Goldsmith University of London
Bsc Computer Science Final Project

Preliminary Report

By
Faizaan Ebrahim

Contents

2.7) Documentation and code quality

2.8) Gantt Chart

3. Prototype Report

3.1 Overview

3.2.1 Evaluation :prototype 1
3.2.2 Future improvements :prototype 1

3.3.1 Evaluation : prototype 2
3.3.2 Future improvements : prototype 2

3.3.1 Evaluation :prototype 3
3.3.2 Future improvements : prototype 3

Introduction

I have chosen *CM3015 Machine Learning and Neural Networks Project Idea Title 1: Deep Learning on a public dataset*.
My reason for choosing this template is due to my fascination with space science.It has the ability to change our perspective on the world and our lives.In terms of data, astronomy is experiencing a data revolution. The volume and variety of astronomical data has increased immensely. This allows for a different branch or route into the field of astronomy whilst still being able to work in the field of data. The sheer volume of data generated by new instruments presents big data challenges. Data management,storage and analysis techniques are needed to handle these datasets.I have a great feeling of anticipation to one day contribute to groundbreaking discoveries, and particularly by new instruments such as the James Webb Space Telescope. What sets the James Webb Space Telescope apart from other telescopes is the use of infrared capabilities. This is important for observing distant galaxies,studying star formation and analysing atmospheres of exoplanets.Due to its size and power it can peer even further into the universe and its past.Redshift ,which is where the light from distant objects is shifted towards longer wavelengths , helps us calculate how far away an object is.Calculating redshifts can help us map out our universe better and therefore better our understanding of the universe.While calculating redshift is not an easy task it is something i would like to attempt either in this project or in the future.

Galaxies are an ever fascinating and mind boggling object in the universe. They are the building blocks of the universe. Machine learning, with its capacity to discern patterns and unveil insights from complex data , is an important aspect of this project. The allure lies in the ability of machine learning algorithms to navigate through data complexities, transforming messy data into coherent patterns. The satisfaction from making sense of intricate datasets is a driving force, compelling exploration into the realm of machine learning. Within the context of astronomy, this technical capability takes on an additional layer of significance. Unravelling the mysteries of the universe requires not only advanced technology but also the help of machine learning to extract meaningful information from massive and intricate datasets.

This report is divided into three chapters. The literature review, Project design and Prototype report. The primary goal of the literature review is to decipher the methodologies in extracting features and classifying astronomical images. Challenges associated and advancements are present, as well as identifying gaps that beckon innovation. Through this exploration, this review not only aims to contribute to the refinement of galaxy images analysis but also to provide insight to future research in astrophysical image classification. The project design gives an overview of the project as well as important information such as the domain,design choices,the overall structure of the project,important technologies and methods,project plan and a test and evaluation plan.The last chapter concerning the prototypes, are prototypes that are seen as the most important to the project. The decision to create these prototypes were based on how confident I would feel on the project as a whole after knowing these features work. Each prototype is evaluated and how they can be improved.

The combination of machine learning and astronomy in this project is an interesting blend of scientific curiosity and advanced technology. Beyond the main goal of the project of creating

a system to categorise galaxies, the project will aim to develop skills and insights that can help me with future tasks. Navigating the vast frontiers of astrophysics, the satisfaction derived from unravelling complex data, along with the potential for remarkable astronomical discoveries, maybe not in this project but in the future, serves as my motivating force driving this project.

1.1 Background

The vastness of space and the unfathomable amount of galaxies in the universe has captivated the imagination of astronomers for centuries. As technological advancements have allowed us to look deeper into the cosmos , the need for accurate and efficient methods of galaxy classification has become increasingly apparent. Understanding the morphological characteristics of galaxies provides important information into their formation , evolution and maybe even something we may be yet to discover.

1.1.1 Motivation

Galaxy classification is a fundamental task in astronomy and has traditionally relied on manual efforts by expert astronomers. However with the rise in the amount of data available to us, there is a need for a more automated approach. An example of this is the recent James Webb Space Telescope, which can peer further into the universe than ever before and use infrared cameras that have not been used before. With billions of new galaxies being discovered, the need for automation is not important but would be necessary. As for personal motivation, as mentioned in the introduction. This field is a passion and hopefully a future endeavour.

1.1.2 Objectives

In this literature review the goal is to examine the current landscape of image analysis in astrophysics, with a specific focus on galaxy classification. His objective is to gain insights into established methodologies for feature extraction and classification of astronomical images, understanding the challenges and advancements in this domain. Exploring the evolution of techniques from traditional methods to more recent machine learning approaches. This review aims to find gaps and areas where there can be improvement as well. The ultimate aim is to contribute to the enhancement of galaxy image analysis and to provide valuable insights for future research in astrophysical image classification.

1.2

Galaxy classification

1.2.1
Manual classification Methods

Galaxies have diverse shapes and sizes. Edwin Hubble devised a classification system in 1936 for galaxies that groups them into four main classes: spirals,barred spirals,ellipticals and irregulars. Spiral and barred spiral galaxies are further categorised based on the size of their central bulge and the texture of their arms. The Hubble classification has been modified over time but the basic structure depicted as the “tuning fork diagram” still remains a tool for describing galaxies. But before Hubble's classification scheme , astronomers primarily relied on visual inspection and manual methods to classify galaxies. There was limited understanding of galaxy morphology.

1.2.2
Challenges in traditional methods.

The following are challenges that come with the traditional methods of classification:

Subjectivity: Visual classification can be subjective and vary between people. Having people with some knowledge on the topic would be helpful. As well as finding willing volunteers.

Data: With new technologies , there is an increase in the amount of data. Manual classification would become very time-consuming. Keeping up with analysis would be challenging. As well as data management and storage.

Limited parameters: Traditional methods often focused on a few key morphological features, which might not capture the full complexity of galaxy structures.

Resources: Manual classification required a lot of human resources and was not always feasible for large datasets. It would also increase the time it would take to discover and find new insights.

Human Bias : Human observers might have biases or preconceptions that could influence their classification. This can lead to errors.

1.2.3
Crowdsourced classification

Galaxy Zoo, launched in 2007, stands out as one of the most renowned and expansive citizen science projects. Over the course of a decade, it has engaged volunteers in exploring galaxies, each containing billions of stars within the observable Universe. The project's

primary focus is unravelling the unique characteristics of individual galaxy systems, relying on the shapes of these cosmic entities for information. Utilising data from the Sloan Digital Sky Survey, Galaxy Zoo employs a manual classification system with three choices for volunteers: Smooth, Features, or Disk, each corresponding to distinct galaxy characteristics. Unlike automated models, Galaxy Zoo follows a decision tree method, tailored to human judgement rather than machine-based algorithms.

Beyond its immediate classification goals, Galaxy Zoo also aims to address scientific inquiries, lay the groundwork for morphological studies using new instruments like the James Webb Telescope (JWST) as well as future instruments.Galaxy Zoo's strength lies in its ability to process large-scale data, leveraging the contributions of numerous volunteers to ensure quality control and minimise the impact of errors or biases. Moreover, the project creates a meaningful connection between the public and the scientific community, making astronomy more accessible and engaging.

Significantly, Galaxy Zoo serves as a benchmark for automated methods. The massive dataset, along with classifications from citizen scientists, becomes a testing ground for refining and enhancing automated classification algorithms. This approach contributes to the ongoing evolution of astronomical research methodologies.

1.3
Automated Classification Techniques

1.3.1
Traditional machine learning approaches

These are the three types of learning mentioned in the ‘Deep Learning with Python’ by FRANÇOIS CHOLLET

Supervised Learning:

Supervised learning is the most common case. It consists of learning to map input data to known targets, given a set of examples. Many applications are within this category, such as speech recognition, image classification along with many others.

Self-Supervised learning :

Self - supervised learning is a specific instance of supervised learning. Self-supervised learning is supervised learning without human-annotated labels. There are still labels involved (because the learning has to be supervised by something), but they’re generated from the input data. Self-supervised learning can be reinterpreted as either supervised or unsupervised learning, depending on whether you pay attention to the learning mechanism or to the context of its application.

Unsupervised learning:

Unsupervised learning consists of finding interesting transformations of the input data without the help of any targets or human-annotated labels, for the purposes of data visualisation, data compression, or data denoising, or to better understand the correlations present in the data at hand. Dimensionality reduction and clustering are well-known categories of unsupervised learning

1.3.2

Keras API

An important feature of the functional api is Layer Weight Sharing. This feature allows the reuse of weights across different processing branches, providing flexibility in model design. With regards to my project this feature can be employed to share common representations for various features extracted from galaxy images, enhancing the models ability to learn from multiple sources simultaneously.

The Keras functional API is indispensable in advanced design of deep neural network architectures . The layer weight sharing and treating models as layers provide a robust framework. This flexibility is important when dealing with diverse data sources . This allows for a more comprehensive learning from various inputs.

For my project this means I can use my model to learn from both the images and any other features such as brightness and shape or features that may be found using a self-supervised learning model.

1.4

Related work

Paper 1

Galaxy Classification: A deep learning approach for classifying Sloan Digital Sky Survey images Sarvesh Gharat,1★ Yogesh Dandawate1

For their data collection the researchers used the Sloan Digital Sky Survey catalogue with 21 785 galaxy samples from Galaxy Zoo. The images were three-band coloured(r,g,i band) and classified into 10 classes.The Python package they used to import data was astroNN.

The data was divided into 10 classes.

They used convolution layers for high -level feature extraction. Applied MaxPoolong for downsampling feature maps , used ReLU (Rectified Linear Unit) activation function for non-linearity and flattened feature maps into 1D arrays.

Their classifier model used a feature extraction model output as input to the classifier .

In their results they mention that previous galaxy classification studies typically focused on fewer than 6 classes, but this study aims to classify into 10 classes.

The accuracy of the model is calculated based on the number of correctly predicted images across all classes.

This paper they classify based on shape and size of the galaxy. This was very insightful and has a close relation to my intended project. They used Convolution layers, which is similar if not the same to my approach of using Keras functional API.

Paper 2

GALAXY CLASSIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS
By Honghui Shi

This paper specifically focuses on morphology. The study of their shape ,structures and visual features. The study primarily utilises machine learning , emphasising convolutional neural networks to automate the process.

- The primary goal is the automated classifier based on the morphology of galaxies.
- The data is from the SLoan DIgital Sky Survey (SDSS) and the Galaxy Zoo Project.
- The study employs supervised learning where machine learning models are trained on a labelled dataset. Galaxy images were paired with known classifications
- He initially addresses the classification task as a regression problem predicting certain properties of galaxies but later transforms it into a multi-class classification task. The study does not mention the specific morphological classes used.
- He concludes that, while traditional methods are effective , CNNs, through automatic feature learning , show a better performance in classifying galaxies based on performance. This study is focused on supervised learning.
- In his conclusion he emphasises the transition from manually designed features to automated features extraction through CNNs, showing the potential for improved galaxy classification.

This study provides valuable insights into the application of machine learning with emphasis on CNNs. The emphasis on morphology and the comparative analysis of traditional and moderns make it a significant contribution to the field.

What makes my project different is the use of unsupervised learning instead of supervised learning. I don't have labelled data but he had access to labelled data. Combining the knowledge of the previous paper which categorises galaxies into more than 3 types, I will use unsupervised learning to hopefully get similar results.

This study only used morphology as classification whereas in my case , while morphology is the primary feature, I will incorporate other features as well.

His use of CNNs is something I will be doing as well but with the difference of different features being extracted and using the functional api as well.

1.5
Conclusion

In conclusion, the reviewed literature highlights the potential of machine learning, especially deep learning models like convolutional neural networks. Future research should focus on refining these models, exploring combinations of image-based and photometric features, and addressing challenges like dataset biases and computational efficiency. As upcoming surveys demand rapid and accurate predictions, collaboration between astronomers and machine learning experts is crucial for developing scalable and efficient algorithms, advancing our understanding of the universe's large-scale structure.

Citation and images

Citation :

Certainly! Here are the revised citations in Harvard style:

Gharat, S. and Dandawate, Y. (2021) 'Galaxy Classification: A Deep Learning Approach for Classifying Sloan Digital Sky Survey Images'. Preprint. [URL or DOI if applicable].

Chollet, F. (2017) Deep Learning with Python. Manning Publications.

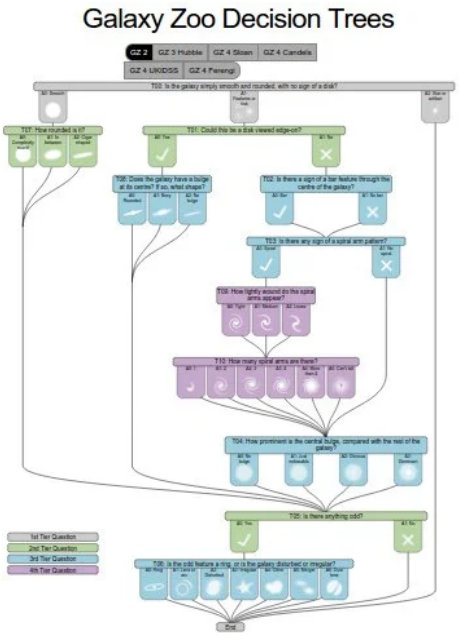
Chollet, F. (2015) 'Keras: The Python Deep Learning library', [Online] Available at: https://keras.io.

Shi, H. (2016) 'Galaxy Classification with Deep Convolutional Neural Networks', Master of Science thesis, University of Illinois at Urbana-Champaign, Urbana, Illinois.

Zooniverse. (n.d.) 'Galaxy Zoo'. Zooniverse. Available at: https://www.zooniverse.org/projects/zookeeper/galaxy-zoo/ (Accessed: 18 November 2023).

Zooniverse. (n.d.) 'Galaxy Zoo Research'. Zooniverse. Available at: https://www.zooniverse.org/projects/zookeeper/galaxy-zoo/about/research (Accessed: 20 November 2023).

Images



Project Design

2.1
Overview of project

This project aims to employ deep learning techniques for comprehensive galaxy analysis in the absence of explicit labels. Focusing on a dataset of galaxy images, the objective is to get labels for the unlabelled dataset using unsupervised learning methods and assigning labels to the images or to implement a self-supervised learning method as well.After attaining labels and getting a new dataset, to the create a galaxy classifier using The Keras functional api. While self-supervised learning was the first idea and method, it would be less reliable due to the quality of the images and can be quite tedious compared to unsupervised learning.The derived features will be used to train a model for galaxy type classification.Employing a dataset of galaxy images, the project aims to create robust models using advanced techniques from “Deep Learning with Python” by Francois Chollet (2017).

2.2
Domain and users

Astrophysicists and Astronomers :

Primary users are scientists in the field of astrophysics and astronomy who are trying to enhance their understanding of galaxies.
They utilise the developed models to extract valuable insights from galaxy images and classifications.

Astronomy Enthusiasts :

People with an interest in astronomy but without expertise in the field can benefit from visualisations and simplified explanations generated by the project.
The project can be an educational tool for enthusiasts to learn more about galaxies.

There may be many other classification models but not very accessible . This project can help me as an individual understand galaxies better and help me develop skills in a field I have an extreme interest in.

2.3
Justification

Challenges in Labelling:

- Obtaining explicit labels for a large dataset of galaxy images is challenging and resource-intensive.Getting labels with human intervention will be close to impossible.

Unsupervised learning provides an alternative by allowing the model to derive implicit patterns and groupings from the data itself.

Feature Learning:

- Unsupervised learning enables the model to learn intricate features directly from the galaxy images. These features can capture important characteristics, shapes, and patterns inherent in the data.

Implicit Label Generation:

- By exploring clustering techniques related to the galaxy images, the model generates groupings during the training process. These groupings will help understand the different galaxy classes.

Downstream Task Flexibility:

- The learned features, serving as implicit groupings, can be flexibly utilised for various tasks further down, such as in this case , galaxy type classification.This flexibility is crucial in scenarios where labels are limited.

Data Efficiency:

- Unsupervised learning is known for its data-efficient nature. Leveraging the abundant unlabeled data available in the galaxy image dataset, the model can still derive meaningful patterns and groupings

Keras functional API:

The reason I am using Keras functional API in this project is because of its adaptability and versatility. This API offers the flexibility so I can handle diverse features like brightness,shape and potentially clustering results. It allows multiple inputs and outputs , shared layers and model reuse.

Deep Learning:

Deep learning models can learn intricate relationships within data , creating more accurate predictions and classifications.

Mixed-input models:

Mixed - input models , combining image data with other features, provides flexibility. This accommodates users who may prefer a hybrid approach.

Automated analysis for Surveys:

Automation is crucial for large -scale astronomical surveys.By automating the estimation and classification, the project will align with the needs of survey organisations and observatories.

2.4
Structure of the project

Data Collection and Preprocessing:

- Obtain a large dataset of galaxy images.
- Normalise and augment images as necessary for unsupervised learning.

Unsupervised Learning:

- Explore clustering techniques such as k-mean to identify patterns and groupings in the images.While k-means clustering is the initial choice , alternative techniques may be explored for improved performance.
- Design an unsupervised learning model to extract features and implicit groupings.
- Train the unsupervised model on the unlabelled galaxy images to gain meaningful representations.

Clustering Analysis/labelling

- Evaluate and interpret results of clustering techniques to understand potential groupings of galaxies.
- Visualise the groupings to help with the labelling process.
- Implement validation techniques to avoid overfitting during unsupervised learning.

Documentation and Version Control:

- Use Jupyter Notebooks for coding, analysis, and documentation.
- Implement version control using Git.

2.5
Key Methods and Technologies

Data Processing and Analysis:

- Python: Utilising Python for data processing and analysis due to its extensive libraries and ecosystem.

- NumPy: Employing NumPy for efficient numerical operations on the galaxy image data.

- PIL (Python Imaging Library) and OpenCV: These libraries are crucial for image processing tasks, such as normalisation, augmentation, and other preprocessing steps.

Deep Learning Frameworks:

- TensorFlow and Keras: Leveraging these powerful frameworks for building and training deep learning models. The Keras functional API provides flexibility in handling various features, making it suitable for mixed-input models.

Unsupervised Learning Techniques:

- Evaluate the performance of clustering techniques in identifying patterns.
- Assess the quality of features extracted by the model
- Explore alternative unsupervised learning methods for comparison

Manual Labelling:

- Creating a method for labelling the images clustered from the unsupervised technique mentioned above.

Clustering analysis:

- Analyse the distribution and characteristics of galaxy grouping identified by clustering

Overall Model Evaluation:

K-Fold Cross-Validation: Implement K-Fold Cross-Validation to assess the model generalisation.

Documentation:

- Jupyter Notebooks: Utilising Jupyter Notebooks for coding, analysis, and documentation. This interactive environment is ideal for presenting the project in a clear and organised manner.

Version Control:

- Git: Implementing version control using Git to track changes, collaborate efficiently, and maintain a clear history of the project's development.

2.6)Evaluation Plan

Unsupervised Learning:

- Performance of Clustering techniques: implement a metric to evaluate the performance.
- Assess the ability of clustering algorithms to identify distinct groups.
- Evaluate quality of features extracted.

Fine-Tuning for Classification:

- Classification Accuracy: Measuring the accuracy of the model on the smaller labelled dataset with galaxy types to assess its capability in correctly classifying galaxies.
- Precision, Recall, and F1 Score: Exploring precision, recall, and F1 score metrics for different galaxy types to understand the model's performance in more detail.

Overall Model Evaluation:

- K-Fold Cross-Validation: Implementing K-Fold Cross-Validation to robustly evaluate the model's performance by partitioning the dataset into multiple folds. This helps in assessing the model's generalisation across different subsets of the data.
- Analysis of Performance: Analysing average performance and variance across folds to ensure consistency and reliability in the model's predictions.

2.7
Documentation and Code Quality

Jupyter Notebooks Review:

- Regularly reviewing Jupyter Notebooks for clarity, completeness, and proper documentation of code and analysis.

Version Control Reflecting Changes:

- Ensuring that version control accurately reflects meaningful changes in the project, making it easier to track the evolution of the model and experiments over time.

Gantt chart

Task	Start Date	End Date	Duration	Dependencies	Progress
Project Design	12/01/2023	12/07/2023	7 days	None	Completed
Prototype 1	12/08/23	12/14/23	7 days	Project Design	Completed
Prototype2	12/15/2023	12/18/2023	3 days	Prototype 1	Completed
Prototype 3	12/19/2023	12/28/2023	10 days	Prototype 1 and 2	Completed
Preliminary Report	01/02/2023	01/05/2023	4 days	Prototype 1,2,3. Project design. Literature review	Completed
Rechecks and finishing up	01/06/2024	01/07/2023	2 days	All work done so far	Completed

Prototype Report

Overview

Prototype 1: Image Clustering
The use of unsupervised learning to cluster my unlabeled galaxy images based on certain features

Prototype 2: Manual Labelling
Manually inspect and label the clusters created in Prototype 1.

Prototype 3: Image Classification
A simple image classification model using the labelled dataset created in Prototype 2 using Keras functional api

Evaluation:Prototype 1(Image Clustering)

The necessary libraries, including TensorFlow, Keras, scikit-learn, pandas, and matplotlib, are imported. Image file paths are specified for processing, and a pre-trained VGG16 model is loaded without the top classification layer to extract features.VGG16, or Visual Geometry Group 16, is a widely used convolutional neural network (CNN) architecture for image classification.It is known for its simplicity. Pre-trained on datasets like ImageNet,which is a large dataset with millions of labelled images used to train and test computer vision models, it is often used as a feature extractor or fine-tuned for specific image-related tasks. There is a function that preprocesses images and extracts VGG features, brightness (mean pixel value), and colour (mean RGB values). K-means clustering is then applied separately to these features, generating cluster labels for each image. The results are organised into a DataFrame, including file paths and cluster labels. Custom functions facilitate the visualisation of images from specified clusters. Images from each cluster are then displayed for VGG features, brightness, and colour, providing a visual representation of the clusters.

Future Improvements:Prototype 1(Image Clustering)

Parameterization:

Parameterizing values like the number of clusters can enhance the code's flexibility and adaptability

Validation and Error Handling:

I could implement validation checks and error handling. This will benefit me if i run into future problems or errors and help prevent them

Efficiency Considerations:

Making the code more efficient will be taken into consideration as I start working on the bigger dataset. Especially in the feature extraction step.

Integration of Domain Knowledge:

I will try integrating domain knowledge into the clustering process for a more meaningful and interpretable set of results.

Evaluation: Prototype 2 (Manual labelling)

The reason why I added this feature as a prototype is because I was unsure how I would implement it and needed that confidence going into the next phase of my project. ITs a simple feature but can be implemented in different ways and why I felt it important to create a prototype. In this prototype, I replace numeric values with string values for a better understanding and human-readability. The labels were updated for the vgg, brightness, and colour columns. The code then saves the labelled DataFrame to the existing CSV file. The visualisation step helps me see the prototype is working well. Overall, this prototype serves as a foundational step in enhancing the interpretability and usability of the dataset.

Future Improvements: Prototype 2 (Manual labelling)

Combined Labels:

I could add more columns such as 'combined_labels,' by concatenating the three cluster labels. This label could give a different representation of the data, capturing potential interactions between different clustering methods.

Dataset Preservation:

I could save the DataFrame with updated labels to a new CSV file .
This way I could reuse the old dataset if necessary.

Visual Inspection:

I could improve on the display by adding the ability to display the images for each unique label, and this would help in visually inspecting the clusters.

Automation and Scalability and integration with ML Pipeline:

I could automate the labelling and integrate it into my Machine learning Pipeline. This could make the transition from labelling to model training and evaluation seamless.

Interactive Visualisations:

I could try to implement interactive visualisation tools or dashboards for dynamic exploration of labelled clusters,to create a more user-friendly analysis experience.

Evaluation: Prototype 3

The prototype begins by loading and preprocessing a dataset that includes both image files and additional features. This combination allows for a more comprehensive understanding of the data and could serve as a better input for the neural network.

The choice of the Keras functional API aligns with the template stating that I should use advanced techniques from chapter 7 of ‘Deep Learning with Python’ by FRANÇOIS CHOLLET . The model consists of separate branches for processing image and feature data, leveraging convolutional layers for image features and dense layers for additional features. The decision to concatenate these branches before the final classification layer enables the model to capture both visual and feature-based patterns in the data.The dataset will consist of multiple types of data which is why this approach is fitting.
I chose the Adam optimizer because it's great at handling different types of data.
Breaking down the model, it's a mix of convolutional layers for handling images and dense layers for extra features. I used ReLU(Rectified Linear Unit) activation functions to capture patterns in the data.ReLu is an activation function that is commonly used in neural networks. It's simple, making it a popular choice. It was also used in paper 1 in my literature review .The final layer uses softmax activation for handling multiple classes, providing probabilities for each class.Softmax is a standard activation function, it is simple and widely used, especially in the output layer.It is easy to interpret and also provides a probability distribution which is very helpful as knowing the models uncertainty or confidence of the prediction is important.

Training the model involves splitting the dataset, loading and preparing images, and putting everything together with the chosen optimizer and metrics.
The final dense layer uses softmax activation for handling multiple classes, providing probabilities for each class.Softmax is a standard activation function, it is simple and widely used, especially in the output layer.It is easy to interpret and also provides a probability distribution which is very helpful as knowing the models uncertainty or confidence of the prediction is important. It is often paired with cross-entropy which I used in my loss function. The loss function is set to 'categorical_crossentropy', which is common for multiclass classification problems

I also added some helpful callbacks, like early stopping and model checkpointing, to make sure our model learns effectively without overfitting.But these will be implemented properly as I continue to work on this prototype.

In summary, this Keras API prototype is a thought-out combination of features, optimizers, and model design in hope to build a strong and efficient image classification model.

Future Improvements: Prototype 3

Ensemble Learning:

Experiment with ensemble techniques, combining predictions from multiple models for potentially improved performance.Ensemble techniques involve combining the predictions from multiple machine learning models to create a more robust and accurate prediction than any individual model. The underlying idea is that different models may capture different aspects or patterns within the data, and by aggregating their predictions, the ensemble model can achieve better generalisation and performance.Potential Benefits include improved generalisation , increased robustness and enhanced performance.

Data Augmentation:

Introduce data augmentation techniques during image preprocessing such as zoom, crop,flip, rotate, to increase model robustness.It takes the approach of generating more training data from existing training samples, by augmenting the samples using the techniques mentioned. The aim is that during training , the model will not see the same picture twice. This will help the model generalise better and expose it to more aspects of the data.

Hyperparameter Tuning:

Fine-tune hyperparameters such as the number of filters, kernel sizes, and dense layer units to optimise model performance.

Regularisation Techniques:

Implement dropout or batch normalisation layers to reduce overfitting.This step could take the most time. I will have to repeatedly modify the model, train it, evaluate on the validation data , modify it again, and repeat.

More Complex Architectures:

Explore deeper or more complex neural network architectures to capture intricate patterns in the data.

TensorBoard Integration:

Add TensorBoard integration for visualising training and validation metrics. TensorBoard helps to visually monitor everything that goes on inside the model during training. TensorBoard would give me access to several helpful features such as , Visually monitoring metrics during training, visualising my model architecture, visualising histograms of activations and gradients and exploring embeddings in 3D.

Experiment with Optimizers:

Try different optimizers apart from ‘Adam’ to observe their impact on convergence. One of these could be ‘Nadam’ which is similar to Adam.

Model Monitoring:

In my prototype and as I stated in my evaluation. I will attempt to include early stopping,model checkpointing and TensorBoard callbacks.

Early Stopping:
Early stopping is like basically watching over the model during training. It monitors the model's performance on a validation set, and if it sees that the performance stops improving or if it starts to degrade, it stops the training process before it overfits.

Model Checkpointing:
Model checkpointing is like getting snapshots of the model's progress at certain points during training. These checkpoints are saved. If the training process is interrupted or if I want to use the model at different stages, I can reload it from a checkpoint instead of starting training from scratch.

TensorBoard Callback:
TensorBoard is like a dashboard for monitoring a model's training. The TensorBoard callback is a tool that runs alongside the training process and logs various metrics, such as loss and accuracy, over time. I could visualise these metrics in TensorBoard for better understanding and analysis.

Early stopping ensures no overtraining.
Model checkpointing saves the best versions of the model during training.
TensorBoard callback creates visualisations to help understand and analyse the training process.