DiagnosisAI

Medical Imaging

TEAM PRODUCT DELIVERY

Chu-Cheng Yu 24798718
Deklin Khan 13918192
Joanna Gunawan 12581738
John-Paul Masson 13195134
Shiyu Wang 13978212
Zihao Huang 13977677

41088 PROFESSIONAL STUDIOS | SPRING 2023

Executive Summary

This product is a Computer Aided Diagnostic (CAD) system that allow for the diagnosis of lung cancer from Computer Tomography (CT) scans. The product aims to improve the current solution of medical professional diagnosis by increasing the rate of diagnosis and preventing the current bottleneck of diagnosing patients, leading to delayed treatment and higher fatality rates. The machine learning model of this product is able to accurately diagnosis lung cancer at 96% accuracy, therefore satisfying the aims of the Product owner. However, the product is limited in its market usability, due to the need for society to accept the diagnosis of AI in a potentially life or death situation.

Table of Contents

List of Figures

List of Tables

1. Product Description

1.1 Team Motivation, Stakeholders & Product Description

Deep learning can be applied to medical imaging to assist in the diagnosis of lung cancer. Lung cancer is the most common cancer to have the highest mortality rate (WHO, 2023). Treating lung cancer is often difficult as it is hard to detect, where most patients are often asymptomatic in the early stages (Leigh, 2022). However, when diagnosed at advanced stages, treatment options are often limited (WHO, 2023). Furthermore, prognosis of lung cancer is significantly lower than other cancers, since only around 20% of cases respond to immunotherapies (Newman & Collier, 2018).

Deep learning is a type of machine learning that enables computers to solve complex problems (Oracle, n.d.) The technology is inspired by the structure of the human brain where neural networks can define input and outputs through multiple layers of transformation and nonlinear functions (Bakator, 2018). For lung cancer detection in CT scans, human detection is often limited to the doctor's experience and capabilities (An et al., 2019). Thus, deep learning can be applied to images to localise objects and classify them to class labels.

DiagnosisAI is an application built to assist in the diagnosis of lung cancer. A user can upload a chest CT scan and deep learning is then applied to the image to identify the lung nodule and classify it as either benign or malignant. The web application enables a user-friendly interface and visualisation of results. The display of metrics, namely the confusion matrix and accuracy reading as well as the localised image results allow users to determine whether the prediction is correct and reliable, thus making the appropriate decision-making towards prognosis.

DiagnosisAI demonstrates the benefits and capabilities of artificial intelligence as an advanced tool. The trending technology motivates the team to upskill and solve real-world problems. The main stakeholder, or target audience, are pulmonologist or any medical practitioner. Alternatively, this application could be publicly available where patients could potentially upload their scans and find an initial reading before going to surgeons to confirm their diagnosis. Furthermore, this application can be utilised in future research.

1.2 Existing Solutions & Product Impact

The topic of deep learning for medical imaging has been a ubiquitous topic for researchers, particularly in the field of deep learning and medicine. However, whilst research has demonstrated potential, there is currently no existing solution utilized in the industry yet. This may be attributed to the high-risk environments within medicine and the possibility of a false diagnosis. The current status quo relies on medical professionals to diagnose lung cancer; however the volume of diagnoses creates a bottleneck leading to delays in diagnosis.

A potential impact of DiagnosisAI is the high accuracy ability of diagnosing lung cancer. The product aims to detect lung cancer at early stages, therefore increasing the probability of a successful prognosis through early treatment. Furthermore, the product and utilisation of deep learning algorithms may be further utilised for other cancer diagnosis.

2. Requirements

2.1 Functional Requirements

Within the context of medical imaging AI, functional requirements are what is required for our project to be considered successful. These requirements help us understand what the system is lacking, and what can be improved. For our project on classifying lung cancer CT scans, the team have proposed five functional requirements that serve as a guideline for the production of the model and the application.

Functional Requirements		
ID	Description	Pass/Fail
FR-01	Application runnable on windows 10.	Pass
FR-02	Application can process an image input.	Pass
FR-03	Results should be easily interpretable, allowing medical professionals to understand the results.	Pass
FR-04	Application should adapt to variations in image quality, resolution and format.	Pass
FR-05	Application should achieve a minimum accuracy rate of 90% in differentiating between benign and malignant nodules.	Pass

Table 1 – Functional Requirements

2.2 Usability/Technical Requirements

As for usability/technical requirements, we put the focus of our requirements towards user experience, quality, privacy and security. Unlike functional requirements, these requirements specifically target attributes aimed at improving and addressing usability concerns, allowing us to evaluate the application and make suitable adjustments, or include new features if necessary. The usability and technical requirements have been stated and addressed below.

Usability/Technical Requirements		
ID	Description	Pass/Fail
UT-01	User friendly interface for medical professionals.	Pass
UT-02	The application should maintain patient data privacy and security.	Pass
UT-03	Application should scale correctly on different screen size.	Pass
UT-04	Application should be able to run multiple consecutive images	Pass
UT-05	Application should be able to classify in real-time and produce a result at a reasonable speed.	Pass

Table 2 – Usability/Technical Requirements

3 Dependencies

3.1 Deep Learning and AI Technology Dependency

Numerous studies have explored the application of deep learning algorithms for CT-based lung cancer screening and diagnosis. Typically, there are distinct image attenuation patterns in CT scans between healthy and unhealthy images. (Thanoon et al., 2023) The core functionality of Diagnosis AI relies on advanced deep learning algorithms capable of analysing CT scans to detect lung cancer.

3.2 Medical Imaging Data Dependency

Research in deep learning-based lung imaging primarily focuses on detecting, segmenting, and classifying pulmonary nodules as benign or malignant. Efforts are centred on developing new network structures and loss functions to improve model accuracy. (Wang, 2022)

3.3 Computational Resources Dependency:

High-performance computing (HPC) resources are essential for processing complex deep learning algorithms. Although deep learning APIs like TensorFlow or PyTorch do not typically guarantee efficient use of HPC resources, the high processing power and fast memory and storage resources of HPC systems are crucial for enhancing the efficiency of deep learning applications. (Brunst & Döbel, 2023) HPC systems usually have between 16 and 64 nodes, with each node running two or more CPUs, offering significantly higher processing power compared to traditional systems. To further increase processing power, many HPC systems incorporate graphical processing units (GPUs), and the combination of CPUs and GPUs, known as hybrid computing, enhances computational capabilities. (Run:AI, n.d.)

3.4 Healthcare Expertise and Collaboration Dependency

In healthcare, collaboration with medical professionals is essential for confirming the accuracy and clinical relevance of AI models, ensuring results are understandable to medical staff, and following new AI regulations (Farah et al., 2023). Explainable AI applications in medicine help in making correct diagnoses and treatment decisions. The transparency and interpretability of these models build clinician trust, enhancing patient care and outcomes. (Yang et al., 2023)

3.5 User Interface Design Dependency

In medical applications, a user-friendly interface designed for medical professionals is crucial. This requires ability in UI/UX design to ensure the application is intuitive and efficient for clinical use. In various healthcare settings, the UX/UI design of medical devices and interfaces plays a critical role. It focuses on supplying a user-friendly experience for healthcare professionals, enabling them to run equipment, interpret data, and make informed decisions efficiently. This contributes to safer and more effective clinical workflows. (Ossmium, 2023)

3.6 Data Privacy and Security Dependency

Ensuring patient data privacy and security is crucial in healthcare, Healthcare data compliance involves adhering to legal and industry standards in managing patient data, encompassing data security, privacy regulations, consent management, risk assessment, and incident response. Diligent adherence to these practices helps mitigate risks of unauthorized access to sensitive information. Healthcare organizations must follow strict government regulations, such as those imposed by HIPAA, to avoid substantial fines and reputational damage. (Group, 2023)

4. Constraints

This semester, our team encountered numerous challenging issues that hindered our progress in completing the final step, which involves combining all our models. The primary issue was related to the project's requirement for accuracy in detection. Consequently, our team decided to divide the work, utilizing different frameworks to determine whose coding yields the most accurate detection. Due to the varying working speeds of team members, we compared results late in the process, leaving insufficient time for combining the models.

Another contributing factor was the heavy workload on each team member, necessitating extensive discussions about project details within a limited time frame. One significant problem our team faced was the difficulty of coordinating everyone due to different subjects and varying timetables, revealing a deficiency in our team's time management that highlights the need for improvement.

The reason our project is limited to scanning CT scans is that it's challenging to train a robot to identify lung cancer. Attempting to insert more models to make the AI read additional medical information would make it harder to train the AI, thereby decreasing the accuracy of lung cancer identification. We've also specified the file types that customers need to attach. To enhance the user-friendliness of our AI, JPEG, JPG, and PNG files are allowed, as these are common file types for everyone to use.

5. Software Tools

5.1 Streamlit

Streamlit is an open-source Python library that allows for the development of interactive web applications, in particular data science and machine learning applications. Streamlit was used to create the functional and shareable web app of our product by combining GUI elements with the image classification model. Streamlit integrated seamlessly with the common Python data libraries that were used for the classification model listed below, making it an ideal tool.

5.2 PyTorch

Was used to build some of the neural network like the MobileNetV2 it is a library used for creating neural networks and has lots of control on variables for training.

5.3 NumPy

NumPy was used as our main data preprocessing tool, as everything has to turned into arrays before it can be trained on the Convolutional Neural Network models. It is also used to split the data into train, test and validation dataset, allowing us to evaluate if the model is over trained, to prevent overfitting and making the data unreliable.

5.4 TensorFlow + Keras

TensorFlow was used to develop the Conventional CNN (Convolutional Neural Networks) and conducting model training and testing. It allows for quick and easy layer construction with many different options for activation function.

5.5 MatPlotLib

Matplotlib was used as the main tool for data visualisation. It allows for the results to be visualised with line graphs, confusion matrix, to show the accuracy and loss over epoch, and to show the confusion matrix allowing for easier interpretation and evaluation on the accuracy of our models.

5.6 YoloV8

YoloV8 is a computer vision model that enables high-accuracy object detection and image segmentation, developed by Ultralytics. The intended use of yolov8 towards this project was to apply it to a CT scan where it would be able to localise the lung nodule. However, after a failed attempt to the implementation, with the requirement of paid third-party accounts, the use of this tool was discontinued.

6. Team Development Tools

6.1 Team Planning

Team planning occurred during phase 0 in which the product proposal was completed. This generated 4 phases to be followed These being Phase 1: Prototyping and research of image classification. This phase lasting a brief period, being completed prior to StuVac with all team members working on their components individually. Phase 2 was the development phase for GUI and further iterations of the models implementing research collated to approach development. Phase 3 was iteration and optimisation for both the GUI and the models, this was aiming towards a Minimum Viable Product for the GUI and improving accuracy overall for the individual models. Phase 4 is the final phase with model comparison occurring to select for the best model which would be integrated with the User interface, Additional testing and validation occurred as to meet our test cases.

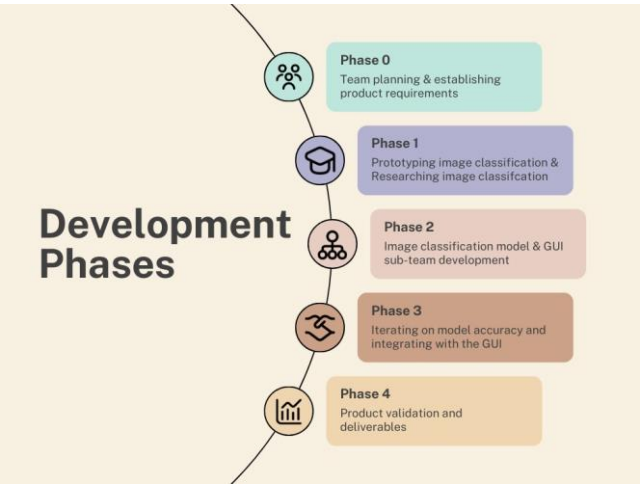


Figure 1 – Development Phases

6.2 Team Coordination

Team coordination occurred primarily through Microsoft Teams due to ease of use and flexibility for those who may be unable to make class, have work, etc. Teams allows for files to be stored and collaboratively worked on. Team meetings had set times allowing for scheduling around them to be in mind for members when necessary. Additional coordination was conducted in classes between available members to help troubleshoot issues.

7. Development

When project development occurred two main components were focused; User Interface and Model Development. The former was undertaken primarily by John Paul-Masson with the models being split into individual models by the remaining members. This method was conducted as to attempt multiple approaches at diagnosis of Lung Cancer. Below each model will be seen and detailed with images for performance visible for each model.

7.1 Image Classification Model Development

The image classification models are different variations of Convolutional Neural Networks, these come in the form of Conventional, ResNet, AlexNet and VGGNet. The variations allow for testing the full range of CNN approach at diagnosing the images. Using differing models would allow for members to develop them individually and then compare performance between them.

7.2 Image Classification Model Training

The training process in the provided code involves loading and preprocessing data using custom dataset classes and PyTorch data loaders. The model architecture is based on MobileNetV2, where the original classifier is replaced with a custom one suitable for the classification task with four classes. Hyperparameters, including the number of epochs, batch size, learning rate, momentum, and the number of classes, are set. The cross-entropy loss function and the Adam optimizer are utilized for optimization.

The training loop iterates over epochs, updating model parameters through backpropagation and gradient descent. Progress is monitored using the tqdm library. After training, the model is evaluated on the test set, and accuracy and loss metrics are reported. The trained model is saved for future use. The overall process involves balancing data loading, model definition, hyperparameter tuning, and optimization to achieve effective image classification. Adjustments to hyperparameters and model architecture can be made based on the specific characteristics of the task and dataset.

```
In [4]: from sklearn.model_selection import train_test_split

        # Reshape the image array and add a channel dimension
        image_array = np.expand_dims(data_images, axis=-1)

        # Splitting the data
        train_images, temp_images, train_labels, temp_labels = train_test_split(image_array, data_la
        validation_images, test_images, validation_labels, test_labels = train_test_split(temp_image

        print("Training Images Shape:", train_images.shape)
        print("Validation Images Shape:", validation_images.shape)
        print("Testing Images Shape:", test_images.shape)
        print("Training Labels Shape:", train_labels.shape)
        print("Validation Labels Shape:", validation_labels.shape)
        print("Testing Labels Shape:", test_labels.shape)

        print(train_labels)

Training Images Shape: (600, 28, 28, 1)
Validation Images Shape: (200, 28, 28, 1)
Testing Images Shape: (200, 28, 28, 1)
Training Labels Shape: (600, 2)
Validation Labels Shape: (200, 2)
Testing Labels Shape: (200, 2)
[[0. 1.]
 [1. 0.]
 [1. 0.]
 ...
 [0. 1.]
 [1. 0.]
 [0. 1.]]
```

Figure 2- Training Parameters

```
while True:
    img = input('Input image filename:')
    try:
        image = Image.open(img)
    except:
        print('Open Error! Try again!')
        continue
    else:
        r_image = test_mydata(img)
```

Figure 3- Loading Images

7.3 Image Classification Model Testing

Since the initial dataset is split into train, validation and test, no additional datasets were required to test if the model is overfitting or not, as the test split already count as unseen data. Another way to see if our model was effective at classifying, or if training is actually working, is by producing an accuracy & loss over epoch graph. This allows us to interpret if the model is actually improving after each epoch, or if the training had completely no effect at all.

```
# Train the model
early_stopping = EarlyStopping(monitor='val_loss', patience=25, restore_best_weights=True)
history = model.fit(train_images, train_labels, epochs=50, batch_size=32, validation_data=(v

# Plot accuracy and loss over epochs
plt.figure(figsize=(12, 5))

# Plot accuracy
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Training and Validation Accuracy')
plt.legend()

# Plot loss
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Training and Validation Loss')
plt.legend()

plt.tight_layout()
plt.show()
```

Figure 4 – Model Training

```
import os
import random


# Set the random seed to ensure that the order of the images is the same every time.
def setup_seed(seed):
    random.seed(seed)


setup_seed(20)

b = 0
dir = './Dataset/'
# The result of os.listdir is a list collection
# Use the sort method of a list to sort. If there are numbers, use numbers to sort.
files = os.listdir(dir)  # Obtain the path of the amplified image folder
files.sort()
# print("files:", files)  #Create txt file for subsequent data storage
train = open('./train.txt', 'w')
test = open('./test.txt', 'w')
a = 0
a1 = 0
while (b < 4):
    label = a  # Set the label to be marked
    ss = './Dataset/' + str(files[b]) + '/'  # training pictures
    pics = os.listdir(ss)  # Get the pictures in the sample00_train folder
    i = 1
    train_percent = 0.8

    num = len(pics)  # Get the total number of samples
    list = range(num)  # get list
    train_num = int(num * train_percent)  # Total number of training sets
    train_sample = random.sample(list, train_num)  # Randomly select train_num lengths in the list and shuffle them
    test_num = num - train_num  # Get the number of test samples

    for i in list:
        name = str(dir) + str(files[b]) + '/' + pics[i] + ' ' + str(int(label)) + '\n'  # Get the names of all picture sequences in the current folder
        if i in train_sample:  # Determine whether i is in the training set
            train.write(name)  # If it is, output the image as training text
        else:
            test.write(name)
    a = a + 1
    b = b + 1
train.close()
test.close()
```

Figure 5 – Training Code

7.4 Image Classification Model Comparison

Below is the results from each model, showing a confusing matrix and an accuracy & loss over epoch graph to each of the respective model.

7.4.1 Resnet

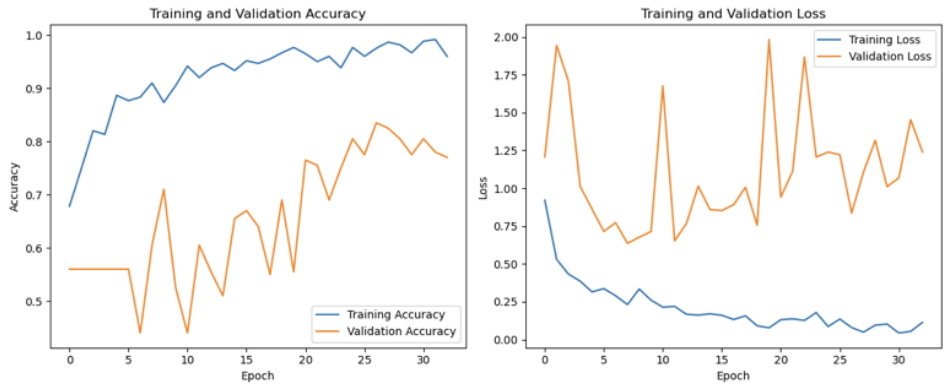


Figure 6 – ResNet Accuracy and Loss

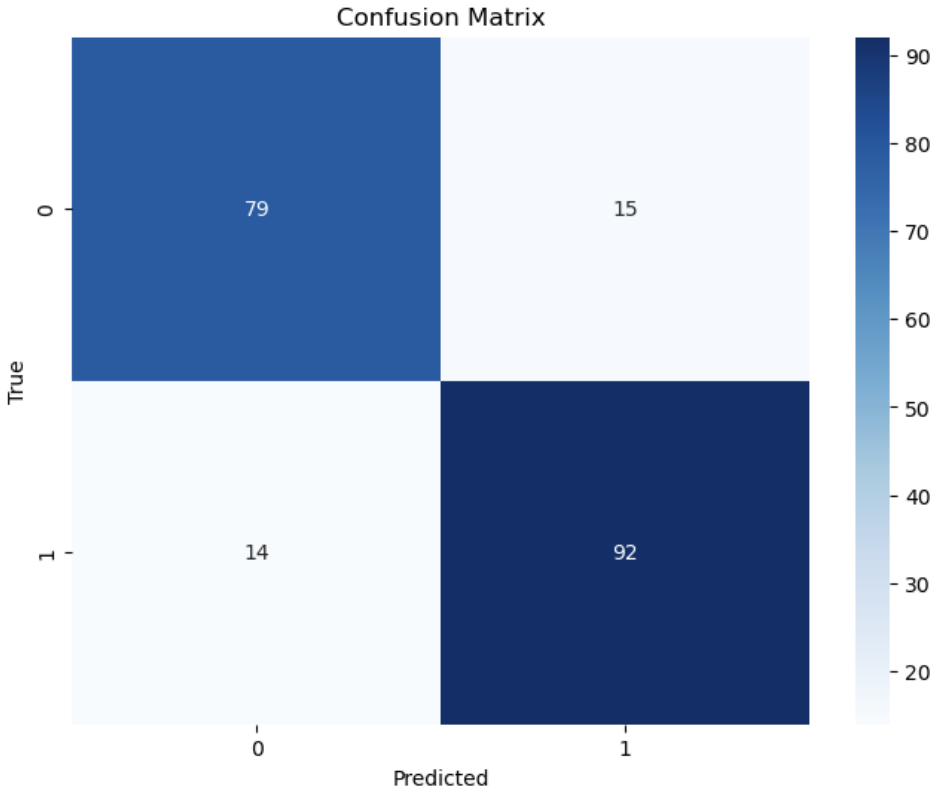


Figure 7 - ResNet Confusion Matrix

7.4.2 VGGNet

Figure 8 – VGGNet Accuracy and Loss

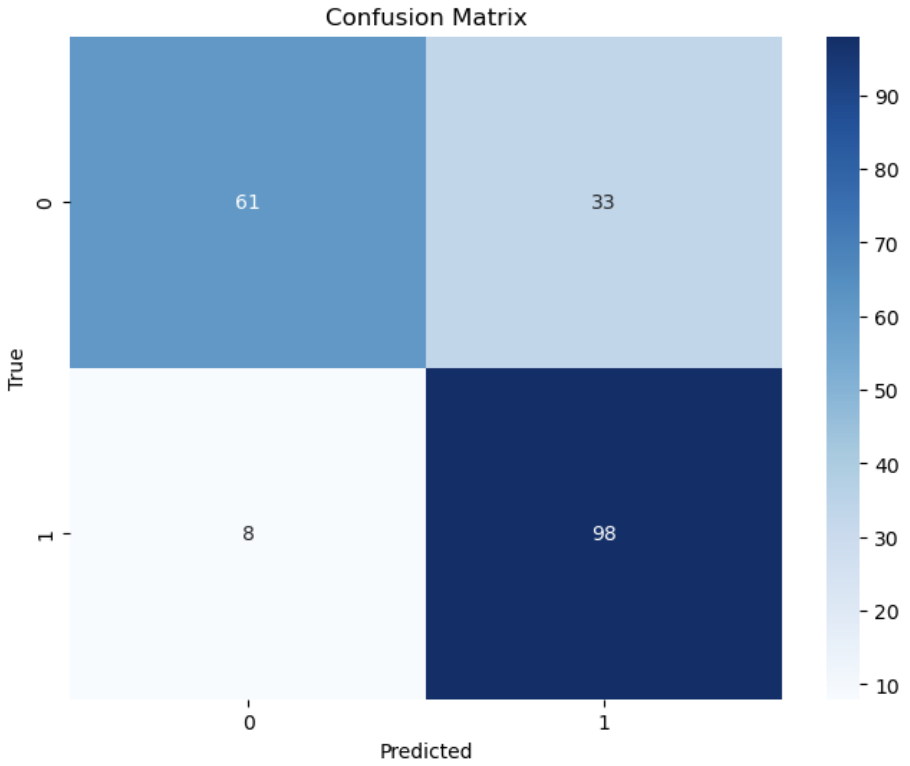


Figure 9 – VGGNet Confusion Matrix

7.4.3 AlexNet

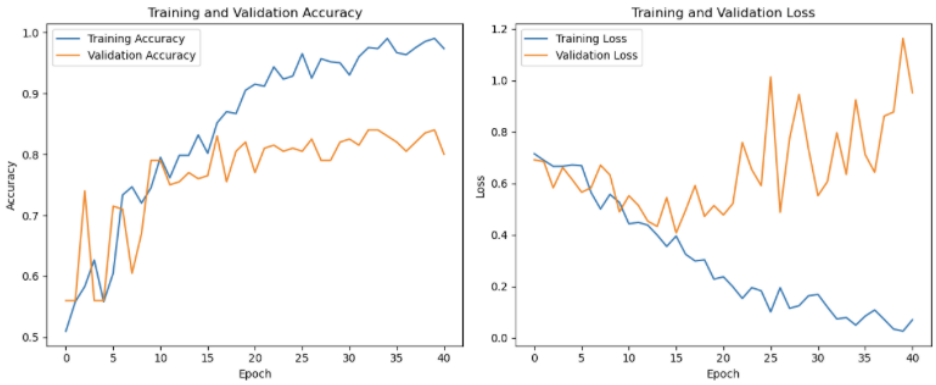


Figure 10 – AlexNet Accuracy and Loss

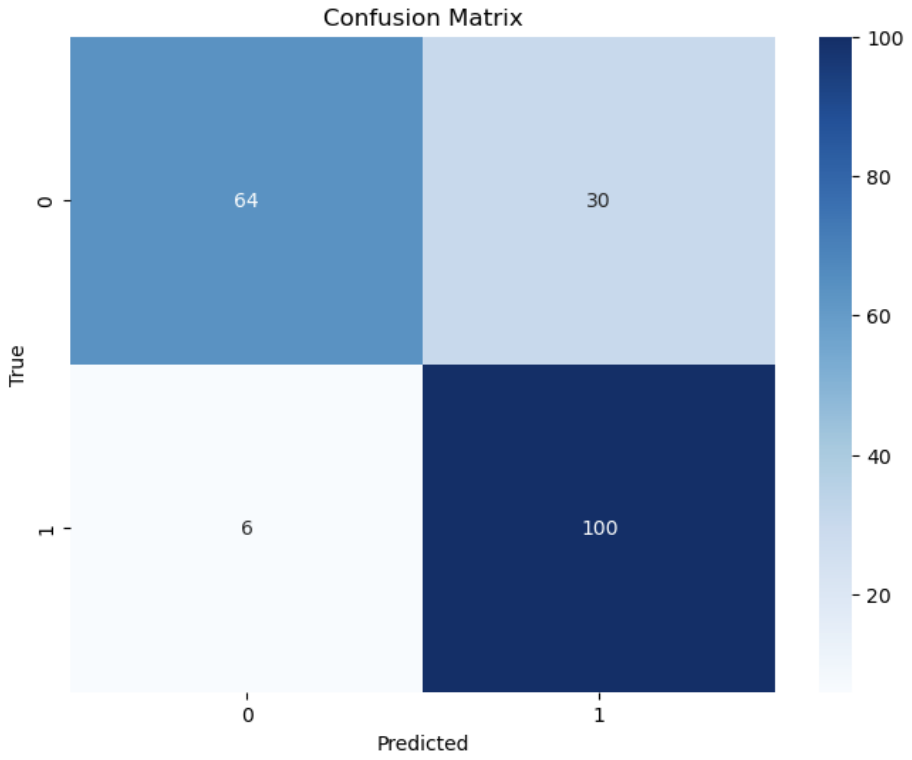


Figure 11 – AlexNet Confusion Matrix

7.4.4 Conventional CNN

This is the basic form of a Convolutional Neural Network and uses the basic format of convolutional layers followed by max pooling and batch normalisation layers. The performance seen is limited due to this fact with an overall lower performance than the other alternatives. This is due to lack of specialisation as it is generally a strong neural network method it needs to be altered to best suit a specific use case.

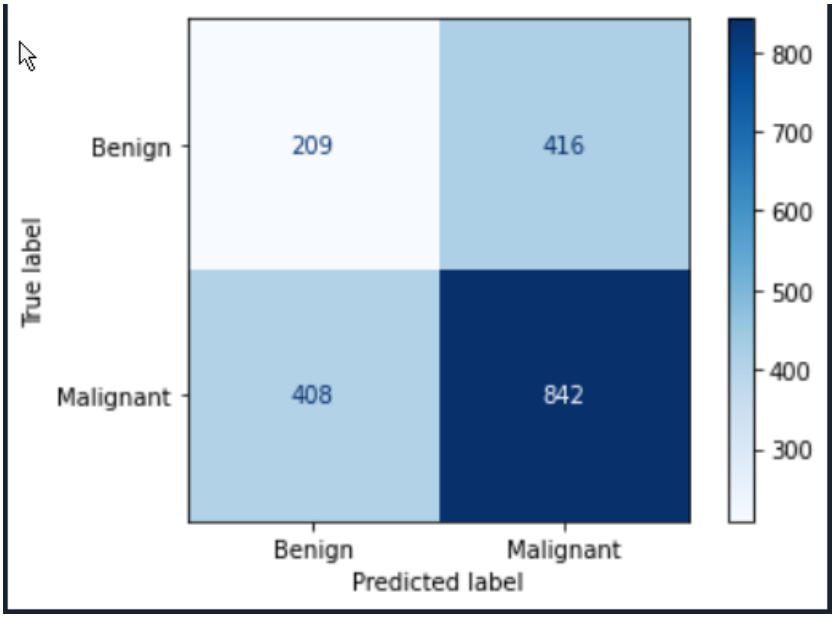


Figure 12 – Conventional CNN Confusion Matrix

7.4.5 MobileNetV2

1. True Positive Rate (TPR) - 0.961 and False Negative Rate (FNR) - 0.039:

The TPR of 0.961 indicates that the model correctly identifies positive samples with a high accuracy of 96.1%.

The FNR of 0.039 complements this by showing that only 3.9% of the positive samples were incorrectly classified as negative. This low FNR is indicative of the model's effectiveness in minimizing missed positive classifications.

2. Implications of Model Architecture and Training:

The MobileNetV2 architecture, known for its efficiency, likely contributes to this balanced performance. Its ability to capture relevant features without being overly complex helps in reducing both false positives and false negatives.

The training process, including data preprocessing, hyperparameter tuning, and the choice of loss function and optimizer, seems to be well-optimized for the task, as evidenced by these metrics.

3. Potential Areas for Improvement:

While the performance metrics are impressive, continuous improvement is always possible. For instance, further fine-tuning of the model, more advanced data augmentation techniques, or even ensemble methods could potentially reduce the FNR further.

It's also important to evaluate the model across various other metrics like precision, recall, and F1-score to ensure a comprehensive understanding of its performance.

4. Overall Assessment:

The combination of a high TPR and a low FNR indicates that the MobileNetV2 model is performing exceptionally well in this image classification task. It is effectively finding many positive cases while minimizing the instances of false negatives.

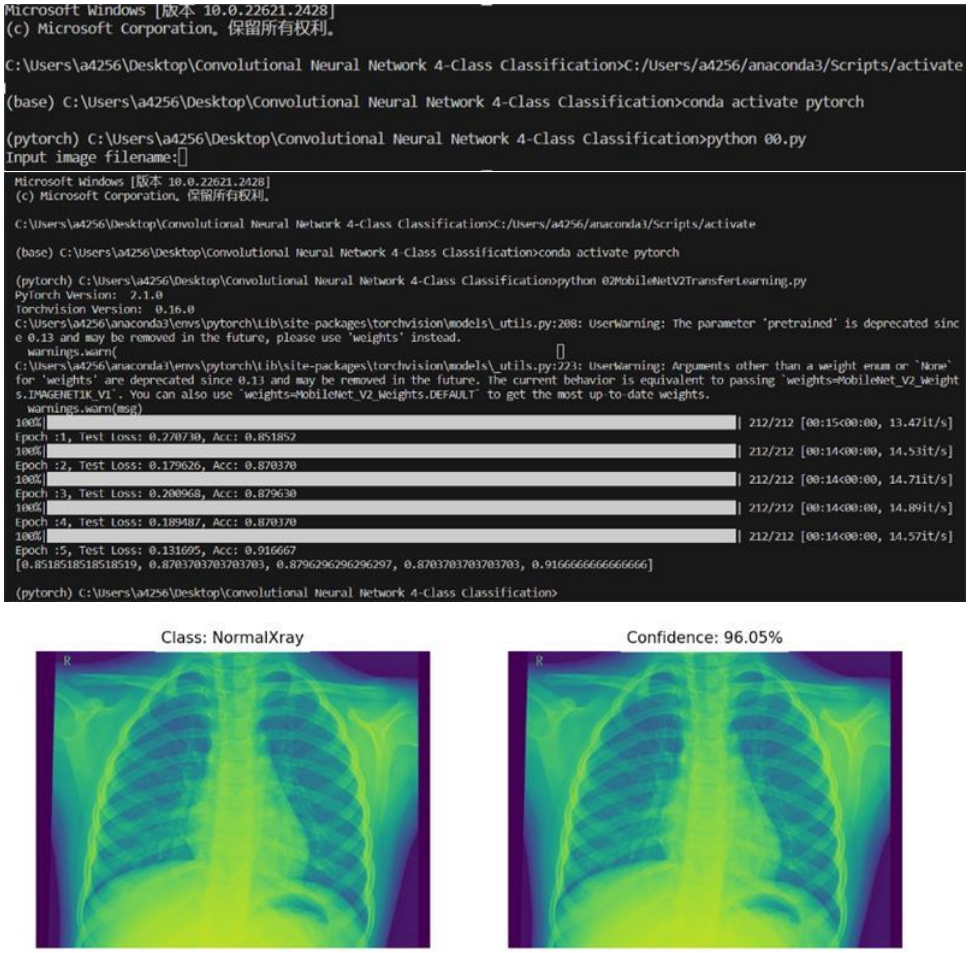


Figure 13 – MobileNetV2 Performance

7.4.6 Model Selection

After our evaluation of all the models built, we have concluded that Shiyu Wang and Zihao Huang's model is the most suitable one to implement into our project for two main reasons. First, their model has one of the highest accuracy of 0.961, and second, this model contains both localization and classification, whereas Chucheng Yu's model only performs classification. Below is a table of all models, and their respective true positive, and false negative value.

Model	**True Positive Rate**	**False Negative Rate**
ResNet	0.868	0.132
AlexNet	0.925	0.075
VGGNet	0.943	0.057
Conventional CNN	0.674	0.326
MobileNetV2	0.961	0.039

Table 3 – Model Comparison

The reason behind only looking at true positive and false negative, while disregarding true negative, and false positive, is because for classifying lung cancer nodules, it is crucial that the application don't classify any positives as negatives, whereas classifying negative as positive is less important.

7.5 GUI (Graphical User Interface) Development

TBC As one of the two main components of the product, the main focus of the GUI was simplistic functionality, with the secondary focus on aesthetic polishing to create an aesthetic design.

7.5.1 GUI design

When designing the GUI, initial wireframing was used to create the proposed layout, shown below in figure 14.

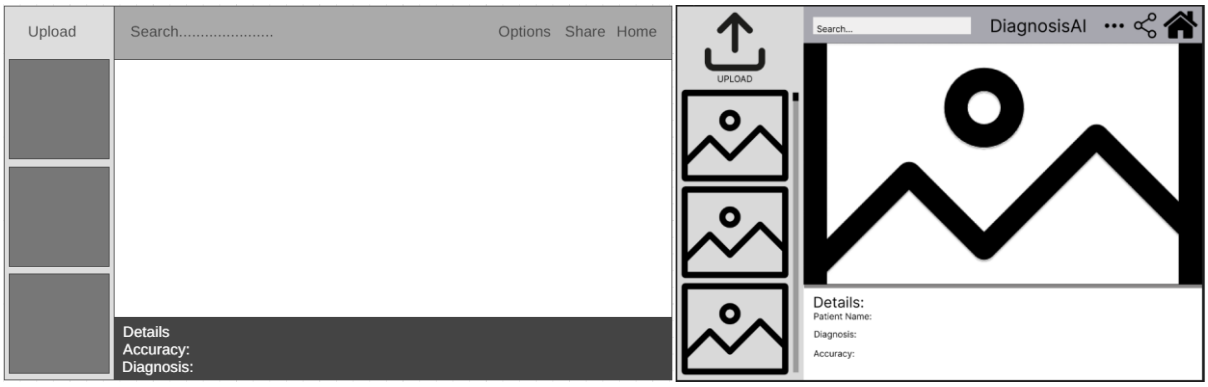


Figure 14 – GUI Wireframe Designs

The initial wireframing focused on the ability to upload images, view the results of the diagnosis from the image classification and then correlating that diagnosis to the patient's name. When consulting with the product owner Steve and considering usability requirements, a simpler design was made, however the introduction of accuracy into the GUI was added in order to give confirmation to the medical professional; Since a low accuracy would prompt the medical professional to get a secondary opinion on the diagnosis.

7.5.2 GUI implementation

The construction of the GUI was achieved through the use of Streamlit, a Python based library designed for "machine learning and data science teams" (Streamlit, 2023), which was discussed in section 5.1. Streamlit handles the model information for the CT scan classification well, however isn't as free form as other GUI development tools. This meant that the design we had planned to implement in 7.5.1 had to be adapted. Keeping all the same components, figure 15 below shows the final layout of the GUI including the DiagnosisAI logo, descriptions of input, expandable menu to upload file, displaying the image and the diagnosis and accuracy below.

Figure 15 – Final GUI Layout

7.5.3. Aesthetic design

With the completion of the GUI functionality, the aesthetic design for a polished product was addressed. With the severity of Lung Cancer that the product aims to aid and the intended target audience of the product being healthcare specialists, a highly vivid and content rich web-app seemed out of place. As such the design focused on minimalism and sleek design. Using colour theory, blue was chosen as the primary colour for the logo and web-app background, since “Blue is calming, soothing and friendly… and can take on a professional or friendly tone…” (Ackerman, 2023). Multiple logos were tested, shown below.

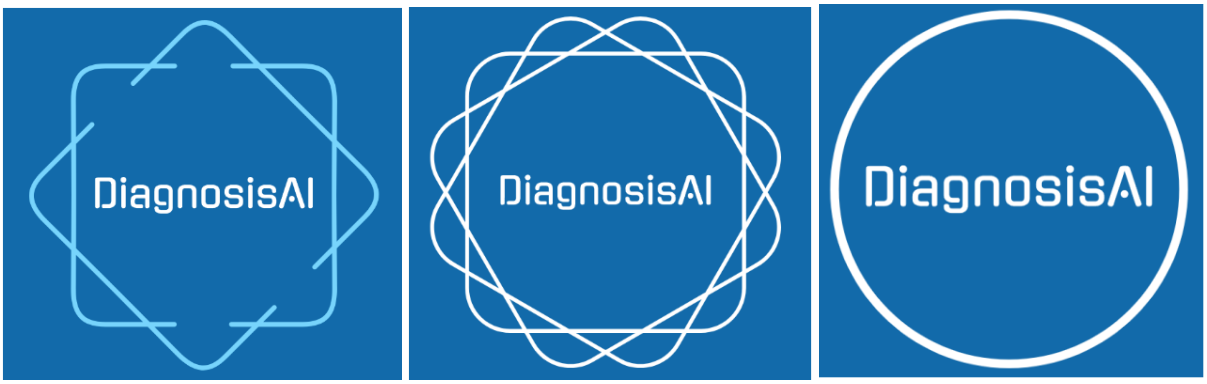


Figure 16 – DiagnosisAI Potential Logo Designs

With the final choice being shown below. Multiple versions of this logo were generated, black background, white background, transparent etc. To allow for a wider use across branding of the product.

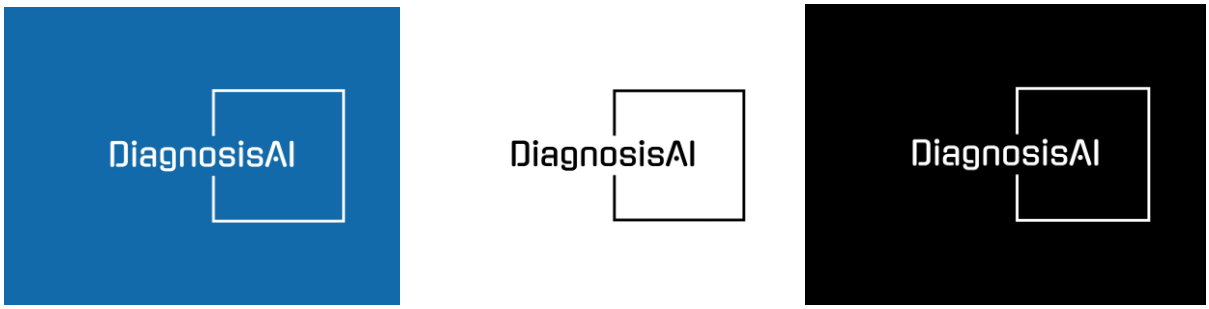


Figure 17 – DiagnosisAI Final Logo Designs

For the background of the web application multiple backgrounds were tested. Some were ruled out due to over complexity for the minimalistc design, while others were ruled out due to the clash between content layout and the background.

Figure 18 – Potential GUI Background Designs

7.6 Validation & Testing

During the process of creating product requirements we established soft testing parameters, which were later solidified into the criteria below. All the tests we established for the product were passed. This resulted in a diagnosis model with a solid accuracy, an easy-to-use GUI and an established threshold for medical reassessment. This means that if the model gives an accuracy lower than 70% a medical professional must re-assess the diagnosis.

Model/GUI Component	Testing	Test	Result
Image classification model	Accuracy	Model accuracy greater than 90% on training dataset	Pass
Image Classification model	Accuracy	Model accuracy greater than 84% on unseen testing dataset	Pass
Image classification model	Precision	Model does not incorrectly predict with an accuracy greater than 70%	Pass
GUI	Versatility	GUI able to determine device and scale display accordingly	Pass
GUI	Functionality	GUI able to upload a CT scan and display on the application	Pass
GUI	Functionality	GUI able to successfully handle multiple file formats	Pass
GUI	Usability	GUI is able to be repeatedly used to diagnose multiple CT scans without a need to refresh or reload.	Pass

Table 4 – Validation & Testing

8. Deliverables

To ensure the successful completion of the project and the effective contributions of each team member, all team members wrote an Individual Learning Contract and a comprehensive Personal Design Journal. Each member was required to share their writing and the progress they have made. These individual deliverables provide our team with an understanding of each member's responsibilities in this project and document personal successes and learning achieved during the project studies. Through these individual deliverables, we can grasp each member's contributions and identify the challenges they faced. This enables us to address problems and allocate tasks accordingly. The individual deliverables were required to be completed using online journaling services (Lab Archives/Trello) and submitted through Canvas fortnightly.

In the context of the completed project "Diagnosis AI," our team successfully created a module and a graphical user interface (GUI). With the GUI, users can easily utilize our design by simply dropping the CT file and waiting for the results. The module allows users to analyse lung cancer CT scans, and the AI will interpret the results for the patient. This initial project prompted discussions on the software and framework needed to train our AI, with a focus on achieving high accuracy in identification. Ultimately, we decided to use Pytorch and successfully achieved an accuracy above 90% on the unseen testing dataset.

8.1 Metrics for success

To assess the success of our team throughout the semester, we need to utilize the following key indicators:

8.1.1 Completion of Requirements

We aim to fulfill the requirements outlined in Table 7.6, which serves as the benchmark for the success of our AI. Meeting all the specified requirements indicates a high level of measurable success for the team.

8.1.2 Project Design Timeline

It is crucial for each team member to adhere to the project design timeline. Regular checks should be conducted to ensure that all work is up to date and progressing according to the established schedule.

8.1.3 Effective Communication with Project Owner (Steve)

Maintaining open and effective communication with Steve, the project owner, is essential. Team members should share their current work with Steve and actively seek feedback from him. This feedback loop enables us to improve our product by incorporating valuable insights from the project owner.

9. Leadership & Meeting Arrangements

9.1 Leadership

John-Paul Masson assumed the leadership role within the team working as the primary chair in meetings, assisted by rotating members recording meeting minutes. This assisted in determining development phases and leading the team focus throughout the semester. Working to ensure members are understanding action items for next meeting and members attend them.

9.2 Meeting Arrangements

Type	**When**	**Participants**	**Attendance**	**Discuss**	**Record**
Team Meetings	Friday Class Time (when possible)	Everyone	Mandatory	- Tracking project progress and roadblocks occurring. - Allocating new tasks to be completed or initiating next phase -Roadshow Preparations	Meeting minutes
Collaborative communication	When needed	People involved	Optional	-Collaboration on completing tasks -Catching up on any missed agendas -Troubleshooting problems -Comparing models	None

Table 5 – Team Meeting Arrangements

10. Details of the Team

The DiagnosisAI team is comprised of 4 Professional, 2 Application and 1 Fundamental student. Given the range of student experience and a diverse range of majors, the team looked to utilise members strengths to lead project areas and guide those with less relevant experience. The professional students handle leadership of the team in both conducting team meetings, steering development direction and mentoring those with less experience in Machine Learning. Table 5 below shows the induvial members of the team, their studio level, experience, strength and role in the product development.

Name	**Studio Level**	**Experience**	**Strengths**	**Group Role**
Chu-Cheng Yu 24798718	Fundamental A	-1st year UTS data engineering student -6-month programming experiences	-Python Programming	-Image Classification model (CNN)
Deklin Khan 13918192	Professional A	-4th year UTS Data engineering Student -5 years' experience programming -Predictive and Classification model experience from studios	Python Programming, Data Visualisation, Machine learning	-Image classification model (CNN)
Joanna Gunawan 12581738	Application B	-2 years programming experience -Basic machine learning -Website development	-Python Programming -Data analysis -Data visualising	-Image Classification (YOLOv8) -Dashboard Creation
John-Paul Masson 13195134	Professional A	- 4th year UTS Data engineering Student. - 3 years' experience programming - Image classification experience from studios - GUI design experience from previous studios	- Python Programming - GUI design - SQL warehousing - Report writing - Presenting	- GUI design - GUI Development - Merging GUI & model into deliverable product
Shiyu Wang 13978212	Professional A	-4th year UTS electrical engineering Student. -3 years' experience programming -Image classification experience	-Python Programming	-Image classification model (CNN)
Zihao Huang 13977677	Professional A	-4th year UTS electrical engineering Student. -2 years' programming experience. -Image classification experience from studio.	-Python Programming -Pytorch	-Image classification model (CNN)

Table 6 – Team Details

11. Development Process

As a new team with a range of different experience levels in studios and university settings, the development process was designed to cater for individuals experience and desired focus area. During our initial weeks of formation, we collaborated and broke down the product requirements from the product owner Steve and identified deliverables and stages of development. These initial collaborative sessions were stage 0 of our 5-stage development process.

The first stage prioritised research on image classification for those who were new to the task and prototype testing of models for those that were familiar. The second stage we split into two teams to work on our two primary deliverables: The GUI and the image classification model. The third stage we combined our two assets into a product ready to be demonstrated in the roadshow and problem solved any issues that arose from the merge. Finally in stage 4 we verified our product met our product requirements and finished the final deliverables of the product and product report.

Development Stage	Development Action	Time Requirement	Status
DS-00	Team & product planning. Product requirement establishment	1 week	Completed
DS-01	Prototyping & researching	3 weeks	Completed
DS-02	Image classification model & GUI sub-team development	5 weeks	Completed
DS-03	Iteration and integration	2 weeks	Completed
DS-04	Product validation and deliverables	2 weeks	Completed

Table 7 – Development Stages

12. Task allocation & Timeline

The DiagnosisAI team has members from all three studio levels and with quite varying skillsets and experience. During our initial planning phase, we allocated 3 weeks for prototyping and researching, This phase exists to give studio members unfamiliar with image classification time to learn the fundamentals. When discussing as a team, two main components of the product were established: The image classification of CT lung scans as malignant, indeterminant & benign and the GUI. The model accuracy is most crucial part and so while splitting into sub-teams to handle both tasks, we allocated more people to the model. Below is a simplified Gantt chart for our development timeline, with a more detailed Gantt chart attached in Appendix H.

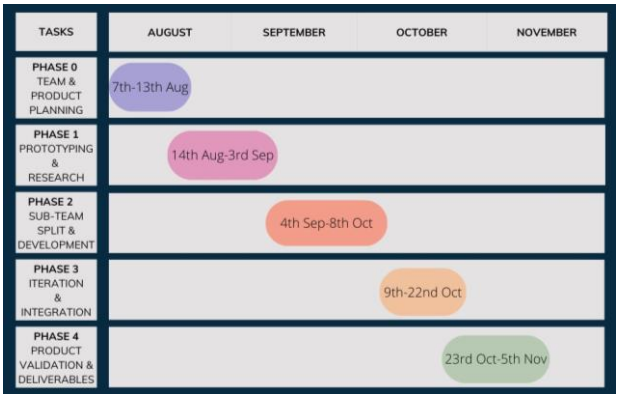


Figure 19 – Simple Gantt Chart Timeline

13. Reflection, Impact & Feedback

13.1 Level of completion reflection

Now that the semester has concluded and our product presented, I believe we have adequately completed the product for a proof of concept, however due to a mix of sociological factors and technical results of the product, it would not be market ready.

At the start of the semester our product owner Steve Ling set the expectation of "Successfully demonstrated using an advanced deep learning approach, achieving an impressive accuracy rate (e.g., 84% or higher)" (Ling, 2023) and to "Develop a GUI tool for this diagnostic system". Both of these expectations have been met, as demonstrated by the results in Table 3 – Validation & Testing. As such we have successfully satisfied the expressed needs from the product owner Steve.

Although we have met Steve's expectations, the product is still deemed as not market ready, due to the nature of society and the medical industry. This product is designed to be a supplement to doctors providing diagnosis and therefore have to compete against their diagnosis. While this product would be able to handle a larger workload of diagnosis than a doctor, the accuracy and trust of the consumer are still left to be questioned. This would result in a mistrust of the consumer (patients potentially with lung cancer) to leave the decision up to technology over a doctor with years of medical expertise.

13.2 Product Impact reflection

The product has the potential to have a drastic impact on the current status quo of Lung Cancer diagnosis, which in turn would impact lung cancer treatment, however it's realistic ability to is hindered. Currently within the Australian context, lung cancer diagnosis is completed by a small handful of medical professionals and is not common to be diagnosed by GPs (General Practitioners), which creates a bottleneck of diagnosing the "estimated…14,529 new cases of lung cancer" (Cancer Australia, 2022) Annually.

"Previous studies of diagnostic delay in lung cancer have shown a median delay between 16 and 33 days…" (Bjerager, 2006) which this product could have the potential to reduce drastically. The issue preventing this from becoming a realistic scenario is the general populus accepting machine learning possible determine the outcome of their life. As examples, a false positive diagnosis by AI could result in suffering the long-term effects of undergoing chemotherapy without a need to. Similarly, a more sever case would be a false negative diagnosis where a patient doesn't receive the treatment needed and they have a "22% chance of surviving at least 5 years" (Cancer Australia, 2022).

Due to the possible implications of a false diagnosis, unless the model has a <99% accuracy, the majority of the public would not accept such a diagnosis and the impact of the product will be non-existent.

13.3 Roadshow Feedback

The roadshow received a lot of positive feedback, indicating that the development of the product has shown significant progress and potential. Furthermore, the team demonstrated high level presentation skills where feedback included "highly informative", "easy to understand", and "good explanation of the problem". Some of the recommended improvements included expanding the application to address other cancers rather than just one as well as connecting the models with the UI and giving a full demonstration. Furthermore, the demonstration of the localisation could be enhanced, and higher accuracies could be achieved with more data collection and finetuning.

14. Conclusion

The product requirements that were establish by the product owner and the product requirements that team established have all been achieved, as demonstrated by Table 1 and Table 2 in section 2. Requirements of this report. The completion of the requirements and the milestones established by the team, leads to the product being deemed a success.

However, the real-world implementation of this product is substantially hindered by the perception of the public and the need to change the status quo of trusting doctors to make a diagnosis of a sever condition like lung cancer, to allowing AI to do that diagnosis. As such the further development of this product would be to conduct market analysis of the stakeholders and adapt the product accordingly. This would likely be increasing the accuracy to greater than 99%, updating the GUI and functionality to suit the medical environment better and changing the narrative of the product through heavy marketing.

Reference list

Ackerman, A. (2023). *Guide to Colour Meaning and Psychology*. Adobe. Retrieved November 12, 2023, from https://www.adobe.com/au/creativecloud/design/discover/color-meaning.html

Bjerager, M. (2006, November 11). *Delay in diagnosis of lung cancer in general practice*. PubMed Central (PMC). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1927095/

Brunst, H., & Döbel, S. (2023, October 24). *Efficiently using & analysing deep learning frameworks on HPC resources*. TU Dresden. https://tu-dresden.de/zih/hochleistungsrechnen/nhr-training/performance-engineering/deepl-hpc?set_language=en

Cancer Australia. (2022). *Lung Cancer in Australia Statistics*. Retrieved November 12, 2023, from https://www.canceraustralia.gov.au/cancer-types/lung-cancer/statistics#:~:text=It%20is%20estimated%20that%20it,7%2C707%20males%20and%206%2C822%20females).

Farah, L., Murris, J., Borget, I., Agathe Guilloux, Martelli, N., & Sandrine Katsahian. (2023). *Assessment of Performance, Interpretability, and Explainability in Artificial Intelligence–Based Health Technologies: What Healthcare Stakeholders Need to Know*. *1*(2), 120–138. https://doi.org/10.1016/j.mcpdig.2023.02.004

Group, C. (2023, September 22). *Healthcare Data Compliance: Security, Privacy & Compliance*. Compliancy Group. https://compliancy-group.com/healthcare-data-compliance/

Newman, T. (2018, May 17). Why is lung cancer so difficult to treat? *Medical News Today*. Retrieved November 12, 2023, from https://www.medicalnewstoday.com/articles/321817

Ossmium. (2023, September 19). *Key UX and UI Design Components During Designing Medical Information System (MIS) Interface*. Medium. https://medium.com/@ossmiumteam/key-ux-and-ui-design-components-during-designing-medical-information-system-mis-interface-ae0071b5da41

Run:AI. (n.d.). *HPC and AI: Better Together*. Run:AI. Retrieved November 12, 2023, from https://www.run.ai/guides/hpc-clusters/hpc-and-ai

Smith, J. L. (2023, March 3). *Why Is Lung Cancer So Deadly? | SERO*. SE Radiation Oncology Group. Retrieved November 12, 2023, from https://treatcancer.com/blog/why-lung-cancer-so-deadly/#:~:text=Lung%20Cancer%20Is%20Hard%20to%20Detect&text=By%20the%20time%20symptoms%20do,tumors%20are%20resistant%20to%20chemotherapy.

Streamlit. (2023). *Streamlit*. Streamlit.io. Retrieved November 12, 2023, from https://streamlit.io/

Thanoon, M. A., Zulkifley, M. A., Mohd Zainuri, M. A. A., & Abdani, S. R. (2023). A Review of Deep Learning Techniques for Lung Cancer Screening and Diagnosis Based on CT Images. *Diagnostics*, *13*(16), 2617. https://doi.org/10.3390/diagnostics13162617

Wang, L. (2022). Deep Learning Techniques to Diagnose Lung Cancer. *Cancers*, *14*(22), 5569. https://doi.org/10.3390/cancers14225569

World Health Organization. (2023, June 26). *Lung cancer*. Retrieved November 12, 2023, from https://www.who.int/news-room/fact-sheets/detail/lung-cancer

Yang, W., Wei, Y., Wei, H., Chen, Y., Huang, G., Li, X., Li, R., Yao, N., Wang, X., Gu, X., Muhammad Bilal Amin, & Byeong Ho Kang. (2023). Survey on Explainable AI: From Approaches, Limitations and Applications Aspects. *Human-Centric Intelligent Systems*, *3*(3), 161–188. https://doi.org/10.1007/s44230-023-00038-y

霹雳吧啦Wz. (2020, March 10). *7.2 使用pytorch搭建MobileNetV2并基于迁移学习训练_哔哩哔哩_bilibili*. Www.bilibili.com. https://www.bilibili.com/video/BV1qE411T7qZ/?spm_id_from=333.337.search-card.all.click

Appendix

Appendix A – Team Contributions Table

Team members	CY	DK	JG	JM	SW	ZH	Total
Background research	20	10	15	11	8	10	74
Team coordination	2	2	5	8	2	2	21
Image classification model development	30	30	10	0	30	30	130
GUI development	0	0	2	23	0	0	25
Roadshow preparation	5	7	5	7	4	5	23
Product validation	3	2	0	3	1	1	10
Report/Presentation preparation and review	6	10	5	10	10	10	51
Total	66	61	42	62	55	58	344

Appendix B – Chu-Cheng Yu Personal Milestone Table

Product specific INDIVIDUAL Milestones: Chu-Cheng Yu	% done				
	W4	W6	W8	W10	W12
W4 Milestone: -ILC contract -Product Proposal -Roadshow planning	100%	100%	100%	100%	100%
W6 Milestone: -Road show presentation -Background research -Data preprocessing	0%	80%	100%	100%	100%
W8 Milestone: -Finish building alexNet, Resnet and VGGNet.	0%	0%	100%	100%	100%
W10 Milestone: -Final roadshow preparation	0%	0%	0%	100%	100%
W12 Milestone: -Intergration of model and UI -Final roadshow presentation -Final report writing	0%	0%	0%	0%	85%

Appendix C – Deklin Khan Personal Milestone Table

Product specific INDIVIDUAL Milestones: Deklin Khan	% done				
	W4	W6	W8	W10	W12
W4 Milestone: - Individual Learning Contract agreements - Product Proposal -Roadshow planning	66%	100%	100%	100%	100%
W6 Milestone: -Testing Supervisors existing model -Roadshow Presentation	0%	75%	100%	100%	100%
W8 Milestone: - Preparing dataset - Prototyping classification model	0%	00%	50%	100%	100%
W10 Milestone: -Troubleshooting prototype model - Optimising model performance - Final Roadshow Preparation	0%	00%	00%	33%	100%
W12 Milestone: -Final Roadshow Presentation -Filming Video -Final Report writing	0%	0%	0%	0%	100%

Appendix D – Joanna Gunawan Personal Milestone Table

Product specific INDIVIDUAL Milestones: Joanna Gunawan	% done				
	W4	W6	W8	W10	W12
W4 Milestone: - Problem Statement - Project Management Plan - Product Proposal	100%	100%	100%	100%	100%
W6 Milestone: - Background research - Brainstorm of ideas - Prototype	0%	85%	100%	100%	100%
W8 Milestone: - Data analysis - ML model creation	0%	10%	75%	100%	100%
W10 Milestone: - Model testing - Model optimisation - Dashboard prototype	0%	20%	95%	100%	100%
W12 Milestone: - Connecting all parts together - Final report	0%	0%	0%	0%	85%

Appendix E – John-Paul Masson Personal Milestone Table

Product specific INDIVIDUAL Milestones: John-Paul Masson	% done				
	W4	W6	W8	W10	W12
W4 Milestone: - ILC contract submission - Product Proposal submission - Research on the issue & product	90%	100%	100%	100%	100%
W6 Milestone: - Establishment of product requirements - GUI wireframing - Roadshow presentation	0%	80%	100%	100%	100%
W8 Milestone: - GUI iterating and feedback assessment. - Feedback of ILC & PDJ	0%	20%	100%	100%	100%
W10 Milestone: - Final Roadshow Presentation - Integration of GUI and image classification model	0%	0%	30%	100%	100%
W12 Milestone: - Connecting all parts together - Final report	0%	0%	0%	0%	100%

Appendix F – Shiyu Wang Personal Milestone Table

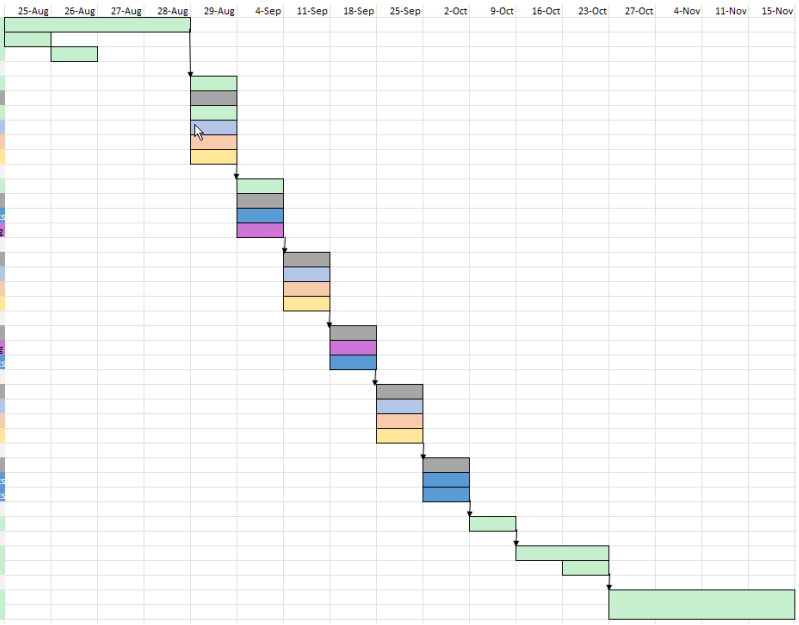
Product specific INDIVIDUAL Milestones: Shiyu Wang	% done				
	W4	W6	W8	W10	W12
W4 Milestone: - Roadshow Planning - Image Processing Learning - Problem statement	70%	100%	100%	100%	100%
W6 Milestone: - Roadshow Presentation - Pytorch Learning - Python Programming	0%	50%	90%	100%	100%
W8 Milestone: - Running Image processing - Improve coding	0%	0%	60%	80%	100%
W10 Milestone: - Model testing - Coding error fixing	0%	0%	0%	100%	100%
W12 Milestone: - Final report	0%	0%	0%	0%	100%

Appendix G – Zihao Huang Personal Milestone Table

Product specific INDIVIDUAL Milestones: Zihao Huang	% done				
	W4	W6	W8	W10	W12
W4 Milestone: - Roadshow Planning - Image Processing Learning - Problem statement	50%	40%	20%	0%	0%
W6 Milestone: - Roadshow Presentation - Pytorch Learning - Python Programming	0%	50%	40%	20%	10%
W8 Milestone: - Running Image processing - Improve coding	0%	0%	70%	50%	10%
W10 Milestone: - Coding error fixing - Final Roadshow Preparation - Coding comment	0%	0%	0%	80%	0%
W12 Milestone: - Final Roadshow Presentation - Final Report Writing	0%	0%	0%	0%	100%

Appendix H – Gantt Chart

Task Name	Duration(days	Start Date	End Date	Predecessor(s)	Team
Product Proposal	4	25-Aug	28-Aug		All
ILC Skill Clarification	1	25-Aug	26-Aug		All
Team Formation	1	26-Aug	27-Aug	3	All
Phase 1 Start(Prototyping and Research)					
Deep Learning Reasearch	6	29-Aug	4-Sep	2,3,4	All
UI Planning	6	29-Aug	4-Sep	2,3,5	UI and UX
Github Created	1	29-Aug	29-Sep	2,3,6	All
Convolution Neural Net Research	6	29-Aug	4-Sep	2,3,7	Convo
Capsule Neural Net Research	6	29-Aug	4-Sep	2,3,8	Capsule
Graph Neural Net Research	6	29-Aug	4-Sep	2,3,9	Graph
Mid Sprint Planning (4-Sep)					
Collating Research	7	4-Sep	11-Sep	6,9,10,11	All
UI & UX schema	7	4-Sep	11-Sep	13	UI and UX
Neural Network prototyping	7	4-Sep	11-Sep	13	All Neual Nets
Pre-processing of imageset	7	4-Sep	11-Sep	13	Preprocessing
Phase 1 Meeting and Phase 2 Start (Development)					
UI & UX Prototyping	7	11-Sep	18-Sep	14	UI and UX
Convolution Neural Net Initial Model	7	11-Sep	18-Sep	15,16	Convo
Capsule Neural Net Initial Model	7	11-Sep	18-Sep	15,16	Capsule
Graph Neural Net Initial Model	7	11-Sep	18-Sep	15,16	Graph
Mid Sprint Planning (18-Sep)					
Front-end Devlopment	7	18-Sep	25-Sep	18	UI and UX
Preprocessing New imageset	7	18-Sep	25-Sep	19,20,21	Preprocessing
Neural Network Testing	7	18-Sep	25-Sep	19,20,21	All Neual Nets
2 Meeting and Phase 3 Start (Iterating and Intergrating)					
UI & UX Visual upgrades	7	25-Sep	2-Oct	23	UI and UX
Convolution Neural Net Final Model	7	25-Sep	2-Oct	25	Convo
Capsule Neural Net Final Model	7	25-Sep	2-Oct	25	Capsule
Graph Neural Net Final Model	7	25-Sep	2-Oct	25	Graph
Mid Sprint Planning (18-Apr)					
Front-end Intergations	7	2-Oct	9-Oct	27	UI and UX
Neural Net optimisations	7	2-Oct	9-Oct	28,29,30	All Neual Nets
Neural Net intergrations	7	2-Oct	9-Oct	28,29,30	All Neual Nets
eting and Phase 4 Start (Polish and Presentation Preperation)					
Polishing product	7	9-Oct	16-Oct	32,33,34	All
Mid Sprint Planning (2-May)					
Presentation Preperation	7	16-Oct	23-Oct	36	All
Presentation	1	23-Oct	23-Oct	38	All
Meeting and Phase 5 Start (Final Report and Tech Transfer)					
Final Report	18	24-Oct	15-Nov	36,39	All
Tech Transfer	18	24-Oct	15-Nov	36,39	All



Appendix I – Video Presentation Link

Video Link: https://youtu.be/2u0evkHJk14

Appendix J – Code GitHub Link

GitHub Link: https://github.com/DeklinKhan/Diagnosis-AI