Skin Cancer Detection

Using

Convolutional Neural Network

Under the Guidance of

**Dr. Agilandeeswari L**

(Associate Professor, School of Information Technology and Engineering)

For the Course

**ITE1015 – Soft Computing**

For Winter Semester 2020-21

**Project Team:**

Aashish Bansal 19BIT0346

Perumalla Sasank 19BIT0338

Keerthi Yasasvi 19BIT0335

Table of Contents

[Abstract 6](#_Toc68564345)

[Ketwords 7](#_Toc68564346)

[Introduction 7](#_Toc68564347)

[Dataset - **ISIC2017: Skin Lesion Analysis** 7](#_Toc68564348)

[Abstract of Dataset 7](#_Toc68564349)

[Background 8](#_Toc68564350)

[Melanoma 8](#_Toc68564351)

[Dermoscopy 8](#_Toc68564352)

[About the Diseases 8](#_Toc68564353)

[Motivation 9](#_Toc68564354)

[General Architecture 9](#_Toc68564355)

[Summary on the General Arhitecture Processes 10](#_Toc68564356)

[Data Selection 10](#_Toc68564357)

[Exploratory Data Analysis 10](#_Toc68564358)

[Checking the Types of Data 10](#_Toc68564359)

[Finding the Outliers 10](#_Toc68564360)

[Data Visualization 10](#_Toc68564361)

[Data Pre-Processing 11](#_Toc68564362)

[Splitting the Data 11](#_Toc68564363)

[Checking for Missing Values 11](#_Toc68564364)

[Checking Categorical Features 11](#_Toc68564365)

[Normalizing Dataset 11](#_Toc68564366)

[Feature Transformation 11](#_Toc68564367)

[Why do we need Feature Transformation and Scaling? 11](#_Toc68564368)

[Feature Transformations used in the Models 12](#_Toc68564369)

[MaxAbs Scalar 12](#_Toc68564370)

[Robust Scalar 12](#_Toc68564371)

[Unit Vector Scaler 12](#_Toc68564372)

[Feature Selection 12](#_Toc68564373)

[Principal Component Analysis 13](#_Toc68564374)

[Linear Discriminant Analysis 13](#_Toc68564375)

[Model Selection 13](#_Toc68564376)

[Model Training 13](#_Toc68564377)

[Model Evaluation 13](#_Toc68564378)

[Comparative study on various subtitles: 14](#_Toc68564379)

[Literature Survey 14](#_Toc68564380)

[Grouping 22](#_Toc68564381)

[On the Basis of the Data used 22](#_Toc68564382)

[On the Basis of the Data Analysis Technique 22](#_Toc68564383)

[ABCDE Rule 23](#_Toc68564384)

[Data Augmentation 23](#_Toc68564385)

[Deep Learning Pipeline 25](#_Toc68564386)

[Machine Learning Is Burgeoning 25](#_Toc68564387)

[Getting Familiar with ML Pipelines 25](#_Toc68564388)

[Challenges Associated with ML Pipelines 26](#_Toc68564389)

[Data Quality 26](#_Toc68564390)

[Data Reliability 26](#_Toc68564391)

[Data Accessibility 26](#_Toc68564392)

[The Value Is In the Metadata 27](#_Toc68564393)

[Object Storage for ML Pipelines 28](#_Toc68564394)

[StyleGANs 29](#_Toc68564395)

[How StyleGAN works 29](#_Toc68564396)

[Mapping Network 29](#_Toc68564397)

[Style Modules (AdaIN) 30](#_Toc68564398)

[Removing traditional input 31](#_Toc68564399)

[Stochastic variation 31](#_Toc68564400)

[Style mixing 32](#_Toc68564401)

[Truncation trick in W 32](#_Toc68564402)

[Fine-tuning 33](#_Toc68564403)

[Results 33](#_Toc68564404)

[Feature disentanglement 34](#_Toc68564405)

[Implementation Details 34](#_Toc68564406)

[Conclusion 34](#_Toc68564407)

[GLCM 34](#_Toc68564408)

[SVM 35](#_Toc68564409)

[On the basis of Data Pre-processing Technique 35](#_Toc68564410)

[Dull Razor Method (common) 35](#_Toc68564411)

[Pros 35](#_Toc68564412)

[Cons 36](#_Toc68564413)

[Transfer Learning (Most Common) 36](#_Toc68564414)

[Using a Pre-Trained Model 36](#_Toc68564415)

[Feature Extraction 36](#_Toc68564416)

[Pros 36](#_Toc68564417)

[Adam 36](#_Toc68564418)

[Pros 36](#_Toc68564419)

[Cons 37](#_Toc68564420)

[Properties of Adam 37](#_Toc68564421)

[Problems with Adam 37](#_Toc68564422)

[RMSprop 37](#_Toc68564423)

[DCNN 38](#_Toc68564424)

[Pros 38](#_Toc68564425)

[Cons 39](#_Toc68564426)

[One-Hot Encoding 39](#_Toc68564427)

[Pros 39](#_Toc68564428)

[Cons 39](#_Toc68564429)

[Noise Removal 39](#_Toc68564430)

[On the Basis of Feature Selection Techniques 39](#_Toc68564431)

[Feature Selection Methods: 39](#_Toc68564432)

[1. Univariate Selection 39](#_Toc68564433)

[2. Feature Importance 40](#_Toc68564434)

[3. Correlation Matrix with Heatmap 40](#_Toc68564435)

[Ways of Feature Selection 40](#_Toc68564436)

[Types of Feature Selection 40](#_Toc68564437)

[Pearson Correlation 41](#_Toc68564438)

[Pros 41](#_Toc68564439)

[Cons 41](#_Toc68564440)

[Decision Tree 41](#_Toc68564441)

[Pros 41](#_Toc68564442)

[Cons 41](#_Toc68564443)

[Ridge Regression 42](#_Toc68564444)

[Pros 42](#_Toc68564445)

[Cons 42](#_Toc68564446)

[Lasso 42](#_Toc68564447)

[Pros 42](#_Toc68564448)

[Cons 42](#_Toc68564449)

[Classification 42](#_Toc68564450)

[Difference Between SoftMax Function and Sigmoid Function 43](#_Toc68564451)

[SoftMax classifier 43](#_Toc68564452)

[Properties of SoftMax Function 43](#_Toc68564453)

[SoftMax Function Usage 43](#_Toc68564454)

[Sigmoid Function 43](#_Toc68564455)

[Properties of Sigmoid Function 43](#_Toc68564456)

[Sigmoid Function Usage 43](#_Toc68564457)

[SoftMax Function Vs Sigmoid Function 44](#_Toc68564458)

[Binary Classifier 44](#_Toc68564459)

[Dense Layer 44](#_Toc68564460)

[The Problem with the Perceptron 44](#_Toc68564461)

[The solution was to add more neurons 44](#_Toc68564462)

[What is multilayer perceptron? 44](#_Toc68564463)

[GLCM 45](#_Toc68564464)

[On the Basis of Model Training Method 45](#_Toc68564465)

[MCNN 45](#_Toc68564466)

[CNN (Most Common) 46](#_Toc68564467)

[Pros 46](#_Toc68564468)

[Cons 46](#_Toc68564469)

[GoogLeNet/Inception 46](#_Toc68564470)

[AlexNet 47](#_Toc68564471)

[PROS 47](#_Toc68564472)

[Why does Dropout work? 48](#_Toc68564473)

[VGG16 48](#_Toc68564474)

[Xception 49](#_Toc68564475)

[What is an XCeption network? 49](#_Toc68564476)

[How does XCeption work? 49](#_Toc68564477)

[Implementation of the XCeption 49](#_Toc68564478)

[The limits of convolutions: 49](#_Toc68564479)

[SGD 49](#_Toc68564480)

[Pros 49](#_Toc68564481)

[Cons 49](#_Toc68564482)

[On the Basis of Model Evaluation Method 49](#_Toc68564483)

[Future Scope 50](#_Toc68564484)

[Conclusions 50](#_Toc68564485)

[References 51](#_Toc68564486)

[COncepts & Information 51](#_Toc68564487)

[Research 51](#_Toc68564488)

Skin Cancer Detection – General Research

General Research Study

# Abstract

The project is a CNN trained model which can predict whether the patient has a suffering from Cancer or not by checking the images of the infected areas on the body. The model has been trained on a variety of images through which it predicts the required.

In this project, the image file of the patient is upload into a software, which is GUI-based interface, developed with the help of Tkinter, and it consists of the model saved as a file and the software uses that to analyze the image and give the prediction which can help doctors to start with the medication way faster instead of waiting for the laboratory reports for the confirmation.

So basically,

* Skin cancer is an abnormal growth of skin cells. Most skin cancers are caused by exposure to ultraviolet (UV) light. When the skin is not protected, UV rays from sunlight or tanning beds can damage and alter skin's DNA that leads to the cancer.
* Deep learning model has been built to classify and identify the binary diagnostic group of melanocytic images obtained through dermoscopy.
* Based on the model, disease detection through dermal cell images has been investigated, and classifications on dermal cell images have been performed.

# Ketwords

* Model
* Convolutional Neural Network
* Cancer
* Malignant
* Benign
* Detection
* Tkinter
* Software
* Analysis
* Uploading
* Training
* Testing
* Validation
* Prediction
* MobileNet
* Inception
* Xception

# Introduction

Skin cancer is the most common form of cancer in the United States. The two most common types of skin cancer—basal cell and squamous cell carcinomas—are highly curable, but can be disfiguring and costly to treat. Melanoma, the third most common skin cancer, is more dangerous and causes the most deaths. The majority of cases of these three types of skin cancer are caused by overexposure to ultraviolet (UV) light.

The most common warning sign of skin cancer is a change on the skin, typically a new [mole](https://my.clevelandclinic.org/health/articles/moles), a new skin lesion or a change in an existing mole.

* Basal cell carcinoma may appear as a small, smooth, pearly, or waxy bump on the face, or neck, or as a flat, pink/red- or brown-coloured lesion on the trunk, arms or legs.
* Squamous cell carcinoma can appear as a firm, red nodule, or as a rough, scaly, flat lesion that may itch, bleed and become crusty. Both basal cell and squamous cell cancers mainly occur on areas of the skin frequently exposed to the sun, but can occur anywhere.
* Melanoma usually appears as a pigmented patch or bump. It may resemble a normal mole, but usually has a more irregular appearance.

# Dataset - **ISIC2017: Skin Lesion Analysis**

## Abstract of Dataset

The goal of the challenge is to help participants develop image analysis tools to enable the automated diagnosis of melanoma from dermoscopic images. Image analysis of skin lesions is composed of 3 parts:

* Part 1: Lesion Segmentation
* Part 2: Detection and Localization of Visual Dermoscopic Features/Patterns
* Part 3: Disease Classification

This challenge provides training data (150 images) and blind held-out test dataset (~600 images) will be provided for participants to generate and submit automated results.

## Background

### Melanoma

Skin cancer is a major public health problem, with over 5 million newly diagnosed cases in the United States each year. Melanoma is the deadliest form of skin cancer, responsible for over 9,000 deaths each year.

### Dermoscopy

As pigmented lesions occurring on the surface of the skin, melanoma is amenable to early detection by expert visual inspection. It is also amenable to automated detection with image analysis. Given the widespread availability of high-resolution cameras, algorithms that can improve our ability to screen and detect troublesome lesions can be of great value. As a result, many centres have begun their own research efforts on automated analysis. However, a centralized, coordinated, and comparative effort across institutions has yet to be implemented.

Dermoscopy is an imaging technique that eliminates the surface reflection of skin. By removing surface reflection, visualization of deeper levels of skin is enhanced. Prior research has shown that when used by expert dermatologists, dermoscopy provides improved diagnostic accuracy, in comparison to standard photography. As inexpensive consumer dermatoscope attachments for smart phones are beginning to reach the market, the opportunity for automated dermoscopic assessment algorithms to positively influence patient care increases.

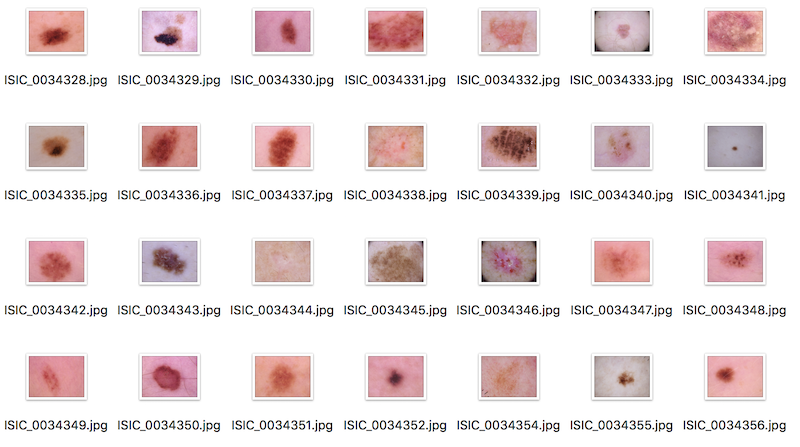


Figure : Sample Dataset Images

# About the Diseases

Skin cancer is the most prevalent type of cancer. Melanoma, specifically, is responsible for 75% of skin cancer deaths, despite being the least common skin cancer. The American Cancer Society estimates over 100,000 new melanoma cases will be diagnosed in 2020. It's also expected that almost 7,000 people will die from the disease. As with other cancers, early and accurate detection—potentially aided by data science—can make treatment more effective.

Currently, dermatologists evaluate every one of a patient's moles to identify outlier lesions or “ugly ducklings” that are most likely to be melanoma. Existing AI approaches have not adequately considered this clinical frame of reference. Dermatologists could enhance their diagnostic accuracy if detection algorithms take into account “contextual” images within the same patient to determine which images represent a melanoma. If successful, classifiers would be more accurate and could better support dermatological clinic work.

As the leading healthcare organization for informatics in medical imaging, the [Society for Imaging Informatics in Medicine (SIIM)](https://siim.org/)'s mission is to advance medical imaging informatics through education, research, and innovation in a multi-disciplinary community. SIIM is joined by the [International Skin Imaging Collaboration (ISIC)](https://www.isic-archive.com/), an international effort to improve melanoma diagnosis. The ISIC Archive contains the largest publicly available collection of quality-controlled dermoscopic images of skin lesions.

In this competition, you’ll identify melanoma in images of skin lesions. In particular, you’ll use images within the same patient and determine which are likely to represent a melanoma. Using patient-level contextual information may help the development of image analysis tools, which could better support clinical dermatologists.

Melanoma is a deadly disease, but if caught early, most melanomas can be cured with minor surgery. Image analysis tools that automate the diagnosis of melanoma will improve dermatologists' diagnostic accuracy. Better detection of melanoma has the opportunity to positively impact millions of people.

# Motivation

* Disease detection plays a very important role in the process of diagnosis. Therefore, the motivation lies in accurate classification and detection of the diseases based on medical images.
* The main aim is to minimize the chances of error that might happen due to the doctor's misjudgement.
* Developing a system that will not only help in detecting the diseases efficiently but will also save the time and effort of the medical practitioners.
* This will also save the patients from running to the doctor to get their medical reports verified.

# General Architecture

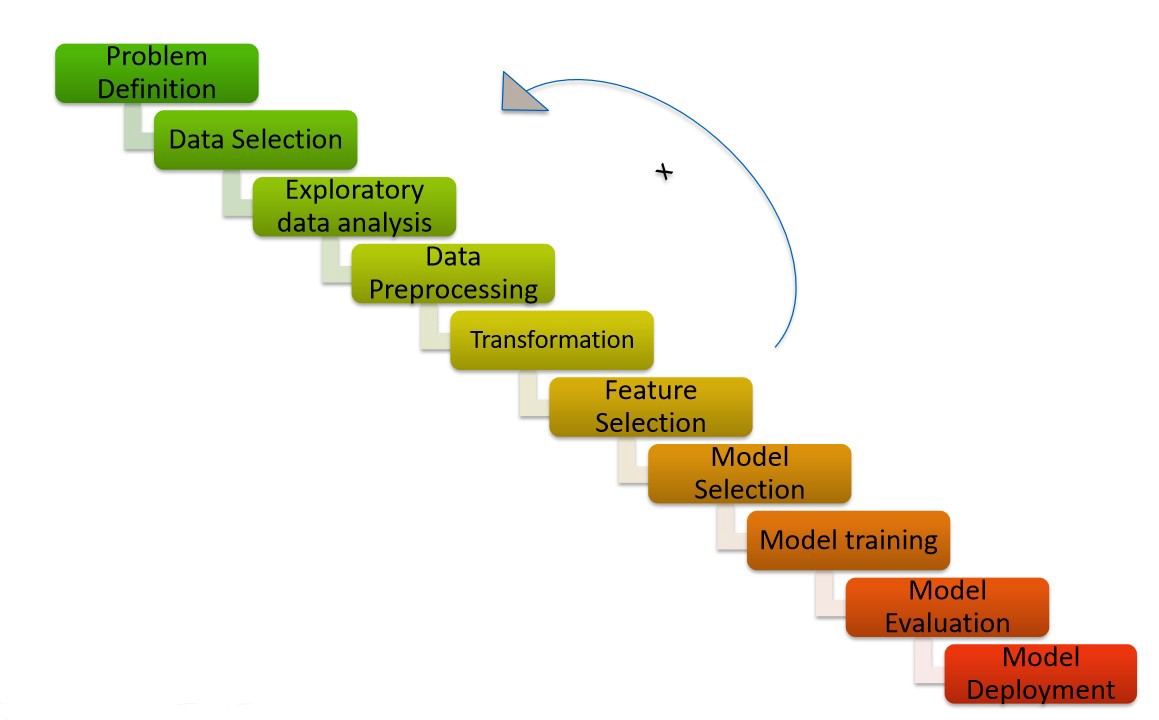
(Analytics Vidya, 2019)

Figure : General Architecture

## Summary on the General Arhitecture Processes

### Data Selection

The data selection is done on the basis of the amount of data and the type of data which is available. The data could be in the form of images, scanned reports, tf records, dcm files, etc. Based on the algorithm selected and the kind of data available, the model will be built. The data obtained can be collected via survey or from public databases.

### Exploratory Data Analysis

#### Checking the Types of Data

To find what all columns it contains, of what types and if they contain any value in it or not, with the help of functions.

#### Finding the Outliers

An outlier is a piece of data that is an abnormal distance from the other points. In other words, it’s data that lies outside the other values in the set. These points can be found by plotting the entire data.

#### Data Visualization

Using this data, we can:

* Analyse individual columns
* Check for missing values
* Perform variable analysis
* Check condition column
* Check quality column
* Plot between different variables and targets

### Data Pre-Processing

#### Splitting the Data

It is very important because your model needs to be evaluated before it has been deployed. And that evaluation needs to be done on unseen data because when it is deployed, all incoming data is unseen. The main idea behind the train test split is to convert the original data into training and testing data. For most of the articles which have been analysed, the data has been split into training and testing data in the range of ration of 75% to 25% (this is an approximate range provided considering all the research papers analysed), respectively.

#### Checking for Missing Values

If your data set is full of NaNs and garbage values, then surely your model will perform on garbage too. So, taking care of such values is important and it mostly done using the Simple Imputer method.

#### Checking Categorical Features

The most common methods used for this are:

* Label Encoding
* One-hot Encoding

#### Normalizing Dataset

The models mostly use the following methods of normalization for the data:

* Standard Scaler
* Variance before Standard Scaler
* Variance after Standard Scaler

### Feature Transformation

Feature pre-processing is one of the most crucial steps in building a Machine learning model. Too few features and your model won’t have much to learn from. Too many features and we might be feeding unnecessary information to the model. Not only this, but the values in each of the features need to be considered as well.

#### Why do we need Feature Transformation and Scaling?

Oftentimes, we have datasets in which different columns have different units – like one column can be in kilograms, while another column can be in centimetres. Furthermore, we can have columns like income which can range from 20,000 to 100,000, and even more; while an age column which can range from 0 to 100(at the most). Thus, Income is about 1,000 times larger than age.

But how can we be sure that the model treats both these variables equally? When we feed these features to the model as is, there is every chance that the income will influence the result more due to its larger value. But this doesn’t necessarily mean it is more important as a predictor. So, to give importance to both Age, and Income, we need feature scaling.

#### Feature Transformations used in the Models

##### MaxAbs Scalar

In simplest terms, the MaxAbs scaler takes the absolute maximum value of each column and divides each value in the column by the maximum value.

Thus, it first takes the absolute value of each value in the column and then takes the maximum value out of those. This operation scales the data between the range [-1, 1].

##### Robust Scalar

If you have noticed in the scalers we used so far, each of them was using values like the mean, maximum and minimum values of the columns. All these values are sensitive to outliers. If there are too many outliers in the data, they will influence the mean and the max value or the min value. Thus, even if we scale this data using the above methods, we cannot guarantee a balanced data with a normal distribution.

The Robust Scaler, as the name suggests is not sensitive to outliers. This scaler-

removes the median from the data

scales the data by the Interquartile Range (IQR)

Are you familiar with the Inter-Quartile Range? It is nothing but the difference between the first and third quartile of the variable.

##### Unit Vector Scaler

Normalization is the process of scaling individual samples to have unit norm. The most interesting part is that unlike the other scalers which work on the individual column values, the Normalizer works on the rows! Each row of the data frame with at least one non-zero component is rescaled independently of other samples so that its norm (l1, l2, or inf) equals one.

Just like MinMax Scaler, the Normalizer also converts the values between 0 and 1, and between -1 to 1 when there are negative values in our data.

However, there is a difference in the way it does so.

* If we are using L1 norm, the values in each column are converted so that the sum of their absolute values along the row = 1
* If we are using L2 norm, the values in each column are first squared and added so that the sum of their absolute values along the row = 1

### Feature Selection

Feature selection is a process where you automatically select those features in your data that contribute most to the prediction variable or output in which you are interested.

Having irrelevant features in your data can decrease the accuracy of many models, especially linear algorithms like linear and logistic regression.

Three benefits of performing feature selection before modelling your data are:

* **Reduces Overfitting**: Less redundant data means less opportunity to make decisions based on noise.
* **Improves Accuracy**: Less misleading data means modelling accuracy improves.
* **Reduces Training Time**: Less data means that algorithms train faster.

The methods used for Feature Selection are:

* Principal Component Analysis
* Linear Discriminant Analysis

#### Principal Component Analysis

Principal Components Analysis is a way of recognizing patterns in data, and expressing the data in such a manner as to focus their differences and similarities. Subsequently patterns in data may be complex to discover in data of high dimension, where the luxury of graphical representation is not available, PCA is a powerful tool for analysing data. The key advantage of PCA is that once we have found the patterns in the data, and you compress the data, i.e., by reducing the number of dimensions, without much loss of information. This technique used in image compression.

#### Linear Discriminant Analysis

There are many possible techniques for classification of data. Principal Component Analysis and Linear Discriminant Analysis are commonly used techniques for dimensionality reduction and data classification. Linear Discriminant Analysis easily handles the case where the within-class frequencies are unequal and their performances have been examined on randomly generated test data. This technique maximizes the proportion of between-class variance to the within-class variance in any specific data set in that way promising maximal separability. The Linear Discriminant Analysis is used for classification issues such as speech recognition. The key difference among LDA and PCA is that PCA perform feature classification and LDA works for data classification. The shape and location of the inventive data sets changes when transformed to a different space in PCA, on the other hand LDA doesn’t change the location but only attempts to offer more class separability and induce a decision region among the given classes. This technique also supports to better recognize the distribution of the feature data.

### Model Selection

Some of the models which were deployed are:

* Simple Convolutional Neural Network Models
* Transfer Learning Models
* Ensemble Models
* Simple K-Means Model
* Generative Automotive Networks

### Model Training

The models deployed one of the following training techniques:

* Infected Area Detection
* Image Classification
* Instance Segmentation

### Model Evaluation

Some of the most common evaluation methods are:

* Accuracy
* Sensitivity
* Specificity
* Recall
* Precision
* F-measure

Some of the rare evaluation methods used are:

* CNN with/without Data Augmentation
* ResNeXt WSL
* ABCD Rule
* GOPS
* L1D Miss Rate
* XGBoost
* GoogLeNet/ResNet/AlexNet/VGGNet Error Rate

# Comparative study on various subtitles:

## Literature Survey

The Literature Survey done for the accomplishment of the project is:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Authors &Year | Methodology or Techniques used | Advantages | Issues | Metrics used | Pros | Cons |
| May-20 | CNN, AlexNet, ResNet-18, VGG16, SVM, Black-hat filter, Inpaint Algorithm, Median Filter, Otsu’s Methodology | SVM Accuracy = 86.21%, ResNet Accuracy = 87% | Accuracy Original Data = 80%, Accuracy Augmented Data = 98.61% | ReLU, CNN with Data Augmentation = 88.87%, CNN without Data Augmentation = 78.96% | Good Accuracy | Small Dataset |
| 2020 | CNN, Inception-v3, Keras, TensorFlow, DCNN, Leaky ReLU, Adamax optimizer, TPR is similar to the positive predictive value | 0.86 AUROC for BKL | 0.78 AUROC for MEL | Accuracy | N/A | N/A |
| 2020 | MVSM classifier, CNN, feature extraction, GLCM, SVM, ABCD | dataset which consists of eight different classes is compressed into 800 images and applied, the accuracy achieved is about 96.25%. | accuracy is lowered if minute amounts of foreign elements are found on the sample | Accuracy | Eight Classes help specify the disease for specific medication | High accuracy on a very small specific training dataset |
| 2020 | CNN SENet154, WSL, Adam, weighted loss-entropy | Efficient Architecture | Not much improved with ensemble strategy | EfficientNet, SENet (T1 = 67.2%, T2 = 70.0%), ResNeXt WSL (T1 = 65.9%, T2 = 68.1%) | Transfer Learning | Not implemented properly with small dataset, parameter tuning required |
| 2020 | GLCM, HOG, GAC | Feature extraction for early detection | Not enough/adequate dataset | ABCD Rule, SVM Classifier, Accuracy, Sensitivity, Specificity using KNN | characterize the texture of an image by calculating how often pairs of pixels with specific values and in a specified spatial relationship occur in an image | high dimensionality of the matrix and the high correlation of the features |
| 2019 | Multiclass SVM, AlexNet, ReLU | Accuracy – 94.016% | Model used is a pre-trained model, robust | GOPS, L1D miss rate | Excellent feature extraction | There are many scenarios in which there are multiple categories to which points belong, but a given point can belong to multiple categories. In its most basic form, this problem decomposes trivially into a set of unlinked binary problems |
| 2019 | CNN, pooling layers, dense network, SVM | AlexNet, VGG16, ResNet-18 | Deep Network but Small Dataset – 3000 images | Accuracy = 74% | N/A | The model does not predict properly. |
| Apr-19 | CNN, pooling layer, dense network | Accuracy – 89.5% | Time consuming | Accuracy = 89.5%, Recall = 0.84, Specificity, Precision = 0.8325, F-measure = 0.8325 | Images have been taken randomly for better training and for wider input category of features. | The model may not predict that well on the given but may be able to identify the features more accurately. |
| Mar-19 | CNN, Inception V2 Net, K-means Cluster, Max-pooling, Sonification Algorithms | No. of K-means Epochs = 100 | F2-score +ve Prediction = 59.9%, High Sensitivity, Low Specificity | F2-score = 81.8%, Sensitivity = 91.7%, Specificity = 41.8%, Precision = 57.3% | Technology improves accuracy  of skin cancer diagnosis and might assist physicians to diagnose  skin cancer and bypass dermoscopy-related experience factors, time  constraints, physical inconvenience of acquiring images | no a priori technology which can identify  whether a suspicious lesion is mild, moderate or severely dysplastic |
| 2019 | CNN, McNemar Test, ResNet50, Bonferroni Correction | MATLAB | Small dataset (11,444), training may be inefficient, class imbalance | Accuracy = 100%, detection rate = 100% | The prediction may be correct most of the time. | The model might have overfit on the data so it might predict properly on a specific kind of data which might be fed to it. |
| 2018-2019 | CNN, VGG16, ImageNet | Accuracy – 92.5% | F1-score = 0.77, VGG16 Accuracy = 78% | Random Forest = 65.9%, XGBoost = 65.15%, SVM = 65.86%, ReLU, Sigmoid | Max-pooling fetches maximum pixel | Low F-score |
| 2018 | MatConvNet & GoogLeNet Inception V3 CNN, GoogLeNet, AlexNet, ResNet, VGGNet, Simple Majority Voting, SMP | 1.28 million NATURAL images 500 epochs, pre-trained models used, MatConvNet provides pre-trained CNN models and some functions to create and initialize new neural networks | Limited computational resources, time-consuming procedures | GoogLeNet Error Rate = 0.1, ResNet Error Rate= 0.02, AlexNet Error Rate = approx. 0.001, VGGNet Error Rate = approx. 0.001 | High Results for all the evaluation metrics | Too many evaluation metrics and parameters used |
| 2018 | CNN, ImageNet, AlexNet, VGG, GoogLeNet, ResNet | Big dataset | There is a risk of overfitting the neural network | Accuracy, Image classification | advantageous for the decision making of dermatologists | There is a risk of overfitting the neural network |
| 2020 | skin lesions using CNNs, Alexnet. deep learning with PNASNet-5 | large dataset with 4 classifications | Less accuracy | Good dataset | efficient in computing time, consume less memory | N/A |
| 2018 | CNN, AlexNet, deep learning, AlexNet, VGG, GoogLeNet, ResNet, Inception V3 | Public dataset and ISIB-2016,2017 | Time-consuming, and errors | Large variety, dataset | advantageous for the decision making of dermatologists | Time-consuming. In addition, errors and the loss of information in the first processing steps have a very strong influence on the classification quality |
| 2020 | deep convolutional neural network, computer image analysis algorithms, CNN  , GoogLeNet Inception V3, AlexNet Deep Learning CNN | More variants from ISIC | Accuracy of less than 75% | Accuracy but small dataset | metric area under the curve of 99.77% was observed. | This would consume time and the patient may advance to later stage |
| 2019 | Machine learning, | Faster identification | Less accuracy | Accuracy, time | Got accuracy of min 85% | highly complex and expensive diagnosis with difficulties and subjectivity of human interpretation |
| 2017 | deep neural networks, Deep convolutional neural networks (CNNs) | dataset of 129,450 clinical images of Malignant and benign | less variants | CNN; melanoma;  skin cancer; image pre-processing | Large dataset and accuracy | Less variants of Malignant and benign |
| 2020 | deep learning, CNN, AlexNet and VGG-16, including VGG-Net, ResNet50, InceptionV3, Xception, and DenseNet121 | 70 % images were used for training and 30% used for testing | Less accuracy | Good dataset | 70 % images were used for training and 30% used for testing | accuracy of 65% to 75%, time consuming |
| 2017 | SVM, CNN, MobileNet | High accuracy | Small dataset | High accuracy | High accuracies in most cases | High Cost of pre trained models which are required |
| 2018 | CNN, GLCM, deep learning, CNN, ResNet, InceptionV2 | Trained on many variants | Small dataset | Accuracy but small dataset | Accuracy increases with bigger dataset | Bigger dataset required, Time consuming process |
| 2020 | CNN, SVM, KNN, Naïve Bayes, and neural network | 97.8 % of Accuracy | High rate of overfitting and misidentification | accuracy | obtained is 97.8 % of Accuracy and 0.94 Area under Curve using SVM classifiers and additionally the Sensitivity obtained was 86.2 % and Specificity obtained was 85 % using KNN. | High rate of overfitting and misidentification |
| 2017 | GANs, CNN, AlexNet, StyleGANs, InceptionV3-StyleGANs, ResNet50- StyleGANs, VGG16BN- StyleGANs | size of 600×600 as input dataset, | sets the weight coefficient w in the SoftMax loss function | Accuracy | Model Automatically learns the feature representations required for the corresponding detection or classification tasks through the dataset, and has a good performance in many applications | Proposed DCGANs, which have clear structural constraints and indicate that they have weak credibility for unsupervised learning and that they are generalized most of the time. |
| 2018 | CNN, SciKit, Keras, TensorFlow, OpenCV, ReLU | 90% accuracy, Convolution maintains the spatial interrelation of the pixels, values of the pixels ranging from 0 - 255 i.e., 256 pixels. | Rectified Linear Unit is a non-linear operation. ReLU acts on an elementary level. | Accuracy | With a large dataset accuracy can be increased to 90% | Averages and accuracy of 70% on standard publicly available dataset and time consuming |
| 2019 | AlexNet, Ordinary CNN, VGG-16, LIN, Inception-v3, and ResNet. Lévy flight, ReLU | size of input images in the input is considered 28×28 pixel. | doesn’t give the best global solution | Accuracy | 97% accuracy | Imbalance of training and testing dataset |
| 2019 | STM32, ROC, CNN, ReLU, NLSC | Accuracy - 99%,  F1-Score - 99% | computing and index loss, poor lesion skin discrimination specificity | Accuracy | This methodology, based on “visual” investigation by the dermatologist and/or oncologist, has the advantage of not being invasive and quite easy to perform | Several approaches proposed in scientific literature increase sensitivity of the pipelines to the disadvantage of ‘specificity’ or vice versa. |
| 2019 | CNN, Feature Extraction, HSV format | Accuracy of 98%. for melanoma skin cancer detection and 93% for melanoma type, TPR of 94.25%, FPR of 3.56%, and EP of 4%, average accuracy of 91.66% | high error rates, 25.6% Caucasian error and 23.2 Xanthous error, validation loss of 57.56% | Accuracy | achieved TPR of 94.25%, FPR of 3.56%, and EP of 4% | With a small dataset an accuracy of 74.76% and validation loss of 57.56% is acquired |
| 2019 | CNN, keras, AlexNet, VGG16, SGD optimiser, | trained on more than 126k images, higher image augmentation (24x) and image resolution (1k), the same performances can be achieved using less than 5000 images, no impact of image resize filters | Experiments at 277x277 pixel resolution, Experiments without transfer learning | Accuracy | 98% specificity | 73% sensitivity and Jaccard Index of 0.69. |
| 2019 | CNN, grad-CAM, TensorFlow, Inception-ResNet-v2, DenseNet121, Xception | consists of 150,223 clinical images from 543 different skin diseases, achieved an accuracy of 87.25 ± 2.24% on the dermoscopic images for four common skin diseases, including SK, BCC, psoriasis and melanocytic nevus. | highest average precision (77.0%) | Accuracy | achieved 92.9%, 89.2%, and 84.3% recalls for the LE, BCC, and SK, respectively, | mean recall and precision reached 77.0% and 70.8%. |
| 2019 | CNN, keras, TensorFlow, Inception V3, ResNet50, VGG16, MobileNet and Inception, Resnet | 7 types of skin lesion diseases identification namely: Benign Keratosis, Dermatofibroma, Vascular Lesion, Melanoma, Melanocytic Nevus, Basal Cell Carcinoma and Actinic Keratosis., Inception, Resnet achieved an average accuracy of 91%, Accuracies of 90 and 91% | low F1 score | Accuracy | This model is advantageous over feed-forward neural networks which cannot understand translation invariance | Low F1 Scores |

## Grouping

### On the Basis of the Data used

The groups for this which can be formed are:

* Dataset is classified into classes and stored separately beforehand
  + Data is classified as:
    - Malignant or Benign
    - 7 classes of Skin Cancer
    - 3 classes of keratosis
* Dataset is not classified into classes and instead CSV file is provided

### On the Basis of the Data Analysis Technique

Some the analysis methods used are:

* ABCDE Rule
* Data Augmentation
* Normal scanning and cropping and photoshop
* CNN (common)
* Deep Learning Pipeline (Morphological Analysis) (common)
* Biopsy, Histopathological Testing, Dermoscopic Assessment
* StyleGANS (common)
* GLCM, SVM

Let’s see each of them one-by-one.

#### ABCDE Rule

Regular self-skin examinations by patients have been shown to decrease the depth of melanomas at the time of diagnosis and facilitate a lower risk of melanoma when coupled with regular visits with a physician.[1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4345927/#B1) Accordingly, the American Academy of Dermatology has sought to improve patient awareness of melanoma through its SPOT Skin Cancer campaign.[2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4345927/#B2) At our institution, we counsel patients to perform regular self-skin examinations in an effort to expedite identification and appropriate treatment of suspicious melanocytic lesions. To aid patients in identifying possible melanoma, we often recommend the use of either the “ABCDE rule” or the “ugly duckling sign.”

The ABCDE rule, first introduced in 1985 as the ABCD rule[3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4345927/#B3) and then expanded in 2004 to the ABCDE rule,[4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4345927/#B4) encompasses several clinical features of melanoma, including Asymmetry, Border irregularity, Color variation (both intralesional color variation as well as a color that is different from the patient’s other nevi), Diameter greater than 6mm, and Evolving (a new or changing lesion). An alternative to the ABCDE rule, the “ugly duckling sign,” was later developed to address the limitations of the ABCDE rule.[5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4345927/#B5) It states that the “spot” that is unlike the others is often the most suspect for malignancy. Their limitations aside, both of these methods provide an easy, general framework from which patients may work in order to identify potential melanomas.

Due to time constraints or personal preference, dermatologists often choose to counsel patients on one of these guidelines while neglecting the other. However, the shortcoming of using only one of these criteria is that the ABCDE and “ugly duckling” rules are complementary to one another and are not necessarily equal alternatives to one another. That is, the ABCDE rule does not account for the added screening benefits of using the “ugly duckling rule,” and vice versa.

Therefore, we propose a modification to the ABCDE rule in order to combine the benefits of this rule with that of the “ugly duckling sign.” The addition of the letter “F,” which stands for “Funny looking,” nicely incorporates the “ugly duckling rule” into one, unified tool—the ABCDEF rule—for patients to use when evaluating their moles at home. The “Funny looking” mole, like the “ugly duckling sign,” implies that there is an overall gestalt one can have about a spot being suspicious.

The addition of another letter to the currently widely used ABCDE rule does not seem to overly complicate the mnemonic for patients. Indeed, in our experience in counseling patients about the ABCDEF rule, we have found that the inclusion of a somewhat humorous and unexpected “capstone” to the mnemonic has been very well received. The ABCDEF rule improves upon the previous iteration of the mnemonic and may prove to be a useful tool for patients and physicians alike in identifying worrisome melanocytic lesions.

#### Data Augmentation

The prediction accuracy of the Supervised Deep Learning models is largely reliant on the amount and the diversity of data available during training. The relation between deep learning models and amount of training data required is analogous to that of the relation between rocket engines (deep learning models) and the huge amount of fuel (huge amounts of data) required for the rocket to complete its mission (success of the deep learning model).

DL models trained to achieve high performance on complex tasks generally have a large number of hidden neurons.  As the number of hidden neurons increases, the number of trainable parameters also increases. The number of parameters in the State-of-the-art Computer Vision models such as RESNET (60M) and Inception-V3 (24M) is of the order of ten million.

In Natural Language Processing models such as BERT (340M), it is of the order of a hundred million. These deep learning models trained to perform complex tasks such as object detection or language translation with high accuracy have a large number of tunable parameters. They need a large amount of data to learn the values for a large number of parameters during the training phase.

In simple terms, the amount of data required is proportional to the number of learnable parameters in the model. The number of parameters is proportional to the complexity of the task.

Oftentimes, when working on specific complex tasks such as classifying a weed from a crop, or identifying the novelty of a patient, it is very hard to get large amounts of data required to train the models. Though transfer learning techniques could be used to great effect, the challenges involved in making a pre-trained model to work for specific tasks are tough.

Another way to deal with the problem of limited data is to apply different transformations on the available data to synthesize new data. This approach of synthesizing new data from the available data is referred to as ‘Data Augmentation’.

Data augmentation can be used to address both the requirements, the diversity of the training data, and the amount of data. Besides these two, augmented data can also be used to address the class imbalance problem in classification tasks.

The questions that come to the mind are; Does data augmentation work? Do we really get better performance from the models when we use augmented data for training? Table 1 shows a few case studies indicating the effect of data augmentation on model performance for different applications.

**Table 1. model performance for different applications with and without data augmentation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **Performance without Augmentation** | **Performance with Augmentation** | **Augmentation method** |
| Image classification | 57% | 78.6% | Simple Image based |
| Image classification | 57% | 85.7% | GAN based |
| NMT | 11 BLEU | 13.9 BLEU | Translation data augmentation |
| Text classification | 79% | 87% | Easy Data Augmentation |

The answer to the questions is an assured and cautious ‘yes’: assured since several case studies indicate that the performance improves and cautious since different augmentation techniques used affect the scale of improvement differently.

#### Deep Learning Pipeline

Machine learning (ML) pipelines consist of several steps to train a model, but the term ‘pipeline’ is misleading as it implies a one-way flow of data.  Instead, machine learning pipelines are cyclical and iterative as every step is repeated to continuously improve the accuracy of the model and achieve a successful algorithm.  To build better machine learning models, and get the most value from them, accessible, scalable and durable storage solutions are imperative, paving the way for on-premises object storage.

##### **Machine Learning Is Burgeoning**

Welcome to the era of [digital transformation](https://blog.westerndigital.com/state-of-object-storage-digital-transformation/), where data has become a modern-day currency.  Tremendous value and intelligence is being extracted from large, captured datasets (big data) that has led to actionable insights through today’s analytics.  Data analytics is uncovering trends, patterns and associations, new connections and precise predictions that are helping businesses achieve better outcomes.  It’s not just about storing data any longer, but capturing, preserving, accessing and transforming it to take advantage of its possibilities and the value it can deliver.  The goal for ML is simple: make faster and more predictive decisions.

Many of today’s ML models are ‘trained’ neural networks capable of executing a specific task or providing insights derived from ‘what happened’ to ‘what will likely happen’ (predictive analysis).  These models are complex and are never completed, but rather, through the repetition of mathematical or computational procedures, are applied to the previous result and improved upon each time to get closer approximations to ‘solving the problem.’  Data scientists want more captured data to provide the fuel to train the ML models.

Machine learning use globally is burgeoning [and its respective market is expected to grow in revenue](https://www.prnewswire.com/news-releases/machine-learning-market---global-forecast-to-2022---market-overview--industry-trends-300531729.html) to $8.81 billion by 2022, at a 44.1 percent CAGR.  Businesses are rethinking their data strategies to include machine learning capabilities, not only to increase competitiveness, but also to create infrastructures that help enable [data to live forever](https://www.hgst.com/products/systems).

##### Getting Familiar with ML Pipelines

A machine learning pipeline is used to help automate machine learning workflows.  They operate by enabling a sequence of data to be transformed and correlated together in a model that can be tested and evaluated to achieve an outcome, whether positive or negative.

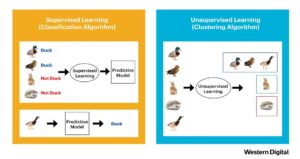
[](https://2s7gjr373w3x22jf92z99mgm5w-wpengine.netdna-ssl.com/wp-content/uploads/2018/09/WD_2.jpg)

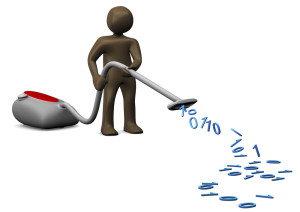
Figure 1. Supervised and unsupervised learning can be useful in machine learning models (Courtesy: Western Digital)

There are generally two types of machine learning approaches (Figure 1).  The first is supervised learning, where a model is built and datasets are provided to solve a particular problem using classification algorithms, and is the most common use of machine learning.  The second approach is unsupervised learning, where a model is built to discover structures within given datasets.  The initial data captured is not necessarily labeled so clustering algorithms are used to group the unlabeled data together.

##### Challenges Associated with ML Pipelines

In creating machine learning pipelines, there are challenges that data scientists face, but the most prevalent ones fall into three categories: Data Quality, Data Reliability and Data Accessibility.

###### Data Quality

If the quality of the data is not accurate, complete, reliable or robust, there is no need to run machine learning models because the outcomes will be wrong.  This places a very high priority on data reliability because data scientists want as much quality data as possible to build and train their ML models.  The more high-quality data they get, the more accurate and better their outcomes.[](https://2s7gjr373w3x22jf92z99mgm5w-wpengine.netdna-ssl.com/wp-content/uploads/2015/11/shutterstock_data_cleaning_TunedIn-by-Westend61.jpg)

###### Data Reliability

Data that will be used to run machine learning pipelines will be generated from a variety of sources.  In order to determine the reliability of the data, collaboration amongst those who have data outcomes is required so that the data itself, its source of generation, and those who assessed the analysis are trusted and viable.  As such, implementing a repository for the data outcomes that serves as a single source of truth is required.  This enables the source data to reside in a single repository that data scientists and analysts can access quickly and use as reference whenever they need to present results.

The single source repository also enables machine learning to be run from various locations within a data center versus administrators having to physically carry or port the ML model to whatever location the analysis is being conducted.  This avoids duplicate and varying versions of data, and makes sure that the analytical teams, from multiple organizations, are always working with the most recent and reliable data.

###### Data Accessibility

Before any machine learning model is run, the data itself must be accessible, requiring consolidation, cleansing and curation (where more qualitative data is added such as data sources, authorized users, project name, and time-stamp references).  As a result of data curation, metadata is updated with the new tags.

Since data can be captured from years or even decades past, it can reside on many forms of storage media ranging from hard drives to memory sticks to hard copies in shoe boxes.  In many cases, it resides on tape that deteriorates over time, can be difficult to find and may require obsolete readers to extract the data.  To analyze big data in the modern world requires that it be captured and stored on reliable media, not only for immediate access, but to validate that it is of the highest integrity and accuracy possible.  As such, enterprise SSDs and HDDs are used extensively to consolidate and store data for machine learning applications.

Cleansing is equally important as it removes irrelevant and redundant data during the pre-analysis stage.  Doing this will not only save compute power, and associated time and costs, but will significantly increase the accuracy and comprehensibility of the ML model itself.  Feature selection is a process used to cleanse unnecessary data by selecting attributes (or features) that are the most relevant in creating a predictive model.  Feature extraction (Figure 2) is an alternate process that extracts existing features (and their associated data transformations) into new formats that not only describe variances within the data, but reduce the amount of information that is required to represent the ML model.

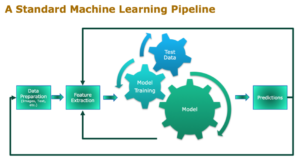
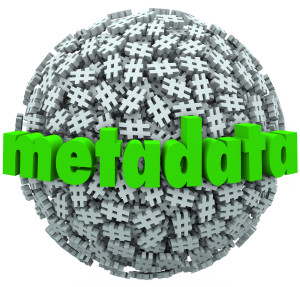
[](https://2s7gjr373w3x22jf92z99mgm5w-wpengine.netdna-ssl.com/wp-content/uploads/2018/09/WD_3.png)

Figure 2: Feature extraction is critical for machine learning pipelines (Courtesy: Western Digital)

Once the data is cleansed, it can be aggregated with other cleansed data.  From a data scientist’s perspective, this is heaven since massive quantities of stored data are needed to successfully run and train analytical models.  Storing data in today’s data-centric world is no longer about just recovering datasets, but rather preserving them and being able to access them easily using search and index techniques.  As such, data curating is part of the cleansing process but worth a separate callout as it requires reference marks as to where the data originated, as well as other forms of identification that differentiate it from other data, so that the information is reliable and trusted.

##### The Value Is In the Metadata

For data scientists and analysts who strive to obtain good outcomes from big data and improve their results over time is really about the metadata.  Metadata extraction and the discovered correlations between metadata insights are the foundation of ML models.  Once a model is sufficiently trained, it can be put into production to deliver faster determinations.  In a traditional file-based network-attached storage (NAS) architecture, directories are used to tag data and must be traversed each time that it needs to be accessed.  So many directories to traverse through in a hierarchical scheme makes it difficult to find files and access them quickly.  But more importantly, the file-based approach has little to no information about the data stored that can help in analysis, or simplify management, or even support the ever-increasing amounts of data at scale.

When a business or operation is at scale is the time that the IT department needs to look at new storage solutions that are affordable, can help keep data forever (for analysis and ML training) and most importantly, easily scalable.  Object storage has made tremendous inroads and is an architecture that manages data as objects (versus traditional block- or file-based approaches), and an exceptional option for storing unstructured data at petabyte scale.  Unlike file-based storage that manages data in a folder hierarchy, or block-based storage that manages disk sectors collectively as blocks, object storage manages data as objects.[](https://2s7gjr373w3x22jf92z99mgm5w-wpengine.netdna-ssl.com/wp-content/uploads/2015/12/shutterstock_metadta_iQocept.jpg)

In an object storage platform, the totality of the data, be it a document, audio or video file, image or photo, or other unstructured data, is stored as a single object.  Metadata resides with the captured data and provides descriptive information about the object and the data itself.  This eliminates the need for a hierarchical structure and simplifies access by placing everything in a flat address space (or single namespace).  The unique identifier assigned to each object makes it easier to index and retrieve data, or find a specific object.

Since metadata resides with captured data, users can tag as many data points as they want, and tag and find groups of objects much faster than file- or block-based storage options.  Object storage also enables versioning — a very important feature of ML pipelines because of the repetitiveness in refining algorithms.  Leveraging this unique feature for object storage, data scientists can version their data such that they or their collaborators can reproduce the results later.  The versioning feature helps to shorten research time, obtain desired results faster, enable reproducible machine learning pipelines and validate data reliability.  And since many users pay for storage per petabyte, one person can manage more petabytes being grouped as objects, resulting in lower total cost of ownership (TCO), especially relating to manpower and power consumption.

##### Object Storage for ML Pipelines

Machine learning gets better over time as more data points are collected and the true value occurs when different data assets from a variety of sources are correlated together.  The act of correlating these new data formats streaming into the data center is quite a challenge as it’s not just about the sheer capacity of data, but more about the disparate data formats and the set of applications that need to access them.  Businesses are now focusing on consolidating their assets into a single petabyte scale-out storage architecture.  On-premises object storage or cloud storage systems serve a great purpose for these environments as they are designed to scale and support custom data formats.

[](https://2s7gjr373w3x22jf92z99mgm5w-wpengine.netdna-ssl.com/wp-content/uploads/2018/09/WD_4.jpg)

Machine learning transforms businesses through data analytics and the insights it delivers (Courtesy: Western Digital)

With data scientists and analysts playing more prominent roles in mapping the statistical significance of key problems, and translate it quickly for business implementation, they also strive to improve their results.  They want to store everything locally because their research is local and not in a public cloud as the time it takes to download an abundance of ML content can be extraordinary.  And they want immediate access to improve their algorithm and re-run the analysis – repeating as necessary so that better comparisons can be made to the original results.

With GPUs residing next to the data on the compute side, results can be produced faster and the technology won’t be blocked from analytical processing, but rather, enabled!  Every step in the ML process is cyclical and iterative as algorithms are being updated, analysis is being reprocessed, more data is being accumulated, and the end result is either improved or worsened.  Once the computer learns, further tests can be taken to see if the results are accurate and whether the analysis needs to be re-run.

#### StyleGANs

Generative Adversarial Networks (GAN) are a relatively new concept in Machine Learning, [introduced](https://arxiv.org/abs/1406.2661) for the first time in 2014. Their goal is to synthesize artificial samples, such as images, that are indistinguishable from authentic images. A common example of a GAN application is to generate artificial face images by learning from a dataset of celebrity faces. While GAN images became more realistic over time, one of their main challenges is controlling their output, i.e. changing specific features such pose, face shape and hair style in an image of a face.

A new paper by NVIDIA, A Style-Based Generator Architecture for GANs ([StyleGAN](https://arxiv.org/abs/1812.04948)), presents a novel model which addresses this challenge. StyleGAN generates the artificial image gradually, starting from a very low resolution and continuing to a high resolution (1024×1024). By modifying the input of each level separately, it controls the visual features that are expressed in that level, from coarse features (pose, face shape) to fine details (hair color), without affecting other levels.

This technique not only allows for a better understanding of the generated output, but also produces state-of-the-art results - high-res images that look more authentic than previously generated images.

##### How StyleGAN works

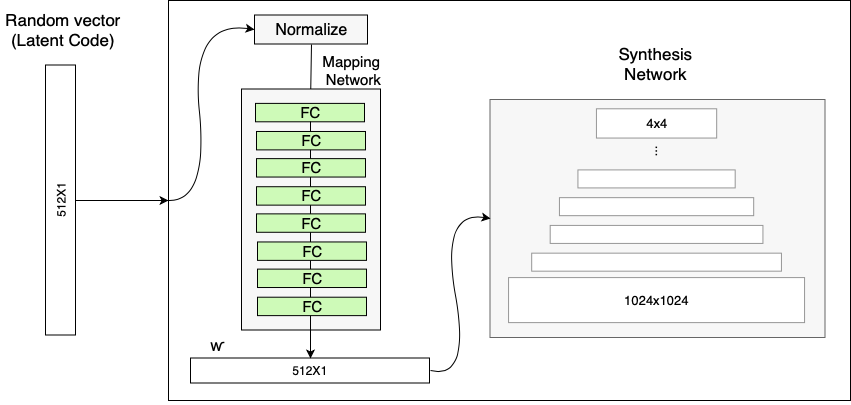
The StyleGAN paper offers an upgraded version of ProGAN’s image generator, with a focus on the generator network. The authors observe that a potential benefit of the ProGAN progressive layers is their ability to control different visual features of the image, if utilized properly. The lower the layer (and the resolution), the coarser the features it affects. The paper divides the features into three types:

1. Coarse - resolution of up to 82 - affects pose, general hair style, face shape, etc
2. Middle - resolution of 162 to 322 - affects finer facial features, hair style, eyes open/closed, etc.
3. Fine - resolution of 642 to 10242 - affects color scheme (eye, hair and skin) and micro features.

The new generator includes several additions to the ProGAN’s generators:

###### Mapping Network

The Mapping Network’s goal is to encode the input vector into an intermediate vector whose different elements control different visual features. This is a non-trivial process since the ability to control visual features with the input vector is limited, as it must follow the probability density of the training data. For example, if images of people with black hair are more common in the dataset, then more input values will be mapped to that feature. As a result, the model isn’t capable of mapping parts of the input (elements in the vector) to features, a phenomenon called features entanglement. However, by using another neural network the model can generate a vector that doesn’t have to follow the training data distribution and can reduce the correlation between features.  
The Mapping Network consists of 8 fully connected layers and its output ⱳ is of the same size as the input layer (512×1).

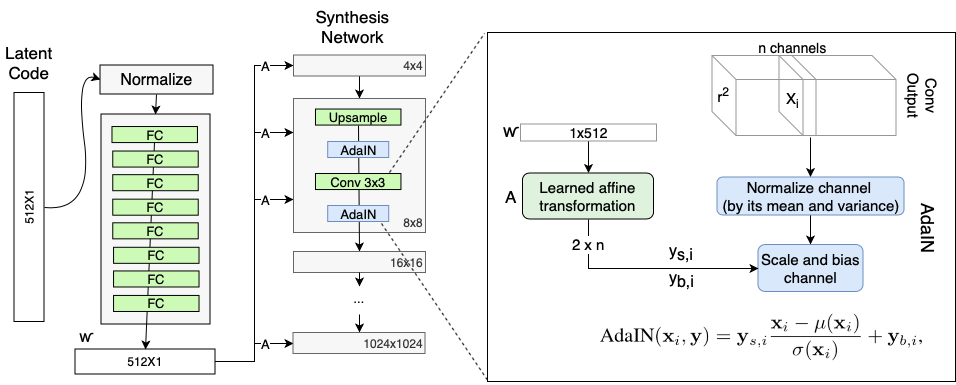


The generator with the Mapping Network (in addition to the ProGAN synthesis network)

###### Style Modules (AdaIN)

The [AdaIN](https://arxiv.org/abs/1703.06868) (Adaptive Instance Normalization) module transfers the encoded information ⱳ, created by the Mapping Network, into the generated image. The module is added to each resolution level of the Synthesis Network and defines the visual expression of the features in that level:

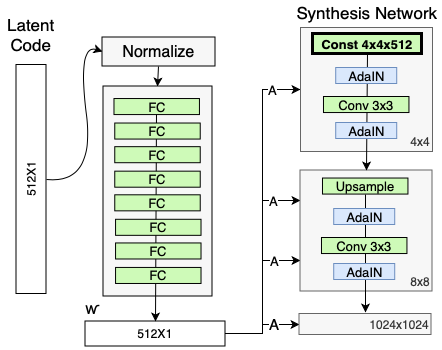
1. Each channel of the convolution layer output is first normalized to make sure the scaling and shifting of step 3 have the expected effect.
2. The intermediate vector ⱳ is transformed using another fully-connected layer (marked as A) into a scale and bias for each channel.
3. The scale and bias vectors shift each channel of the convolution output, thereby defining the importance of each filter in the convolution. This tuning translates the information from ⱳ to a visual representation.



The generator’s Adaptive Instance Normalization (AdaIN)

###### Removing traditional input

Most models, and ProGAN among them, use the random input to create the initial image of the generator (i.e. the input of the 4×4 level). The StyleGAN team found that the image features are controlled by ⱳ and the AdaIN, and therefore the initial input can be omitted and replaced by constant values. Though the paper doesn’t explain why it improves performance, a safe assumption is that it reduces feature entanglement — it’s easier for the network to learn only using ⱳ without relying on the entangled input vector.

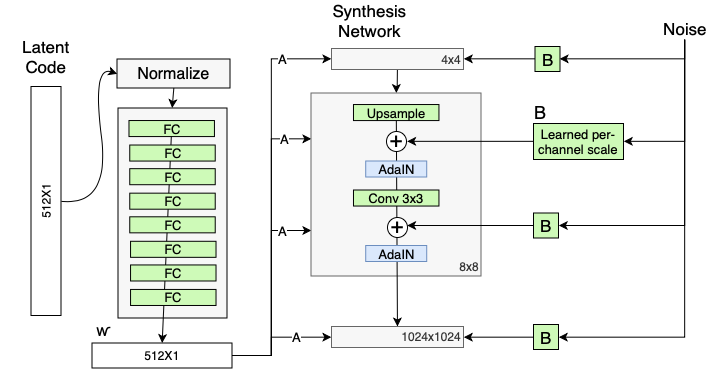


The Synthesis Network input is replaced with a constant input

###### Stochastic variation

There are many aspects in people’s faces that are small and can be seen as stochastic, such as freckles, exact placement of hairs, wrinkles, features which make the image more realistic and increase the variety of outputs. The common method to insert these small features into GAN images is adding random noise to the input vector. However, in many cases it’s tricky to control the noise effect due to the features entanglement phenomenon that was described above, which leads to other features of the image being affected.

The noise in StyleGAN is added in a similar way to the AdaIN mechanism — A scaled noise is added to each channel before the AdaIN module and changes a bit the visual expression of the features of the resolution level it operates on.



Adding scaled noise to each resolution level of the synthesis network

###### Style mixing

The StyleGAN generator uses the intermediate vector in each level of the synthesis network, which might cause the network to learn that levels are correlated. To reduce the correlation, the model randomly selects two input vectors and generates the intermediate vector ⱳ for them. It then trains some of the levels with the first and switches (in a random point) to the other to train the rest of the levels. The random switch ensures that the network won’t learn and rely on a correlation between levels.

Though it doesn’t improve the model performance on all datasets, this concept has a very interesting side effect — its ability to combine multiple images in a coherent way (as shown in the video below). The model generates two images A and B and then combines them by taking low-level features from A and the rest of the features from B.

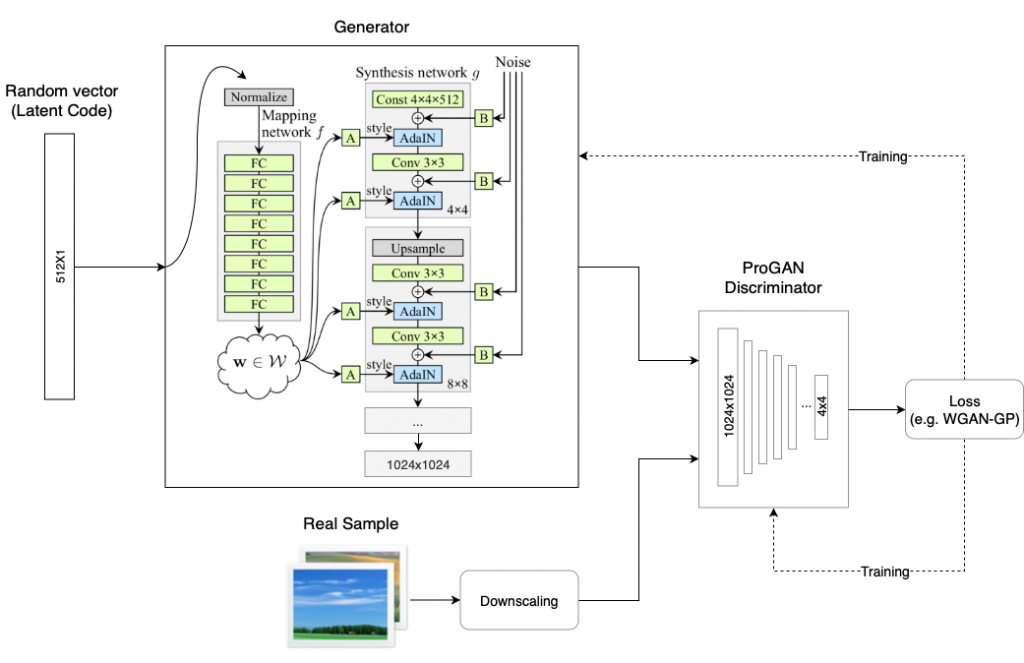
###### Truncation trick in W

One of the challenges in generative models is dealing with areas that are poorly represented in the training data. The generator isn’t able to learn them and create images that resemble them (and instead creates bad-looking images). To avoid generating poor images, StyleGAN truncates the intermediate vector ⱳ, forcing it to stay close to the “average” intermediate vector.

After training the model, an “average” ⱳavg is produced by selecting many random inputs; generating their intermediate vectors with the mapping network; and calculating the mean of these vectors. When generating new images, instead of using Mapping Network output directly, ⱳ is transformed into ⱳ\_new=ⱳ\_avg+𝞧(ⱳ -ⱳ\_avg), where the value of 𝞧 defines how far the image can be from the “average” image (and how diverse the output can be). Interestingly, by using a different 𝞧 for each level, before the affine transformation block, the model can control how far from average each set of features is, as shown in the video below.

###### Fine-tuning

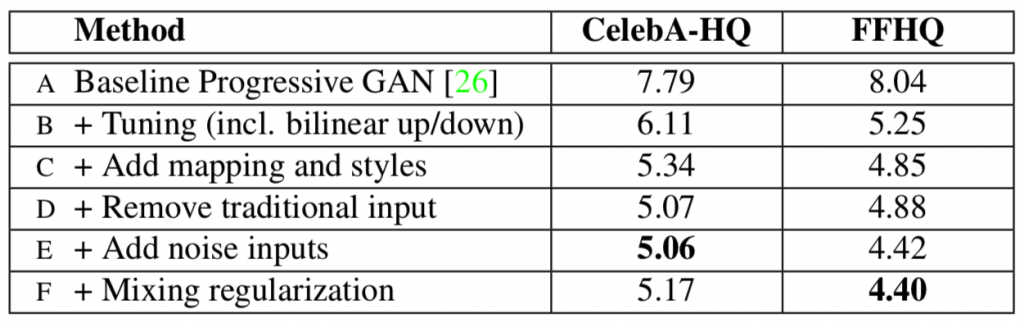
Additional improvement of StyleGAN upon ProGAN was updating several network hyperparameters, such as training duration and loss function, and replacing the up/downscaling from nearest neighbors to bilinear sampling. Though this step is significant for the model performance, it’s less innovative and therefore won’t be described here in detail (Appendix C in the paper).



An overview of StyleGAN

##### Results

The paper presents state-of-the-art results on two datasets — CelebA-HQ, which consists of images of celebrities, and a new dataset Flickr-Faces-HQ (FFHQ), which consists of images of “regular” people and is more diversified. The chart below shows the Frèchet inception distance ([FID](https://arxiv.org/abs/1706.08500)) score of different configurations of the model.



The performance (FID score) of the model in different configurations compared to ProGAN. The lower score the better the model (Source: [StyleGAN](https://arxiv.org/abs/1812.04948))

In addition to these results, the paper shows that the model isn’t tailored only to faces by presenting its results on two other datasets of bedroom images and car images.

##### Feature disentanglement

In order to make the discussion regarding feature separation more quantitative, the paper presents two novel ways to measure feature disentanglement:

1. Perceptual path length — measure the difference between consecutive images (their VGG16 embeddings) when interpolating between two random inputs. Drastic changes mean that multiple features have changed together and that they might be entangled.
2. Linear separability — the ability to classify inputs into binary classes, such as male and female. The better the classification the more separable the features.

By comparing these metrics for the input vector z and the intermediate vector ⱳ, the authors show that features in ⱳ are significantly more separable. These metrics also show the benefit of selecting 8 layers in the Mapping Network in comparison to 1 or 2 layers.

##### Implementation Details

StyleGAN was trained on the CelebA-HQ and FFHQ datasets for one week using 8 Tesla V100 GPUs. It is implemented in TensorFlow and will be open-sourced.

##### Conclusion

StyleGAN is a ground-breaking paper that not only produces high-quality and realistic images but also allows for superior control and understanding of generated images, making it even easier than before to generate believable fake images. The techniques presented in StyleGAN, especially the Mapping Network and the Adaptive Normalization (AdaIN), will likely be the basis for many future innovations in GANs.

#### GLCM

The GLCM texture method is a way of extracting second order statistical texture features from gray-level images (in this case images obtained from a Scanning Electron Microscope (SEM)). The statistical measures from this matrix are extracted from how often pairs of pixels with a specified value occur together in a spatial area. Textural features are perceived patterns, variations, and randomness across image pixels (journal). Imaging devices, such as computer-aided diagnostics rely on quantitative measurements of textures as input that aim to characterize biological or disease status. It is unclear if these features behave similarly to unconventional imaging biomarkers such as tumour volume etc.

#### SVM

Support vector machines so called as SVM is a ***supervised learning algorithm*** which can be used for classification and regression problems as support vector classification (SVC) and support vector regression (SVR). It is used for smaller dataset as it takes too long to process. In this set, we will be focusing on SVC. SVM is based on the idea of finding a hyperplane that best separates the features into different domains.

### On the basis of Data Pre-processing Technique

Some of the pre-processing methods can be grouped as:

#### Dull Razor Method (common)

Dull Razor software [10] is a medical imaging software for hair removal. In this, a special type of filter is used, which replaces hair pixels by neighboring pixels. It improves classification results Hair removal is done using Dull Razor software. The dermoscopic images may contain hairs. These hairs somehow will give erroneous classification. So, it is desirable to do the hair removal before proceeding to further steps.

##### Pros

* One of the main advantages of the simulation that the ground truth pixel values and hair mask are known

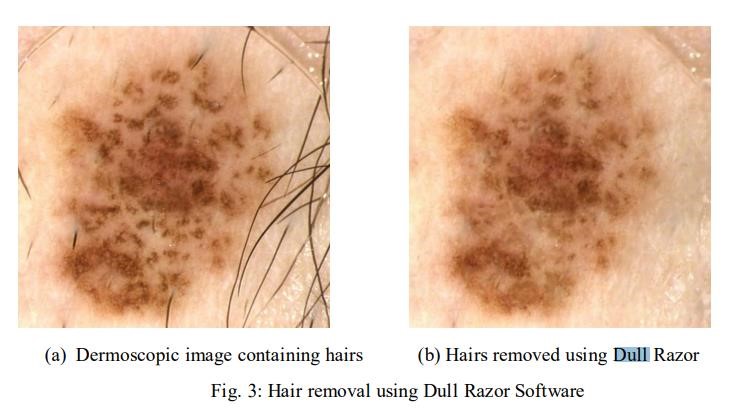


Figure : Sample Image for Example

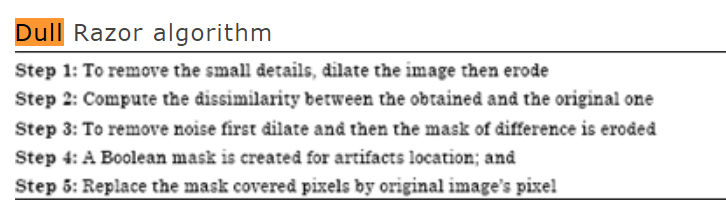


Figure : Dull Razor Algorithm

##### Cons

With modern techniques the primary disadvantage is that it’s a very technical domain understood by very few people and thus it is difficult to execute on. Importantly, every modern ML model shipped by a major company in the space (i.e., Google, FB, OpenAI) relies on transfer learning at its core.

#### Transfer Learning (Most Common)

Transfer learning in a modern context requires a very large, general dataset. It’s extremely important that you not build your base model from domain-specific data. If you build your base model on domain-specific data (god forbid data in your training or test sets) then your entire experiment is invalid. Even if you don’t present labels to the base model, by seeing the “correct” base data you’ve given it information that it shouldn’t have access to.

Transfer learning is the reuse of a pre-trained model on a new problem. It's currently very popular in deep learning because it can train deep neural networks with comparatively little data.

In transfer learning, the knowledge of an already trained machine learning model is applied to a different but related problem.

With transfer learning, we basically try to exploit what has been learned in one task to improve generalization in another. We transfer the weights that a network has learned at "task A" to a new "task B."

The general idea is to use the knowledge a model has learned from a task with a lot of available labelled training data in a new task that doesn't have much data. Instead of starting the learning process from scratch, we start with patterns learned from solving a related task.

##### Using a Pre-Trained Model

The second approach is to use an already pre-trained model. There are a lot of these models out there, so make sure to do a little research. How many layers to reuse and how many to retrain depends on the problem.

#### Feature Extraction

Another approach is to use deep learning to discover the best representation of your problem, which means finding the most important features. This approach is also known as representation learning, and can often result in a much better performance than can be obtained with hand-designed representation.

##### Pros

Transfer learning has several benefits, but the main advantages are saving training time, better performance of neural networks (in most cases), and not needing a lot of data. Additionally, training time is reduced because it can sometimes take days or even weeks to train a deep neural network from scratch on a complex task.

* Hyper-parameters for Image Augmentation and CNNs (common)
  + CNN to Binary Output

#### Adam

##### Pros

* Easy to implement.
* Quite computationally efficient.
* Requires little memory space.
* Good for non-stationary objectives.
* Works well on problems with noisy or sparse gradients.
* Works well with large data sets and large parameters.

##### Cons

There are few disadvantages as the Adam optimizer tends to converge faster, but other algorithms like the Stochastic gradient descent focus on the datapoints and generalize in a better manner. Thus, the performance depends on the type of data being provided and the speed/generalization trade-off.

##### Properties of Adam

Here I list some of the properties of Adam, for proof that these are true refer to the paper:

1. Actual step size taken by the Adam in each iteration is approximately bounded the step size hyper-parameter. This property adds intuitive understanding to previous unintuitive learning rate hyper-parameter.
2. Step size of Adam update rule is invariant to the magnitude of the gradient, which helps a lot when going through areas with tiny gradients (such as saddle points or ravines). In these areas SGD struggles to quickly navigate through them.
3. Adam was designed to combine the advantages of Adagrad, which works well with sparse gradients, and RMSprop, which works well in on-line settings. Having both of these enables us to use Adam for broader range of tasks. Adam can also be looked at as the combination of RMSprop and SGD with momentum.

##### Problems with Adam

When Adam was first introduced, people got very excited about its power. Paper contained some very optimistic charts, showing huge performance gains in terms of speed of training

#### RMSprop

The RMSProp algorithm full form is called **Root Mean Square Prop**, which is an adaptive learning rate optimization algorithm proposed by Geoff Hinton.

​RMSProp has several advantages; for one, it is a very robust optimizer which has pseudo curvature information. Additionally, it can deal with stochastic objectives very nicely, making it applicable to mini batch **learning**. Works with gnumpy. (float or array\_like) Step rate of the optimizer.

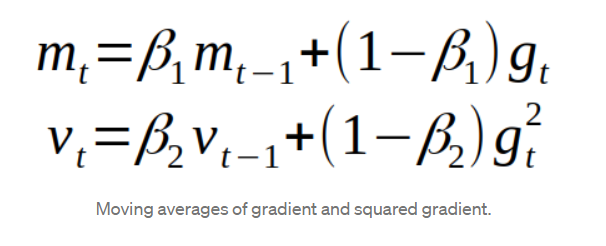


Figure : RMSProp Formulae

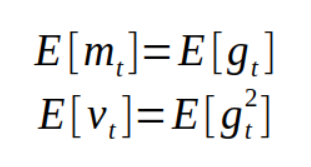


Figure : RMSProp Formulae

#### DCNN

A **deep convolutional neural network** (**DCNN**) consists of many neural network layers. Two different types of layers, convolutional and pooling, are typically alternated. The depth of each filter increases from left to right in the network. The last stage is typically made of one or more fully connected layers:

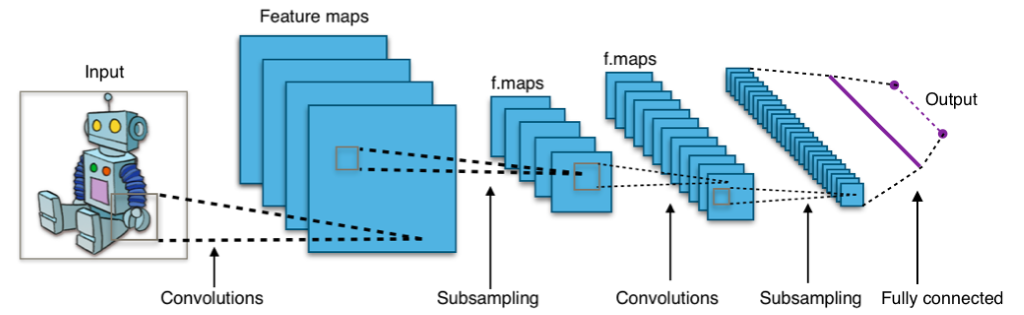


Figure : DCNN Architecture

There are three key intuitions beyond ConvNets:

* Local receptive fields
* Shared weights
* Pooling

##### Pros

CNNs is **weight sharing**. Let’s take an example to explain this. Say you have a one layered CNN with 10 filters of size 5x5. Now you can simply calculate parameters of such a CNN, it would be 5\*5\*10 weights and 10 biases i.e., **5\* 5\*10 + 10 = 260 parameters**. Now let’s take a simple one layered NN with 250 neurons, here the number of weight parameters depending on the size of images is ‘250 x K’ where size of the image is P X M and K = (P \*M). Additionally, you need ‘M’ biases. For the MNIST data as input to such a NN we will have (**250\*784+1 = 19601) parameters.**Clearly, CNN is more efficient in terms of memory and complexity.

In terms of performance, CNNs outperform NNs on conventional image recognition tasks and many other tasks. Look at the Inception model, Resnet50 and many others for instance.

For a completely new task / problem CNNs are very good **feature extractors.**This means that you can extract useful attributes from an already trained CNN with its trained weights by feeding your data on each level and tune the CNN a bit for the specific task.

##### Cons

* CNN do not encode the position and orientation of object
* Lack of ability to be spatially invariant to the input data
* ANN, SVM, Naïve-Bayes Algorithm

#### One-Hot Encoding

##### Pros

One-hot encoding ensures that machine learning does not assume that higher numbers are more important. For example, the value '8' is bigger than the value '1', but that does not make '8' more important than '1'.

##### Cons

The disadvantage is that for high cardinality, the feature space can really blow up quickly and you start fighting with the curse of dimensionality.

#### Noise Removal

Noise removal algorithm is the process of removing or reducing the noise from the image. The noise removal algorithms reduce or remove the visibility of noise by smoothing the entire image leaving areas near contrast boundaries.  
One of the most popular methods is wiener filter. In this work four types of noise (Gaussian noise, Salt & Pepper noise, Speckle noise and Poisson noise) is used and image de-noising performed for different noise by Mean filter, Median filter and Wiener filter. Further results have been compared for all noises.

The noise degrades performance of image processing algorithms in brain imaging. Image denoising methods are important image processing algorithms which are used to reduce the noise.

The noise degrades performance of image processing algorithms in brain imaging. Image denoising methods are important image processing algorithms which are used to reduce the noise:

* Segmentation
* Resize, Feature Extraction, Classification (Common)

### On the Basis of Feature Selection Techniques

Some of the feature selection methods can be grouped as:

#### Feature Selection Methods:

I will share 3 Feature selection techniques that are easy to use and also gives good results.

1. Univariate Selection
2. Feature Importance
3. Correlation Matrix with Heatmap

##### Univariate Selection

Statistical tests can be used to select those features that have the strongest relationship with the output variable. The scikit-learn library provides the [SelectKBest](http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html%22%20/l%20%22sklearn.feature_selection.SelectKBest%22%20/o%20%22http:/scikit-learn.org/stable/modules/generated/sklearn.feature_selection.selectkbest.html" \l "sklearn.feature_selection.selectkbest" \t "_blank) class that can be used with a suite of different statistical tests to select a specific number of features.

##### Feature Importance

You can get the feature importance of each feature of your dataset by using the feature importance property of the model.

Feature importance gives you a score for each feature of your data, the higher the score more important or relevant is the feature towards your output variable.

Feature importance is an inbuilt class that comes with Tree Based Classifiers, we will be using Extra Tree Classifier for extracting the top 10 features for the dataset.

##### Correlation Matrix with Heatmap

Correlation states how the features are related to each other or the target variable.

Correlation can be positive (increase in one value of feature increases the value of the target variable) or negative (increase in one value of feature decreases the value of the target variable)

Heatmap makes it easy to identify which features are most related to the target variable, we will plot heatmap of correlated features using the seaborn library.

​There are two main types of feature selection techniques: supervised and unsupervised, and supervised methods may be divided into wrapper, filter and intrinsic:

* Filter-based feature selection methods use statistical measures to score the correlation or dependence between input variables that can be filtered to choose the most relevant features.
* Statistical measures for feature selection must be carefully chosen based on the data type of the input variable and the output or response variable

#### Ways of Feature Selection

Select a subset of input features from the dataset, then:

* **Unsupervised**: Do not use the target variable (e.g., remove redundant variables).
  + Correlation
* **Supervised**: Use the target variable (e.g., remove irrelevant variables).
  + **Wrapper**: Search for well-performing subsets of features.
    - RFE
  + **Filter**: Select subsets of features based on their relationship with the target.
    - Statistical Methods
    - Feature Importance Methods
  + **Intrinsic**: Algorithms that perform automatic feature selection during training.
    - Decision Trees

#### Types of Feature Selection

There are three **types of feature selection**: Wrapper methods (forward, backward, and stepwise **selection**), Filter methods (ANOVA, Pearson correlation, variance thresholding), and Embedded methods (Lasso, Ridge, Decision Tree).

##### Pearson Correlation

###### Pros

**Pearson's Correlation Coefficient** helps you find out the relationship between two quantities. It gives you the measure of the strength of association between two variables. The value of **Pearson's Correlation Coefficient** can be between -1 to +1. 1 means that they are highly **correlated** and 0 means no **correlation**.

###### Cons

The **disadvantages** of the **Pearson** r **correlation** method are:

* It assumes that there is always o linear relationship between the variables which might not be the case at all times
* It can be easily misinterpreted as a high degree of **correlation** from large values of the **correlation coefficient**.

##### Decision Tree

Decision Tree is a very popular machine learning algorithm. Decision Tree solves the problem of machine learning by transforming the data into a tree representation. Each internal node of the tree representation denotes an attribute and each leaf node denotes a class label.

A decision tree algorithm can be used to solve both regression and classification problems.

###### Pros

* Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
* A decision tree does not require normalization of data.
* A decision tree does not require scaling of data as well.
* Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
* A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.

###### Cons

* A small change in the data can cause a large change in the structure of the decision tree causing instability.
* For a Decision tree sometimes, calculation can go far more complex compared to other algorithms.
* Decision tree often involves higher time to train the model.
* Decision tree training is relatively expensive as the complexity and time has taken are more.
* The Decision Tree algorithm is inadequate for applying regression and predicting continuous values.

##### Ridge Regression

Regularized methods such as Ridge Regression can be used to select only relevant features in the training dataset. The process of transforming a dataset in order to select only relevant features necessary for training is called dimensionality reduction.

###### Pros

We can use a regularized model to reduce the dimensionality of the training dataset. Dimensionality reduction is important because of three main reasons:

* **Prevents Overfitting**: A high-dimensional dataset having too many features can sometimes lead to overfitting (model captures both real and random effects).
* **Simplicity**: An over-complex model having too many features can be hard to interpret especially when features are correlated with each other.
* **Computational Efficiency**: A model trained on a lower dimensional dataset is computationally efficient (execution of algorithm requires less computational time).

###### Cons

* Regularization leads to dimensionality reduction, which means the machine learning model is built using a lower dimensional dataset. This generally leads to a high **bias error**.
* If regularization is performed before training the model, a perfect balance between **bias-variance trade-off** must be used.

##### Lasso

###### Pros

* As any regularization method, it can avoid overfitting. It can be applied even when number of features is larger than amount of data.
* It can do feature selection.
* It is fast in terms of inference and fitting.

###### Cons

* The model selected by lasso is not stable. For example, on different bootstrapped data, the feature selected can be very different.
* The model selection result is not intuitive to interpret: for example, why lasso select a feature?
* When there are highly correlated features, lasso may randomly select one of them of part of them. The result depends on the implementation. To improve, people introduced elastic net.
* Based on my experience, its prediction performance is usually worse than ridge regression in terms of MSE.

### Classification

* CNN, Pooling Layer (common)
  + Max pooling (most common)
  + Sum pooling
  + Average pooling
* Autoencoders
  + Stacked Deep Autoencoders
* No. of Hidden layers in Dense and Sparse network
* Multilayer Perceptron
* Algorithms
  + SGNN
  + Genetic
  + Skin Lesion Segmentation’

#### Difference Between SoftMax Function and Sigmoid Function

##### SoftMax classifier

**SoftMax** extends this idea into a multi-class world. That is, SoftMax assigns decimal probabilities to each class in a multi-class problem. Those decimal probabilities must add up to 1.0. This additional constraint helps training converge more quickly than it otherwise would.

Consider the following variants of SoftMax:

* **Full SoftMax** is the SoftMax we've been discussing; that is, SoftMax calculates a probability for every possible class.
* **Candidate sampling** means that SoftMax calculates a probability for all the positive labels but only for a random sample of negative labels. For example, if we are interested in determining whether an input image is a beagle or a bloodhound, we don't have to provide probabilities for every non-doggy example.

Full SoftMax is fairly cheap when the number of classes is small but becomes prohibitively expensive when the number of classes climbs. Candidate sampling can improve efficiency in problems having a large number of classes.

###### Properties of SoftMax Function

Below are the few properties of SoftMax function.

* The calculated probabilities will be in the range of 0 to 1.
* The sum of all the probabilities is equals to 1.

###### SoftMax Function Usage

* Used in multiple classification logistic regression model.
* In building neural networks SoftMax functions used in different layer level.

##### Sigmoid Function

###### Properties of Sigmoid Function

* The sigmoid function returns a real-valued output.
* The first derivative of the sigmoid function will be non-negative or non-positive.
  + **Non-Negative:** If a number is greater than or equal to zero.
  + **Non-Positive:** If a number is less than or equal to Zero.

###### Sigmoid Function Usage

* The Sigmoid function used for **binary classification** in logistic regression model.
* While creating artificial neuron’s sigmoid function used as the **activation function**.
* In statistics, the **sigmoid function graphs** are common as a cumulative distribution function

##### SoftMax Function Vs Sigmoid Function

While learning the logistic regression concepts, the primary confusion will be on the functions used for calculating the probabilities. As the calculated probabilities are used to predict the target class in [logistic regression model](https://dataaspirant.com/2017/03/02/how-logistic-regression-model-works/). The two principal functions we frequently hear are SoftMax and Sigmoid function.

Even though both the functions are same at the **functional level.** (Helping to predict the target class) many noticeable mathematical differences are playing the vital role in using the functions in deep learning and other fields of areas.

SoftMax function calculates the probabilities distribution of the event over ‘n’ different events. In general way of saying, this function will calculate the probabilities of each target class over all possible target classes. Later the calculated probabilities will be helpful for determining the target class for the given inputs.

The main advantage of using SoftMax is the output probabilities range. The range will **0 to 1**, and the sum of all the probabilities will be **equal to one**. If the SoftMax function used for multi-classification model it returns the probabilities of each class and the target class will have the high probability.

#### Binary Classifier

It is one of the most frequently used problems in machine learning. In simplest form the user tries to classify a unit into 1 of the 2 possible categories. For example, take the attributes of the fruits like color, peel texture, shape etc. A linear classifier that the perceptron is classified as is a classification algorithm, which depends on a linear predictor function to make the predictions and predictions are based on the union that includes weights and feature vector.

#### Dense Layer

Neural network dense layers (or fully connected layers) are the foundation of nearly all neural networks.

##### The Problem with the Perceptron

Neural networks come in many different variations these days, from convolutional and recurrent, to homogenous and heterogeneous, to linear and branching.

But the original neural networks were a single neuron: the perceptron. Perceptron’s showed some promise, but came up short when attempting to handle some of the simplest logical operations. Unfortunately, perceptron didn't have enough complexity to approximate many of the functions that neural networks can approximate today.

###### The solution was to add more neurons

What is multilayer perceptron?

The [perceptron](https://deepai.org/machine-learning-glossary-and-terms/perceptron) is very useful for [classifying](https://deepai.org/machine-learning-glossary-and-terms/classifier) data sets that are linearly separable.  They encounter serious limitations with data sets that do not conform to this pattern as discovered with the XOR problem.  The XOR problem shows that for any classification of four points that there exists a set that are not linearly separable.

The algorithm for the MLP is as follows:

1. Just as with the perceptron, the inputs are pushed forward through the MLP by taking the dot product of the input with the weights that exist between the input layer and the hidden layer (WH).  This dot product yields a value at the hidden layer. We do not push this value forward as we would with a perceptron though.
2. MLPs utilize [activation functions](https://deepai.org/machine-learning-glossary-and-terms/activation-function) at each of their calculated layers.  There are many activation functions to discuss: [rectified linear units](https://deepai.org/machine-learning-glossary-and-terms/rectified-linear-units) ([ReLU](https://deepai.org/machine-learning-glossary-and-terms/relu" \t "_blank)), [sigmoid function](https://deepai.org/machine-learning-glossary-and-terms/sigmoid-function), tanh.  Push the calculated output at the current layer through any of these activation functions.
3. Once the calculated output at the hidden layer has been pushed through the activation function, push it to the next layer in the MLP by taking the dot product with the corresponding weights.
4. Repeat steps two and three until the output layer is reached.
5. At the output layer, the calculations will either be used for a [backpropagation](https://deepai.org/machine-learning-glossary-and-terms/backpropagation) algorithm that corresponds to the activation function that was selected for the MLP (in the case of training) or a decision will be made based on the output (in the case of testing).

#### GLCM

The **GLCM functions** characterize the texture of an image by calculating how often pairs of pixels with specific values and in a specified spatial relationship occur in an image, creating a **GLCM**, and then extracting statistical measures from this matrix. GLCM set of features are based on second order statistics... they can be used to reflect, the overall average for degree of correlation between pairs of pixels in different aspects (in terms of homogeneity, uniformity...etc.). One of the main factors affects that affects the discrimination capabilities of GLCM is the separation distance between pixels... When you take the distance 1 it leads to reflect the degree of correlation between adjacent pixels (i.e., short range neighborhood connectivity). While, increasing the distance value leads to reflect the degree of correlation between distant pixels

### On the Basis of Model Training Method

Some of the training methods can be grouped as:

* MVSM
* Transfer Learning: (used as a group of 2-3 with CNN)
  + **Inception (V1, V3)**
  + ResNet
  + ResNet50
  + MobileNet
* Deep Pipeline

#### MCNN

Traditional approaches typically involve extracting discriminative features from the original time series using dynamic time warping (DTW) or shape transformation, based on which an off-the-shelf classifier can be applied. These methods are ad-hoc and separate the feature extraction part with the classification part, which limits their accuracy performance. Plus, most existing methods fail to take into account the fact that time series often have features at different time scales. To address these problems, we propose a novel end-to-end neural network model, Multi-Scale Convolutional Neural Networks (MCNN), which incorporates feature extraction and classification in a single framework. Leveraging a novel multi-branch layer and learnable convolutional layers, MCNN automatically extracts features at different scales and frequencies, leading to superior feature representation. MCNN is also computationally efficient, as it naturally leverages GPU computing. MCNN advances the state-of-the-art by achieving superior accuracy performance than other leading methods.

#### CNN (Most Common)

##### Pros

In terms of architecture, the key building block of CNN is the convolutional layer. According to a MathWorks post, a CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images. Since CNNs eliminate the need for manual feature extraction, one doesn’t need to select features required to classify the images. How CNN work is by extracting features directly from images and the key features are not pretrained; they are learned while the network trains on a collection of images, the post notes. It is the automated feature extraction that makes CNNs highly suited for and accurate for computer vision tasks such as object/image classification.

##### Cons

Convolutional neural networks like any neural network model are computationally expensive. But that is more of a drawback than a weakness. This can be overcome with better computing hardware such as GPUs and Neuromorphic chips.

#### GoogLeNet/Inception

While VGG achieves a phenomenal accuracy on ImageNet dataset, its deployment on even the most modest sized GPUs is a problem because of huge computational requirements, both in terms of memory and time. It becomes inefficient due to large width of convolutional layers.

For instance, a convolutional layer with 3X3 kernel size which takes 512 channels as input and outputs 512 channels, the order of calculations is 9X512X512.

In a convolutional operation at one location, every output channel (512 in the example above), is connected to every input channel, and so we call it a dense connection architecture. The GoogLeNet builds on the idea that most of the activations in a deep network are either unnecessary (value of zero) or redundant because of correlations between them. Therefore, the most efficient architecture of a deep network will have a sparse connection between the activations, which implies that all 512 output channels will not have a connection with all the 512 input channels. There are techniques to prune out such connections which would result in a sparse weight/connection. But kernels for sparse matrix multiplication are not optimized in BLAS or CuBlas (CUDA for GPU) packages which render them to be even slower than their dense counterparts.

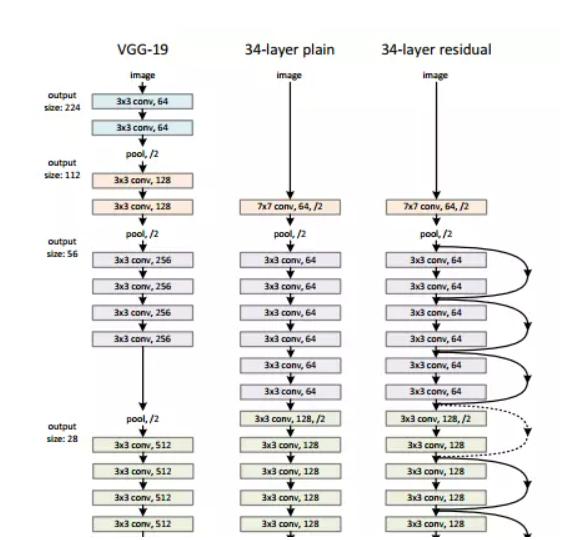


Figure : GoogLeNet/Inception Architecture

#### AlexNet

This architecture was one of the first deep networks to push ImageNet Classification accuracy by a significant stride in comparison to traditional methodologies. It is composed of 5 convolutional layers followed by 3 fully connected layers.

AlexNet, proposed by Alex Krizhevsky, **uses ReLU (Rectified Linear Unit) for the non-linear part, instead of a Tanh** **or Sigmoid function which was the earlier standard for traditional neural network**s. ReLU is given by

f(x) = max (0, x)

##### PROS

**The advantage of the ReLU over sigmoid is that it trains much faster** than the latter because the derivative of sigmoid becomes very small in the saturating region and therefore the updates to the weights almost vanish. This is called **vanishing gradient problem**.

In the network, ReLU layer is put after each and every convolutional and fully-connected layers (FC).

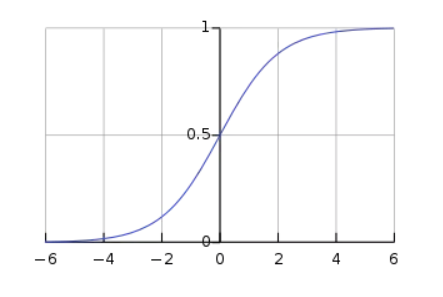


Figure : ReLU Function Graph

Another problem that this architecture solved was reducing the **over-fitting** by using a Dropout layer after every FC layer. Dropout layer has a probability, **(p)**, associated with it and is applied at every neuron of the response map separately. It randomly switches off the activation with the probability**p**

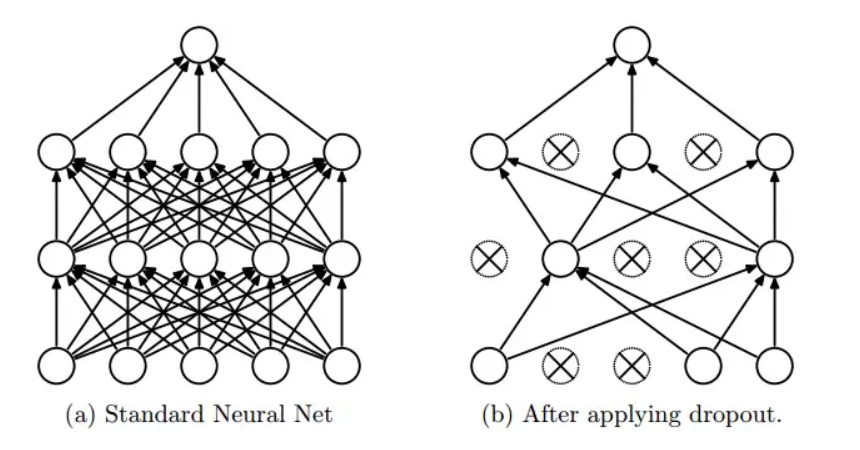


Figure : ReLU Function Net

#### Why does Dropout work?

The idea behind the dropout is similar to the model ensembles. Due to the dropout layer, different sets of neurons which are switched off, represent a different architecture and all these different architectures are trained in parallel with weight given to each subset and the summation of weights being one. For n neurons attached to Dropout, the number of subset architectures formed is 2^n. So, it amounts to prediction being averaged over these ensembles of models. This provides a structured model regularization which helps in avoiding the over-fitting. Another view of Dropout being helpful is that since neurons are randomly chosen, they tend to avoid developing co-adaptations among themselves thereby enabling them to develop meaningful features, independent of others.

#### VGG16

This architecture is from VGG group, Oxford. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3X3 kernel-sized filters one after another. With a given receptive field (the effective area size of input image on which output depends), multiple stacked smaller size kernel is better than the one with a larger size kernel because multiple non-linear layers increase the depth of the network which enables it to learn more complex features, and that too at a lower cost.

For example, three 3X3 filters on top of each other with stride 1 ha a receptive size of 7, but the number of parameters involved is 3\*(9C^2) in comparison to 49C^2 parameters of kernels with a size of 7. Here, it is assumed that the number of input and output channel of layers is C. Also, 3X3 kernels help in retaining finer level properties of the image. The network architecture is given in the table.

You can see that in VGG-D, there are blocks with same filter size applied multiple times to extract more complex and representative features. This concept of blocks/modules became a common theme in the networks after VGG.

The VGG convolutional layers are followed by 3 fully connected layers. The width of the network starts at a small value of 64 and increases by a factor of 2 after every sub-sampling/pooling layer. It achieves the top-5 accuracy of 92.3 % on ImageNet.

#### Xception

Xception is a deep convolutional neural network architecture that involves Depth wise Separable Convolutions.

##### What is an XCeption network?

The data first goes through the entry flow, then through the middle flow which is repeated eight times, and finally through the exit flow. Note that all Convolution and Separable Convolution layers are followed by batch normalization

##### How does XCeption work?

Xception is an efficient architecture that relies on two main points:

* Depth wise Separable Convolution
* Shortcuts between Convolution blocks as in ResNet

##### Implementation of the XCeption

Xception offers an architecture that is made of Depth wise Separable Convolution blocks + MaxPooling, all linked with shortcuts as in ResNet implementations.

The specificity of Xception is that the Depth wise Convolution is not followed by a Pointwise Convolution.

#### The limits of convolutions:

First of all, Convolution is a really expensive operation. To overcome the cost of such operations, depth wise separable convolutions have been introduced. They are themselves divided into 2 main steps:

* Depth wise Convolution
* Pointwise Convolution

#### SGD

##### Pros

* Memory requirement is less compared to the GD algorithm as derivative is computed taking only 1 point at once.

##### Cons

* The time required to complete 1 epoch is large compared to the GD algorithm.
* Takes a long time to converge.
* May stuck at local minima.

### On the Basis of Model Evaluation Method

Some of the evaluation methods can be grouped as:

* Most Used:
  + Precision & Recall
  + F-score
  + Accuracy (Most Common)
  + Random Forest, XGBoost, SVM
  + Specificity & Sensitivity
  + Confusion Matrix
  + ABCDE Criteria
* Rare:
  + Jaccard similarity coefficient (JSC)
  + geometric mean (G-mean)
  + Matthew’s correlation coefficient (MCC)
  + Cohen’s kappa score (CKS)
  + AUROC
  + precision-recall curve (PR-AUC)
  + evaluation time

# Future Scope

* Implementation of various other algorithms and using several optimization techniques. Also, more data will be collected in order to recognize the features more accurately.
* Major attention will be given to increase the accuracy such that our proposed system can be used to detect a large number of chronic and critical diseases.
* When these enhancements are done, the system can be integrated with an android application to make it more convenient and easily portable. This will allow people from all strata to use it effectively even if they do not have a personal computer.

# Conclusions

* A “health discernment system” has been proposed for medical image classification that will work in real-life scenarios.
* The proposed method is based on ***Convolutional Neural Network*** architecture.
* Different sub-models pertaining to the two diseases (skin cancer: Melanoma, Benign) have been designed using convolutional neural network (CNN) and they have all been tested separately.
* For pre-Processing method, we have come to transfer learning as the best algorithm as it works best with most neural networks, doesn’t require a lot of data and consumes very less time as compared to others
* For Feature selection method, we have arrived at correlation matrix with heatmap as the heatmap makes it easy to identify which features are most related to the target variable and it can be trained under supervised and unsupervised methods with three subclasses under each method.
* For model selection or the activation function we think the SoftMax function is the best option as it used in multiple classification logistic regression model and in building neural networks SoftMax functions used in different layer level. SoftMax assigns decimal probabilities to each class in a multi-class problem. Those decimal probabilities must add up to 1.0. This additional constraint helps training converge more quickly than it otherwise would.
* For model training the best method out of all the listed one’s is the CNN algorithm as a CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images and also CNNs highly suited for and accurate for computer vision tasks such as object/image classification.

# References

## COncepts & Information

* ResearchGate.com
* ScienceDirect.com
* GeeksforGeeks.com
* TutorialsPoint.com
* cs.stanford.edu
* Springer.com
* towardsdatascience.com
* academictorrents.com
* kaagle.com
* ncbi.nlm.nih.gov
* springml.com

## Research

* <https://www.researchgate.net/publication/334751850_Skin_Cancer_Detection_Using_Convolutional_Neural_Network>
* Peer-review under responsibility of the scientific committee of the 16th International Learning & Technology Conference 2019. 10.1016/j.procs.2019.12.090
* https://thesai.org/Publications/IJACSA
* <https://www.thelancet.com/journals/ebiom/article/PIIS2352-3964(19)30294-4/fulltext>

* <https://www.sciencedirect.com/science/article/pii/S2215016120300832?via%3Dihub>
* https://www.sciencedirect.com/science/article/pii/S1532046418301618?via%3Dihub
* https://www.ejcancer.com/article/S0959-8049(19)30381-8/fulltext
* https://www.researchgate.net/publication/335321267\_Bio-Inspired\_DeepCNN\_Pipeline\_for\_Skin\_Cancer\_Early\_Diagnosis
* https://www.researchgate.net/publication/347927500\_Automated\_Multiclass\_Classification\_of\_Skin\_Lesions\_through\_Deep\_Convolutional\_Neural\_Network\_with\_Dermoscopic\_Images
* https://www.researchgate.net/publication/339804392\_Deep\_Learning\_Solutions\_for\_Skin\_Cancer\_Detection\_and\_Diagnosis?enrichId=rgreq-cc1ddabe639ba17298c595c27b48021c-XXX&enrichSource=Y292ZXJQYWdlOzMzOTgwNDM5MjtBUzo4ODM0MjE3NjUxNzMyNTFAMTU4NzYzNTU3MDg4Nw%3D%3D&el=1\_x\_3&\_esc=publicationCoverPdf
* https://ieeexplore.ieee.org/document/9062473
* https://www.sciencedirect.com/science/article/pii/S0933365719301460
* https://papers.ssrn.com/sol3/papers.cfm?abstract\_id=3352407
* https://www.researchgate.net/publication/335337495\_A\_CNN\_toolbox\_for\_skin\_cancer\_classification?enrichId=rgreq-91f014447a4c29cec1b8c22e6f8930fb-XXX&enrichSource=Y292ZXJQYWdlOzMzNTMzNzQ5NTtBUzo4MDExNDUyMDMyMjQ1NzdAMTU2ODAxOTMwOTE5Nw%3D%3D&el=1\_x\_3&\_esc=publicationCoverPdf
* https://ieeexplore.ieee.org/document/8720210
* https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6231861/
* https://iopscience.iop.org/article/10.1088/1757-899X/982/1/012005
* https://www.researchgate.net/publication/327539277\_Skin\_Cancer\_Classification\_using\_Convolutional\_Neural\_Networks\_Systematic\_Review\_Preprint
* https://www.sciencedirect.com/science/article/pii/S2352914819302047
* https://www.researchgate.net/publication/334123580\_Melanoma\_Skin\_Cancer\_Detection\_using\_Image\_Processing\_and\_Machine\_Learning
* https://www.nature.com/articles/nature21056
* https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9113301
* <https://pubmed.ncbi.nlm.nih.gov/28117445/>
* https://pubmed.ncbi.nlm.nih.gov/30333097/
* https://ieeexplore.ieee.org/document/9198489