**CHAPTER 1**

**INTRODUCTION**

In the recent times, the amount of data computers generate has grown exponentially, and the need to store them has become paramount. The industry has started to develop specialised category of softwares for handling what is now called as Big Data. Human intervention for data segregation is no more feasible and effecient for such large scale data. Images are one of the most generated and used type of data and it needs to be handled intelligently as well as effeciently. Hence image segregation is a necessity that software engineers need to address.

Our project aims to segregate image files effeciently using machine learning techniques.It carries out image file segregation based on the similar and dissimilar features present in it. The aim of this project is to map a given set of files based on their similar and dissimilar features i.e map similar ones nearer and vice versa to the dissimilar ones. The above project falls under the category of system software. Currently we have options like classification of files based on type, date, size and so on. We do not have a system software that classifies the files based on the content.

This is achieved using dimensionality reduction algorithms like T-SNE (t-distributed stochastic neighbour embedding). This falls under Unsupervised Machine Learning wherein the programmer does not manually teach the program what exactly the file contents are, rather the program itself picks certain features and maps it nearer to some files and farther to some other files based on the similarities and dissimilarities.

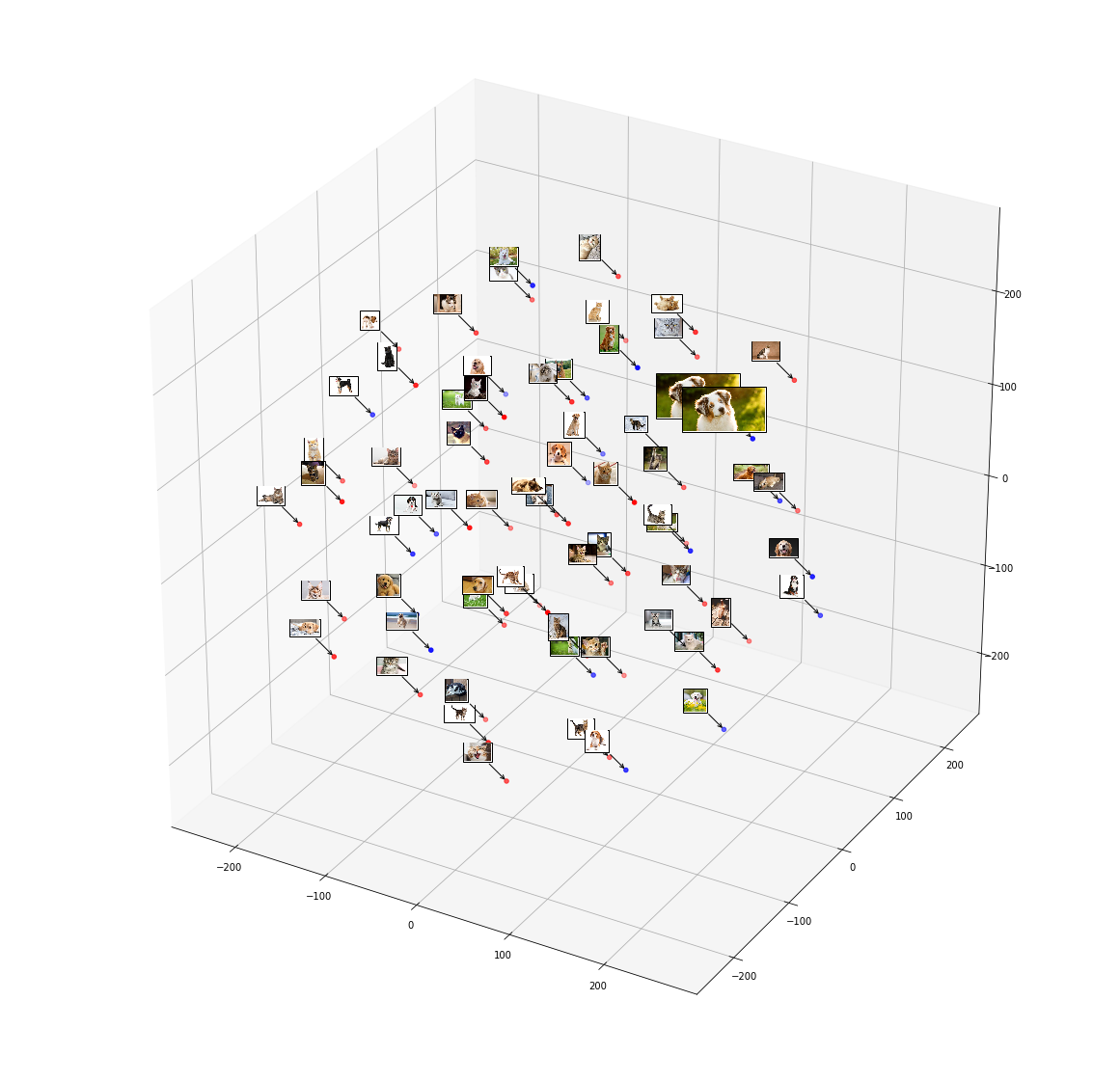
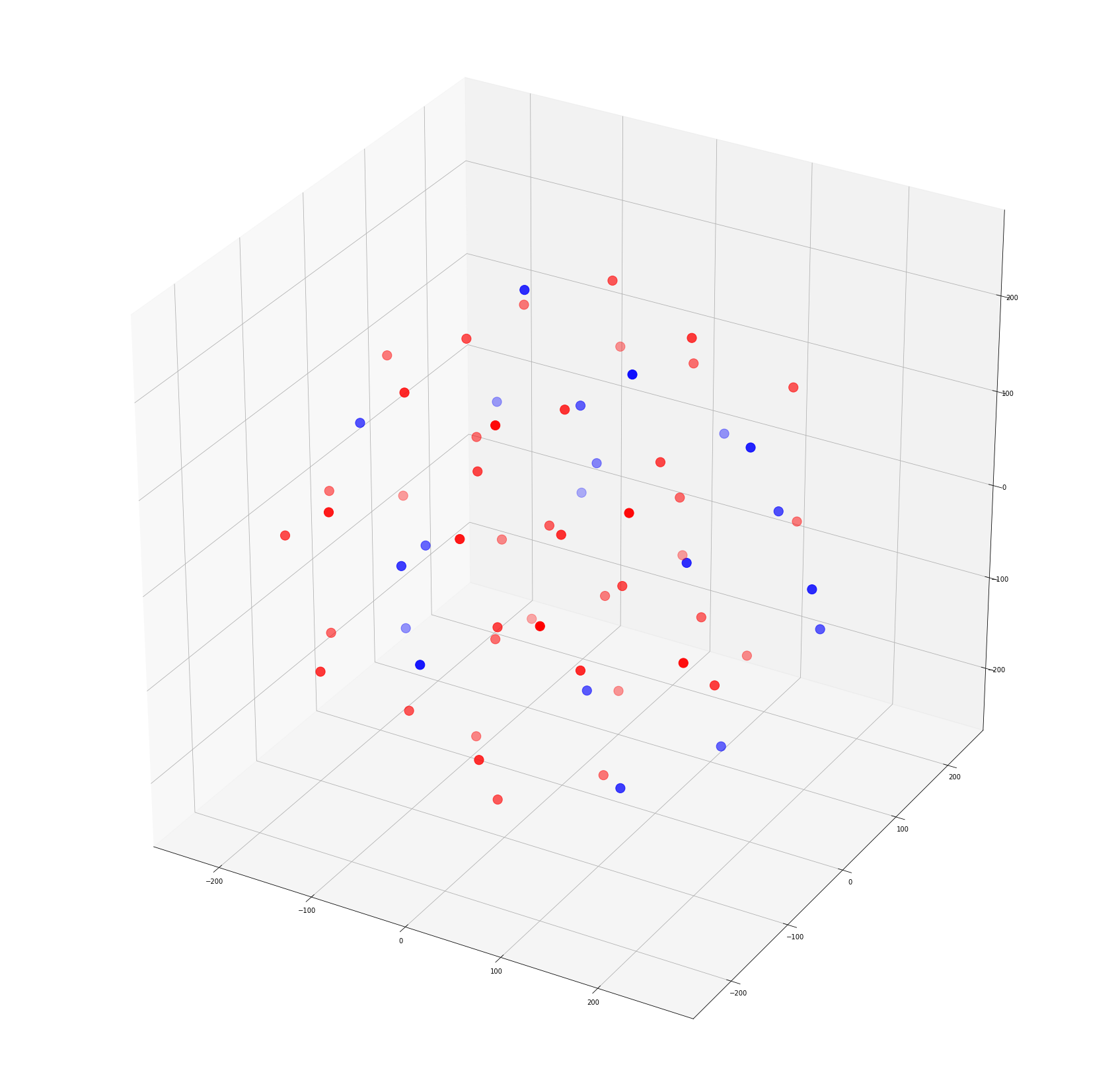
However in the process, we use convolutional neural networks that needs to be trained (supervised machine learning), but this is only for practical convenience and greater effeciency and not a requirement or a flaw.

In this project we are using image files as one can easily identify how the images have been classified. Images are basically made up of pixlels and each pixel has a certain RGB value and a set of pixels making up a part of an image may constitue a feature. And every image is considered as a high dimensional dataset space.

However, passing raw RGB pixel values may not be the most favourable way for dimensionality reduction algorithms. Hence we use a pre trained convolutional neural network of an image classifier. Instead of obtaining the labels for the image(s) we extract values from the penultimate layer which contains feature sets of the image. The feature sets are basically array of floating point values.

This large array of floating points containg the feature sets is provided to the T-SNE algorithm which maps them into lower dimensions as specified which maybe a cluster of images (3D), 2D Map or linear collection(1D). T-SNE returns what is called as embeddings. Embedding(s) are basically floating point values which effectively represent the contents of a dataset in a lower dimension.

We need to take into account the fact that reducing the mappings to lower dimensions results in significant loss of spatial information. However the accuracy of mappings can be improved by training the image classifier CNN with larger and more diverse dataset.

****

(i)Representation of Images as points in 2D plane (ii)Representation of Images in 2D plane on a 3D plot.   
 on a 3D plot.

**OBJECTIVE:**

* To sort or segregate image files based on their relative similarities and dissimilarities in order to help the users organise their file system in a more systematic way.
* This project aims to change the organisational paradigm of user’s file system.
* It enables quick and easy way of sorting large sets of images.
* Machine learning picks up many patterns and features that a person might have missed (color distribution etc.) i.e it gives a thorough comparison and segregation.

**CHAPTER 2**

**LITERATURE SURVEY**

A literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic.

**Area chosen**: **System Software**

**2.1 Convolutional Neural Network**

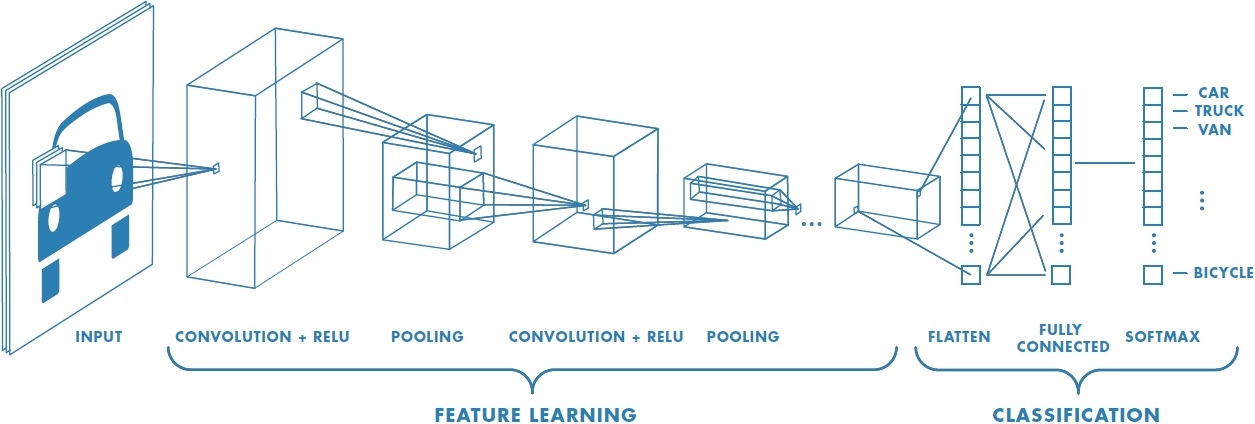
In [machine learning](https://en.wikipedia.org/wiki/Machine_learning), a convolutional neural network (CNN, or ConvNet) is a class of deep, feed forward [artificial neural networks](https://en.wikipedia.org/wiki/Artificial_neural_network) that has successfully been applied to analyzing visual imagery.

CNNs use a variation of [multilayer perceptrons](https://en.wikipedia.org/wiki/Multilayer_perceptron) designed to require minimal [preprocessing](https://en.wikipedia.org/wiki/Data_pre-processing).They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and [translation invariance](https://en.wikipedia.org/wiki/Translation_invariance) characteristics.

Convolutional networks were [inspired](https://en.wikipedia.org/wiki/Mathematical_biology) by [biological](https://en.wikipedia.org/wiki/Biological) processes in that the connectivity pattern between [neurons](https://en.wikipedia.org/wiki/Artificial_neuron) resembles the organization of the animal [visual cortex](https://en.wikipedia.org/wiki/Visual_cortex). Individual [cortical neurons](https://en.wikipedia.org/wiki/Cortical_neuron) respond to stimuli only in a restricted region of the [visual field](https://en.wikipedia.org/wiki/Visual_field) known as the [receptive field](https://en.wikipedia.org/wiki/Receptive_field). The receptive fields of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other [image classification algorithms](https://en.wikipedia.org/wiki/Image_classification). This means that the network learns the [filters](https://en.wikipedia.org/wiki/Filter_(signal_processing)) that in traditional algorithms were [hand-engineered](https://en.wikipedia.org/wiki/Feature_engineering). This independence from prior knowledge and human effort in feature design is a major advantage.They have applications in [image and video recognition](https://en.wikipedia.org/wiki/Computer_vision), recommender systems and [na](https://en.wikipedia.org/wiki/Natural_language_processing)tural language processing.

CNN in this project is used for extracting feature sets from an image. Feature sets identified by a CNN depends on how well and how diversely it has been trained. As mentioned earlier we extract values from the penultimate layer which contains feature sets of the image. The feature sets are basically array of floating point values.

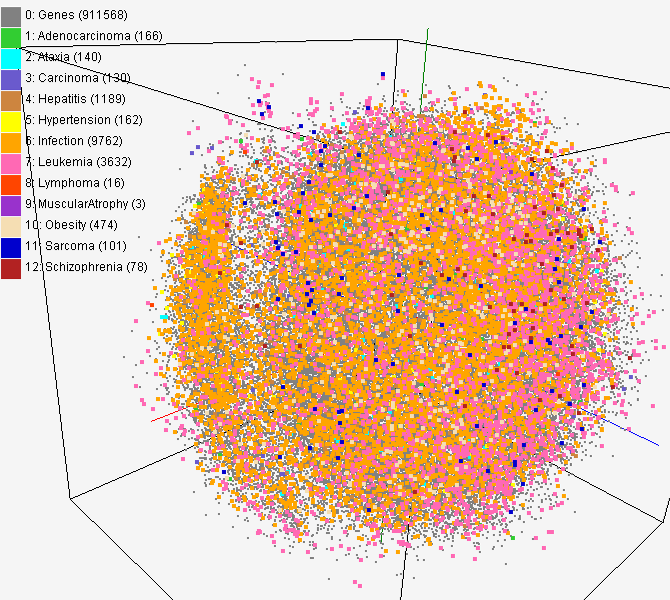


**2.2 High dimensional dataset**

Suppose we have *n* data points and *p* features (attributes for every data point).

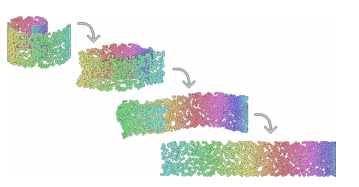
It is said to be a high dimensional dataset when p>n, but usually p>>n. Meaning, the number of features per data point is very much more than the number of data points.

The adjoinging figure shows a pictorial representation of a high dimensional dataset.

****

**2.3 t-distributed stochastic neighbor embedding (t-SNE)**

It is a [machine learning](https://en.wikipedia.org/wiki/Machine_learning) algorithm for dimensionality reduction was developed by [Geoffrey Hinton](https://en.wikipedia.org/wiki/Geoffrey_Hinton) and Laurens van der Maaten. It is a [nonlinear dimensionality reduction](https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction) technique that is particularly well-suited for embedding high-dimensional data into a space of two or three dimensions, which can then be visualized in a [scatter plot](https://en.wikipedia.org/wiki/Scatter_plot). Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points.



Reducing higher dimension data to lower dimensions

The t-SNE algorithm comprises two main stages. First, t-SNE constructs a [probability distribution](https://en.wikipedia.org/wiki/Probability_distribution) over pairs of high-dimensional objects in such a way that similar objects have a high probability of being picked, whilst dissimilar points have an extremely small probability of being picked. Second, t-SNE defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the [Kullback–Leibler divergence](https://en.wikipedia.org/wiki/Kullback–Leibler_divergence) between the two distributions with respect to the locations of the points in the map. Note that whilst the original algorithm uses the [Euclidean distance](https://en.wikipedia.org/wiki/Euclidean_distance) between objects as the base of its similarity metric, this should be changed as appropriate.

t-SNE has been used in a wide range of applications, including [**computer security**](https://en.wikipedia.org/wiki/Computer_security) research, [music analysis](https://en.wikipedia.org/wiki/Music_analysis),[cancer research](https://en.wikipedia.org/wiki/Cancer_research),[bioinformatics](https://en.wikipedia.org/wiki/Bioinformatics), and biomedical signal processing.It is often used to visualize high-level representations learned by an [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network).

**2.4 Caffe Library**

Caffe is a machine learning library developed by BVLC(Berkley Vision and Learning Center) at UC (University of California) Berkeley as a part of their reasearch on machine learning. It is completely written in C++ and hence priorotize on deployment than reasearch unlike most other machine learning libraries. This library is used for edge case, air gapped(no internet) computer deployments. This library is widely adopted by tech giants like Facebook.

This mini project uses caffe’s models for feature extraction. The model used is called bvlc\_reference\_caffenet. This extracts features i.e elements,colours and colour gradients and various other entities present in an image that can be used to for deducing similarities and differences between images present in the dataset.

**2.5 QT**

Qt is a cross-platform application framework that is used for developing application software that can be run on various software and hardware platforms with little or no change in the underlying codebase, while still being a native application with native capabilities and speed. Qt is currently being developed both by The Qt Company, a publicly listed company, and the Qt Project under open-source governance, involving individual developers and firms working to advance Qt.

**CHAPTER 3**

**REQUIREMENT SPECIFICATIONS**

**Hardware requirements:**

* Architecture: intel i386, x86\_64
* Processor: intel core i3
* RAM: 4GB DDR3 SDRAM
* Secondary storage: 512GB SATA
* Graphics: 256 Mb onboard graphics or above
* Mouse, Keyboard and LCD Display

**Software requirements**

* Operating System: Linux x86\_64 – 3.16 and above or Windows 7 and above
* Image viewer and File explorer
* Caffe 1.0

**CHAPTER 4**

**DESIGN**

**CHAPTER 5**

**IMPLEMENTATION**

**CHAPTER 6**

**TESTING**

**TESTING CASES:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl No** | **Test case** | **Description** | **Expected result** | **Status** |
| **1.** | Unit Testing: Extracting Embeddings | Getting the values for images based on their features | The function should return a vector of vectors having floating point values. | Passed |
| **2.** | Unit Testing:  File Dialog - Image selection | Checking whether the ui gets the details of the selected images | The function should return list of strings having file absolute path. | Passed. |
| **3.** | Unit Testing  File Dialog -  Folder selection | Checking whether the ui gets the details of the image files in the folder selected. | The function should return list of strings having the file names selected from the folder. | Passed. |
| **4.** | Unit Testing -  Creating 2D Plot | Displaying the image files at the respective coordinates. | Creates a new window with images at their respective coordinates. | Passed. |
| **5.** | Integration Testing -  UI and Back-End Integration | Passing the List of files selected to the Back-End | The function returns a Vector of Vectors having 2 floating points in each vector, that are used for plotting on 2D plane. | Passed. |
| **6.** | Validation Testing | Output Window Test | The window displays images at different coordinates in a 2D plane, based on their features. | Passed. |

**SNAPSHOTS**

**CONCLUSION**

**REFERENCES**