	Fillet Tower Burger	650 kcal
KFC	Popcorn Chicken	285 kcal
KFC	Zinger Burger	450 kcal
Mcdonalds	Big Mac Burger	508 kcal

The 'Testing' directory also contains the same eight folders with their respective labelled names. The calorie value information is taken from the official website of KFC and Mcdonalds Nutrition Pages.

5.2.1 Splitting the data sets

As mentioned earlier the data sets are split into two directories: 'Training' and 'Testing'. The 'Training' directory contains samples on which the CNN can train and learn the features of the specified classification. The 'Testing' provides images for the trained CNN model to test its learning performance. A common rule of thumb in the field of Deep Learning is to divide the total data sets into training and testing in the ratio 80:20. Following the convention, the data sets are divided into 80 percent 'Training' set which are 4152 images and 20 percent 'Testing' set which are 1032 images. The 'Testing' set will be kept hidden from the model during training and only be exposed during testing stage to evaluate the validation accuracy.

5.3 Transfer Learning on custom data using Xception

At first the necessary libraries are imported from Keras and the training and testing directories are defined as well. The appropriate data and algorithm holding variables are initialised as well. Then the pre-trained algorithm is initialised with 'inputShape = (299, 299, 3)', imagenet weights and excluding the top layer/classifier. The input shape specifies the dimensions of the image and the '3' represents the number of channels which in this case is Red, Green and Blue. If the images were black and white then the value should be 1.

ImageNet is a competition that runs each year where different algorithms with their trained weight values are presented [96, 97]. The models with the best accuracy are recorded and kept for future use of importing pre trained algorithms. The weight values from imagenet are imported as they are trained on millions of images and been tested to perform well. It has been observed through research that using ImageNet weights achieves the same accuracy as random weight values initialisation. But on the plus side, importing the pre trained values scaled down the training time and data set by orders of magnitude [98]. In the same paper it was established that ImageNet models with better architectures perform better when transferring features from one domain to another [98].

Removing the last layer is the one of the main steps of Transfer Learning as the layer will be replaced with a custom classifier.

```
preTrainedAlgorithm = Xception(input_shape = (299, 299, 3),
weights = imagenetWeights, include_top = False)
```

As explored earlier in Section 4.1, in the event of having different small data sets for classification, it is best to fine tune the pre trained algorithm. The fine tune technique consists of choosing how many and which layers to keep frozen during training phase. After multiple training experiments it was established that keeping the first 20 layers in the algorithm is the best option for the project.

```
for layer in preTrainedAlgorithm.layers:
layer.trainable = False
```

The figure below shows a visual representation of the architecture of the custom deep CNN built in the project. The choices of the layers are explained through out the section.

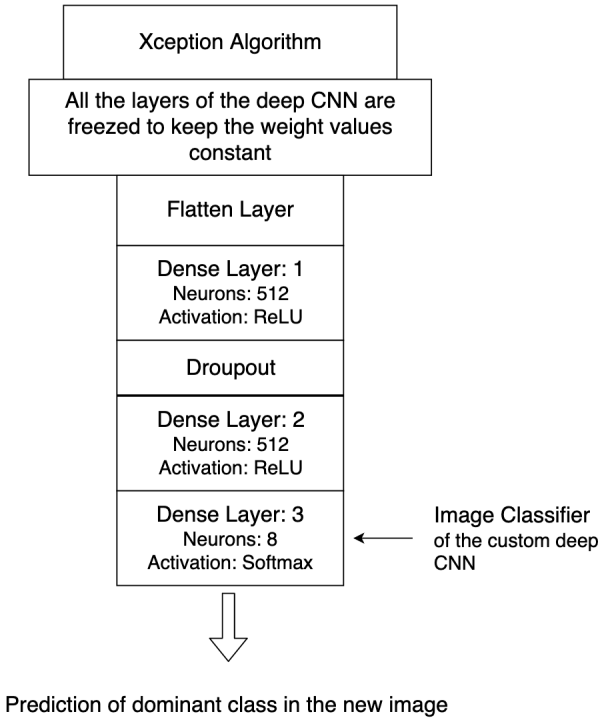


Figure 5.2: Architecture of the custom deep CNN for CalorieTracker.ai

A sequential model in Keras is initialised and the pre trained algorithm is added to it to make a custom model. The output of the pre trained model is flattened.

```
customModel = Sequential()
customModel.add(preTrainedAlgorithm)
customModel.add(layers.Flatten())
```

A dense layer is added on top of the network with 512 neurons with activation algorithm, ReLU. In the presence of large parameters, a neural network is prone to known problems like overfitting. To address that a technique, Dropout is explored. Dropout usually prevents overfitting by dropping units from the network while training [99]. A second dense layer is added with the same number of neurons and activation algorithm. The last dense layer performs a logistic regression function with the number of image classification classes. The activation algorithm implemented here is 'Softmax'. The format shown is a very standard notation of defining custom CNN layers in Keras.

```
customModel.add(layers.Dense(512, activation = reluAlgorithm))
customModel.add(layers.Dropout(0.5))
customModel.add(layers.Dense(512, activation = reluAlgorithm))
```

```
customModel.add(layers.Dense(classificationClasses, activation = softmaxAlgorithm))
```

As mentioned before in Section 3.7.4, EarlyStopping algorithm keeps tracks of the validation accuracy through the training process. With the end of each epoch, it will monitor whether the validation loss value increased or decreased. According to MATLAB Answers, an epoch is the number of times all the training images are used to update the weight values [100]. If the validation loss increases after each epoch, it means that the model is running into over fitting issue. To prevent that from happening, EarlyStopping is implemented and the best model of the training process is observed and saved by ModelCheckpoint algorithm. ModelCheckpoint algorithm has the following parameters [101]:

- monitor: It monitors the validation loss value during training.
- save_best_only: If set to 'True', it will save the best model from the training with the least validation loss value.
- mode: It can be either of auto, min and mode. If the previous argument 'save_best_only' is set to 'True', then overwriting the saved file only depends on the minimization of validation loss if the mode is set to min. The system automatically decides on the direction of training when chosen auto.
- period: This is the interval between checkpoints during training.

```
callbacks_list = [
keras.callbacks.ModelCheckpoint(
filepath='best_model.{epoch:02d}-{val_loss:.2f}.h5',
monitor= validationLoss, save_best_only=True, mode = min, period = 0),
keras.callbacks.EarlyStopping(monitor= validationAccuracy,
patience= 0, mode = max)
]
```

According to Keras documentation, the mode argument in EarlyStopping is one of either auto, min, max. Training will be stopped in 'min' mode when the validation accuracy stopped decreasing. In 'max' mode the training stops when the validation accuracy stopped increasing and in 'auto' default mode, the system infers the statistics of the validation accuracy and makes choices of the direction in training [101].

Then the custom model is compiled with loss function/algorithm 'Categorical Crossentropy' and optimizer algorithm 'Adam'. Adam is used with a low learning rate of 0.0001. As the

number of classification classes is more than 2, Categorical Crossentropy algorithm fits well to serve the purpose. In the event of two classification labels, binary crossentropy algorithm can be used. The metrics used here is 'accuracy'.

```
CustomModel.compile(loss = categoricalCrossentropyAlgorithm, optimizer =
adamOptimizer,
metrics = ['accuracy'])
```

The training data is augmented through the in built ImageDataGenerator class of Keras. The class comes with augmentation features like rescale, horizontal flip, rotation range and many others. For the purpose of the research the augmentations allowed here are rescale, horizontal flip, rotation range, shear range and zoom range. The data sets are trained and tested according to their respective batch sizes. Batch size is the number of examples or in this case images utilized in one iteration. The batch sizes for the training set are set to 128 and 32 as the number of training examples are larger in quantity. Both the training and testing data sets are rescaled to treat all images in the same manner. Another reason is because all the images are taken from a phone, the pixel values of the images are quite high in pixel value. So rescaling enforces typical learning rate of the classifier.

```
augmentedTrainingData = ImageDataGenerator(rescale = 1./255, shear_range = 0.2,
zoom_range = 0.2, horizontal_flip = True, rotation_range = 20)
augmentedTestingData = ImageDataGenerator(rescale = 1./255)
```

The training process begins when the custom model is fit with the generator with the appropriate training, testing, epochs number and steps per epoch. If not yet clear, a very important factor in deep learning is the value of weights and the process back propagation for updating the weight values to desired outputs. The number of epochs is kept 5 as a high number of epochs will result in a problem known as Overfitting where the custom model learns more about the data than it is required. As a result, it does not perform well in the testing images. The steps per epochs is kept at 10. Argument callbacks is also passed through the fit generator method.

```
r = customModel.fit_generator(augmentedTrainingData, validation_data =
                              augmentedTestingData, epochs = epochNumber,
                              steps_per_epoch = epochSteps, validation_steps =
                              len(augmentedTestingData), callbacks= callbacks_list,
                              verbose = 1)
```

The verbose variable is kept 1 so that progress of the training session will be animated and open to view for the user to keep track. Setting the variable 0 will result in no display of the progress.

5.3.1 Training Results

Updates can be seen on the performance of the model during training. The progress shown is in the form of 'loss', 'accuracy', 'validation loss' and 'validation accuracy'. The loss and accuracy refers to the training performance and rest two specifies the validation performance. All of them are in built in 'TensorFlow' through 'tf.Session()'. The user is updated on the progress of the training through this framework. For example, a decrease in loss values will result in increase of validation accuracy which means that the model is not over fitting. The figure below shows the training and the validation results.

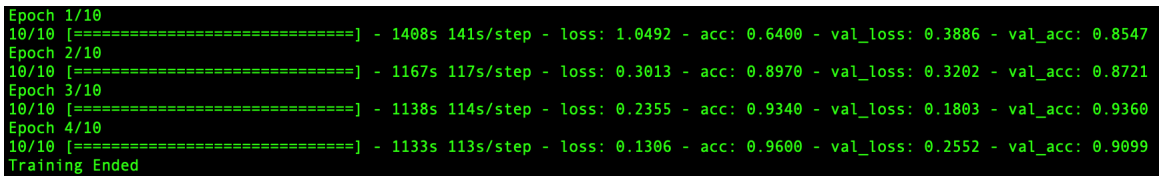


Figure 5.3: Output of the training process in MacOS terminal

```
loss: training loss

acc: training accuracy

val_loss: validation loss

val_acc: validation accuracy
```

As seen in the figure above, the training reached until epoch 4/10 epochs and was stopped by the EarlyStopping algorithm as the validation accuracy started to decrease. The algorithm is programmed to take an argument, 'patience' which is the number of epochs the algorithm would wait before interrupting training. The patience variable was set to 0, so if the validation accuracy decreases compared to the last epoch, the training would stop. The validation accuracy decreased in epoch 4 from epoch 3 and the training was stopped. If the validation accuracy did not show a downward trend then the training would have continued as normal and finish after 10 epochs.

5.4 Transfer Learning on custom data using Apple VisionFeaturePrint.Scene

Apple only released a very small explanation of VisionFeaturePrint.Scene in public. It is only known to be present for many years underneath the user's photo album application in Iphones and had been trained on a large mammoth of large visual data for weeks. Currently it is one of the largest Artificial Deep Network in Apple ecosystem and provides Transfer Learning for Create ML.

The process of implementing Transfer Learning technique in Xcode has been much more easier than Keras. The CreateMLUI is imported and then training process is being displayed as GUI(Graphical User Interface) by the second and third lines code commands below.

```
import CreateMLUI
let builder = MLImageClassifierBuilder()
builder.showInLiveView()
```

In order to match the robustness of the deep CNN built in the first approach, powerful augmentations are applied which increased the training time significantly. The training dataset is augmented through Rotate and Flip as these two are the only ones close to augmentation settings provided by Keras in Section 5.3 . The maximum of iterations for back-propagation process is kept to 50 after testing it with 10 and 30 iterations before. The results from different number of iterations are shared in the Appendix section A.3. Plus in the previous approach using Xception algorithm, the model was could only continue till 4 epochs (40 Iterations). The training folders containing the labelled data sets is dropped into the GUI and the model starts training in the Xcode native playground.

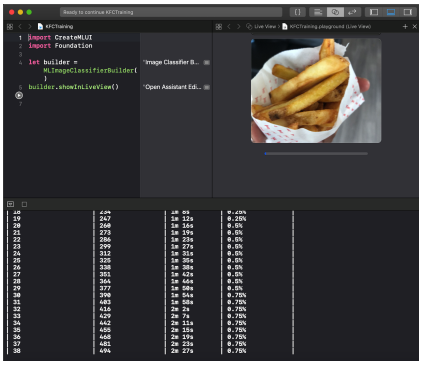


Figure 5.4: Training session in Apple Xcode playground

5.4.1 Results of the Training Process

Xcode is capable of showing the progress of the training in its built in playground tool. The figure below demonstrates the playground output.

Iteration	Elapsed Time	Training Accuracy	Validation Accuracy
0	1.579302	0.124397	0.140097
1	6.075623	0.778761	0.811594
2	15.609237	0.873318	0.893720
3	18.830498	0.876072	0.903382
4	22.368848	0.881715	0.908213
5	25.427250	0.886949	0.908213
6	28.444988	0.892495	0.917874
7	31.561825	0.897905	0.917874
8	34.587895	0.903080	0.917874
9	37.611545	0.907610	0.922705
10	40.832049	0.913156	0.922705
11	43.882396	0.918078	0.927536
12	46.816718	0.923253	0.932367
13	49.924235	0.928174	0.937198
14	53.434661	0.933056	0.951691
15	56.531828	0.938114	0.961353
16	59.543243	0.943035	0.966184
17	62.698869	0.950026	0.980676
18	65.701770	0.953815	0.985507
19	68.654335	0.956861	0.985507
20	71.982349	0.960787	0.985507
21	74.980834	0.964382	0.985507
22	78.023152	0.967212	0.990338
23	81.353661	0.969946	0.990338
24	84.723042	0.972738	0.990338
25	87.730468	0.975589	0.990338
26	90.693799	0.977308	0.990338
27	93.934180	0.979671	0.990338
28	96.874677	0.981760	0.990338
29	99.924725	0.983596	1.000000
30	103.242348	0.985412	1.000000
31	106.283776	0.986584	1.000000
32	109.121790	0.988361	1.000000
33	112.434886	0.989416	1.000000
34	115.837531	0.990783	1.000000
35	118.867466	0.991778	1.000000
36	121.881044	0.992853	1.000000
37	125.036307	0.993438	1.000000
38	128.001696	0.994083	1.000000
39	131.093778	0.994473	1.000000
40	134.429166	0.994962	1.000000
41	137.464920	0.995469	1.000000
42	140.684655	0.996153	1.000000
43	143.745878	0.996387	1.000000
44	147.098768	0.996602	1.000000
45	150.185282	0.996914	1.000000
46	153.070619	0.997129	1.000000
47	156.347641	0.997305	1.000000
48	159.147551	0.997559	1.000000
49	162.433014	0.997774	1.000000
50	165.525963	0.998067	1.000000

Figure 5.5: Output of the training process in MacOS terminal

For the purpose of the research, the validation accuracy is a good measure of accuracy here as it demonstrates how the model performed on images it has not been trained on. As seen above, the algorithm did really well and made a big jump in the 2nd iteration.After that the both training and validation values kept rising slowly. It reached validation accuracy of 100% after the 28th Iteration while the training accuracy was still at 98.4%. At the end the training accuracy went to a highest of 99.8%.

5.4.2 Predictions

Predictions are being made from the mobile iOS app CalorieTracker.ai and the results are displayed below. Since building an iOS application was not the requirement of the research rather a model which can perform robust image classification, the code for the template is taken from a teaching course on Udemy and Apple documentation. The code is modified to better serve the needs of the project and is referenced in the Appendix section. Apple has provided a template application in their documentation to let developers build and test their deep learning

models [102]. The samples used for predictions are taken from device in aeroplane mode on real KFC samples. This is a good measure to test on images which the model have not seen both during Training and Validation stages.

Correct

Correct

Correct

Chapter 6

Evaluation

6.1 Comparing both Transfer Learning approaches

In this section the performance of both the Transfer Learning approaches described above are reviewed in terms of validation accuracy, validation loss and training time.

6.1.1 Results from Xception

For the first Transfer Learning approach using pre-trained deep learning model Xception, the validation accuracy showed an upward curve at first and then started to depreciate after the third epoch. EarlyStopping algorithm monitors the validation accuracy values through out each epoch and stops the training after the fourth epoch as the value went down from 'r = 0.94' to '0.91'. The changes in accuracy values during training process is shown in Figure 6.1 below.

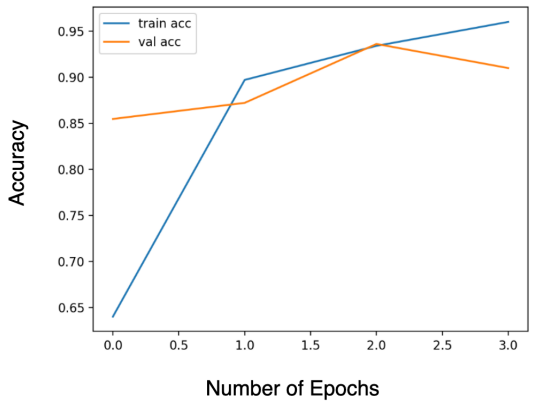


Figure 6.1: Comparison of Training and Validation accuracy VS Number of Training Epochs

The figure provided below shows the change in validation loss VS change in training loss throughout the training process.

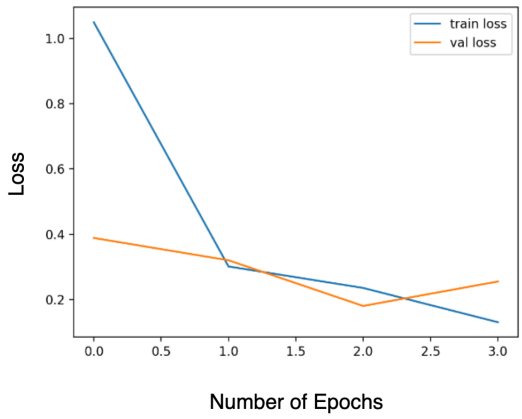


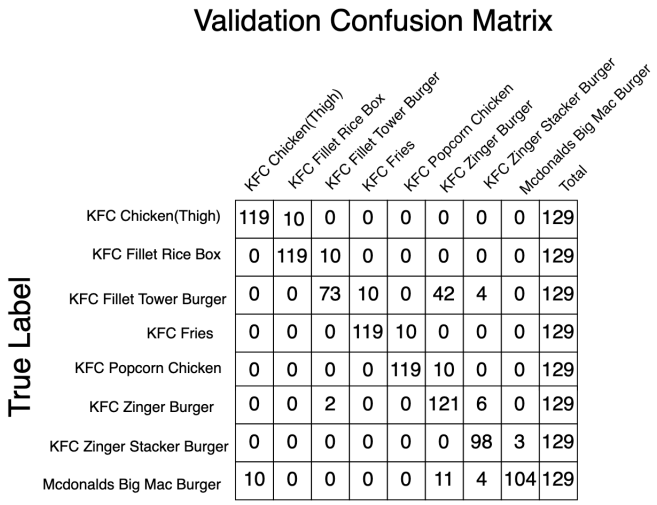
Figure 6.2: Comparison of Training and Validation loss VS Number of Training Epochs

As observed in the figure above, the validation loss started on a decreasing trend and continued to decrease until the end of the third epoch, 'r = 0.18'. From the fourth epoch it started increasing which inversely affects the validation accuracy value. Hence the training process was stopped by Early Algorithm.

Looking at the two plots above, the presence of minor overfitting is established during the training process until the third epoch. After that the overfitting ratio increased by significant amount.

The figure belows shows the confusion matrix for validation.

Figure 6.3: Validation Confusion Matrix for Xception



As seen above, the model showed good accuracy when performing on validation images of 'KFC Chicken(Thigh)', 'KFC Fillet Rice Box', 'KFC Fries', 'KFC Popcorn Chicken' and best with 'KFC Zinger Burger'. The confusion matrix is very good at demonstrating the times the model predicted wrongly and also the wrong classification labels. The model performed poorly when predicting 'KFC Fillet Tower Burger' as it mistakenly taken the object to be either 'KFC Zinger Burger', 'KFC Zinger Stacker Burger' or 'KFC Fries'. The figure below shows the classification report of the trained model.

Figure 6.4: Classification Report for Xception

Classification Report

	precision	recall	f1-score	support
KFC Chicken(Thigh) 285 kcal	0.92	0.92	0.92	129
KFC Fillet Rice Box 448 kcal	0.92	0.92	0.92	129
KFC Fillet Tower Burger 650 kcal	0.86	0.57	0.68	129
KFC Fries 250 kcal	0.92	0.92	0.92	129
KFC Popcorn Chicken 285 kcal	0.92	0.92	0.92	129
KFC Zinger Burger 450 kcal	0.57	0.94	0.71	129
KFC Zinger Stacker Burger 780 kcal	0.88	0.76	0.81	129
Mcdonalds Big Mac Burger 508 kcal	0.97	0.81	0.88	129
micro avg	0.84	0.84	0.84	1032
macro avg	0.87	0.84	0.85	1032
weighted avg	0.87	0.84	0.85	1032

As seen above, the model performs least favourable when classifying KFC Fillet Tower Burger. In the confusion matrix it can be seen that it wrongly predicted the food as KFC Zinger Burger which looks similar. KFC Zinger Stacker burger and Mcdonalds Big Mac burger both have the second and third least favourable recall values.

6.1.2 Results from VisionFeaturePrint.Scene

The project will now evaluate the results of the second Transfer learning approach which was training the model on VisionFeaturePrint.Scene. The validation accuracy achieved is a good value depicting that the model was able to train well and in theory, will predict correctly most of the time. The prediction process was done manually with some hand selected images from the internet which is different from the types of images used in training. The model demonstrated good inference capabilities and was able to correctly predict most images. By observing the training and validation curves in the figure below, it is quite clear that there was no over fitting problem during the training process.

The green curve is represented as training accuracy and the blue is represented as the validation accuracy. As seen in the graph, both the accuracy values increase with the number of iterations. The algorithm's performance on the validation data is better than training. This proves less chances of overfitting during training. The accuracy values went high after the first iteration which is a mark of the algorithm efficiency.

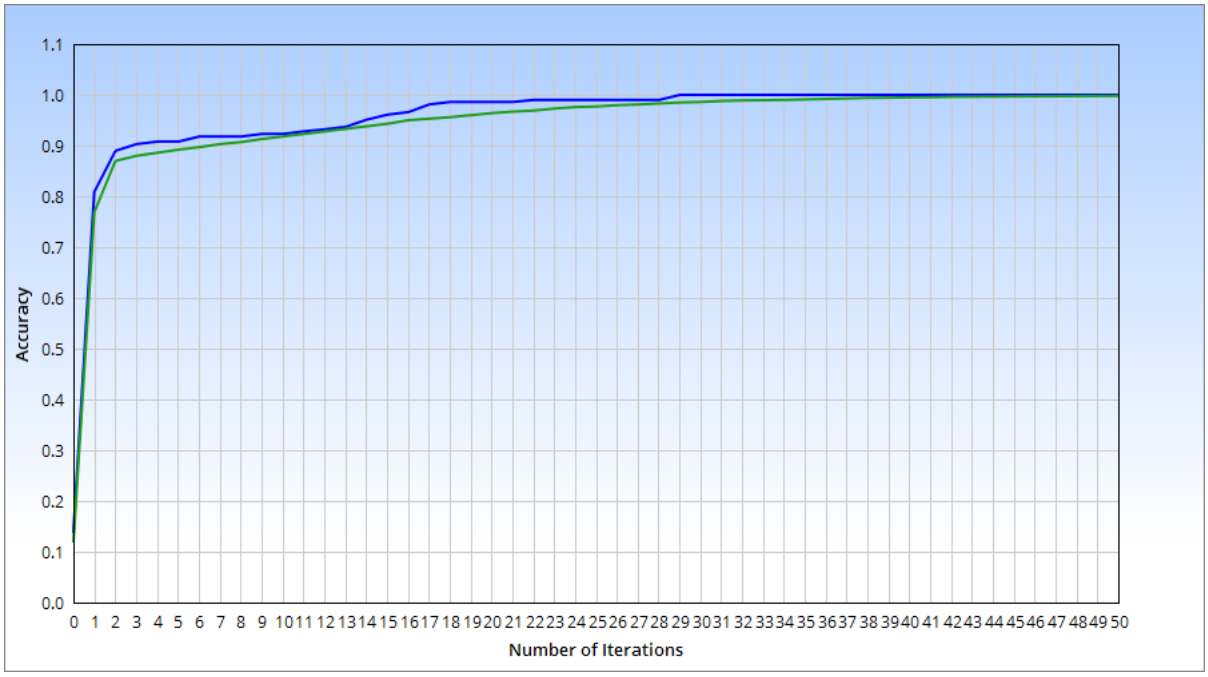
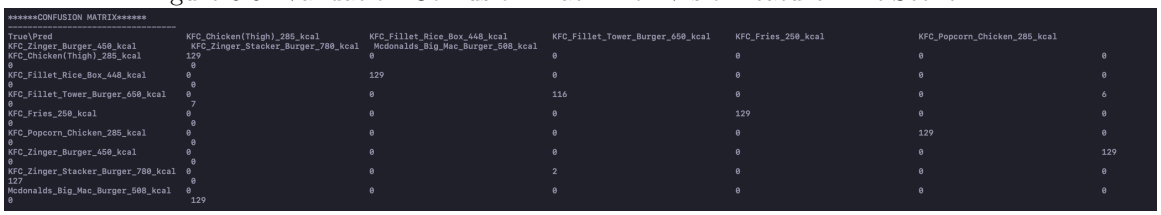


Figure 6.5: Variations of Training and Validation Accuracy during training process

Figure 6.6: Validation Confusion Matrix for VisionFeaturePrint.Scene



******CONFUSION MATRIX******

True\Pred	KFC_Chicken(Thigh)_285_kcal	KFC_Fillet_Rice_Box_448_kcal	KFC_Fillet_Tower_Burger_650_kcal	KFC_Fries_250_kcal	KFC_Popcorn_Chicken_285_kcal	
KFC_Zinger_Burger_450_kcal	KFC_Zinger_Stacker_Burger_780_kcal	Mcdonalds_Big_Mac_Burger_508_kcal				
KFC_Chicken(Thigh)_285_kcal	129	0	0	0	0	0
0	0					
KFC_Fillet_Rice_Box_448_kcal	0	129	0	0	0	0
0	0					
KFC_Fillet_Tower_Burger_650_kcal	0	0	116	0	0	6
0	7					
KFC_Fries_250_kcal	0	0	0	129	0	0
0	0					
KFC_Popcorn_Chicken_285_kcal	0	0	0	0	129	0
0	0					
KFC_Zinger_Burger_450_kcal	0	0	0	0	0	129
0	0					
KFC_Zinger_Stacker_Burger_780_kcal	0	0	2	0	0	0
127	0					
Mcdonalds_Big_Mac_Burger_508_kcal	0	0	0	0	0	0
0	129					

Figure 6.7: Classification Report for VisionFeaturePrint.Scene

******PRECISION RECALL******

Class	Precision(%)	Recall(%)
KFC_Chicken(Thigh)_285_kcal	100.00	100.00
KFC_Fillet_Rice_Box_448_kcal	100.00	100.00
KFC_Fillet_Tower_Burger_650_kcal	98.31	89.92
KFC_Fries_250_kcal	100.00	100.00
KFC_Popcorn_Chicken_285_kcal	100.00	100.00
KFC_Zinger_Burger_450_kcal	95.56	100.00
KFC_Zinger_Stacker_Burger_780_kcal	100.00	98.45
Mcdonalds_Big_Mac_Burger_508_kcal	94.85	100.00

As seen in both validation confusion matrix and classification report above, the trained model performed the worst when presented with KFC Fillet Tower Burger as the recall value for that is 89.92%. Rest of recall scores are acceptable for the study.

6.1.3 Training time for both approaches

It is important to acknowledge first that no GPU was used for the training process. Since it was only the Macbook CPU handling the whole process, training time was significantly longer in both approaches. Training time for the first Transfer Learning approach took 1 hours 21 minutes altogether. If the training finished all 10 epochs, then the time would have increased significantly. Given the time taken to train, achieving more than 90% validation accuracy value is a good indication. The architectural choices were robust enough to tone down the computational training time to a more acceptable value as robust training process usually takes more time. The second Transfer Learning took 3 hours and 32 minutes which is not bad given the achieved validation accuracy. But since Create ML takes of all the abstractions under the hood for selecting the best deep learning framework and algorithm, many developers would be keen to this approach. Investing 3 and half hours in training time without any knowledge of deep learning and still manage to achieve more than 98% accuracy stands as a decent trade off.

6.1.4 Sizes of the custom models from the Transfer Learning approaches

The size of the deep CNN model from the first approach using pre-trained Xception algorithm is 1.35 gb. This an alarming size for a model which needs to be deployed in a mobile device.

The second Transfer Learning approach using VisionFeaturePrint.Scene yielded much better result as the size of the model is only 116 kb. This is an ideal size for deploying in a mobile which can make predictions without the need for an Internet connection.

6.2 Project Requirement Completion

The section examines whether the functional and non functional requirements of the project from Section 3.1 are satisfied.

6.2.1 Functional Requirements

Having five main functional requirements, the first one was that the model should be a custom deep convolutional neural network. As seen in the implementation section, a custom deep CNN is used to train the dataset in the project for each Transfer Learning approach.

The second functional requirement was that the model will be able to train on a small training dataset. The number of total images used in the project is 5184 for both implementation approaches. Both the custom deep convolutional neural networks implemented in the project managed to train on a small dataset and achieve good accuracy value on the validation set.

Applying augmentation to training images was the third functional requirement. The model was able to satisfy that and apply various augmentations to the 4152 training images for both approaches.

The fourth requirement of the project was that the model has to stop training process has to stop automatically once the validation accuracy started to decrease. The requirement was only valid for the first Transfer learning approach as Create ML does not provide any mechanism for that. The VisionFeaturePrint.Scene was very efficient through out the whole training as it managed to keep an upward curve for the validation accuracy value. In case of Xception, as seen in Section 5.3.2, the training process was stopped in the fourth epoch when the performance started to decrease. Therefore the model satisfied the functional requirement.

The last requirement of the project was that the iOS built with the trained model will be able to make offline predictions. As seen in Section 5.4.2, the model was able to make correct predictions when the device was in aeroplane mode.

6.2.2 Non Functional Requirements

The project also had one non functional requirements. The first one was that the software must be able update the stakeholder with training and validation accuracy and loss values. The requirement is satisfied for the first approach, as the terminal output in Section 5.3.2 shows that.

In case of the second approach, Xcode only displays training, elapsed time and validation accuracy values.

Chapter 7

Future Work

7.1 Minor Improvements

Although the validation accuracy of the model is quite well in both approaches, there are some minor improvements to improve the performance. Both the models struggle to generate correct predictions when there is a lack of sufficient light in the picture. A dark image with a burger or any other items would sometimes fail to achieve the correct predicted label. Therefore in a darker environment existing generic calorie detection systems can be used instead of CalorieTracker.ai.

7.1.1 Augmentation

For the first approach, the model could be made to perform better by generating more different types of augmented pictures. Keras API provides more than 20 different types of augmentations which will require high training time and compute power. But in the long run it can significantly help in increasing the accuracy of the model.

7.1.2 Using a different pre-trained Deep CNN algorithm

For the first approach, a different pre-trained model like NASNetLarge which is very popular in the machine learning community for its performance. If size of the model is not an issue then this pre-trained model can be used to improve the performance. For the purpose of the research, it was important to keep the size of the model into consideration so there was a minor tradeoff in accuracy.

7.1.3 Increase the number of training images with different lighting conditions

Although the research tested the validity of the concept Transfer Learning by providing less samples, it is always better to have supply of more samples for training. The classifiers in both approaches will surely be more robust and faster in performing with a new picture.

7.1.4 Converting the Keras model into CoreML format

The Deep Learning custom CNN implemented in Keras can be converted to Coreml for using the model in Apple devices. CoreML format is acceptable in the Apple ecosystem. The same demo application used in section 5.4.2 can be used with just the model changed into the Keras one. Unfortunately the software which is required to do the conversion, Coremltools, have some specific limitations. Coremltools require Python 2.7 and TensorFlow 1.12 to function properly. The model developed in the project is built from Python 3.7 and TensorFlow 1.13. As a result it was not possible to downgrade to the required Python and TensorFlow version in order to use Coremltools. Finding a solution for the problem was beyond the scope of the time frame for the research and as a result was not explored.

7.2 Major Improvements

The research aim was to make two models for solving image classificaiton problems. But it would be nice to take the concept now one step further with object localizations, which can be achieved by using Faster R-CNN with Transfer Learning. More about this is described in the coming sections.

7.2.1 Problems to consider when implementing CNNs

One of the biggest problem of a CNN approach to image classification is that the objects in the image can have various aspect ratios and spatial locations. For example, in some cases the object might be dominating in the image or in other cases it can be covering a small percentage in the image. As a result the CNN has to explore the whole input image for detecting one or more objects present [103].

From the neural network context it means that the output layer of the network is not constant in length as it varies depending on the number of the objects present. Hence the process is computationally too expensive to process all the regions.

The concept of Convolutional Neural Networks can be extended to provide a more robust output which will not only classification but also the location of the object in the frame. The next few sections will depict various advancements in the field over a short period of time.

7.2.2 R-CNN

To tackle the problem described above, Ross Girshick proposed a way of using of selective search to extract only 2000 regions from the image in the form of region proposals. So instead of processing a large of regions, the algorithm now only looks at 2000 region proposals for detection [104]

```
Selective Search:
1. Generate initial sub-segmentation, we generate many candidate
regions
2. Use greedy algorithm to recursively combine similar regions into
larger ones
3. Use the generated regions to produce the final candidate region
proposals
```

Figure 7.1: Implementation of selective search algorithm - Source: TowardsDataScience, Medium

The region proposals are warped into a square and fed into a CNN producing a 4096 dimensional feature vector. As seen previously, CNNs are good at extracting features. Hence the CNN extracts features and forms a dense layer which is then fed into a SVM(Support Vector Machine), to identify the presence of objects within those regions.

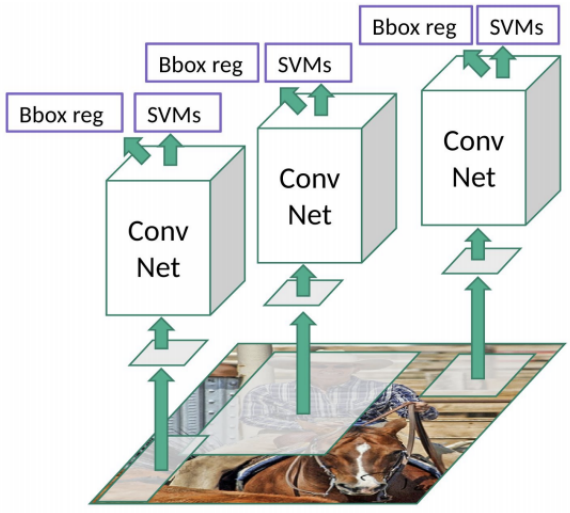


Figure 7.2: R-CNN - Source: TowardsDataScience, Medium

The algorithm also makes a prediction of four offset values to increase the accuracy of the bounding boxes, in addition to identifying the object in the proposed regions [104].

Problems with R-CNN:

- Classifying 2000 region proposals for each image takes a lot of time to train the network.
- It takes approximately 47 seconds to process any input image and classify it.
- No learning occurs during the execution of the selective search algorithm which in turn can lead to inaccurate region proposals.

7.2.3 Fast R-CNN

To tackle the issues in R-CNN the same author Ross ..... came up with an approach called Fast R-CNN where it is not required to feed the CNN with 2000 region proposals every time. The input image is passed into the CNN and a convolutional feature map is generated. Region of proposals are identified and warped into squares from the feature map. The proposals are reshaped into fixed size using RoI(Region of Interest) pooling and then fed into a fully connected layer. Using 'Softmax' function the class of the proposed region and the offset values for the bounding box are predicted from the RoI feature vector [104].

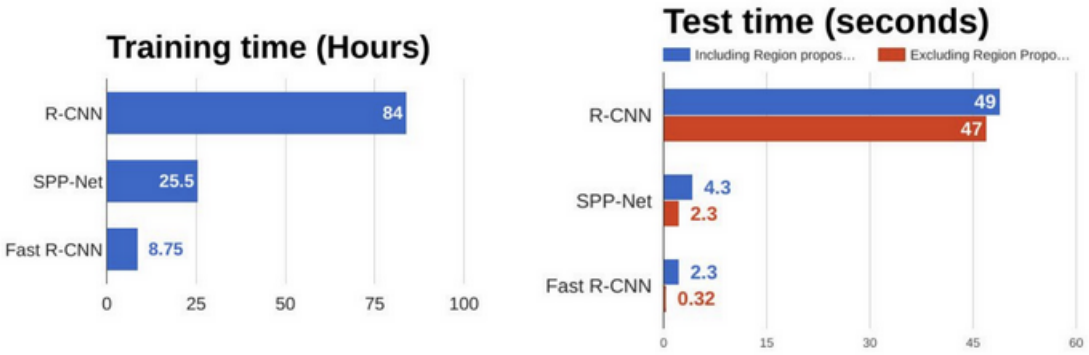


Figure 7.3: Comparison of object detection algorithms

Problem with Fast R-CNN:

The use of selective search algorithm to identify the Regions of Interest is still a slow and time consuming process which takes around 2s to detect objects [103].

7.2.4 Faster R-CNN

This algorithm is basically a modified version of Fast R-CNN. The only difference is that instead of generating regions of interest, the algorithm uses Regional Proposal Network(RPN). It repeats the same step of Fast R- CNN of generating feature maps from an input image and then applied RPN on those to return object proposals along with their objectness score. To bring down all the proposals to equivalent size, a RoI pooling layer is applied. After that the proposals are fed into a fully connected layer which uses 'Softmax' and 'Linear Regression' functions to classify and display the bounding boxes for objects [103].

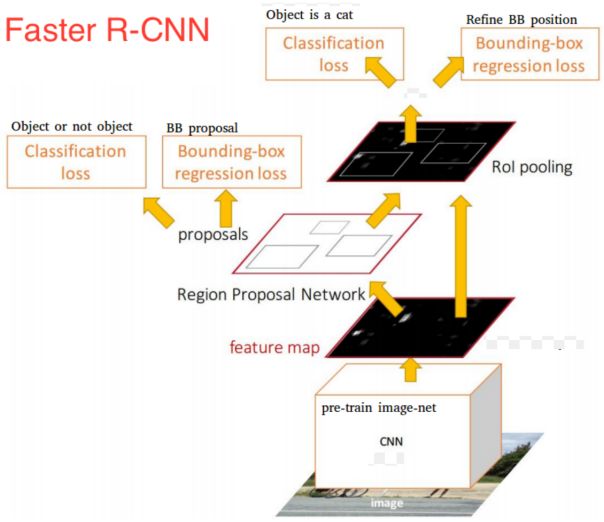


Figure 7.4: Faster R-CNN - Source: Earthmagic, Github

Chapter 8

Conclusions

The project demonstrated a powerful technique in Deep Learning which can be harnessed to achieve effective results for classification without the presence of Internet. The state of the art technique, Transfer Learning was first explored and then implemented to test the robustness of the theory part. Before this project, no attempts have been made commercially with such a small number of data set to train fast food models. Every company with a deep learning back-end has invested enormous amount of data to train their models so that there is no room for error. With access to double digits thousands of images, any good Convolutional Neural Network can achieve benchmarks results. But through this project it is established that low samples data set can also very effective when implemented with Transfer Learning. Of course the choice of a good pre-trained algorithm and its architecture was also very crucial in achieving the benchmarks. Also the strategy used from Transfer Learning section 4.1, proved to be effective as well.

When compared with the current industry methods discussed in Section 2.2, the proposed model demonstrated some strength and some weaknesses. It has successfully been able to make effective classifications without any internet connection through either network or WIFI. The process required a small data set and also a small comparatively small training time. In case for Apple, the training process even did not require any deep learning knowledge from user. The only challenge in that was to collect different types of images of the classes. Without Augmentation, the whole training process takes only less than 15 minutes, compared to 10 hours of augmented training but it will lose 5-10 percent accuracy value.

The notable weaknesses in the model is the variations in the type of data. Augmentation does increase variations but it is not the same as gathering raw data. For the Keras model,

size is a concern that cannot be overlooked. For a model with very desirable results but big in size, it might turn out less feasible when deploying it on phones. This is an area currently on going research and hopefully deep learning frameworks like Keras will be able to optimise models suitable for phones in the future.

Chapter 9

Legal, Social, Ethical and Professional Issues

9.1 British Computing Society: Code of Conduct and Code of Good Practice

During the project, all the good practices from British Computing Society were being explored and kept in close check with the research. The guidelines provided [105] were read and acknowledged before writing the report.

9.2 Ethical Issue

One of the biggest reason for favouring Apple Create ML platform because of the importance of data security. With big data starting to power up smart cities, it is very important to protect user privacy in all aspects. The longer the data is allowed to travel through the web to various cloud servers, the more it is exposed to cyber threats. But using Create ML implies trusting the deep Convolutional Neural Network algorithm powering it. An algorithm about which Apple has released no official information. As long as Apple does not release any information working with Create ML will be dealing with a black box. The platform has its own pre defined settings and architecture which is not allowed to change, limiting the user from other choices. From a research view point it seemed alright to experiment the algorithm's performance. But what are the chances of the algorithm under performing when implemented commercially with much

more classification choices?

Apple has been seen making out of the box innovations from time to time and Create ML is definitely one of the latest ones. With a market capitalisation of 182.8 billion, it is currently the most successful and valuable brand in the world [106]. So far the language swift has been able to impress developers by its swifty speed feature and easy syntax [107]. It will be clear in the upcoming years to come whether Create ML will be successful in building a machine learning on the go market.

9.3 Professional Issue

It is very important to attribute all the third party libraries used in the project. The iOS mobile application template used for predictions is taken from the Udemy course 'iOS 12 and Swift 4: From Beginner to Paid Professional' [108]. Since the objective of the project was to build two custom deep learning models, the template was used to speed up the research. The code is tweaked to meet the needs of the study.

References

[1] A. Dennanni and A. Dennanni, "Speed up your deep learning projects with pre-trained neural networks," Oct 2018.

[2] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1389–1397, 2017.

[3] "Create ml."

[4] S. A. Borkowski, "Using apple machine learning algorithms to detect and subclassify non-small cell lung cancer," *arXiv preprint arXiv:1808.08230*, 2018.

[5] "Kdnuggets."

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[7] B. Stephenson, "Have you tried counting calories with an app?," Jan 2019.

[8] "7 best calorie counter app in 2019."

[9] A. Weiner, "Samsung adds food image recognition to bixby through calorie mama api," Mar 2018.

[10] Greene, "Here's what ai experts think will happen in 2019," Jan 2019.

[11] "Artificial intelligence," Apr 2019.

[12] G. Press, "Ai in 2019 according to recent surveys and analysts' predictions," Dec 2018.

[13] "Linkedin 2018 emerging jobs report."

[14] S. G. S. o. Business, "Andrew ng: Artificial intelligence is the new electricity," Feb 2017.

[15] "Machine learning applications: How they work," Oct 2017.

[16] "Machine learning," Apr 2019.

[17] D. Newman, "Three ai and machine learning predictions for 2019," Jul 2018.

[18] "What is supervised learning ? - definition from whatis.com."

[19] "Supervised and unsupervised machine learning algorithms," Sep 2016.

[20] DeepLearning.TV, "What is a neural network - ep. 2 (deep learning simplified)," Dec 2015.

[21] "Nvidia blog: Supervised vs. unsupervised learning," Sep 2018.

[22] "What is regression in supervised learning?."

[23] B. Marr, "Supervised v unsupervised machine learning – what's the difference?," Mar 2017.

[24] L. Pan, S. Pouyanfar, H. Chen, J. Qin, and S.-C. Chen, "Deepfood: Automatic multi-class classification of food ingredients using deep learning," in *2017 IEEE 3rd international conference on collaboration and internet computing (CIC)*, pp. 181–189, IEEE, 2017.

[25] K. Yanai and Y. Kawano, "Food image recognition using deep convolutional network with pre-training and fine-tuning," in *2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1–6, IEEE, 2015.

[26] Y. Kawano and K. Yanai, "Food image recognition with deep convolutional features," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pp. 589–593, ACM, 2014.

[27] S. Pouyanfar and S.-C. Chen, "Semantic event detection using ensemble deep learning," in *2016 IEEE International Symposium on Multimedia (ISM)*, pp. 203–208, IEEE, 2016.

[28] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, "Deep learning for content-based image retrieval: A comprehensive study," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 157–166, ACM, 2014.

[29] A. Thomas, *An introduction to neural networks for beginners.* 2017.

[30] 187009171801728, "A gentle introduction to neural networks series - part 1," Aug 2017.

[31] S. Milhomem, T. d. S. Almeida, W. G. da Silva, E. M. da Silva, and R. L. de Carvalho, "Weightless neural network with transfer learning to detect distress in asphalt," *arXiv preprint arXiv:1901.03660*, 2019.

[32] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, pp. 647–655, 2014.

[33] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE Journal of Selected topics in applied earth observations and remote sensing*, vol. 7, no. 6, pp. 2094–2107, 2014.

[34] A. Hern, "Computers are now better than humans at recognising images," May 2015.

[35] "How long does it take to train deep neural networks? would it be feasible for an individual to replicate the performance of deep neural networks on the mnist dataset?."

[36] D. Cornelisse and D. Cornelisse, "An intuitive guide to convolutional neural networks," Apr 2018.

[37] P. M. Cheng and H. S. Malhi, "Transfer learning with convolutional neural networks for classification of abdominal ultrasound images," *Journal of digital imaging*, vol. 30, no. 2, pp. 234–243, 2017.

[38] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pp. 253–256, IEEE, 2010.

[39] "What is deep learning?," Sep 2016.

[40] B. Marr, "What is deep learning ai? a simple guide with 8 practical examples," Dec 2018.

[41] X. Li, H. Xiong, H. Wang, Y. Rao, L. Liu, and J. Huan, "Delta: Deep learning transfer using feature map with attention for convolutional networks," *arXiv preprint arXiv:1901.09229*, 2019.

[42] "Transfer learning - ml strategy (2)."

[43] "Create ml tutorial: Getting started."

[44] "Functional requirements vs non functional requirements," Oct 2018.

[45] O. 1, L. 103k30193251, pulasthipulasthi 1, S. M. M. 34.9k17188208, M. S. S. 1991418, A. 391, and R. D. S. D. S. 212, "What is the difference between functional and non functional requirement?."

[46] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, pp. 818–833, Springer, 2014.

[47] Uniqtech and Uniqtech, "Understand the softmax function in minutes," Jan 2018.

[48] Prabhu and Prabhu, "Understanding of convolutional neural network (cnn) - deep learning," Mar 2018.

[49] N. Jmour, S. Zayen, and A. Abdelkrim, "Convolutional neural networks for image classification," in *2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, pp. 397–402, IEEE, 2018.

[50]

[51] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014.

[52] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[53] H. Ide and T. Kurita, "Improvement of learning for cnn with relu activation by sparse regularization," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2684–2691, IEEE, 2017.

[54] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[55] "A gentle introduction to the rectified linear activation function for deep learning neural networks," Dec 2018.

[56] S. Saha and S. Saha, "A comprehensive guide to convolutional neural networksâĂŁâĂŁthe eli5 way," Dec 2018.

[57]

[58] 1245284952193863, "AdamâĂŁ-âĂŁlatest trends in deep learning optimization.," Oct 2018.

[59] M. Rizwan, "Adam optimization algorithm," May 2018.

[60] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[61] "What's new in core ml 2?."

[62] A. Creuzot and A. Creuzot, "Reducing core ml 2 model size by 4x using quantization in ios 12," Nov 2018.

[63] S. Nayak, "Home," Oct 2018.

[64] S. Kambampati, "Introduction to create ml: How to train your own machine learning model in xcode 10."

[65] S. Jagtap and S. Jagtap, "Wwdc18: Create ml-complete machine learning in swift," Jun 2018.

[66] "News."

[67] S. Sinha, "Apple's core ml 2 vs google's ml kit - which one is better for you?," Oct 2018.

[68] 1738555939573683, "Deep dive into google's ml kit and apple's core ml: Which one to use?," Jul 2018.

[69] "Machine learning on mobile: Core ml and tensorflowlite frameworks comparison."

[70] "What are the main differences between watson and tensorflow?," Dec 2016.

[71]

[72] J. Novet, "Apple's a.i. strategy stands apart from the rest of big tech, for better or worse," Jun 2018.

[73] "Python vs swift: Which language is better to learn."

[74] 10153643990339491, "Top 3 programming language to watch out in 2019," Nov 2018.

[75] A. Goel, "10 best programming languages to learn in 2019 (for job future)," Apr 2019.

[76] "Python vs java vs swift 2019 comparison," Nov 2018.

[77] "How python offers the best business advantages in 2019."

[78] S. S. Bhowmik, "Is there any advantage of using tensorflow for neural networks."

[79] "Tensorflow - google's artificial intelligence system for large scale machine learning," Oct 2018.

[80] J. McCaffrey03/04/2019, "Neural anomaly detection using keras."

[81] C. K. keras.io, "Keras reviews 2019 — g2," May 2018.

[82] "Waterfall model: What is it and when should you use it?," Nov 2017.

[83] "Next generation agile: Guide to continuous development."

[84] "What is agile methodology," Mar 2019.

[85] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 806–813, 2014.

[86] P. Marcelino and P. Marcelino, "Transfer learning from pre-trained models," Oct 2018.

[87]

[88] 1404033991, "A comprehensive hands-on guide to transfer learning with real-world applications in deep learning," Nov 2018.

[89] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in neural information processing systems*, pp. 3320–3328, 2014.

[90] Tsang, "Review: Xception - with depthwise separable convolution, better than inception-v3 (image...," Sep 2018.

[91] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[92] E. Bobrov, A. Georgievskaya, K. Kiselev, A. Sevastopolsky, A. Zhavoronkov, S. Gurov, K. Rudakov, M. d. P. B. Tobar, S. Jaspers, and S. Clemann, "Photoageclock: deep learning algorithms for development of non-invasive visual biomarkers of aging," *Aging (Albany NY)*, vol. 10, no. 11, p. 3249, 2018.

[93] M. Islam, S. Maheshwari, and N. Nunta, "Cs23o project: What's on my plate? identifying different food categories,"

[94] "Create ml tutorial: Getting started."

[95] A. Inc, "Vision with core ml - wwdc 2018 - videos."

[96] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[97] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[98] S. Kornblith, J. Shlens, and Q. V. Le, "Do better imagenet models transfer better?," *arXiv preprint arXiv:1805.08974*, 2018.

[99] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[100] "Charu (view profile)."

[101] "Create a callback."

[102] "Classifying images with vision and core ml."

[103] P. Sharma, "A step-by-step introduction to the basic object detection algorithms (part 1)," Oct 2018.

[104] R. Gandhi and R. Gandhi, "R-cnn, fast r-cnn, faster r-cnn, yolo - object detection algorithms," Jul 2018.

[105] "Bcs code of conduct."

[106] F. F. S. Board, "Top 10 world's most valuable brands in 2019," Apr 2018.

[107] A. Inc, "Swift."

[108] "Online courses - anytime, anywhere."