International Institute of Professional Studies

Devi Ahilya Vishwa Vidyalaya

Indore, MP

Project Report on

Investigation of Efficient Approaches for Image Classification through Deep Learning

Project Submitted in the Partial fulfilment of the Requirement for the Award of the Degree of

Master in Computer Application

Semester VI

Session Jan - May, 2023

Under the guidance of
Dr. Shaligram Prajapat

Submitted By:
Shruti Khandelwal
IC - 2K20 - 77

International Institute of Professional Studies
Devi Ahilya Vishwa Vidyalaya, Indore, MP

DECLARATION

I hereby declare that the project entitled “Investigation for Efficient Approaches for Image Classification through Deep Learning” submitted by me for the partial fulfilment of the requirement for the award of the degree of Master of Computer Application(MCA) Semester VI to International Institute of Professional Studies, Devi Ahilya Vishwa Vidyalaya is an authentic and original work and due acknowledge has been made in text to all other materials submitted.

The matter embodied in this project is genuine work done by me and has not been submitted whether to this University or to any other University/Institution for the fulfilment of the requirements for any course of study.

Signature of Student:
Date:
Place:

International Institute of Professional Studies
Devi Ahilya Vishwa Vidyalaya, Indore, MP

CERTIFICATE

This is to certify that we have examined the dissertation on “Investigation for Efficient Approaches for Image Classification through Deep Learning”, submitted by Ms. Shruti Khandelwal to International Institute of Professional Studies, Devi Ahilya Vishwa Vidyalaya, Indore and hereby accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfilment for the award of the degree of “Master of Computer Application (5 Years) Semester VI”.

Internal Examiner	**External Examiner**
Signature: ______________	Signature: _____________
Name : ______________	Name : _____________
Date : ______________	Date : _____________

ACKNOWLEDGEMENT

I would like to take this opportunity to acknowledge my sincere gratitude to everyone who has significantly contributed towards this project.

I acknowledge my sincere thanks to those who have contributed significantly to this project. It is a pleasure to extend my deepest gratitude to **Dr. Shaligram Prajapat**, for his valuable guidance and support as well as providing necessary information regarding the project and also for his support in completing the project.

A special mention needs to be made for **Dr. B.K.Tripathi**, director of IIPS, DAVV, **Mr. Jogendra Dongre** and **Mrs. Shraddha Soni** who have helped very willingly as often as required and cooperated with interest.

My classmates also deserve a mention for their constant review, praise and critical appraisal.
I thank and acknowledge each and everyone's efforts that have helped me in some way or another for their small and significant efforts.

Before ending, I would like to thank my dear mother, **Ms. Archana Khandelwal** and friends for their unwavering support and encouragement throughout this journey.
I sincerely regret any inadvertent omissions.

With my heartiest thanks to all.
Shruti Khandelwal
IC-2K20-77

Abstract

This project aims to investigate deep learning methods that provide efficient approaches for Image Classification through Deep Learning methods. In recent years, deep learning methods have shown unprecedented success in image classification tasks. The specific aim of this project is to investigate different deep learning methods for image classification and compare their performance. This project will specifically explore Convolutional Neural Networks (CNN) and Recurrent Neural Networks(RNN) for image classification.

The project will begin with a comprehensive literature review of existing deep learning methods for image classification. We will then implement and evaluate these methods on benchmark datasets, such as MNIST, CIFAR-10 and ImageNet. This project will also evaluate the impact of various hyperparameters, such as learning rate, batch size, and activation functions based on the performance of these methods.

Finally, we will conduct a comparative analysis of the different methods used and provide insights into the strengths and weaknesses of each approach.

Table of Contents

Table of Figures

Introduction

1 Introduction

1.1 Problem Definition

Image classification is a foundational task in computer vision that has many practical applications such as object recognition, facial recognition, and autonomous driving. Traditional methods of machine learning, such as Support Vector Machines(SVM), Decision Trees and Random Forests have been widely used for image classification. However, with increasing availability of large scale datasets and advancements in deep learning methods, deep neural networks have shown exceptional success in image classification tasks.

However, despite the impressive performance of deep neural networks, there are still several challenges that need to be addressed. These challenges are:

1. Deep neural networks require heavy computation and are computationally expensive.
2. They require large amounts of datasets for training.
3. Choosing the appropriate network architecture and hyperparameters can be a time consuming and a challenging task.
4. Deep neural networks are prone to overfitting, which may result in poor generalisation performance.

Therefore, the problem this project aims to address is to investigate the effectiveness of multiple deep learning methods for image classification and identify the challenges associated with them. The project will traverse the effectiveness of different deep learning methods for image classification and identify and address the challenges associated with them.

Therefore, the problem that this project aims to address is to investigate the effectiveness of different deep learning methods for image classification and identify the challenges associated with them. The project will explore the performance of different deep learning neural network architectures, namely Convolutional Neural Networks and Recurrent Neural Networks on benchmark datasets. The project will also investigate the impact of various hyperparameters on the performance of these models. By addressing these challenges, the project aims to improve the accuracy and efficiency of deep learning models for image classification.

1.2 Aim

This project aims to compare the performance of different deep learning methods for image classification. Specifically, the project will investigate Convolutional Neural Networks and Recurrent Neural Networks and evaluate the impact of various hyperparameters on their performance. Our goal is to identify the most effective techniques for image classification and improve the accuracy and efficiency of deep learning models.

1.3 Objective

The following are the objectives of this project on the investigation of methods of deep learning for image classification:

1. To conduct a comprehensive literature review of existing deep learning methods for image classification.
2. To implement and evaluate different deep learning methods, including CNNs and RNNs, on benchmark datasets such as MNIST, CIFAR-10, and ImageNet.
3. To investigate the impact of various hyperparameters such as learning rate, batch size, and activation functions on the performance of these models.
4. To compare the performance of different methods and provide insights into strengths and weaknesses of each approach.
5. To identify the most effective deep learning methods for image classification and provide recommendations for selecting appropriate techniques for different types of image datasets.

1.4 Project Goals

The following are the project goals of the investigation of methods of deep learning for image classification:

1. To identify the most effective deep learning methods for image classification and provide recommendations for selecting appropriate techniques for different types of image datasets.
2. To provide insights into the strengths and weaknesses of different types of deep learning methods for image classification.
3. To explore the impact of various hyperparameters on the performance of deep learning models for image classification.
4. To compare the performance of different deep learning methods for image classification on benchmark datasets.
5. To provide a comprehensive literature review of existing deep learning methods for image classification.

1.5 Benefits

The investigation of methods of deep learning for image classification can provide several benefits, which are listed below:

1. **Improved Accuracy**
   By identifying the most effective deep learning methods for image classification and exploring the impact of various hyperparameters, the project can help improve the accuracy of image classification models.
2. **Increased Efficiency**
   By investigating and comparing different deep learning methods, the project can help identify techniques that are more effective than others, leading to faster and resource-efficient image classification models.
3. **Practical Applications**
   The findings of this project can have practical applications in fields such as object recognition, facial recognition, and autonomous driving, where accurate and efficient image classification is crucial.
4. **Insights into strengths and weaknesses**
   The project can provide insights into the strengths and weaknesses of different deep learning methods for image classification, which can help guide the development of future research in this area.
5. **Transferable knowledge**
   This project can contribute to the body of knowledge on deep learning methods, which can be applied to areas beyond image classification.

1.6 Methodology

The project plan is to carry out the design and implementation of the project in a complete step-by-step manner. The project is divided into the following phases:

Phase 1 - Study and Analysis Phase

1. A comprehensive review was conducted of existing methods of deep learning for image classification, focusing on strengths and weaknesses of different models and techniques. This involved reviewing relevant books, online resources and research papers.

Phase 2 - Design Phase

In this phase, the design of the proposed study is prepared. The design is carried out in the following manner:

1. Entities and their relationship is identified from the scenario.
2. ER Diagram is designed for the proposed system and relationship among entities is documented in the form of tables, and fields having attributes.
3. The data flow diagram of the whole study is constructed. For detailed information, DFD is prepared for each level.
4. Flow chart for each process in the DFD is prepared.
5. The physical design of the system is proposed.
6. Defined the hardware and software requirements.

The experimental design was developed, including selection of models, hyperparameters, evaluation metrics, and experimental methodology.

Phase 3 - Coding Phase

1. The research to be done on the methods is implemented through actual code.
2. Proper train and test data is used to train and test the model.
3. The datasets: MNIST, CIFAR-10, and ImageNet, are collected and prepared for experiments.

Phase 4 - Testing and Implementation

Testing is done on various training and testing datasets. Testing is applied on:

1. Implementing the deep learning models: CNN and RNN using Python programming language, and popular deep learning libraries such as TensorFlow, PyTorch, and Keras.
2. Training the models on the image data and evaluating their performance on validation datasets.
3. Fine tuning models to improve their performance.
4. Selecting the best performing model using evaluation metrics such as accuracy, precision, recall and F1 score.
5. Testing the models on a separate dataset to evaluate its generalisation ability.

The above plan follows the System Development Life Cycle, also known as SDLC, approach. These five phases provide a structured and comprehensive approach for system development, ensuring that systems are built to meet the needs and requirements of the target audience, and are tested and maintained to provide optimal performance and reliability.

1.7 Achievements

The student completing this project will achieve the following:

1. Gain deeper understanding of deep learning and image classification techniques
2. Develop technical skills in programming, data analysis and machine learning
3. Develop expertise in using deep learning libraries and tools such as TensorFlow and Keras.
4. Learn to work with large datasets and popular deep learning frameworks
5. Gain experience in project management and software development methodologies
6. Have a tangible project to showcase to potential employers or academic institutions.

1.8 Organisation of Report

This report is a comprehensive document detailing the procedure, finding and conclusion of the project. It contains the following chapters:

1. **Current and Proposed System**
   This section details the current methodologies for image classification and the proposed method for the same.
2. **Feasibility Study**
   This section details the economic, technical and behavioural feasibility of the project.
3. **Analysis**
   This section details the study and analysis that was conducted for this project.
4. **Project Planning**
   This section details the scope of the project, its development plan, the team structure, project deliverables and a gantt chart highlighting the project timeline.
5. **Design**
   This section details the logical design of the project, detailing its entities and attributes relationships.
6. **Implementation**
   This section details how the project was implemented, and its step-by-step procedure.
7. **Code Snippet**
   This section contains the code snippet, which the code used to implement the project.
8. **Testing**
   This section details how the project was tested against various benchmark datasets.
9. **Future Prospects**
   This section reflects upon how the project can be carried further from this stage on.

Current and Proposed System

2 Current and Proposed System

2.1 Current System

There is no *single* current system used for image classification using deep learning, as there are multiple approaches and techniques that can be used depending on the problem being addressed.

1. **Convolutional Neural Networks(CNN)**
   It is a type of deep learning model and is commonly used for image and video recognition, natural language processing, and other tasks involving multi-dimensional data.
2. **Recurrent Neural Networks**
   It is a type of neural network in deep learning that is designed to work with sequential data, such as time series, natural language or speech. The main idea behind RNNs is that they use the output from the previous step as input for the current step. They have a wide array of applications, including speech recognition, sentiment analysis and stock prediction.
3. **Deep Belief Networks**
   It is a type of deep neural network in deep learning that is composed of multiple layers of Restricted Boltzmann Machines(RBMs). The key idea behind DBNs is to train each layer of the network using unsupervised learning to learn a compressed representation of input data. Its applications include image and speech recognition, natural language processing and drug discovery.

2.2 Limitations of the Current System

There is no single current system used for image classification using deep learning, as there are multiple approaches and techniques that can be used depending on the problem being addressed. However, each approach has disadvantages of its own.

1. **Convolutional Neural Networks(CNN)**
   Although CNNs are widely used in many areas of deep learning, they have some limitations:
   a. Computationally Expensive
      CNNs require a large number of computations to train and evaluate, which requires a large amount of computations, thus making the model computationally expensive and time consuming, especially for large datasets.
   b. High Memory Requirements
      CNNs require a large amount of memory space to store the model's parameters, especially for deep networks with many hidden layers. This can be challenging when training on devices with limited memory.
   c. Lack of Interpretability
      CNN is also referred to as a "black box" because the model makes it difficult to understand how it's making its decisions. This can be challenging in applications where it's important to understand how the network is making its predictions.
   d. Susceptible to overfitting
      Like a lot of machine learning models, CNNs are susceptible to overfitting, especially when training on small datasets.
   e. Limited Ability to handle Sequential Data
      CNNs are mainly designed for processing grid data, such as images, and are ill-fitted for processing sequential data such as time series and neural networks.
2. **Recurrent Neural Networks**
   Although RNNs have many advantages in deep learning in deep learning, they also has their share of disadvantages:

a. Vanishing and Exploding gradients

   RNNs tend to suffer from vanishing gradients and exploding gradient problems, making it difficult to train the network efficiently.

b. Computationally Expensive

   RNNs can be computationally expensive to train and run, especially when dealing with large datasets and complex models.

3. Deep Belief Networks

Although DBNs have many advantages in deep learning in deep learning, they also has their share of disadvantages:

a. Computationally Expensive

   Training a DBN requires heavy computations, particularly if the network or dataset is large.

b. Dependency on Pre-Training

   DBNs often depend on pre-training to initialise the model parameters. This is very time consuming, and computationally expensive.

2.3 Proposed System

The following points give an overview of the proposed study:

Aim:

To explore and analyse different deep learning techniques for image classification.

Objective:

1. To explore and evaluate the performance of different deep learning methods, including CNNs and RNNs.
2. To identify strengths and limitations of each technique.

Methodology:

1. Study and Analysis Phase

   Conducting a literature review to identify the current techniques and provide a foundation for the experimental phase.
2. Experimental Phase

   Train and test a variety of deep learning models on image classification datasets, using techniques such as cross-validation, to evaluate their performance. Use a range of metrics to evaluate the performance of the models, including accuracy, prediction, recall and F1 score.

Outputs:

1. Comparison of different deep learning techniques for image classification tasks.
2. Identification of strengths and limitations of each technique.
3. Insights into the practical considerations for using these techniques in applications.

Benefits:

1. Identification of the most effective techniques and their potential applications.

2.4 Objectives of Proposed System

The aim of the proposed study is to study the limitations of the current models. Following are the objectives for the proposed study:

1. To explore and analyse different methods of deep learning used for image classification tasks.
2. To identify strengths and limitations of each method.
3. To evaluate and compare the performance of different methods, such as CNNs, and RNNs.
4. To determine the optimal parameters for each method to achieve its best performance in image classification tasks.

Feasibility Study

3 Feasibility Study

3.1 Feasibility Analysis

The feasibility study examines how truly feasible it is for the project to be commenced. The feasibility is verified in terms of economic feasibility, technical and non-technical feasibility. The project is completely feasible, and it satisfies all the criteria for feasibility.

3.2 Economic Feasibility

The economic feasibility analysis evaluates the effectiveness of a system using the cost/benefit analysis method.

The economic feasibility of the project is measured using the following attributes:

1. The project fits under the proposed budget.
   The project has not exceeded the cost constraints.
2. The project satisfies the economic constraints.
   The project has been created with the cost constraints in mind, and does not exceed those constraints.
3. The existing hardware meets the proposed budget.
   The current hardware on which the project is being run comprises the hardware made available by Google Colab.

This project can be considered as economically feasible, because the candidate conducting the research possesses a computer with themself. Thus, there is no need to bear any costs because of the resources needed for the development of the system.

The only costs incurred during the project are those during the research phase. The resources needed to conduct proper research

The only resource required for proper commencement of this project is a computer system that meets the software and hardware requirements of the study.

3.3 Technical Feasibility

The criterias for technical feasibility of the project are:

1. Availability of the required hardware and software
    a. Availability of CPUs and GPUs that are powerful enough to train the deep learning models.
    b. Availability of deep learning libraries such as TensorFlow or PyTorch for implementing deep learning models.
    c. Availability of other software tools and libraries such as NumPy and Pandas for data processing and analysis.
2. Availability of large and diverse image datasets
    a. Availability of large and diverse image datasets that are suitable for training and testing deep learning models.
    b. Availability of tools for data preprocessing and cleaning.
3. Computing resources and performance
    a. The time and computing resources required for training and testing deep learning models.
    b. Availability of sufficient memory, storage and processing power to large datasets and complex models.
4. Technical expertise
    a. Availability of support resources, such as online forums and documentation, for resolving technical issues.

Technology

Python

Python is a very powerful and highly versatile general purpose language. It is an interpreted, interactive, object-oriented and high-level programming language. It is the primary language used for developments in data science, machine learning, deep learning and artificial intelligence, due to its easy to read and learn syntax and vast libraries.

Python is used in deep learning due to its following features:

1. <u>Concise and readable code</u>
   Python is a high-level programming language that is easy to read and write. Python's syntax is more intuitive and closer to natural language than many other programming languages. This makes it easy to prototype and experiment with deep learning models.
2. <u>Portability</u>
   Python is a portable language, which means that code written in Python can be easily moved across different platforms and operating systems. Code from Google Colab can be moved onto a computer in different formats and vice versa.
3. <u>Extensive libraries and frameworks</u>
   Python has a large and robust system of libraries and tools designed for deep learning, such as TensorFlow, Keras, PyTorch and Theano. These libraries facilitate easy development and training of deep learning models, and provide a wide range of functionalities for various tasks.
4. <u>Community support</u>
   Python has a large and open source community, which means developers can easily find support and resources online. This community also contributes to the development of new libraries, frameworks and tools, which has helped facilitate advancements in deep learning.
5. <u>Educational Resources</u>
   Python is often taught in universities and online courses, meaning there is a large pool of developers who contribute to its online community.

Python Libraries

The main Python libraries involved in this Project are:

1. **Numpy**
   It is a library for Python programming, and gives support to large, multi-dimensional arrays and matrices. It stands for Numerical Python. It is a fundamental package for scientific computing with Python. It is an open source software.
2. **Pandas**
   It is a software library for Python programming, and is particularly used for data manipulation and analysis. It particularly offers data structures and operations for facilitating manipulation of numerical tables and time series.
3. **Matplotlib**
   It is a plotting library for the Python programming language. It is a comprehensive library that is used for creating static, animated and interactive data visualisation graphics. It is an extension of Numpy. Pyplot, a submodule of matplotlib, is mainly used for this project.
4. **Seaborn**
   It is a python library, built on top of matplotlib, for visually appealing and informative statistical graphs. It is designed to work with pandas dataframes. It offers a variety of powerful tools for visualising data, including scatter plots, line graphs, bar plots, and many more. It also provides support for advanced statistical analysis, such as regression analysis, distribution plots and categorical plots.
5. **Scikit**
   It is a machine learning library used in the Python language. It features various algorithms, such as classification, regression, random forests, k-means clustering, etc. It is developed to work alongside Numpy and SciPy.
6. **Tensorflow**
   It is an end to end open source software library for machine learning and artificial intelligence. It can be used for a wide range of tasks, but is particularly focused on training and inference of deep neural networks. It is developed and maintained by Google.

Google Colab

Colaboratory by Google, also known as Google Colab, is a Jupyter notebook based runtime environment which allows the user to run python programs entirely on cloud. It can be used to train large scale Machine Learning and Deep Learning models, even if the user does not have access to powerful machines. It allows both GPU and TPU instances, making it the perfect tool for deep learning and data analytics. Since it is based on cloud, it can be remotely accessed from any machine through a browser.

Services that are offered by Google Colab and are used in this project:

- ➢ Write and execute Python programs
- ➢ Document programs that support mathematical equations
- ➢ Create/Upload/Share notebooks
- ➢ Import/Share notebooks from/to Google Drive
- ➢ Import external datasets, for example, from Kaggle.
- ➢ Integrate TensorFlow, Keras
- ➢ Free cloud service with free GPU.

Image Datasets

1. **ImageNet**

   It is a comprehensive image repository consisting of millions of labelled images, which is extensively used in areas of computer vision and machine learning. It is also used to assess the performance of deep neural networks, with the focus being on image classification tasks.

2. **MNIST**

   The MNIST database(Modified National Institute of Standards and Technology) is a large database that consists of handwritten digits. It is commonly used for image processing systems. This database of training and testing data is also widely used in training machine learning and deep learning models.

3. **CIFAR 10**

   CIFAR-10 is a benchmark dataset, used for mainly training and testing machine learning models, particularly for image classification tasks. It is used as a standard benchmark dataset for image classification, and is used to test the performance of machine learning models and deep neural networks.

3.4 Behavioural Feasibility

The behavioural feasibility involves assessing the practicality of the project from a behavioural perspective. This includes the project's potential impact on behaviours of users and beneficiaries.

The following criterias were evaluated to ensure behavioural feasibility of this project:

1. **User Acceptance**
   The project was designed in such a way that it is user friendly, easy to use and requires minimal training towards the users' side.
2. **Ethical Considerations**
   The project was conducted in such a way that it followed ethical protocols, and complied with relevant regulations.
3. **Sustainability**
   The project's sustainability was taken into account, with a long term plan for maintenance and support.
4. **Accessibility**
   The project is accessible to all through and complies with the accessibility standards.

The above criteria were evaluated to ensure that the project is feasible from a behavioural point of view.

Analysis

4 Analysis

Study and Analysis

Image classification mainly finds its use in computer vision. It is used to categorise images into classes or categories that are predefined. Deep learning and machine learning techniques are commonly used for image classification due to their ability to automate learning complex features from a large dataset.

Image classification is popularly achieved through the following techniques:

1. **Data Preparation**
   The first step for any image classification task is to gather and process the data. This usually involves collecting a large dataset consisting of images, each of which is associated with a class label. The dataset is then split into training and testing dataset.
2. **Feature Extraction**
   In machine learning approaches, features can be extracted from the pixelated data using SIFT, HOG and other such techniques. However, in deep learning, Convolutional Neural Networks(CNN) are capable of capturing complex patterns and representations from images.
3. **Model Training**
   Once the data is prepared and the features are extracted, the next step is to train the model. The training process involves several epochs, with the model iterating over the training set multiple times to improve its performance.
4. **Model Evaluation**
   After training, the model is evaluated on the validation dataset to assess its performance. Common evaluation metrics for image classification include its accuracy, precision, recall and F1-score.
5. **Model Testing**
   Once the model is assessed on the validation set, it is then tested on the test dataset, which is a separate dataset not used during training or validation.

Project Planning

5 Project Planning

The most critical phase of SDLC is the planning phase. It allows a structured and organised approach to achieving project objectives. Planning ensures that resources are allocated efficiently, timelines are realistic, and potential risks and challenges are addressed. Proper planning can also help avoid delays, and ensure that the project stays on track to achieve its objectives within the specified timeframe and budget.

5.1 Project Scope

The scope of the project can be described using the following points:

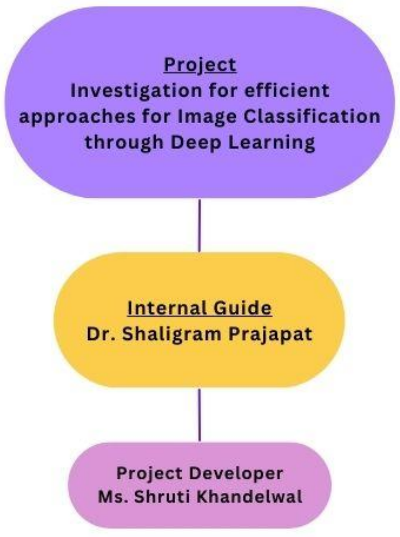
1. Investigate various methods of deep learning for Image Classification.
2. Conduct a comparative study of different models for image classification.
3. Develop and implement deep learning models for image classification.
4. Test and evaluate the performance of the model on various datasets.
5. Document the entire process to be presented in the form of a report.

5.2 Development Plan

The design and implementation of this project can be carried in a step by step manner through the following steps:

1. Literature Review
2. Design document
3. Coding
4. Testing
5. Project report

5.3 Team Structure


Project
Investigation for efficient approaches for Image Classification through Deep Learning
Internal Guide
Dr. Shaligram Prajapat
Project Developer
Ms. Shruti Khandelwal

5.4 Project Deliverables

Project deliverables are the tangible outcomes that are produced as a result of completing the project. The following are the deliverables for this project:

Project Proposal

A project proposal is a document that outlines the proposed project, including its objectives, methodology, deliverables and budget.

Project Software Requirements Specifications (SRS)

A Software Requirements Specification(SRS) is a document detailing the descriptions of the requirements of the software project. It is usually created during the planning phase of the System Development Life Cycle(SDLC). It serves as a guide for developers, testers and other stakeholders in this project.

Project Software Design Document (SDD)

A Software Design Document(SDD) is a document detailing the technical specifications of the project. It is usually created after the requirements of the software have been defined in the Software Requirements Specification(SRS) document. It serves as a blueprint for developers to follow during the implementation of the project.

Project Software Project Management Plan (SPMP)

It is a comprehensive document that outlines the processes and procedures that will be used during the management of the project. It describes how the project will be planned, executed ,monitored, controlled, and closed. The document should be updated and reviewed throughout the project lifecycle.

Project Report

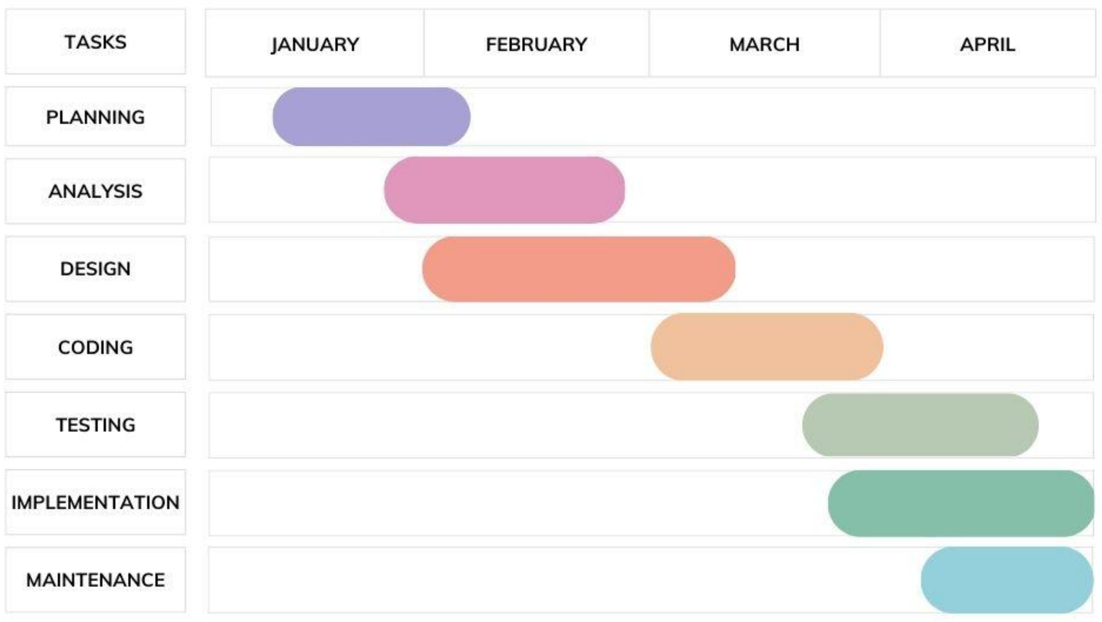
A project gives a detailed insight to the current system as well as the proposed system. It provides a detailed account of the project's activities, including planning, execution, and evaluation. It serves as a record of the project's accomplishments and can be used to assess the project's impact.

Source Code

The source code for the deep learning models is developed as part of the project.

5.5 Gantt Chart

Gantt Chart is a visual representation of the project's schedule. It is used for project management, as it allows the team to visualise the timeline of the project, indemnify critical activities and adjust the schedule as and when needed to meet project goals and deadlines.



Design

6 Design

6.1 Logical Design

The logical design describes the functions that are required of a system.

6.1.1 Entities Definition

Entities represent real world objects or concepts that are relevant to the system being modelled. They are typically represented as rectangles in an ER Diagram. Entities have attributes that describe their features or characteristics. Attributes are represented by ovals in an ER Diagram. Entities are typically connected to each other through relationships.

- Project
  It is the concept that is being modelled.
- Project Developer
  They are a real world entity who is assigned to work on the project.
- Institute
  It is a real world entity, who has a relationship with the project developer and assigns the project to the project developer.
- Users
  They are a world entity that requires the project.

6.1.2 Attribute Definition

An attribute is used to describe the features or characteristics of an entity. They provide information about the entity. They are represented by ovals in an ER Diagram.

1. **Name of Entity**: Users

Attribute	Datatype	Description	Length	Allow Null
name	text	User name	50	no
address	text	User Address	100	no
phone_no	numeric	User Phone Number	double	no

2. **Name of Entity**: Institute

Attribute	Datatype	Description	Length	Allow Null
name	text	Institute name	50	no
location	text	Institute Address	100	no
landline_no	numeric	Institute Phone Number	double	no

3. **Name of Entity**: Project Developer

Attribute	Datatype	Description	Length	Allow Null
name	text	Developer name	50	no
address	text	Developer Address	100	no
unique_id	numeric	Developer’s unique ID	double	no

4. **Name of Entity**: Project

Attribute	Datatype	Description	Length	Allow Null
name	text	Project name	50	no

unique_id	text	Project ID	100	no
start_date	numeric	Project start date		no
end_date	numeric	Project end date		no

6.13 Relationships

One to many

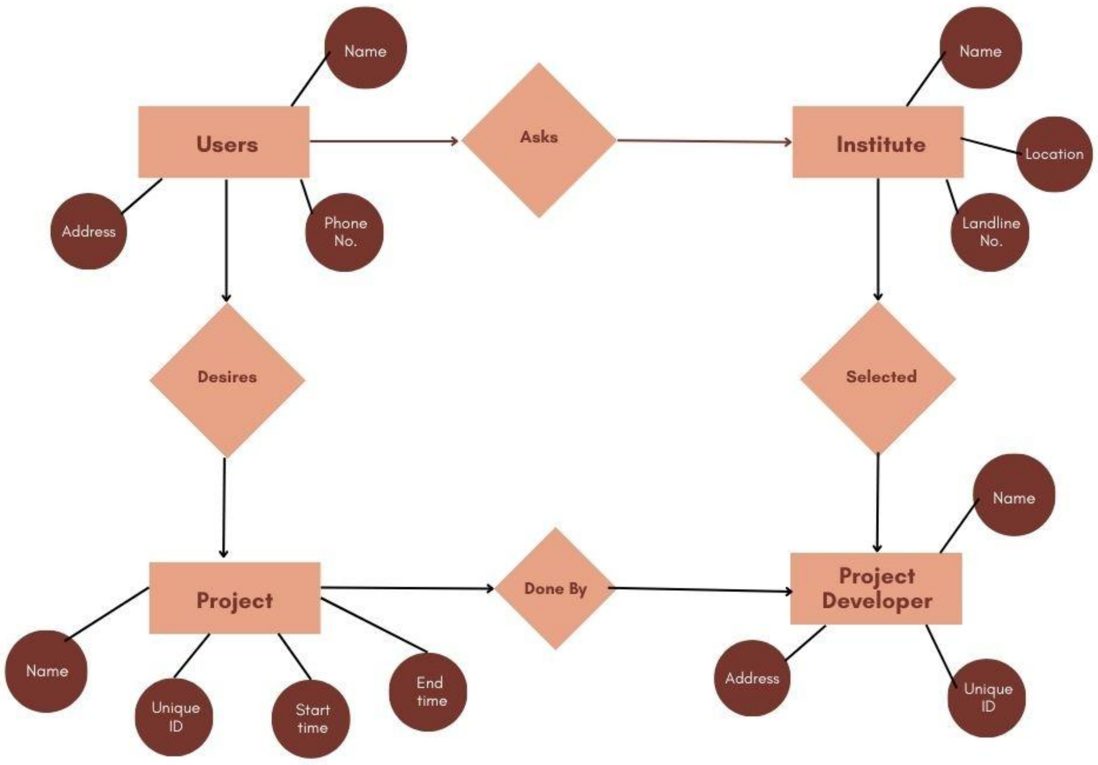
User <-> Project

User <-> Institute

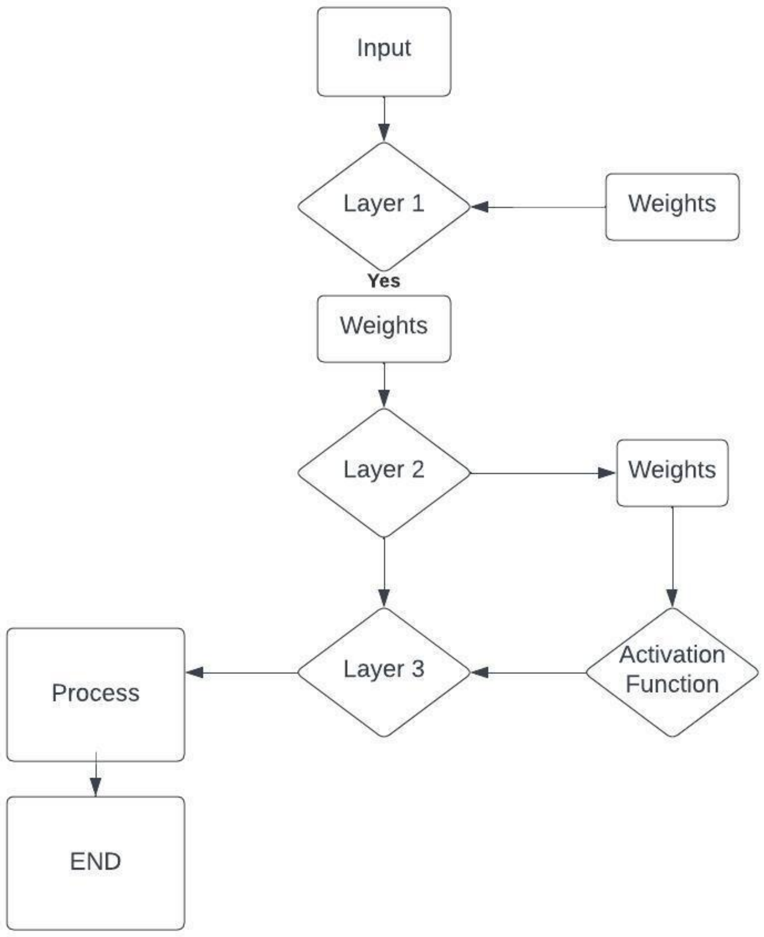
Project Developer <-> Institute

Project Developer <-> Project

6.1.4 ER Diagram



6.1.5 Data Flow Diagram



Implementation

7 Implementation

The implementation of this project involves several steps:

1. **Data collection and preprocessing**
   Gathering relevant datasets, cleaning and preprocessing data to ensure it is suitable for training and testing the selected deep learning models.
2. **Model selection and training**
   Choosing appropriate models for image classification, and training them on preprocessed data. This involves model selection, hyperparameter tuning, and model training.
3. **Model evaluation and validation**
   Evaluating the trained model on validation and testing datasets to assess its performance.
4. **Reporting and documentation**
   Documenting the project results and overall progress of the project, including timeline, resources and tasks.

Code Snippet

Convolutional Neural Network

Activation function - Linear function

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
(X_train,y_train),(X_test,y_test) = datasets.cifar10.load_data()
X_test.shape
X_train.shape
y_train[:5]
y_train = y_train.reshape(-1,)
y_train[:5]
y_test = y_test.reshape(-1,)
classes = ['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']
def plot_sample(X,y,index):
  plt.figure(figsize=(15,2))
  plt.imshow(X[index])
  plt.xlabel(classes[y[index]])
plot_sample(X_train,y_train,5)
plot_sample(X_train,y_train,501)
X_train = X_train/255.0
X_test = X_test/255.0
ann = models.Sequential([
    layers.Flatten(input_shape = (32,32,3)),
    layers.Dense(3000,activation = 'relu'),
    layers.Dense(3000,activation = 'relu'),
    layers.Dense(10,activation = 'softmax')
])

ann.compile(optimizer='SGD',
          loss = 'sparse_categorical_crossentropy',
          metrics = ['accuracy'])
```

```
ann.fit(X_train,y_train,epochs = 5)
from sklearn.metrics import confusion_matrix, classification_report

y_pred = ann.predict(X_test)
y_pred_classes = [np.argmax(element) for element in y_pred]

print('Classification report: \n',classification_report(y_test,y_pred_classes))
import seaborn as sns
plt.figure(figsize = (14,7))
sns.heatmap(y_pred,annot = True)
plt.ylabel('Truth')
plt.xlabel('Prediction')
plt.title('Confusion Matrix')
plt.show()
cnn = models.Sequential([
            layers.Conv2D(filters = 32,kernel_size = (3,3),activation = 'relu',input_shape=(32,32,3)),
   layers.MaxPooling2D((2,2)),


   layers.Conv2D(filters = 64,kernel_size = (3,3),activation = 'relu'),
   layers.MaxPooling2D((2,2)),


   layers.Flatten(),
   layers.Dense(64,activation = 'relu'),
   layers.Dense(10,activation = 'softmax')
])
cnn.compile(optimizer='adam',loss = 'sparse_categorical_crossentropy',metrics = ['accuracy'])
cnn.fit(X_train,y_train,epochs=10)
cnn.evaluate(X_test,y_test)
y_pred = cnn.predict(X_test)
y_pred[:5]
```

```
y_classes = [np.argmax(element) for element in y_pred]
y_classes[:5]
y_test[:5]
plot_sample(X_test,y_test, 60)
plot_sample(X_test,y_test,100)
classes[y_classes[60]]
```

Activation function - Step function

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D

(X_train,y_train),(X_test,y_test) = datasets.mnist.load_data()
X_test.shape
X_train.shape
y_train[:5]
y_train = y_train.reshape(-1,)
y_train[:5]
y_test = y_test.reshape(-1,)
classes= ['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']
def plot_sample(X,y,index):
 plt.figure(figsize=(15,2))
 plt.imshow(X[index])
 plt.xlabel(classes[y[index]])
```

```
plot_sample(X_train,y_train,98)
X_train = X_train/255.0
X_test = X_test/255.0

x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_train = x_train.astype('float32') / 255.0
y_train = tf.keras.utils.to_categorical(y_train, num_classes=10)

model.fit(x_train, y_train, epochs=10)

x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_test = x_test.astype('float32') / 255.0
y_test = tf.keras.utils.to_categorical(y_test, num_classes=10)
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

Activation Function - Sigmoid Function

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D

model = tf.keras.Sequential([
  tf.keras.layers.Conv2D(32, (3,3), activation='sigmoid', input_shape=(28,28,1)),
  tf.keras.layers.MaxPooling2D((2,2)),
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
          loss='categorical_crossentropy',
          metrics=['accuracy'])

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_train = x_train.astype('float32') / 255.0
y_train = tf.keras.utils.to_categorical(y_train, num_classes=10)

model.fit(x_train, y_train, epochs=10)

x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_test = x_test.astype('float32') / 255.0
y_test = tf.keras.utils.to_categorical(y_test, num_classes=10)
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

Activation Function - Tanh Function

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D

model = tf.keras.Sequential([
  tf.keras.layers.Conv2D(32, (3,3), activation='tanh', input_shape=(28,28,1)),
  tf.keras.layers.MaxPooling2D((2,2)),
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
        loss='categorical_crossentropy',
        metrics=['accuracy'])

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_train = x_train.astype('float32') / 255.0
y_train = tf.keras.utils.to_categorical(y_train, num_classes=10)

model.fit(x_train, y_train, epochs=10)

x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_test = x_test.astype('float32') / 255.0
y_test = tf.keras.utils.to_categorical(y_test, num_classes=10)
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

Activation Function - ReLU Function

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
```

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D

(X_train,y_train),(X_test,y_test) = datasets.mnist.load_data()
X_test.shape
X_train.shape
y_train[:5]
y_train = y_train.reshape(-1,)
y_train[:5]
y_test = y_test.reshape(-1,)
classes= ['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']
def plot_sample(X,y,index):
 plt.figure(figsize=(15,2))
 plt.imshow(X[index])
 plt.xlabel(classes[y[index]])
plot_sample(X_train,y_train,98)
X_train = X_train/255.0
X_test = X_test/255.0
ann = models.Sequential([
   layers.Flatten(input_shape = (32,32,3)),
   layers.Dense(3000,activation = 'relu'),
   layers.Dense(3000,activation = 'relu'),
   layers.Dense(10,activation = 'softmax')
])
```

```
ann.compile(optimizer='SGD',
        loss = 'sparse_categorical_crossentropy',
        metrics = ['accuracy'])

ann.fit(X_train,y_train,epochs = 5)
y_pred = ann.predict(X_test)
y_pred_classes = [np.argmax(element) for element in y_pred]

print('Classification report: \n',classification_report(y_test,y_pred_classes))
plt.figure(figsize = (14,7))
sns.heatmap(y_pred,annot = True)
plt.ylabel('Truth')
plt.xlabel('Ptrediction')
plt.title('Confusion Matrix')
plt.show()
cnn = models.Sequential(([
               layers.Conv2D(filters = 32,kernel_size = (3,3),activation = 'relu',input_shape=(32,32,3)),
  layers.MaxPooling2D((2,2))

  layers.Conv2D(filters = 64,kernel_size = (3,3),activation = 'relu'),
  layers.MaxPooling2D((2,2)),



  layers.Flatten(),
  layers.Dense(64,activation = 'relu'),
  layers.Dense(10,activation = 'softmax')
  ]))
cnn.compile(optimizer='adam',loss = 'sparse_categorical_crossentropy',metrics = ['accuracy'])
cnn.fit(X_train,y_train,epochs=10)
cnn.evaluate(X_test,y_test)
y_pred = cnn.predict(X_test)
y_prediction[:5]
```

```
y_classes = [np.argmax(element) for element in y_pred]
y_classes[:5]
y_test[:5]
plot_sample(X_test,y_test, 60)
plot_sample(X_test,y_test,100)
Classes[y_classes[60]]
```

Recurrent Neural Network

1. **Activation function - Linear function**

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
model = tf.keras.Sequential([
  tf.keras.layers.SimpleRNN(128, activation='linear', input_shape=(28, 28)),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
          loss='sparse_categorical_crossentropy',
          metrics=['accuracy'])

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
```

```
model.fit(x_train, y_train, epochs=10)

test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

2. **Activation function - Sigmoid function**

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
model = tf.keras.Sequential([
  tf.keras.layers.SimpleRNN(128, activation='sigmoid', input_shape=(28, 28)),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
         loss='sparse_categorical_crossentropy',
         metrics=['accuracy'])

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0

model.fit(x_train, y_train, epochs=10)
```

```
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

3. Activation function - Tanh function

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
model = tf.keras.Sequential([
  tf.keras.layers.SimpleRNN(128, activation='tanh', input_shape=(28, 28)),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
         loss='sparse_categorical_crossentropy',
         metrics=['accuracy'])

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0

model.fit(x_train, y_train, epochs=10)
```

```
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

4. Activation function - Relu function

```
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
model = tf.keras.Sequential([
  tf.keras.layers.SimpleRNN(128, activation='relu', input_shape=(28, 28)),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
         loss='sparse_categorical_crossentropy',
         metrics=['accuracy'])

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0

model.fit(x_train, y_train, epochs=10)

test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

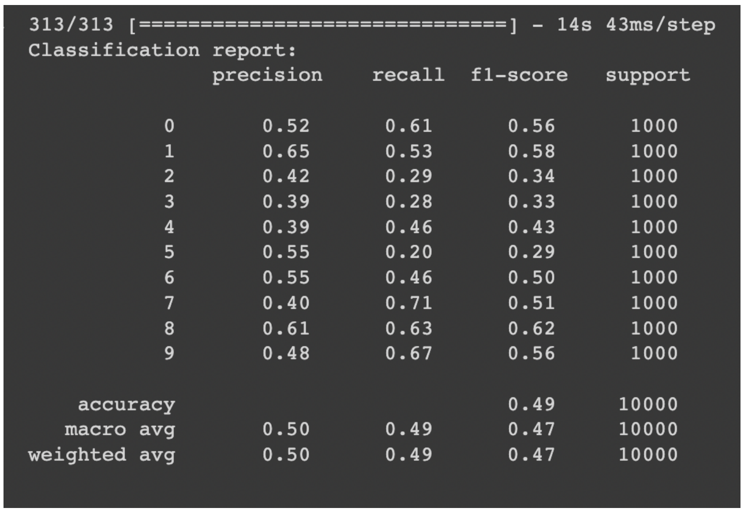
Testing

9 Testing

The system is tested using the following testing strategies:

1. **Accuracy Testing**

   This involves measuring the accuracy of the model in correctly classifying images. The accuracy can be measured using metrics such as precision, recall and F1 score.



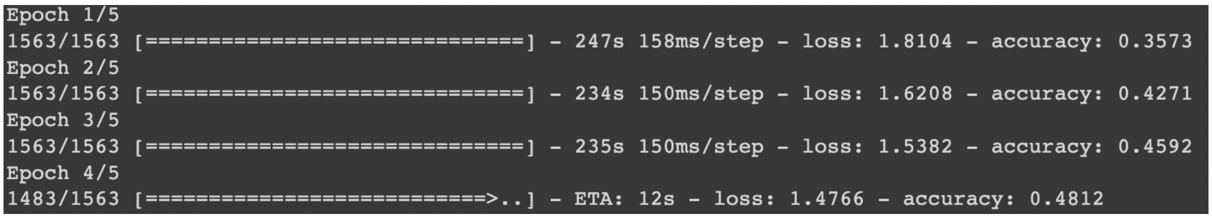
```
313/313 [==============================] - 14s 43ms/step
Classification report:
              precision    recall  f1-score   support

           0       0.52      0.61      0.56      1000
           1       0.65      0.53      0.58      1000
           2       0.42      0.29      0.34      1000
           3       0.39      0.28      0.33      1000
           4       0.39      0.46      0.43      1000
           5       0.55      0.20      0.29      1000
           6       0.55      0.46      0.50      1000
           7       0.40      0.71      0.51      1000
           8       0.61      0.63      0.62      1000
           9       0.48      0.67      0.56      1000

    accuracy                           0.49     10000
   macro avg       0.50      0.49      0.47     10000
weighted avg       0.50      0.49      0.47     10000
```

2. **Cross Validation**

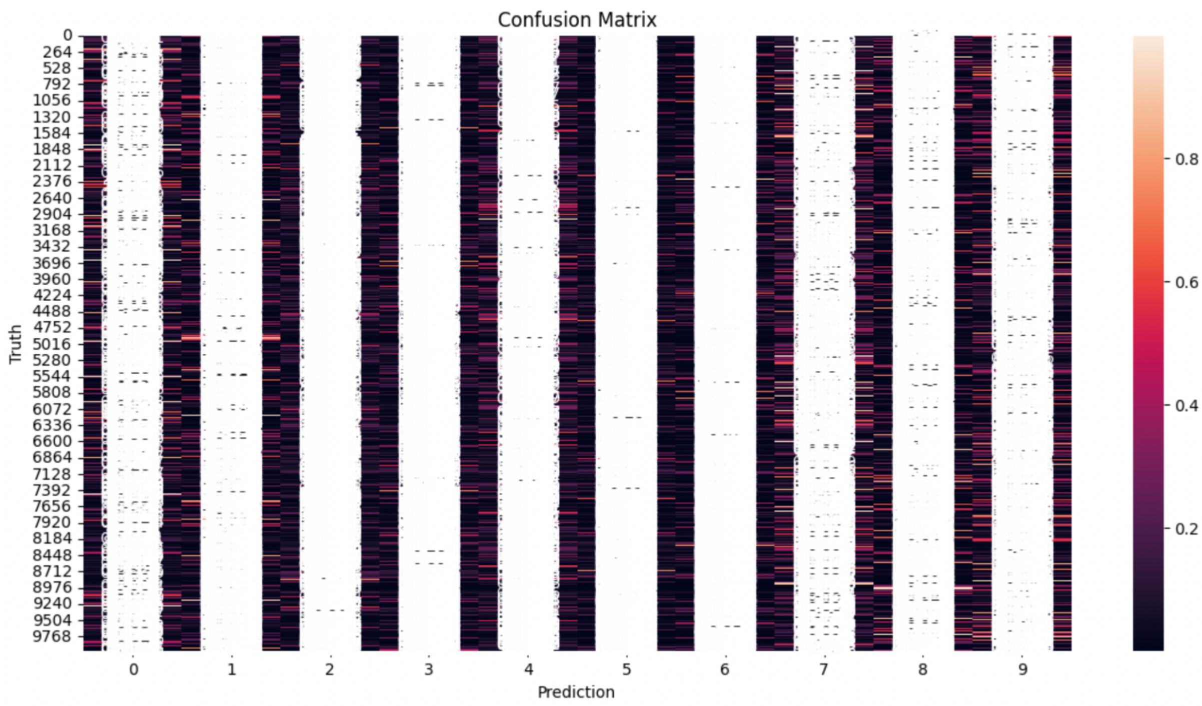
   This involves dividing the dataset into multiple subsets and training the model on each subset while testing it on the remaining subset.



```
Epoch 1/5
1563/1563 [==============================] - 247s 158ms/step - loss: 1.8104 - accuracy: 0.3573
Epoch 2/5
1563/1563 [==============================] - 234s 150ms/step - loss: 1.6208 - accuracy: 0.4271
Epoch 3/5
1563/1563 [==============================] - 235s 150ms/step - loss: 1.5382 - accuracy: 0.4592
Epoch 4/5
1483/1563 [===========================>..] - ETA: 12s - loss: 1.4766 - accuracy: 0.4812
```

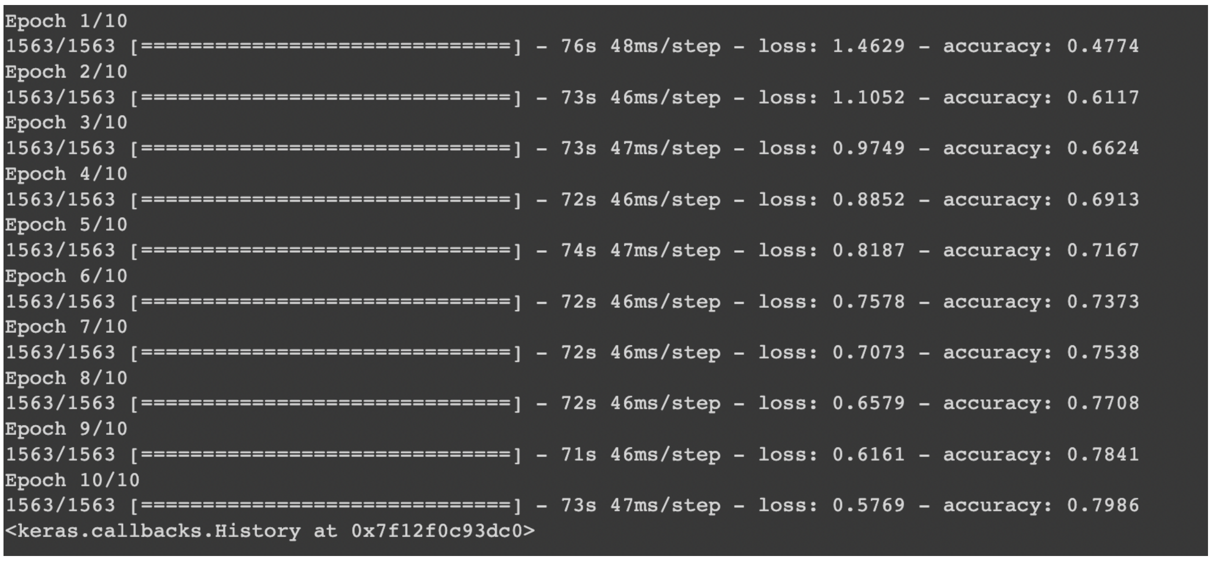
3. **Confusion matrix**

   This is a matrix that shows the number of correct and incorrect predictions made by the model. It can be used to evaluate the performance of the model and identify areas for improvement.



4. **Stress Testing**

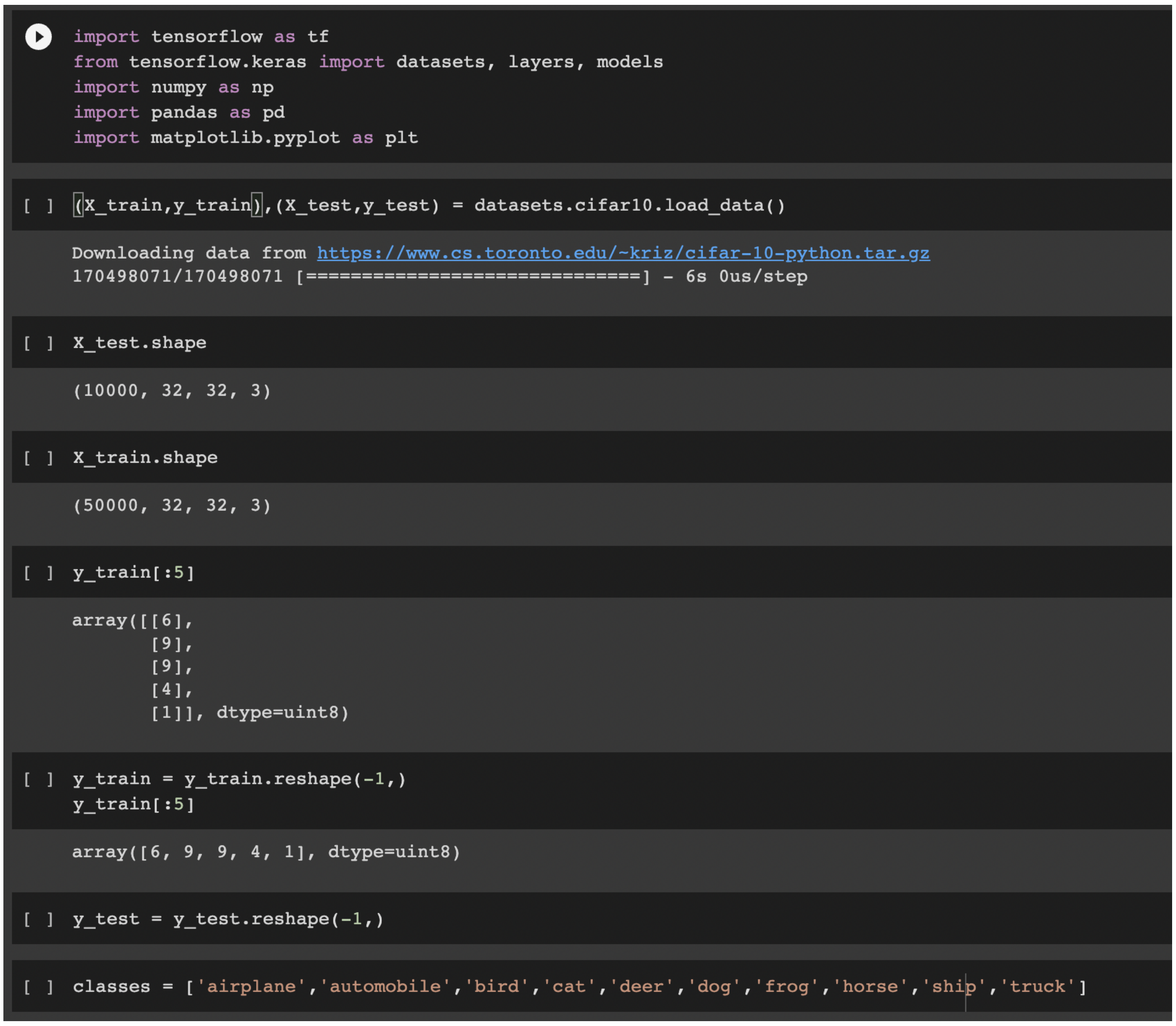
   This involves testing the model under extreme circumstances, such as a high load to determine its robustness and reliability.



```
Epoch 1/10
1563/1563 [==============================] - 76s 48ms/step - loss: 1.4629 - accuracy: 0.4774
Epoch 2/10
1563/1563 [==============================] - 73s 46ms/step - loss: 1.1052 - accuracy: 0.6117
Epoch 3/10
1563/1563 [==============================] - 73s 47ms/step - loss: 0.9749 - accuracy: 0.6624
Epoch 4/10
1563/1563 [==============================] - 72s 46ms/step - loss: 0.8852 - accuracy: 0.6913
Epoch 5/10
1563/1563 [==============================] - 74s 47ms/step - loss: 0.8187 - accuracy: 0.7167
Epoch 6/10
1563/1563 [==============================] - 72s 46ms/step - loss: 0.7578 - accuracy: 0.7373
Epoch 7/10
1563/1563 [==============================] - 72s 46ms/step - loss: 0.7073 - accuracy: 0.7538
Epoch 8/10
1563/1563 [==============================] - 72s 46ms/step - loss: 0.6579 - accuracy: 0.7708
Epoch 9/10
1563/1563 [==============================] - 71s 46ms/step - loss: 0.6161 - accuracy: 0.7841
Epoch 10/10
1563/1563 [==============================] - 73s 47ms/step - loss: 0.5769 - accuracy: 0.7986
<keras.callbacks.History at 0x7f12f0c93dc0>
```

5. **Unit Testing**

   This involves testing the individual components of the model to ensure that each of them are functioning correctly.



```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
[ ] (X_train,y_train),(X_test,y_test) = datasets.cifar10.load_data()

    Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
    170498071/170498071 [==============================] - 6s 0us/step
```

```
[ ] X_test.shape

    (10000, 32, 32, 3)
```

```
[ ] X_train.shape

    (50000, 32, 32, 3)
```

```
[ ] y_train[:5]

    array([[6],
           [9],
           [9],
           [4],
           [1]], dtype=uint8)
```

```
[ ] y_train = y_train.reshape(-1,)
    y_train[:5]

    array([6, 9, 9, 4, 1], dtype=uint8)
```

```
[ ] y_test = y_test.reshape(-1,)
```

```
[ ] classes = ['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']
```

Conclusion

10 Conclusion

10.1 Conclusion

As the project comes to a conclusion, it is evident that deep learning models perform remarkably in image classification tasks. The purpose of this project was to investigate various methods of deep learning for image classification.

Through the implementation and testing of different models, the strengths and weaknesses of each project was identified. We also demonstrated the importance of selecting the correct parameters and hyperparameters for gaining high accuracy in image classification.

The project has provided insights into many techniques which can be used in a variety of applications, such as self-driving cars, medical diagnosis, and surveillance systems.

10.2 Limitations of the system

The limitations of this project are:

1. Limited access to high quality and diverse datasets, which affects the accuracy and generalisability of the model.
2. Limited commuting resources limit the scope and complexity of the project.
3. Limited experience and expertise with deep learning methods and technologies, which impacted the project's ability to develop and evaluate effective models.
4. The time given was limited, which affected the exploration in the subject matter of deep learning.
5. The comparison between different deep learning models across different datasets proved to be a challenge, which made it difficult to draw definite conclusions about the effectiveness of different methods.

Future Prospects

10 Future Prospective

The project can be carried out further into the following possibilities:

1. This project can lead to adoption of such technologies in real world situations such as:
    a. Traffic rule breaking image analysis
    b. Analysing images for security
    c. Self Driving Cars
    d. Augmented and Virtual Reality
    e. Medical Imaging
    f. Robotics
    g. Environmental monitoring

Bibliography

1. Chistopher M. Bishop, Pattern Recognition and Machine Learning
2. Geron Aurilien, Hands on Machine Learning with Scikit-learn, Keras and Tensorflow

References

1. FreeCodeCamp. 2020. Deep Learning Crash Course for Beginners. Video. (July 30, 2020). Retrieved from https://www.youtube.com/watch?v=VyWAvY2CF9c
2. FreeCodeCamp. 2020. TensorFlow 2.0 Complete Course - Python Neural Networks for Beginners Tutorial. Video. (July 30, 2020). Retrieved from https://www.youtube.com/watch?v=tPYj3fFJGjk
3. Daniel Bourke. 2021. Learn TensorFlow and Deep Learning fundamentals with Python (code-first introduction) Part 1/2. Video. (March 16, 2021). Retrieved from https://www.youtube.com/watch?v=tPYj3fFJGjk
4. Daniel Bourke. 2021. Learn TensorFlow and Deep Learning fundamentals with Python (code-first introduction) Part 2/2. Video. (March 17, 2021). Retrieved from https://www.youtube.com/watch?v=ZUKz4125WNI\
5. Unfold Data Science. 2020. Google Colab Tutorial for Beginners | Using Google Colab for machine learning and Deep learning. Video. (October 10, 2020). Retrieved from https://www.youtube.com/watch?v=6Xt6L1I5jSc
6. GeeksForGeeks. 2020. Computer Vision. (6 February 2020). Retrieved from https://www.geeksforgeeks.org/computer-vision-introduction/

Annexure 1

Project SRS

Table of Contents

1 Introduction

The Software Requirement Specification(SRS) of the project provides an introduction of the current and proposed systems. The project is a study on deep learning methods for image classification tasks.

1.1 Purpose

It describes why the project is being undertaken and the problems associated with the current system:

1. The project aims to explore and investigate current methods and techniques for image classification tasks through deep learning methods.
2. The project aims to look into practical solutions for image classification using deep learning, such as medical imaging, robotics, autonomous driving, and many more.
3. The project aims to provide an educational and learning experience for the project team, allowing them to gain head on experience with deep learning techniques, image classification tasks and project management skills.

1.2 Project Scope

The scope of the project can be described using the following points:

1. Investigate various methods of deep learning for Image Classification.
2. Conduct a comparative study of different models for image classification.
3. Develop and implement deep learning models for image classification.
4. Test and evaluate the performance of the model on various datasets.
5. Document the entire process to be presented in the form of a report.

1.3 Project Objectives

The following are the objectives of this project on the investigation of methods of deep learning for image classification:

1. To conduct a comprehensive literature review of existing deep learning methods for image classification.

2. To implement and evaluate different deep learning methods, including CNNs, DBNs and RNNs, on benchmark datasets such as MNIST, CIFAR-10, and ImageNet.
3. To investigate the impact of various hyperparameters such as learning rate, batch size, and activation functions on the performance of these models.
4. To compare the performance of different methods and provide insights into strengths and weaknesses of each approach.
5. To identify the most effective deep learning methods for image classification and provide recommendations for selecting appropriate techniques for different types of image datasets.

1.4 Abbreviations

1. SRS - Software Requirement Specification
2. CNN - Convolutional Neural Network
3. RNN - Recurrent Neural Network
4. DBN - Deep Belief Networks.

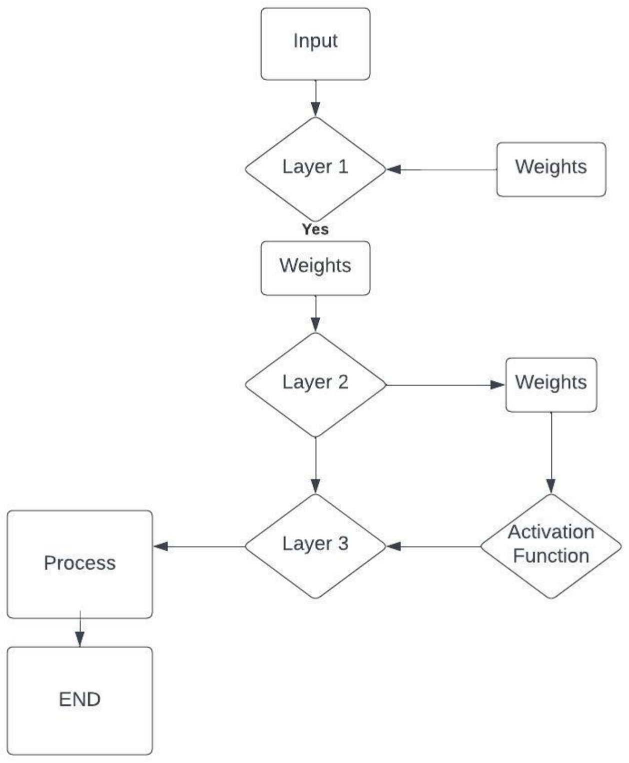
2 Overall Description

This section describes the general factors that affect the product and its requirements. This section consists of five subsections. This section does not state specific requirements. Each of these subsections makes those requirements easier to understand, it does not specify design or express specific requirements.

2.1 Project Perspective

This section describes the content and origin of "Investigation of Efficient Approaches for Image Classification through Deep Learning" being specified. It is a new, self contained product for the system. It allows users to look into different methods of deep learning for image classification.

The project diagram shows the overview of the system modules and the relationship between them.



Functions of System Components:

1. **Input**

   Input, in the form of image data is prepared that can be fed into the neural network.
2. **Layer 1 - Convolutional Layers**

   In this step, the image data is passed through multiple convolutional layers that apply filters to extract data from the image.
3. **Layer 2 - Pooling Layers**

   The image data is then passed into pooling layers that reduce the spatial dimension of data, and preserve the features at the same time.
4. **Layer 3 - Dense Layers**

   After the pooling layer, the image data is then passed through dense layers that make predictions based on the features extracted from the image.
5. **Process - Training**

   The model is then trained using labelled image data.

2.2 Project Functions

The functions of a project on investigation of deep learning for image classification include:

1. Conducting thorough research on existing methods of deep learning for image classification.
2. Collecting relevant image data for training, validation and testing of deep learning models.
3. Developing and implementing appropriate deep learning models for image classification tasks, such as CNN, RNN and DBN.
4. Training the models on training data and evaluating the model's performance on the validation dataset.
5. Testing the model performance on the test dataset, compare the models and select the best performing model.
6. Document the project's methodologies, results and findings in a report.

2.3 Design and Implementation Constraints

The constraints to be implemented and encountered during the design and implementation of this project are:

1. Hardware Limitations

   The performance of deep learning models heavily depends on the hardware used. Availability of high performance hardware, such as GPUs, can significantly affect the model.
2. Availability of labelled datasets

   Deep learning models require a large amount of training data to be trained effectively. The availability of high quality labelled data can be a developmental constraint for the models.
3. Complexity of deep learning models

   Deep learning models are complex and require significant computational resources to train and tune. This is a constraint as there is a limit to the access of computational resources.
4. Technical expertise

   Developing and implementing deep learning models requires specialised technical expertise. The availability of skilled personnel is a constraint for this project.
5. Time constraints

   The development and testing of these models is a time consuming task. The project has time constraints due to the short time frame given to the project.

2.4 User Documentation

The user documentation of a set of written material intended to help the users understand and use a system or a product. It usually includes manuals, quick-start guides, tutorials and other reference material. It depends on the specific design and implementation of the project.

It includes:

1. Overview of the project

A brief introduction to the project, its objectives and its goals.

2. Installation Instructions

   A guide to installing all necessary software and dependencies required to run the project.
3. User Manual

   A comprehensive guide to using the project's features and functions, including step-by-step instructions and screenshots to provide examples.
4. Troubleshooting Guide

   A list of common issues that users may encounter and how to resolve or troubleshoot them.
5. Frequently Asked Questions(FAQs)

   A list of common questions that the user may have and their answers about the project.
6. Glossary

   A list of technical terms and their definitions that are used in the project.
7. References

   A list of all sources and references used in the project, such as academic papers, or online resources.

2.5 Assumptions and dependencies

Assumptions and dependencies are important factors to consider when planning and implementing a project.

The *assumptions* of this project are:

1. Sufficient computing resources will be available to train and test deep learning models.
2. Appropriate datasets for image classification will be available for use in the project.
3. The team will have necessary knowledge and skills to implement and test deep learning models for image classification.

The *dependencies* of this project are:

1. The project timeline depends on the availability of certain datasets or computing resources.

2. The success of this project depends on external Python libraries and tills for deep learning image classification.
3. The project depends on the cooperation of team members and stakeholders, such as project mentor and subject matter experts.

3 Interface Requirements

3.1 Hardware Interface

The hardware interface for this project involves a computer system with high computational power to train deep learning models. This includes CPUs with multiple cores, GPUs, and a large amount of RAM. In addition, high quality datasets are required for training and testing the models.

3.2 Software Interface

The software interface is a point of interaction between different components of a system. The software components of this project include:

1. Programming languages
2. Deep learning libraries
3. Development environment
4. Hardware requirements
5. Visualisations

4 System Features

System features are the functionalities and capabilities of a software or a system. These features are designed to enable the system to perform its specific tasks.

The system features of this project include:

1. Preprocessing of image data
   The system should be able to preprocess the image data to make it compatible with deep learning algorithms.
2. Training of deep learning models
   The system should be able to train deep learning models using preprocessed data.
3. Evaluation of model performance
   The system should be able to evaluate the performance of the trained model using metrics such as accuracy, prediction and F1 score.
4. Prediction of class labels
   The system should be able to predict the class labels of test data based on the trained model.
5. Visualisation of results
   The system should be able to visualise the results of trained data in the form of charts, graphs and heatmaps.

5 Non-Functional Requirements

5.1 Performance Requirements

The performance requirements for this project are:

1. Accuracy

   The program should be able to accurately classify the images based on their content.
2. Speed

   The program should be able to classify images in real time.
3. Robustness

   The program should be able to handle different types of images, including those images which have different degrees of illumination, angles, etc.
4. Scalability

   The program should be able to handle a large amount of images and classify them efficiently.
5. Resource utilisation

   The program should be designed to use the system resources efficiently.

5.2 Safety Requirements

It is important to note that any data acquired during this project is obtained legally and ethically.

5.3 Security Requirements

It is important that this project does not use any sensitive or confidential data. If such sensitive data is used, appropriate measures must be taken to protect the data from unauthorised access or disclosure. Additionally, the production environment should be secured to prevent attacks from unauthorised access or malware.

Annexure 2

Project SDD

Table of Contents

1 Introduction

1.1 Purpose

It describes why the project is being undertaken and the problems associated with the current system:

1. The project aims to explore and investigate current methods and techniques for image classification tasks through deep learning methods.
2. The project aims to look into practical solutions for image classification using deep learning, such as medical imaging, robotics, autonomous driving, and many more.
3. The project aims to provide an educational and learning experience for the project team, allowing them to gain head on experience with deep learning techniques, image classification tasks and project management skills.

1.2 Project Scope

The scope of the project can be described using the following points:

1. Investigate various methods of deep learning for Image Classification.
2. Conduct a comparative study of different models for image classification.
3. Develop and implement deep learning models for image classification.
4. Test and evaluate the performance of the model on various datasets.
5. Document the entire process to be presented in the form of a report.

1.3 Project Objectives

The following are the objectives of this project on the investigation of methods of deep learning for image classification:

1. To conduct a comprehensive literature review of existing deep learning methods for image classification.
2. To implement and evaluate different deep learning methods, including CNNs, DBNs and RNNs, on benchmark datasets such as MNIST, CIFAR-10, and ImageNet.

3. To investigate the impact of various hyperparameters such as learning rate, batch size, and activation functions on the performance of these models.
4. To compare the performance of different methods and provide insights into strengths and weaknesses of each approach.
5. To identify the most effective deep learning methods for image classification and provide recommendations for selecting appropriate techniques for different types of image datasets.

2 System Overview

2.1 Overview

The following points give an overview of the proposed system:

Aim:

To explore and analyse different deep learning techniques for image classification.

Objective:

1. To explore and evaluate the performance of different deep learning methods, including CNNs and RNNs.
2. To identify strengths and limitations of each technique.

Methodology:

1. Study and Analysis Phase

   Conducting a literature review to identify the current techniques and provide a foundation for the experimental phase.
2. Experimental Phase

   Train and test a variety of deep learning models on image classification datasets, using techniques such as cross-validation, to evaluate their performance. Use a range of metrics to evaluate the performance of the models, including accuracy, prediction, recall and F1 score.

Outputs:

1. Comparison of different deep learning techniques for image classification tasks.
2. Identification of strengths and limitations of each technique.
3. Insights into the practical considerations for using these techniques in applications.

Benefits:

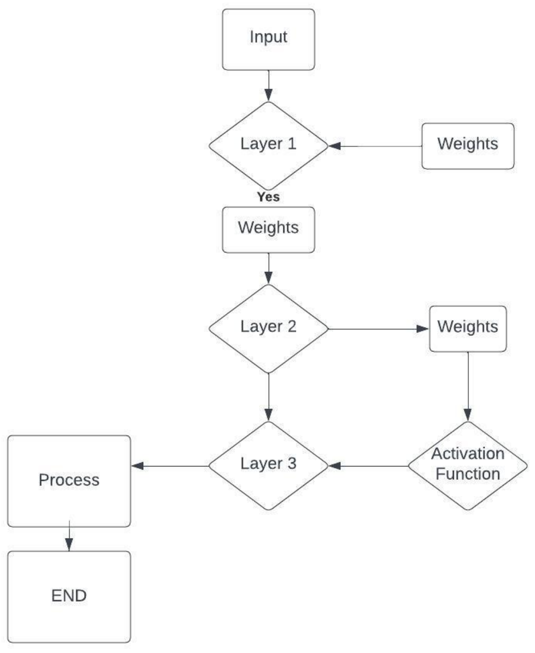
1. Identification of the most effective techniques and their potential applications.

2.2 Functionality

The functionalities of the project are:

1. Data preprocessing
   The program should be able to process input data and convert them into a suitable format for the deep learning algorithms.
2. Model development
   The project should be able to develop deep learning models that can effectively classify image data.
3. Model evaluation
   The project should be able to evaluate the performance of deep learning models using appropriate metrics.

2.2 Data Flow Diagram



3 System Architecture

3.1 Architectural Design

Convolutional Neural network

CNNs are a type of deep learning neural network that are specifically designed for image recognition and classification tasks.

CNNs are composed of multiple layers, each of which performs a specific type of operation on input data.

1. Convolutional Layer
   The first layer applies a set of filters to the input image. The filters slide over the image, computing a dot product between the filter and local pixel values at each location. The output of this layer is a feature map, which captures the presence of specific features in the input image.
2. Pooling Layers
   They downsample the feature maps to reduce their spatial size and make the neural network more computationally efficient.
3. Dense Layers
   They perform the classification task. They form a series of matrix multiplications to transform the output of convolutional layers into a vector of probabilities for each possible class.
4. Training
   During training, the weights of CNNs are adjusted using back-propagation and gradient descent to minimise a loss function that measures the difference between the predicted and the true class labels for each image in the training dataset.

Recurrent Neural network

RNNs are a type of deep learning neural network that are specifically used for sequential modelling tasks, such as natural language processing and speech recognition, rather than image classification. But, it can be used in image classification for certain tasks, such as video classification.

General steps for using RNNs for image classification:

1. Preprocessing
   The input image is preprocessed to break it down into smaller patches or frames in a video.
2. Encoding
   Each patch or frame in the input sequence is encoded into a fixed length vector representation, such as CNN or a feature extraction method such as Histogram of Oriented Gradients.
3. Sequence Modelling
   The sequence of encoded vectors is fed into an RNN, which processes each vector in the sequence in a sequential manner, updating its hidden state with each new input.
4. Classification
   The final hidden state of the RNN is passed through one or more dense layers to perform a classification task, where the output is a probability distribution over the possible classes.

3.2 Decomposition Description

3.2.1 Convolutional Neural Network

A Convolutional Neural Network is typically composed of different components that work together to extract and classify features from input images.

1. Input Layer
   The input layer is where the raw image is fed into the network.

2. Convolutional Layer
   The convolutional layer applies a set of filters to each input image to extract special features. Each filter generates a feature ma, which is a 2D array of values representing the activation of the filter across the image.
3. Pooling Layer
   The pooling layer reduces the spatial size of the feature maps generated by the convolutional layer by performing down sampling. Common pooling layers are MaxPooling and L2 Pooling.
4. Activation Layer
   The activation function applies a non-linear activation function to the output of the pooling layer.
5. Dense layer
   It takes the flattened output of the previous layers and performs classification. It consists of neurons that are fully connected to each activation in the previous layer. Its output is a probability distribution over the possible classes.
6. Softmax Layer
   It is a special type of activation layer that is used to generate a probability distribution over the possible classes. Its output represents the probability of each class given the input image.
7. Output Layer
   It is the final layer of the network that produces the output prediction from an image.

3.2.2 Recurrent Neural Network

A Recurrent Neural Network consists of the following components:

1. Input Layer
   The input layer is where the raw sequential data is fed into the network.
2. Hidden State
   The hidden state is a vector that represents the internal state of the RNN at a particular time step. It is updated at each step based on the current input and the previous hidden state.

3. <u>Recurrent Layer</u>

   It is the core component of the RNN, which processes the sequential data by updating the hidden state at each time step.

4. <u>Activation Function</u>

   It is the nonlinear function applied to the output of the recurrent layer to introduce non-linearities into the model.

5. <u>Output Layer</u>

   This is where the final prediction for the sequential data is generated.