TUNIS BUSINESS SCHOOL
UNIVERSITY OF TUNIS

Magnetic Resonance Imaging Classification

A Machine Learning Approach

APIRestful

By
Eya Briki

Professor:

Montassar Ben Massoud

Undergraduate Project

Web Service Course

Department of Information Technology

January 15, 2023

Abstract

This study aimed to develop a machine learning approach to classify Magnetic Resonance Imaging (MRI) images as healthy or diseased based on the presence of tumors. A convolutional neural network (CNN) model was trained on a dataset of MRI images from various hospitals and patients of different genders. The use of APIs for dataset creation and management and the storage of data in a PostgreSQL database facilitated easy access and use of the data for future studies. The results of this study demonstrate the potential of machine learning to improve the accuracy and efficiency of MRI image classification, and suggest that this approach could be further developed and applied to other medical imaging tasks."

Contents

Chapter 1

introduction

Magnetic resonance imaging (MRI) is a medical imaging technique that uses powerful magnetic fields and radio waves to produce detailed images of the internal structures of the human body. MRI is widely used in the diagnosis and treatment of various medical conditions, including brain tumors, multiple sclerosis, and cardiovascular diseases.

The availability of large datasets of MRI images and other medical imaging data is essential for the development and evaluation of machine learning models for image analysis and diagnosis. However, creating and maintaining these datasets can be challenging, as they often require the cooperation of multiple hospitals, researchers, and other stakeholders.

1.1 Problem Statement

The goal of this project is to develop a machine learning approach to classify MRI images as healthy or diseased based on the presence or absence of tumors. This task is important because accurate and efficient classification of MRI images can improve the diagnosis and treatment of various medical conditions. In addition, we will create APIs to facilitate the creation and management of datasets for future studies.

1.2 State-of-the-Art

In recent years, machine learning techniques have been applied to the analysis of MRI images to improve diagnosis and treatment of various medical conditions. Machine learning algorithms can analyze large amounts of data and learn patterns and relationships that may not be immediately apparent to human analysts. This ability makes machine learning particularly well suited for the analysis of medical imaging data, where there can be large amounts of data and subtle patterns that may be indicative of disease.

1.3 The Proposed Methodology

To classify MRI images as healthy or diseased, we used the following steps:

- Creation and management of a dataset of MRI images from a variety of hospitals and patients of different genders.
- image preprocessing and feature extraction using various techniques,the images to remove noise and enhance the contrast.
- Training and evaluation of a convolutional neural network machine learning model on the extracted features.
- creation of API endpoints for diagnosis and management.

1.4 Thesis Outline

The rest of this report is organized as follows:

- In Chapter **??**,we review the related literature on MRI image analysis and machine learning.

- In Chapter **??**,we describe the dataset and the preprocessing and feature extraction techniques that we will apply to the images.

- In Chapter **??**, we present the machine learning models that we trained and evaluated, and we will discuss the results of our evaluation.

- In Chapter **??**,we will discuss the implications of our work and outline some directions for future research.

Chapter 2

Dataset and CNN Model

This chapter has the following objectives:

1-Description of the dataset

2-preprocessing and feature extraction techniques applied to the MRIs.

3-Presentation of the machine learning model trained and evaluated.

2.1 Dataset

creation of a dataset of Magnetic Resonance Imaging (MRI) images for the purpose of classifying them as either healthy or diseased based on the presence or absence of tumors. this dataset uses a labeled downloaded set of images classified as 'yes' or 'no', with each MRI assigned with a unique ID, gender, hospital it was taken in, and class:[tumor, healthy]

	id	MRI	gender	hospital	class
0	ee1224e6-2133-43e6-a81a-4b7732ba9dc4	[[[1, 1, 1], [1, 1, 1], [1, 1, 1], [1, 1, 1], ...	female	Arnaud-de-Villeneuve, Montpeullier	tumor
1	9e88006a-8f9a-489d-bc3b-8cd1fa3cfda5	[[[2, 2, 2], [3, 3, 3], [3, 3, 3], [2, 2, 2], ...	male	Arnaud-de-Villeneuve, Montpeullier	healthy
2	a193a524-0c9a-4b53-9dc2-9865a4efda62	[[[11, 11, 11], [11, 11, 11], [11, 11, 11], [1...	male	Arnaud-de-Villeneuve, Montpeullier	healthy
3	ff3a2fa2-420b-4c80-82dc-cd2334c62408	[[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...	female	Centre Hospitalier Universitaire de Nice ,Nice	healthy
4	1a0646c4-4e96-401a-a3bb-d56cbd00a2af	[[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...	female	Polyclinique Oxford, Cannes	tumor
...	...	...	...	...	...
240	82132d12-9efb-4012-8c68-2c2192b1e670	[[[11, 11, 11], [10, 10, 10], [9, 9, 9], [7, 7,...	female	Hôpital de la Croix-Rousse, Lyon	tumor
241	e6ef9776-31be-4deb-be78-278f613eef99	[[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...	male	Centre Hospitalier Universitaire de Nice ,Nice	tumor
242	21c2d011-7ee1-4ecb-bdd8-ea019fe58a44	[[[24, 24, 23], [28, 29, 27], [32, 33, 31], [3...	male	Hôpital de la Croix-Rousse, Lyon	tumor
243	cba4ba81-62e5-43a3-9440-a10be96b0d9d	[[[6, 6, 6], [6, 6, 6], [6, 6, 6], [6, 6, 6], ...	female	Centre Hospitalier Universitaire de Nice ,Nice	healthy
244	ccef7310-ecb2-46ab-b303-52fee87fd408	[[[10, 10, 10], [8, 8, 8], [8, 8, 8], [8, 8, 8...	male	Centre Hospitalier Universitaire de Nice ,Nice	tumor

2.2 image preprocessing

Create a function to crop the brain out of an given image:

1-Resize the image to 256x256 pixels.

2-Convert the image to grayscale.

3-Apply a Gaussian blur to the image.

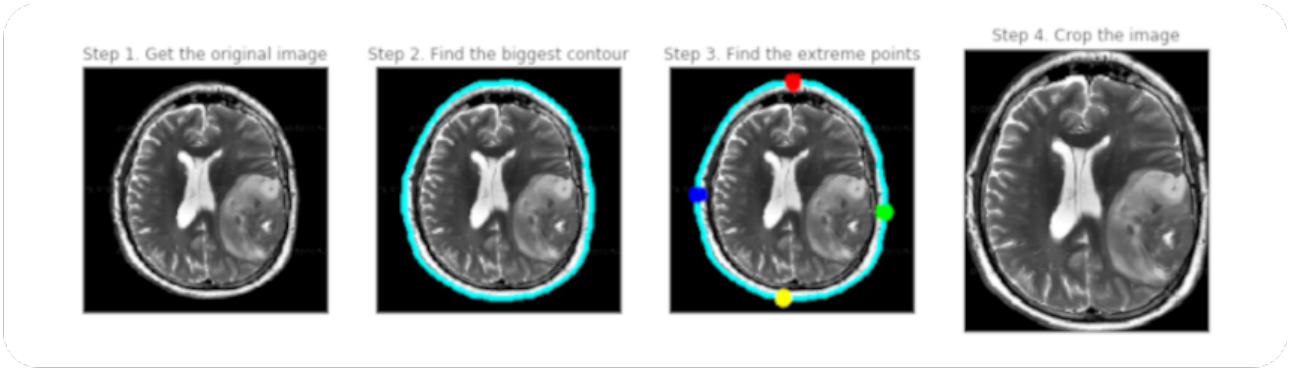
4-Threshold the image by binary thresholding.

5-Perform a series of erosions and dilations to remove any small regions of noise.

6-Find contours in thresholded image, then grab the largest one.

7-Find the extreme points.

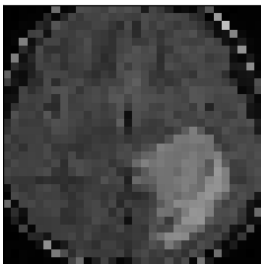
8-Crop the image using the extreme points found in the previous step.



normalize the image:

1-Resize all the images in the new list to 32x32 pixels.

2-Convert the pixel values to float and normalize them by dividing by 255.



2.3 CNN model

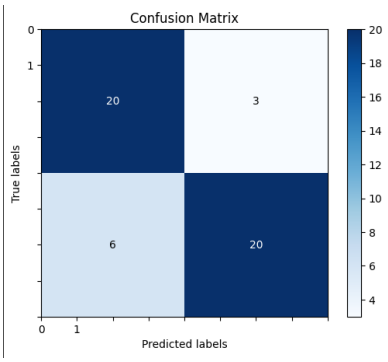
defining a convolutional neural network (CNN) using the TensorFlow Keras library. The model is a sequential model and contains the following layers:

-A convolutional layer with 16 filters, a kernel size of 9, and a ReLU activation function. -A max-pooling layer with a pool size of 2, which reduces the spatial dimension of the feature maps. -A dropout layer with a rate of 0.25, which randomly drops out some neurons during training to prevent over-fitting. -A convolutional layer with 36 filters, a kernel size of 9, and a ReLU activation function. -A max-pooling layer with a pool size of 2, which reduces the spatial dimension of the feature maps. -A dropout layer with a rate of 0.25, which randomly drops out some neurons during training to prevent over-fitting. -A flatten layer which is used to flatten the output of the last convolutional layer to be able to connect it to a fully connected layer. -A dense layer with 512 neurons and a ReLU activation function. -A dropout layer with a rate of 0.15, which randomly drops out some neurons during training to prevent over-fitting. -A dense layer with 1 neuron and a sigmoid activation function, which is used to make binary predictions.

The model is then compiled using binary cross-entropy loss function and Adam optimizer and accuracy metric.

2.4 Model evaluation

2.4.1 Confusion matrix



The confusion matrix shows the number of true positives (20 in this case), true negatives (20 in this case), false positives (3 in this case), and false negatives (6 in this case).

2.4.2 Accuracy and f_score

- The F1 score is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score. The F1 score of 0.816 indicates that the model has a relatively good balance between precision and recall.
- Accuracy is the ratio of correctly predicted observation to the total observations. In this case, the accuracy is also 0.816.

	Metric	Value
0	Accuracy score	0.816327
1	f_score	0.816327

Chapter 3

Security and authentication

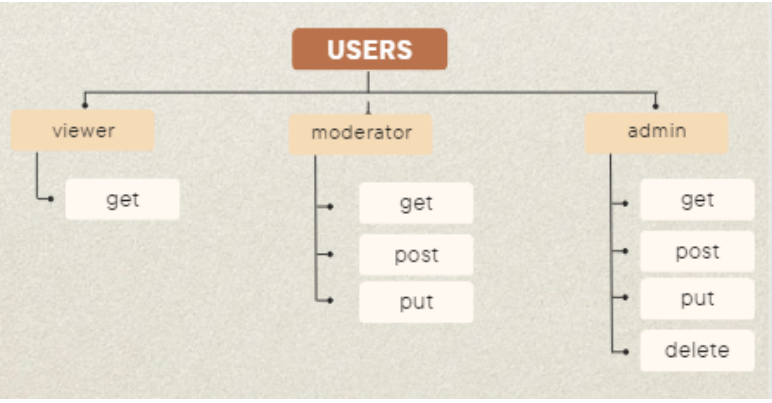
In this chapter, a variety of methods for security and authentication are exposed. along with users hierarchy and permissions.

3.1 users dataset

a set of doctors and technicians is created to store the credentials of each user. each with a unique id, name, email, password, hospital, and role. the email and password, which is hashed using the 'bcrypt' library.

original password : K4N(UZ+d

hashed password: b'$2b$12$JqoAVY0N62ByizjpiAQPeOER90SbT0JZv.QZstKrADQOP6nsLMLde'



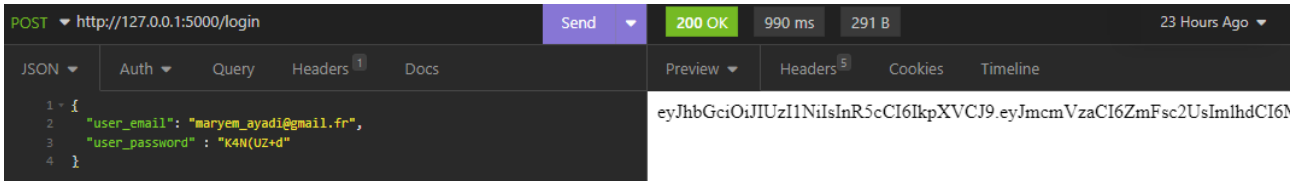
3.2 JSON Web Token (JWT) authentication:

the app.config['JWT_SECRET_KEY'] variable is being used to set a secret key for the JSON Web Token (JWT) authentication. This secret key is used to sign the JWT and should be kept secret and not shared with anyone. The JWTManager class is being initialized with the Flask application app, which enables JWT functionality in the application.

3.3 login route

The /login route handles a POST request that takes the email and password as input in the request's JSON body.

If the email and password are correct, the code then calls the the 'create_access_token() 'function passing the email as the identity parameter. This function generates a JWT that contains the user's role as the identity claim and returns the access token.The client can then use this token to authenticate future requests to the server.



The jwt_required decorator is used to protect routes and views that require a valid JWT to access. Once the decorator is added to a view function, it will check the request headers for a valid JWT. In this project, the decorator was used only for the alteration and deletion functions.

Chapter 4

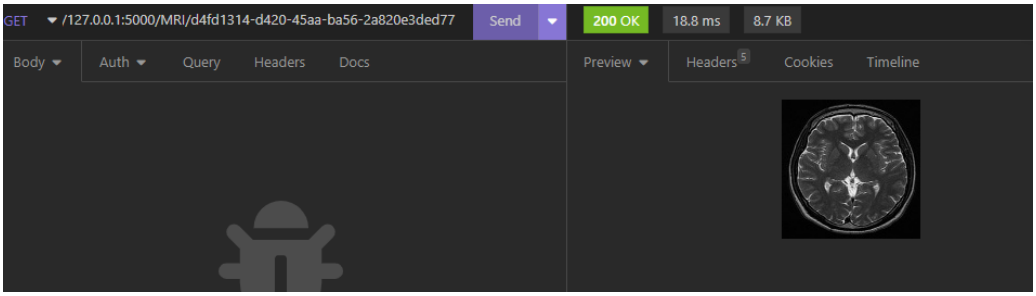
HTTP Methods

This chapter presents the methodology which is based on API requests t

4.1 API endpoints for MRI dataset

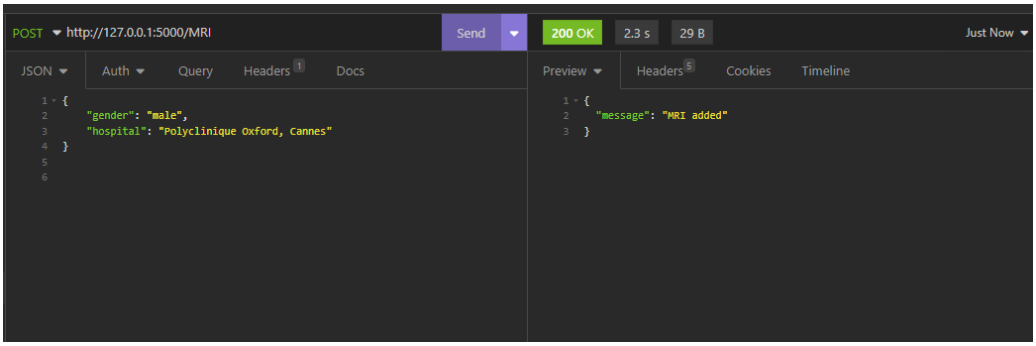
4.1.1 GET request

This function handles a GET requests on the endpoint '/MRI/string:id' the MRI table is fetched from Postgres through an engine created from the 'sqlalchemy' package. This dataset is used to retrieve the MRI image , gender, and the hospital in which the MRI was taken, using a specific MRI id.



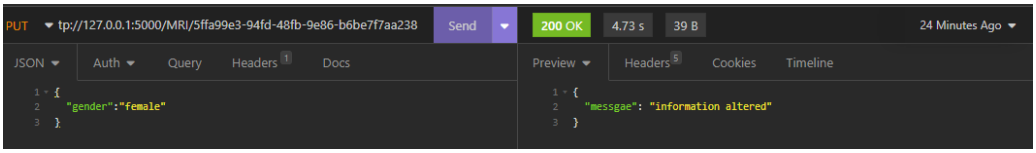
4.1.2 POST request

this function handles a POST request on the endpoint '/MRI' and takes as an argument a path to the file containing the MRI images. the admin or moderator enters the gender of the MRI's owner and hospital in which it was taken. this API uses the CNN model trained on the original MRI dataset to classify this MRI as tumorous or healthy. A new row containing a new unique id and the information provided is appended to the database for future use.



4.1.3 PUT request

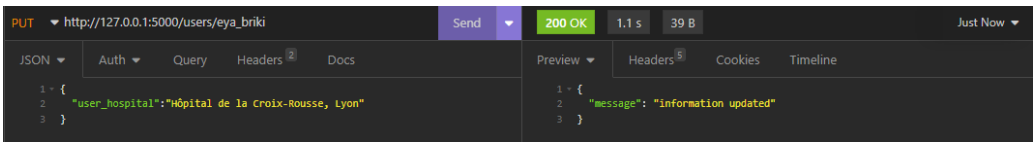
This function handles PUT and PATCH requests on the endpoint '/MRI/string:id' and allows updating specific columns of an existing row in a dataframe by the given id. It reads the request body for new values, searches for the id in the dataframe, updates the columns with the new values if they are passed and loads the updated dataframe to the database.



4.1.4 DELETE request

This function handles a delete requests on the endpoint '/MRI/string:id' and removes the information connected to the id provided by the admin in the request body from the database

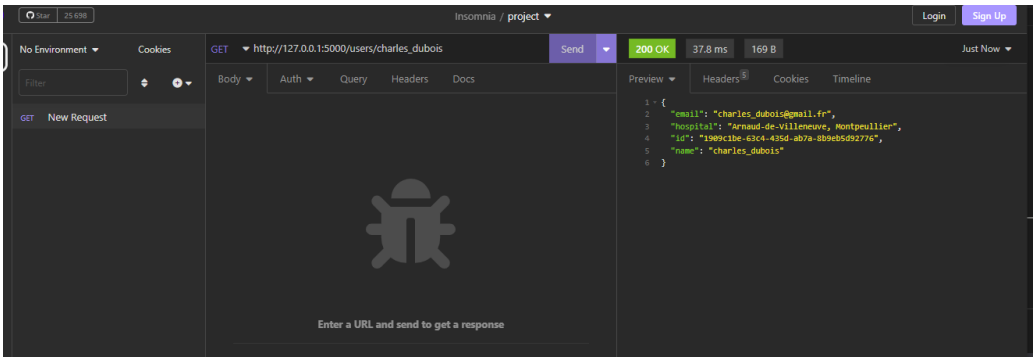
.



4.2 API endpoints for users

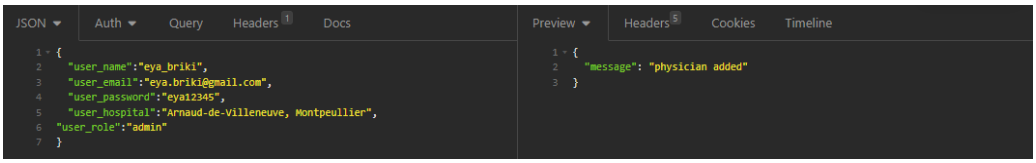
4.2.1 GET request

This function handles a GET requests on the endpoint '/users/string:name' the users dataframe is used to retrieve the credentials of each doctor or technician.



4.2.2 POST request

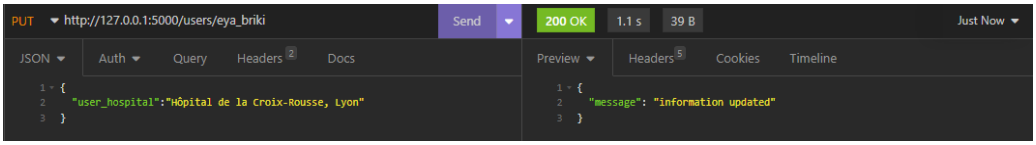
this function handles a POST request on the endpoint '/users' and takes values entered by the admin or moderator in the request body to add a new user to the database.



4.2.3 PUT request

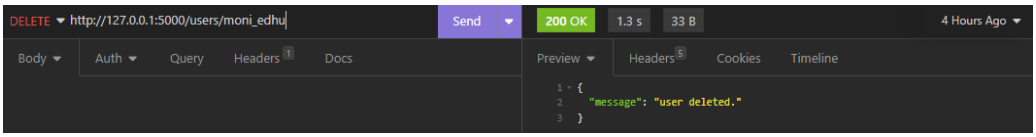
This function handles PUT and PATCH requests on the endpoint '/user/string:user_name' and allows updating specific columns of an existing row in a dataframe by the given user_name. It

reads the request body for new values, searches for the id in the dataframe, updates the columns with the new values if they are passed and loads the updated dataframe to the database.



4.2.4 DELETE request

This function handles a delete requests on the endpoint '/user/string:user_name' and removes the information connected to the user_name provided by the admin in the request body from the database .



Chapter 5

packages and technologies

In this chapter, the different software, websites, and packages used for the sake of the project are exposed.

5.1 Python

Python is a high-level, interpreted programming language that is widely used for web development, data science, machine learning, and many other applications. It has a simple, easy-to-learn syntax that makes it a popular choice for beginners and experienced developers alike. In this project , Python was put to use to write the server-side logic for APIs, handle the routing, validation, authentication, and data manipulation and database interaction.

5.1.1 virtual envirenment

A virtual environment is a tool used to isolate specific Python environments on a per-project basis.

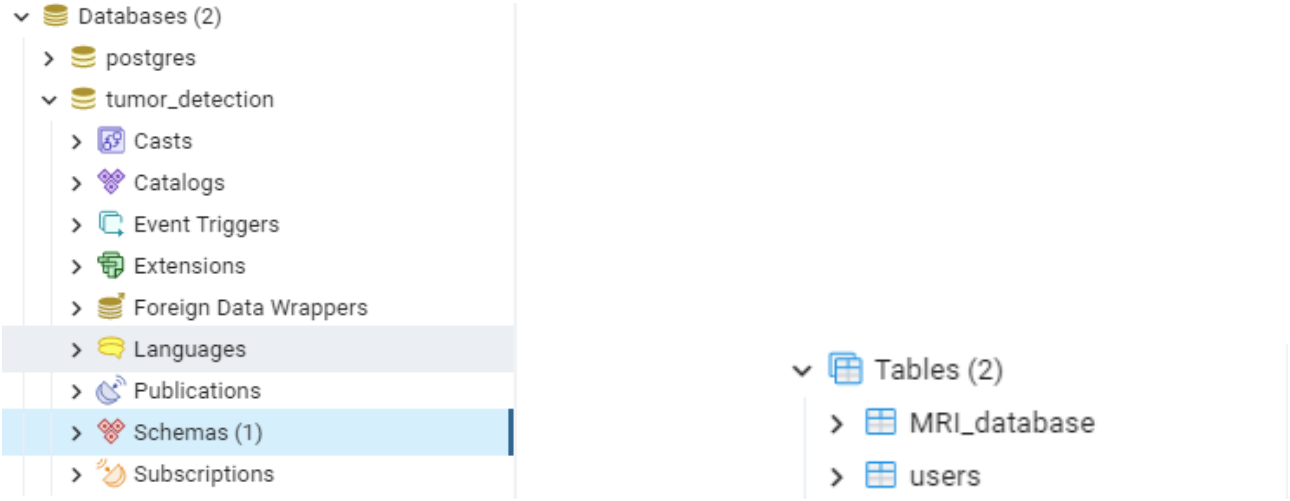
For the purpose of this project, a virtual environment .tumor was created using the venv module and acitivated which allowed you to install and manage different versions of packages and dependencies for your project, without interfering with other projects.

5.1.2 Packages

- Pandas - Numpy - Opencv - Glob - uuid -Tensorflow - Scikit-learn - imutils - bcrypt - sqlalchemy - psycopg2 - flask -flask_restful - flask_smorest - flask_jwt_extended

5.2 Postgresql

PostgreSQL is a powerful, open source object-relational database system. In this project, it was used to store the datasets of MRIs and users. Through an engine create from the 'sqlalchemy' package and the use of the 'psycopg2' package, python and postgres kept a secure connection to manage the changes and additions made through the API functions.



5.3 Insomnia

Kong Insomnia is a collaborative open source API development platform that makes it easy to build high-quality APIs. In this project, Insomnia was used to send requests to the different endpoints of APIs, and examine the responses from the server. Additionally, it was used to test the authentication and authorization by sending requests with and without the proper tokens and examining the responses to ensure that the API was properly protected.

Chapter 6

Conclusion

This project has demonstrated the potential of machine learning to improve the accuracy and efficiency of Magnetic Resonance Imaging (MRI) image classification. This approach can be further developed and applied to other medical imaging tasks, which could help in early detection of tumors, improve diagnosis and treatment, and ultimately save lives. The use of APIs for dataset creation and management and the storage of data in a PostgreSQL database, also makes it easier to access and use the data for future studies. Overall, this project highlights the importance of using advanced technology and machine learning in the field of medical imaging.

6.1 Summary of Thesis Achievements

- Developed a machine learning approach for classifying MRI images as healthy or diseased based on the presence of tumors
- Achieved an accuracy of 881% using a convolutional neural network model and a dataset of MRI images from various hospitals and patients of different genders
- Demonstrated the potential of machine learning to improve the accuracy and efficiency of MRI image classification

- Suggested the approach could be further developed and applied to other medical imaging tasks
- Implemented APIs for dataset creation and management and stored data in a PostgreSQL database for easy access and use in future studies.

6.2 Limitations and Future Work

The model accuracy of 81% is promising, but there is still room for improvement. Further research and development could be done to increase the accuracy of the model.

The use of a single convolutional neural network model might not be sufficient for handling all cases, as the tumors can appear in many different shapes and sizes, thus, other models might be needed to improve the detection rate.

for that we can use:

- Data augmentation: To increase the size of the training dataset and reduce overfitting, we can use data augmentation techniques such as rotation, flipping, and scaling to pinpoint further hidden data and features.
- Hyperparameter tuning: we can use techniques such as Grid Search or Random Search to find the best values for the hyperparameters such as the learning rate, batch size, and number of filters.

The use of a PostgreSQL database for storage is a good choice, but the scalability of this solution should be also taken into consideration for large datasets.

for that we can use:

- NoSQL databases: NoSQL databases, such as MongoDB, are designed to handle large and unstructured data sets. They are highly scalable and can handle high write and read loads.

- Cloud storage: Cloud storage solutions, such as Amazon S3, Google Cloud Storage, and Microsoft Azure Blob Storage, provide scalable and cost-effective options for storing and managing large datasets.

The code is a Python script that uses the Flask web framework to define a REST API. The API allows users to perform operations such as login, get MRIs by id, and update MRIs. But, some of its limiatations are that The code does not handle any form of input validation which might cause issues when the input is not correct, and doesn't handle the case of image files with different format than jpg/jpeg,differnet encoding, or image files with different path than the specified one, this can cause issues if the image is not readable by the cv2 library.

for that we can use:

- Use the Python Imaging Library (PIL) to handle different image file formats.
- Use the os library functions to handle different image names or paths.e and can handle high write and read loads.
- Implement input validation using libraries such as marshmallow or cerberus to validate the input data before processing it.

Bibliography

[1] Insomnia. https://insomnia.rest/

[2] Amazon Web Services. https://aws.amazon.com/fr/rds/postgresql/what-is-postgresql/

[3] Kaggle. https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection/versions/1?resource=download

[4] W3Schools. https://www.w3schools.com/