Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Renal Calculi Detection Using Convolutional Neural Networks

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Glennie England, Liam James

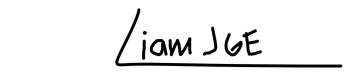
Tutor: Rodríguez Álvarez, María José

Tutor Externo: Nolan, Keith

2021-22

Declaration

I hereby certify that the material, which I now submit for assessment on the programmes of study leading to the award of Bachelor of Science, is entirely my own work and has not been taken from the work of others except to the extent that such work has been cited and acknowledged within the text of my own work. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution.



Liam Glennie
1 May 2022

Acknowledgements

I would like to thank my supervisor, Dr Keith Nolan, for his continuous guidance and support throughout the writing of this thesis. Additionally, I would like to thank my family for their support and time spent proofreading this thesis.

List of Figures

1 Traditional Neural Network Structure 11
2 Computer Vision Application 12
3 X-ray, CT Scan & Ultrasound Examples 14
4 CT Scan Cuts 18
5 Sagittal Cut CT Scan 19
6 Transverse Cut Unhealthy Kidney CT Scan 20
7 Image Grayscaling Before & After Comparison 20
8 Normalisation Code 21
9 Image Normalisation Before & After Comparison 21
10 Image Scaling Before & After Comparison 22
11 Image Rotation Comparison 22
12 Image Contrast Comparison 23
13 Model 1 Structure 25
14 Model 2 Structure 26
15 Model 3 Structure 27
16 Model 4 Structure 28
17 Transverse & Coronal Cut CT Scans 29
18 Model 5 Structure 30
19 Model 6 Structure 31

List of Tables

1 Model 1 Parameters 25
2 Model 2 Parameters 26
3 Model 3 Parameters 27
4 Model 4 Parameters 27
5 Results With Transverse Cut 28
6 Results With Coronal Cut 29
7 Model 5 Parameters 30
8 Model 6 Parameters 31
9 Results With Transverse & Coronal Cut 32
10 Model 7 Parameters & Structure 33
11 Model 7 Results With Unseen Data 33
12 Industry-standard Models Results With Test Data 35
13 Industry-standard Models Results With Unseen Data 36

Contents

1 Introduction **9**

2 Literature Review **10**

2.1 Deep Learning 10
2.2 Neural Networks 10
2.3 Computer Vision 12
2.4 Convolutional Neural Networks 12
2.5 Medical Image Analysis 13
2.6 CNNs In Medical Image Analysis 14
2.7 Measures Of Performance For Medical Imaging 14
2.8 Related Work 16

3 Methodology **17**

3.1 Data Collection 17
3.2 Data Preprocessing 19
3.2.1 Grayscaling 20
3.2.2 Image Normalisation 21
3.2.3 Image Scaling 21
3.3 Data Augmentation 22
3.3.1 Image Rotation 22
3.3.2 Applying Contrast 23

4 Model Development **23**

4.1 Tools 24
4.2 Phase I 24
4.2.1 Model 1 25
4.2.2 Model 2 25
4.2.3 Model 3 26
4.2.4 Model 4 27
4.2.5 Results 28
4.3 Phase II 29
4.3.1 Model 5 30

4.3.2 Model 6 . . . 31
4.3.3 Results . . . 32
4.4 Phase III . . . 32
4.4.1 Model 7 . . . 33
4.4.2 Results . . . 33

5 Comparison With Industry-Standard CNNs 34
5.1 VGG16 . . . 34
5.2 VGG19 . . . 34
5.3 ResNet . . . 35
5.4 Xception . . . 35
5.5 Comparing Models . . . 35

6 Discussion 36
6.1 ALTAI Ethical Assessment . . . 36
6.1.1 Human Agency . . . 37
6.1.2 Technical Robustness . . . 37
6.1.3 Privacy And Data Governance . . . 38
6.1.4 Transparency . . . 38
6.2 Is Model 7 Ready To Be A Medical Application? . . . 39

7 Conclusion 40
7.1 Summary . . . 40
7.2 Future Improvements . . . 40

Abstract

Throughout their lives, 3 in 20 men and up to 2 in 20 women develop renal calculi at some stage. Renal calculi are solid masses that originate in the kidneys but can develop along the urinary tract. They appear when solutes from urine crystalise to form calculi. Calculus formation is related to diet, urinary tract infections and medications.

In most cases, renal calculi can result in excruciating pain and agony. Moreover, although it does not lead to kidney failure, recurrent renal calculi can result in a functional loss of the kidney.

With the help of convolutional neural networks, image preprocessing, data augmentation and Python, TensorFlow, and Keras libraries, we have built a classification model to detect renal calculi from abdominal CT scans. Consequently, we go through an iterative process of adjusting the models and preprocessing techniques to improve their performance. Finally, we compare the best performing model against industry-standard architectures such as VGG16, VGG19, ResNet and Xception. We conclude that our model outperforms the industry-standard CNNs, but it is not ready to become a medical application.

1 Introduction

Renal calculi are solid masses that originate in the kidneys but can develop along the urinary tract. The development of renal calculi can result in excruciating pain with possible consequences being nausea, severe lower back pain, blood in urine, fever and chills. If not identified in time, the treatment for renal calculi extraction is removed by surgery. However, if, when identified, the calculus is small enough, it can be passed without surgery. Over the past decades, the increase in high-salt diets could have also led to the rise in cases. Renal calculi can also be formed due to hereditary kidney problems, medications and urinary tract infections.

In recent decades, Ireland and the rest of the world have witnessed an increase in radiologists' demand. However, this demand has not been entirely satisfied, leading to a shortage of radiologists and long waiting periods for patients in most hospitals. It is predicted that this steady increase in demand will continue to rise in the coming years. [8]

The most common case is that a radiologist will assess the CT scan and identify any renal calculi. There exists literature that documents previous attempts to automate this form of diagnosis. We will cover this in Section 2.

This project aims to develop a model that can accurately identify renal calculi from CT scans using deep learning techniques such as convolutional neural networks. By achieving this, we hope that this would reduce the workload of medical staff, particularly radiologists while increasing the number of successful diagnoses and reducing waiting periods for patients.

We will start with an overview of technical terms explaining matters like deep learning, neural networks, CNNs, their use in the medical field, and previous work similar to our project. In the next section, we will explain the methods applied to collect and prepare the data used to train the models. In the following chapters, we will cover the three phases of model development, describing the models that we have built and a review of the performance of said models. Next, we will compare our best performing model to industry-standard CNNs, and, finally, we will provide our conclusions and future work to be done.

2 Literature Review

This section will discuss the technologies that we have used throughout this project, their application to medical analysis, and previous work related to our project.

2.1 Deep Learning

Deep learning has been around for many decades now, with the first implementation dating back to 1943 with McCulloch and Pitts. However, they only started attracting interest in the late 2000s and early 2010s when we finally had the computational power to truly take advantage of this technology [11]. Deep learning is a subset of machine learning that attempts to mimic the human brain in the form of artificial neural networks. These networks are used to find patterns in data.

The main difference between machine learning and deep learning is the amount of human intervention and data needed for training the models. Several preprocessing techniques are done manually to extract the essential features from a dataset in machine learning. However, preprocessing is not as crucial in deep learning because neural networks can find the most important patterns and features within a dataset. Nonetheless, this comes at a price. Usually, deep learning algorithms need much more data to learn than machine learning algorithms.

Deep learning has many applications like image recognition, natural language processing, recommendation systems, financial fraud detection and medical analysis.

2.2 Neural Networks

Neural networks (NN) are architectures that represent how the human brain learns and are used in many applications to automate processes. First, we will explain the components of a NN, then describe the typical structure of one.

- **Node:** A simple data structure that can connect to other nodes via edges.
- **Neuron:** A node in a neural network. A neuron receives one or more weighted inputs and sums them using an activation function to produce an output.

- **Edge:** Link between two neurons that has a weight associated with it.
- **Weight:** A real value represents the influence one neuron has on the output of another neuron.
- **Activation function:** Calculates the output of each neuron by applying a nonlinear function.

Neural networks consist of 3 different types of layers formed by neurons connected by edges, observed in Figure 1. The first layer is the input layer, where each neuron represents the data fed to the network. These could be words, pixels or numerical values. Then we have the hidden layers, where each component is considered a neuron that calculates an output from its weighted inputs and activation function. There can be as many hidden layers as necessary. The more neurons, the taller the network is, and the more layers, the deeper it is. Deep networks tend to be preferred over tall networks because they achieve the same level of performance while using fewer parameters, making them much more efficient in terms of computation. Finally, we have the output layer, which represents each of the possible classes to which the data can belong. If given enough data and time, neural networks are powerful enough to represent any possible existing mapping function. [10]

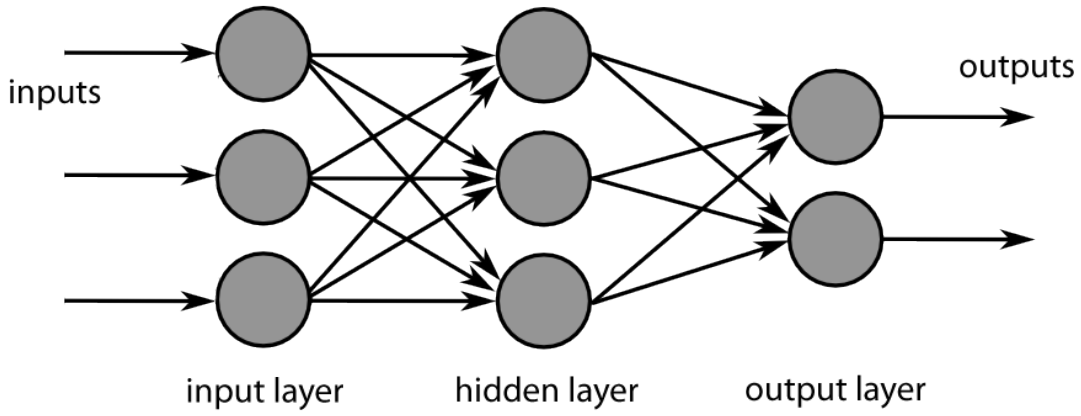


Figure 1: Traditional Neural Network Structure

2.3 Computer Vision

An application of deep learning is computer vision, a field of artificial intelligence that allows computer systems to gain information from digital data such as videos or images and make decisions based on the conclusion gained from such data. In recent times, computer vision has gained significant importance in AI. Its rise in deep learning can be attributed to convolution neural networks. This technology has allowed computer vision to embrace its full potential and be used for medical analysis, autonomous vehicles and facial recognition. Figure 2 shows an example of computer vision.

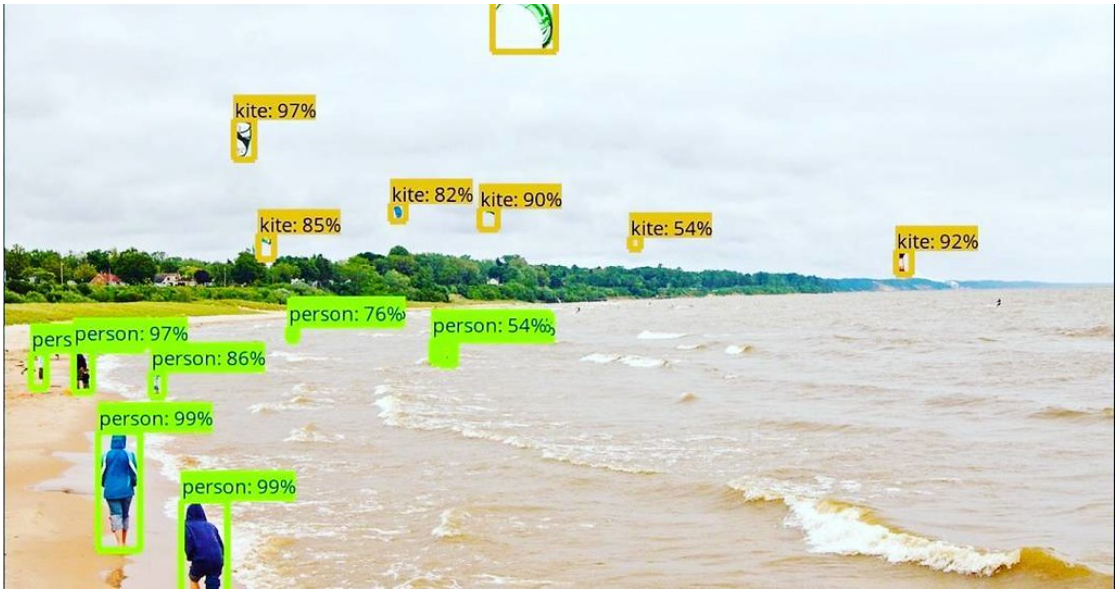


Figure 2: Computer Vision Application

2.4 Convolutional Neural Networks

CNNs are traditionally structured in the following manner. First, we have one or more convolutional layers that process the input data and extract features from it. These features are then fed to a neural network. Convolutional layers have components known as receptive fields and filters.

The receptive field can be explained as the region that extracts features from the input image data. In other words, the number of pixels covered from the input

data to extract relevant features. The receptive field is modified depending on the size of the image. If we were looking to recognise an object within a large image, we would probably use a large receptive field. Moreover, we would likely use a smaller receptive field for smaller images.

A filter is a 2D array of real values that slides over the input image and is used to generate a feature map. This map is calculated with the dot product of the filter and the input image data. It is very common to use many filters in parallel to produce different feature maps. The number of filters varies depending on the complexity of the shapes or features found within the image. For example, for complex shapes, like the parts of a car, we would introduce more filters to produce more feature maps because complex shapes tend to need more features to be extracted to classify them correctly. On the other hand, we would use fewer filters to produce feature maps for simple shapes. These feature maps can then be used to visualise the characteristics that the model is extracting from the input data. Essentially, they can be used to analyse whether the model is focusing on the correct features to classify the model.

It is worth mentioning that we also implemented pooling layers in our models, which reduce the dimensionality of the feature maps. Therefore, reducing the number of parameters and computation power needed.

After the data has been fed through the convolutional layers, we obtain a 2D array of weights representing a feature and its importance when classifying the image. We apply this output to a neural network that processes the information and, depending on the implementation, it returns the class calculated from the input. Thus, CNNs essentially break down an image into minor features or characteristics and use a neural network to classify such images. [11]

2.5 Medical Image Analysis

Medical imaging provides visual information about the human body and its organs, aiding medical professionals in diagnosing. Some examples of medical imaging, such as X-rays, computed tomography (CT), and ultrasound can be found in Figure 3. These technologies have played a crucial role in the advancement of modern

healthcare. They allow medical professionals to visualise the human body's interior and identify any anomalies. [3]

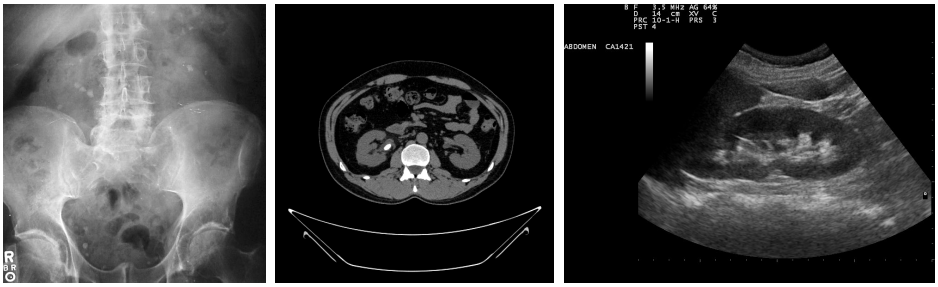


Figure 3: X-ray, CT Scan & Ultrasound Examples

2.6 CNNs In Medical Image Analysis

In recent years, CNNs have gained attraction in medical image analyses. In traditional machine learning, the algorithms work off the hand-picked features selected by a human. In contrast, CNNs are fed the raw data and extract the most important features to classify the data with the help of the neurons or perceptrons and convolution layers. This allows the CNN model to learn complex information that could be vital in identifying an image as healthy or unhealthy.

However, there are not only positive aspects to the use of CNNs. There are limitations, such as the computational power needed to train these models. Also, for a CNN model to learn well, it needs a large amount of image data. Nevertheless, if there is a large amount of data, it takes the model more time to learn. Finally, the ethical problem. CNN models are known as black-box models. Therefore, the inputs and outputs of the model are known, but the internal representation is not. In other words, it is difficult to understand why a CNN model classifies an image as healthy or unhealthy. [3]

2.7 Measures Of Performance For Medical Imaging

Although it is important in other fields, it is critical in the medical world to achieve high accuracy with an equal balance in the model's performance for each class. That is why accuracy should not be the only measure of performance. Other

measures such as sensitivity, specificity, precision, recall, and F1-score should be calculated.

- **Sensitivity:** Ratio of the positive class correctly classified. It is also known as recall.

$$Sensitivity = \frac{TP}{TP + FN} \tag{1}$$

- **Specificity:** Ratio of the negative class correctly classified.

$$Specificity = \frac{TN}{TN + FP} \tag{2}$$

- **Precision:** Ratio of instances that were correctly classified as positive.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

- **$\mathbf{F1_{score}}$:** Weighted average of precision and recall. It measures the balance between the two metrics. A score of 1 means both precision and recall are 1, and a score of 0 means that either the precision or recall are 0.

$$F1_{score} = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

- **Accuracy:** Ratio of instances classified correctly.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

These metrics should be calculated and analysed to avoid producing models that are biased towards one class. In the following example, the unhealthy class is considered the positive class, and the healthy class the negative. So, we could train a model to detect pancreatic cancer from ultrasound images and only train it with images of healthy patients. If we were to release this model and test it on a random sample of the whole population on Earth, it would probably have very high

accuracy. However, its sensitivity would likely be extremely low because the model has never been exposed to images of patients with pancreatic cancer. Therefore, it is very probable that this model would classify patients with pancreatic cancer as healthy. This type of error is known as type-II error, and it is the worst error a model could produce because it could have fatal consequences. Perhaps, a patient incorrectly classified as healthy does not get the treatment they need and does not survive because the model was biased toward the healthy class. Another type of error could derive from model bias: type-I errors. These errors occur when a model classifies a healthy subject as unhealthy. Type-I errors are not as bad as those of type-II. However, they are still not desirable. [3]

2.8 Related Work

Several projects have been published related to renal calculi detection from CT scans. For example, [13] built a classification model using k-Nearest Neighbours (KNN) and Support Vector Machines (SVM) to identify renal calculi from ultrasound images (US). An unknown number of images were used to train the models. However, the paper used preprocessing techniques such as filters to combat US images' low resolution and quality. Next, they applied Principal Component Analysis (PCA) and image segmentation to extract the most relevant features to feed to the machine learning models. Finally, this data is fed to the KNN and SVM to be classified. The results show that KNN classified the US images with 89% accuracy and SVM with 84%. We should note that accuracy is the only metric of performance used, meaning that we do not know which class it is incorrectly classifying. It could be the healthy class, the unhealthy one, or both. With the accuracy alone, we do not have enough information on the true performance of these two models.

Another related paper, [1], used Fuzzy C-means clustering (FCM) to identify and locate renal calculi from CT scans. The image dataset used for this project is composed of CT scans from 50 patients with CT image slices of the renal calculus selected and stored for each patient. Therefore, we can say that all the data used for this work are from patients with renal calculi. Preprocessing techniques such as cropping, grayscaling and matrix calculations were used. Then, this data

was fed to the FCM model, which returned the location and size in pixels of the calculus. Throughout this paper, this model's only indication of performance is a table where renal calculi were detected and their location in the image. However, we do not know how accurate this location is or whether there were cases in which the model incorrectly detected calculi in the image. All in all, there is a lack of information when it comes to how good this model truly is.

[4] is probably the most similar to the project we have proposed. They used a dataset of 206 images of kidney CT scans. Preprocessing techniques such as median filtering, grayscaling, and gamma correction were used to reduce the noise produced by speckles and enhance the image's quality. Following this, K-means filtering was applied to obtain the area of interest in the image and the 12 most important features for classification. These features were used to train an Artificial Neural Network (ANN), which returned an accuracy of 80%. However, by using 3 of the 12 features extracted from the 206 images, the SVM classifier obtained an accuracy of 95%. Nonetheless, there is not much information about the origin of the images, breakdown of classes or number of patients.

3 Methodology

This section will cover how the data was collected, and which pre-processing and data augmentation techniques were used to improve the dataset.

3.1 Data Collection

The dataset we have used for this project contains 12,446 unique images of abdomen CT scans. The dataset is divided into healthy, calculi, cysts, and tumours. There are 5,077 instances of healthy kidneys, 1,377 kidneys with calculi, 3,709 with cysts, and 2,283 with tumours. The dataset contains images for each of the different cuts produced by a CT scan, Figure 4.

The images were collected from the Picture Archiving and Communication System (PACS) of different hospitals in Dhaka, Bangladesh. The patients' information and the images' metadata were removed from the data allowing it to be anonymous.

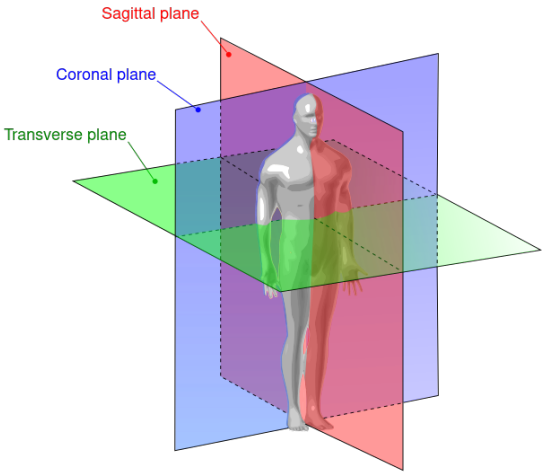


Figure 4: CT Scan Cuts

The original format of the photos was Digital Imaging and Communications in Medicine (DICOM), but they were converted to lossless JPG to train the model. DICOM images can reach the size of 35 MB, making training the model costly in terms of time and computation. After the conversion, each image was verified by a radiologist and medical technologist, assuring that the data continued to be correct.

Given that the dataset contains images of renal calculi, cysts and tumours, we were only concerned with images of healthy kidneys or those with renal calculi. Thus, we removed instances of kidneys with cysts and tumours. Furthermore, we removed all scans taken on the sagittal cut, Figure 5, as there were only 14 instances for kidneys with calculi and 0 for healthy ones. It would not have been correct to train the models with only images for one of the classes because the model would be biased toward that class. In other words, it is very likely that if the model received a sagittal cut image as an input, it would classify as an unhealthy kidney regardless of its true class. We will refer to kidneys with renal calculi as unhealthy kidneys from now on.

Figure 5: Sagittal Cut CT Scan

3.2 Data Preprocessing

Data preprocessing is a set of techniques used to clean and normalise data ready to be fed to a machine or deep learning model. It is a vital process that can significantly improve the performance of a model. Typically, it involves removing any erroneous or noisy data to ensure that the model can learn from the data optimally without it being negatively affected by a few incorrect instances. With image data, one of the techniques usually applied is to resize all the images to the same size because the model always expects the data to be in the same dimension.

In our case, we could say that the image data we possess is almost too clean. That is, it is not representative of real-world data. So, if we trained our model on the data without preprocessing, it may perform well in training and testing, but when it is shown real-world data, it could perform terribly because it was not prepared for that type of data. Therefore, we will apply preprocessing techniques to make the models we build more robust to real-world scenarios.

Initially, we chose to train our models only on transverse cut images, Figure 6, because this cut was the most predominant in the dataset. Therefore, we began preprocessing with 4,111 images. Nevertheless, this number of instances is relatively small regarding the data needed to train a CNN. Consequently, as well as preprocessing, we performed data augmentation to achieve a more significant number of instances and improve the model's performance.

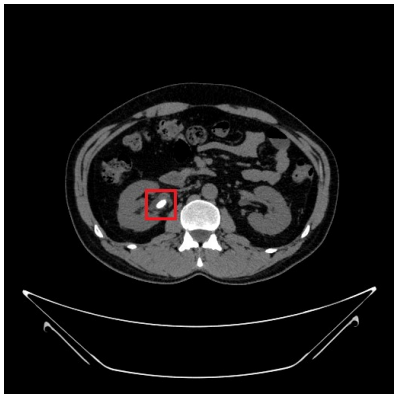


Figure 6: Transverse Cut Unhealthy Kidney CT Scan

3.2.1 Grayscaling

All the images are in black and white, as observed in Figure 7. However, they are in RGB format, meaning that each image was a three-dimensional array with three values for each pixel. Therefore, we converted the images to a one-dimensional array with one value per pixel. In other words, we converted each RGB image to grayscale. This operation was performed because the RGB data provided no additional information than the grayscale images. Also, by reducing the number of dimensions, we transform the data to be computationally more efficient and reduce its size in terms of storage space.

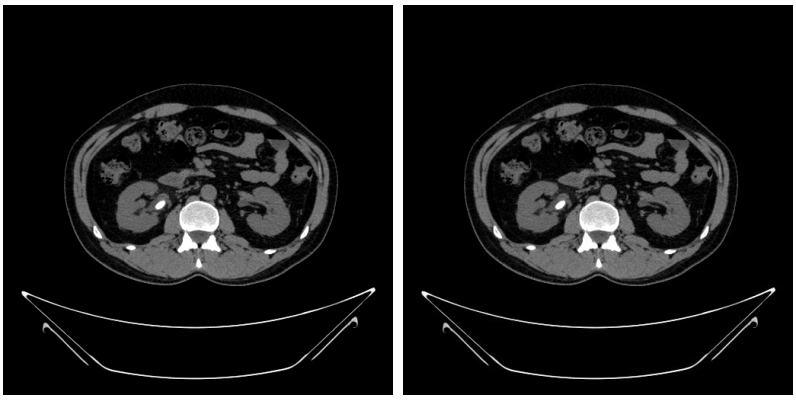


Figure 7: Image Grayscaling Before & After Comparison

3.2.2 Image Normalisation

Image normalisation is applied to ensure that all the data is within the same scale. In our case, by using it, we guaranteed that the pixels in every image have a value between 0 and 255. Each image has a pixel with a value of 0 and a pixel with a value of 255. As we can see, there is a very slight difference in contrast and colours between the images compared in Figure 9. Essentially, image normalisation is used to improve the contrast of image data. We took this step to enhance the performance of the model. Figure 8 is the code used to normalise an image's pixels.

```
img = np.array(image)
norm = (img - np.min(img)) / (np.max(img) - np.min(img))*255
```

Figure 8: Normalisation Code

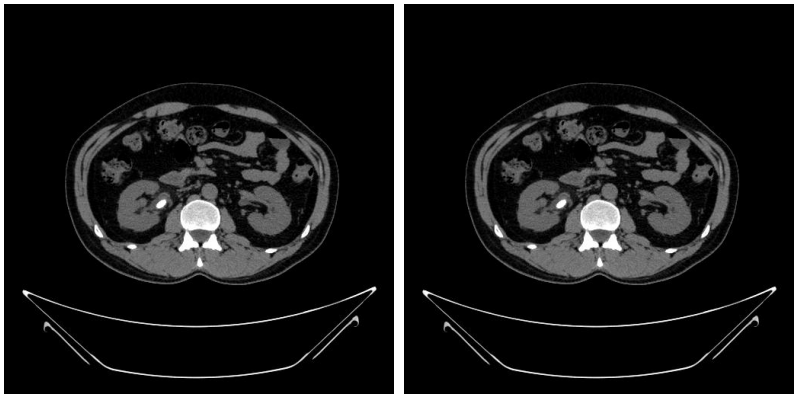


Figure 9: Image Normalisation Before & After Comparison

3.2.3 Image Scaling

The original data was composed of images of different sizes. So, to feed this data to a model and due to computational limitations, we scaled the images to 128x128 pixels. These dimensions were chosen because it was the first size where the renal calculi could be visibly differentiated. Additionally, we did not apply antialiasing as this moved the kidneys to similar coordinate ranges, and we wanted to maintain as much variability as possible. Figure 10 showcases the difference in the resolution after scaling the images.

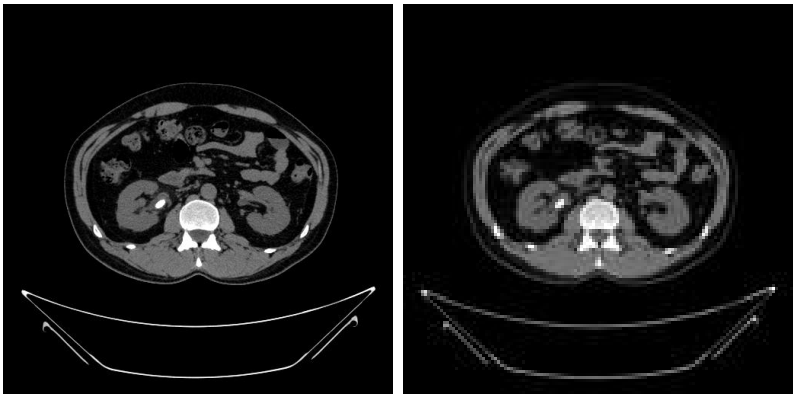


Figure 10: Image Scaling Before & After Comparison

3.3 Data Augmentation

Data augmentation is a set of techniques to achieve more data instances by creating slightly modified copies of the original data and adding them to the dataset. It is used to increase the performance of models and reduce overfitting. As mentioned earlier, 4,111 images are not enough data to optimally train a CNN model. Therefore, we applied some data augmentation techniques to achieve more data to train the model.

3.3.1 Image Rotation

The first technique we applied is image rotation, Figure 11. So, we rotated each instance in intervals of 60^{o} starting from 0^{o}, increasing the number of cases by six times the original size and introducing more variety into the dataset.

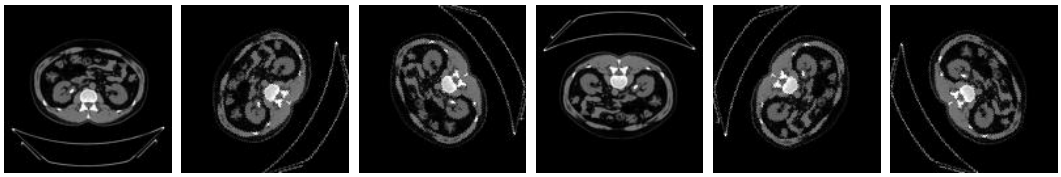


Figure 11: Image Rotation Comparison

3.3.2 Applying Contrast

The images captured using computer tomography may have low contrast as it is a dependent variable that can be influenced by the subject or the machinery used. Some causes of low contrast images can be using a low-radiation dose during the examination or using different storage, transmission, or display devices. Another cause can be noise in the captured image. [2]

To emulate these imperfections, we applied different contrast to each of the scaled rotated images from 40% to 160% in intervals of 20%, Figure 12. This form of data augmentation, like rotation, brings more variety to the data set, making the model more robust to inputs with different contrasts. Thus, improving the overall performance of the model and reducing overfitting.

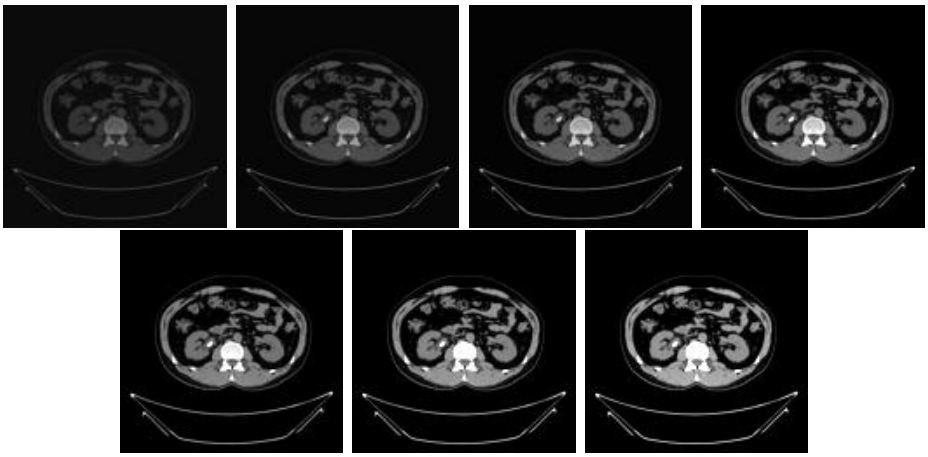


Figure 12: Image Contrast Comparison

After performing the preprocessing and data augmentation techniques mentioned, we increased the size of our dataset from 4,111 images to 172,666. The increase in instances allowed us to train the model with more diverse data, resulting in a more well-rounded and robust model.

4 Model Development

We have opted to implement models using convolutional neural networks to classify the data as healthy or unhealthy kidneys. In the following sections, we will discuss

the tools used and the architecture of the models.

4.1 Tools

We have elected to use TensorFlow as the library to develop the models because it offers NVIDIA GPU support. It has a large and helpful community, making debugging more manageable, and it is easy to use. Most importantly, TensorFlow offers Keras support. Keras is a high-level API that allows us to build machine and deep learning solutions using TensorFlow's capabilities.

4.2 Phase I

Four models were built to classify abdomen CT scans using the transverse cut. All the models were trained with a random subset of the augmented data set. The said subset is composed of 10,000 healthy kidneys and 5,000 unhealthy kidney instances. We applied a validation split of 60%-40%. 9,000 images were used to train the model and 6,000 to test it.

The four models contain convolution, max-pooling and dense layers and implement the Adam algorithm for optimisation, a stochastic gradient descent algorithm based on adaptive estimates of lower-order moments. It is a computationally efficient algorithm with small memory requirements suited for machine learning problems with extensive data or many parameters.

As this is a classification problem with two classes, we have opted for the binary cross-entropy loss function at the output layer. For all the convolutional and dense layers, we have used the ReLU activation function, except for the output layer, which implements the Sigmoid activation function. We have chosen ReLU (6) over Sigmoid (7) as the activation function because it is more efficient, has better gradient propagation, and converges quicker than the Sigmoid function.

$$f(x) = max(0, x) \tag{6}$$

$$S(x) = \frac{1}{1 - e^{-x}} \tag{7}$$

4.2.1 Model 1

Model 1 is a convolutional neural network that consists of 6 layers which was trained for 6 epochs with a batch size of 120. Its parameters and structure can be seen in Table 1 and Figure 13.

2D Conv	Max-pooling	2D Conv	Max-pooling	Hidden Dense	Output Dense
Input: 128x128 **Filters:** 64 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid	**Filters:** 128 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid	**Neurons:** 128 **Act. func.:** ReLu	**Neurons:** 1 **Act. func.:** Sigmoid

Table 1: Model 1 Parameters

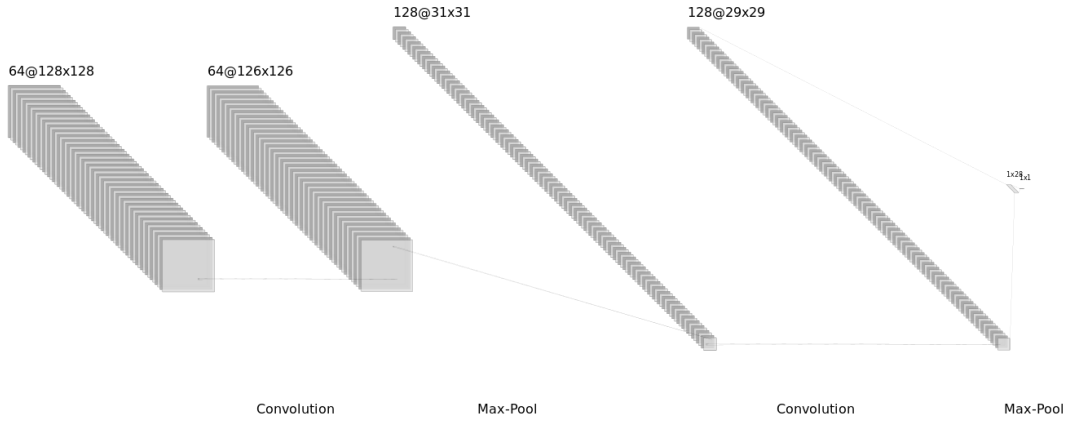


Figure 13: Model 1 Structure

4.2.2 Model 2

Model 2 is a convolutional neural network that consists of 6 layers which was trained for 10 epochs with a batch size of 120. This second model attempts to

obtain similar performance to Model 1 while being computationally more efficient. Its parameters and structure can be seen in Table 2 and Figure 14.

2D Conv	Max-pooling	2D Conv	Max-pooling	Hidden Dense	Output Dense
Input: 128x128 **Filters:** 32 **Filter size:** 5x5 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 4x4 **Padding:** Valid	**Filters:** 64 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid	**Neurons:** 28 **Act. func.:** ReLu	**Neurons:** 1 **Act. func.:** Sigmoid

Table 2: Model 2 Parameters

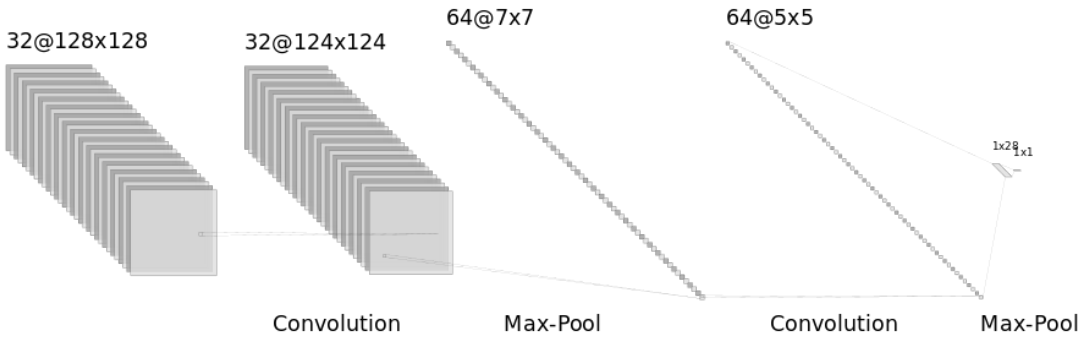


Figure 14: Model 2 Structure

4.2.3 Model 3

Model 3 is a convolutional neural network that consists of 4 layers which was trained for 15 epochs with a batch size of 120. Much like Model 2, this version's objective is to obtain similar performance while reducing the computational complexity. Model 3 has fewer layers and produces fewer features to be fed to the dense layers. Its parameters and structure can be seen in Table 3 and Figure 15.

2D Conv	Max-pooling	Hidden Dense	Output Dense
Input: 128x128 **Filters:** 32 **Filter size:** 9x9 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 5x5 **Padding:** Valid	**Neurons:** 256 **Act. func.:** ReLu	**Neurons:** 1 **Act. func.:** Sigmoid

Table 3: Model 3 Parameters

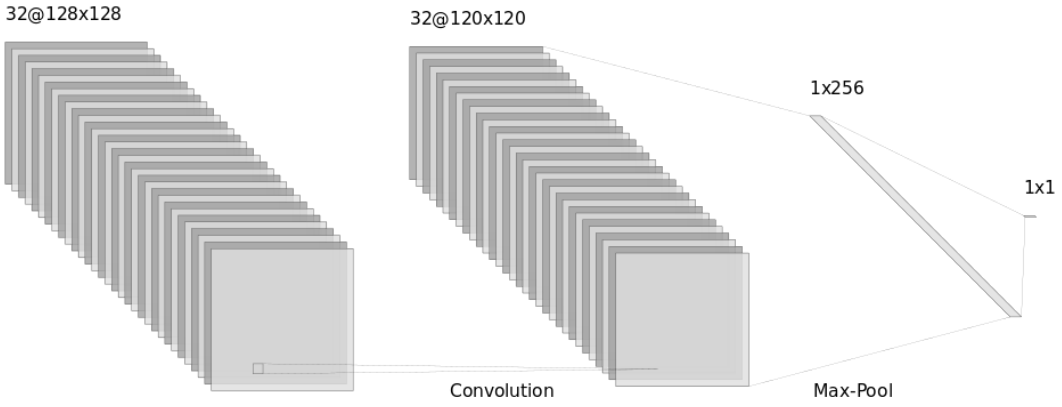


Figure 15: Model 3 Structure

4.2.4 Model 4

Model 4 is a convolutional neural network that consists of 6 layers which was trained for 15 epochs with a batch size of 120. This version reverts to a similar architecture to Models 1 and 2. This model has fewer filters per layer and fewer neurons than Model 1 in the hidden dense layer, making it more efficient. Its parameters and structure can be seen in Table 4 and Figure 16.

2D Conv	Max-pooling	2D Conv	Max-pooling	Hidden Dense	Output Dense
Input: 128x128 **Filters:** 16 **Filter size:** 7x7 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 4x4 **Padding:** Valid	**Filters:** 32 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid	**Neurons:** 32 **Act. func.:** ReLu	**Neurons:** 1 **Act. func.:** Sigmoid

Table 4: Model 4 Parameters

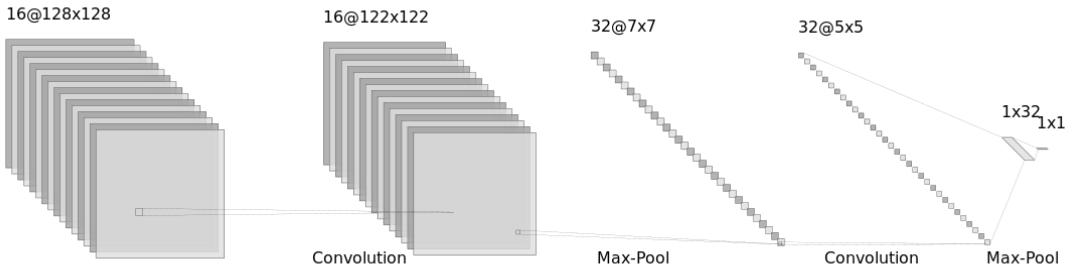


Figure 16: Model 4 Structure

4.2.5 Results

Transverse Cut Data

As we can see in Table 5, all four models showcase a high performance in detecting renal calculi in transverse cut images with very little bias toward the healthy class.

	Accuracy	**Precision**	**F1-Score**	**Specificity**	**Sensitivity**
Model 1	99.96%	99.98%	0.9983	99.99%	99.89%
Model 2	99.95%	99.87%	0.9988	99.97%	99.90%
Model 3	99.67%	99.24%	0.9920	99.80%	99.16%
Model 4	99.89%	99.60%	0.9972	99.90%	99.84%

Table 5: Results With Transverse Cut

Coronal Cut Data

After observing the high performance of all four models, we tested the most efficient and best model on different data. We performed the same preprocessing and data augmentation techniques on the coronal cut data, transforming 2,328 images to 97,776. We tested this data on Model 2, which was trained using only transverse

cut images.

Table 6 shows that Model 2's performance dropped considerably compared to when tested on transverse cut images. This is expected as the model was only trained for transverse cut images. We observed that Model 2 struggled with false negatives during testing, classifying unhealthy kidneys as healthy. Below we can see the overall statistics of this model.

	Accuracy	**Precision**	**F1-Score**	**Specificity**	**Sensitivity**
Model 2	73.96%	41.58%	0.3164	86.38%	25.54%

Table 6: Results With Coronal Cut

4.3 Phase II

Intending to build complete models to identify CT scans of kidneys as healthy or unhealthy, we retrained the models described above with transverse and coronal cut data, Figure 17. Additionally, we built two more models that were more robust and offered slightly higher performance than the previous models. We will only describe the new models in the following sections and compare their performance to the previous ones.

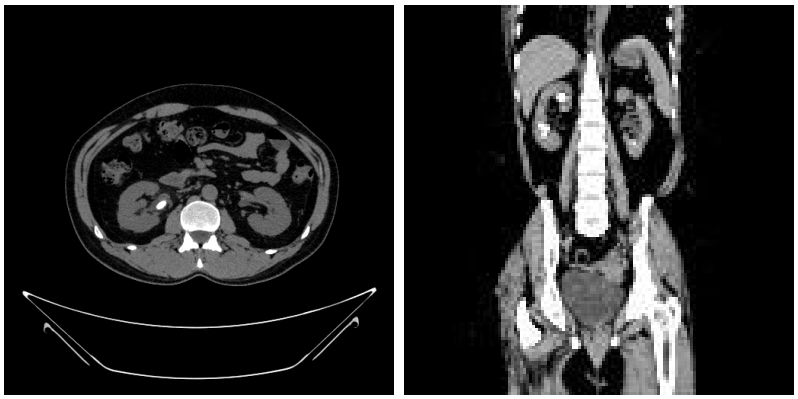


Figure 17: Transverse & Coronal Cut CT Scans

4.3.1 Model 5

Model 5 is a convolutional neural network that consists of 7 layers which was trained for 15 epochs with a batch size of 120. The architecture is very similar to Model 4. The difference can be found in the hidden dense layers, as this model has two. Its parameters and structure can be seen in Table 7 and Figure 18.

2D Conv	Max-pooling	2D Conv	Max-pooling
Input: 128x128 **Filters:** 16 **Filter size:** 7x7 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 4x4 **Padding:** Valid	**Filters:** 32 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid

Hidden Dense	Hidden Dense	Output Dense
Neurons: 48 **Act. func.:** ReLu	**Neurons:** 16 **Act. func.:** ReLu	**Neurons:** 1 **Act. func.:** Sigmoid

Table 7: Model 5 Parameters

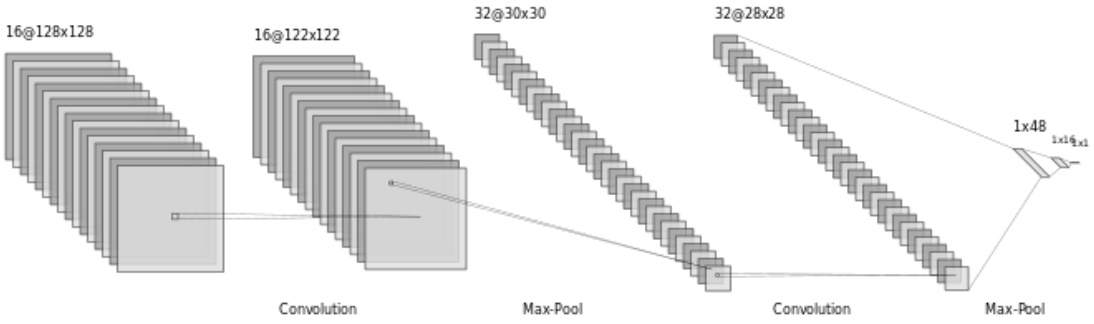


Figure 18: Model 5 Structure

4.3.2 Model 6

Model 6 is a convolutional neural network that consists of 7 layers which was trained for 12 epochs with a batch size of 120. Again, this version is very similar to Model 5. However, to improve performance, we made the hidden dense layers taller. That is, we added more neurons per layer. Its parameters and structure can be seen in Table 8 and Figure 19.

2D Conv	Max-pooling	2D Conv	Max-pooling
Input: 128x128 **Filters:** 16 **Filter size:** 7x7 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 4x4 **Padding:** Valid	**Filters:** 32 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid

Hidden Dense	Hidden Dense	Output Dense
Neurons: 128 **Act. func.:** ReLu	**Neurons:** 32 **Act. func.:** ReLu	**Neurons:** 1 **Act. func.:** Sigmoid

Table 8: Model 6 Parameters

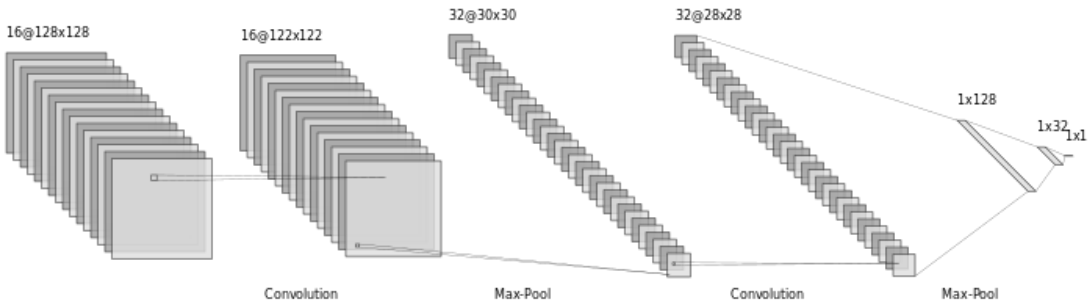


Figure 19: Model 6 Structure

4.3.3 Results

In Table 9, we can observe that, when retrained on transverse and coronal cut data, the original models, 1 to 4, mostly recovered to their initial performance. Nonetheless, models 5 and 6 slightly outperform them in terms of accuracy.

	Accuracy	**Precision**	**F1-Score**	**Specificity**	**Sensitivity**
Model 1	99.13%	98.45%	0.9795	99.59%	97.45%
Model 2	99.35%	98.05%	0.9850	99.46%	98.96%
Model 3	98.77%	97.12%	0.9713	99.21%	99.14%
Model 4	99.28%	98.60%	0.9829	99.61%	97.99%
Model 5	99.64%	99.10%	0.9916	99.75%	99.21%
Model 6	99.69%	99.19%	0.9928	99.78%	99.37%

Table 9: Results With Transverse & Coronal Cut

Up until now, we have been testing the models on images from the dataset [9]. As we mentioned earlier, these CT scans were all obtained from hospitals in the same city, and it is likely they are all captured with the machine model, making the images very similar. Of course, the preprocessing and data augmentation techniques we have used to introduce variety into the data should make the models more robust and maintain an equivalent level of performance when tested with real-world data.

To truly observe the performance of our best model (Model 6), we must test it against completely unseen data. Therefore, we collected 45 images from the Internet, 24 unhealthy and 21 healthy kidneys. After testing Model 6, its accuracy dropped to 64.44%, with a sensitivity of 72% and a precision of 66.67%.

4.4 Phase III

We changed the preprocessing and the healthy-unhealthy ratio used for training to improve the model's performance. First, we observed that unseen data instances were more cropped and had more noise than the training data. So, we randomly cropped 10,710 images during preprocessing and introduced Gaussian noise layers into the model. Then, we noticed that the model learned healthy kidneys much better than unhealthy ones. Therefore, we changed the number of healthy and

unhealthy images used during training to 12,500 healthy and 15,000 unhealthy kidneys. Previously the ratio had been 2/3 healthy kidneys and 1/3 unhealthy. Now, there are more unhealthy kidney instances used than healthy instances. Finally, we introduced dropout layers between the hidden layers to reduce overfitting. After many iterations of models, we have built the following:

4.4.1 Model 7

Model 7 is a convolutional neural network that consists of 12 layers which was trained for 22 epochs with a batch size of 120. It is a considerably larger model, but most additional layers have been implemented to reduce overfitting. Its parameters and structure can be seen in Table 10.

2D Conv	Max-pooling	2D Conv	Max-pooling	2D Conv	Max-pooling
Input: 128x128 **Filters:** 16 **Filter size:** 9x9 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 4x4 **Padding:** Valid	**Filters:** 64 **Filter size:** 5x5 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid	**Filters:** 32 **Filter size:** 3x3 **Stride:** 1 **Padding:** Valid **Act. func.:** ReLU	**Filter size:** 2x2 **Padding:** Valid

Gaussian Noise	Hidden Dense	Dropout	Hidden Dense	Dropout	Output Dense
Value: 0.2	**Neurons:** 768 **Act. func.:** ReLu	**Value:** 0.2	**Neurons:** 96 **Act. func.:** ReLu	**Value:** 0.2	**Neurons:** 1 **Act. func.:** Sigmoid

Table 10: Model 7 Parameters & Structure

4.4.2 Results

In the Table 11, we can observe the performance metrics for Model 7 with the unseen data.

	Accuracy	**Precision**	**F1-Score**	**Specificity**	**Sensitivity**
Model 7	75.56%	76.92%	0.7843	70.00%	80.00%

Table 11: Model 7 Results With Unseen Data

Initially, we developed six models that could classify kidneys as healthy or unhealthy from transverse cut CT scans. Consequently, we started an iterative process of testing the models and retraining them to adapt to different data. We began with retraining the models so they could detect renal calculi in coronal and transverse cut CT scans. Then, after testing the best performing model on unseen data, we adapted the training process and the preprocessing of the data to achieve

higher performance. These changes showed an accuracy increase from almost 50% to 75.56%. It is also worth noting that the final model, Model 7, has only a small amount of bias toward the positive class. It is slightly better at classifying unhealthy kidneys than healthy ones.

5 Comparison With Industry-Standard CNNs

The next step is to analyse how our best performing model compares to industry-standard models like VGG16, VGG19, ResNet and Xception. We trained all four models on the same images as our Model 7 and tested them with the same unseen data to compare their performance differences. Also, these models were trained using a validation split of 0.4 and a batch size of 120. We applied the early stopping technique to monitor the validation accuracy with a patience of 3 to avoid overfitting.

5.1 VGG16

VGG16 is a convolutional neural network presented at the 2014 ImageNet competition sponsored by Google and Facebook. VGG16 introduced a new idea in the form of stacked convolution layers with small filter sizes, which innovated the world of computer vision.

VGG16 is a CNN formed by a combination of 16 layers that include convolution max-pooling and dense layers. The original model was built for 224x224 RGB images, but we modified the input layer to accept grayscale images of 128x128. [12]

5.2 VGG19

Like VGG16, this model was also submitted to the 2014 ImageNet. The main difference between the configuration of VGG16 and VGG19 is the number of layers. VGG19 has three extra convolution layers. This model was also built for 224x224 RGB images. [12]

5.3 ResNet

ResNet is a residual network that reformulates layers as residual functions referencing the input layers. These types of networks are seemingly easier to optimise than networks that learn unreferenced functions. ResNet was the winner of the 2015 ImageNet classification task. The configuration we have used is ResNet50V2 which is composed of 50 layers, and despite it having more layers, it is more efficient than both the VGG configurations we have elected. [6]

5.4 Xception

Xception is a depthwise separable convolutional neural network that takes inspiration from the Inception models. It separates 36 convolution layers into 14 modules. In the output layer, it uses logistic regression for classification. Xception and Inception models have the same number of parameters showcasing that the increase in performance obtained from Xception is due to more efficient use of model parameters. [5]

5.5 Comparing Models

In this section, we will compare the performance of Model 7 against the industry-standard architectures. Table 12 shows the performance with testing data, and Table 13 is against unseen data.

	Accuracy	**Precision**	**F1-Score**	**Specificity**	**Sensitivity**
VGG16	95.09%	96.41%	0.9549	95.71%	94.59%
VGG19	85.52%	89.02%	0.8644	87.38%	84.00%
ResNet	81.44%	80.56%	0.8378	74.37%	87.26%
Xception	98.94%	98.96%	0.9903	98.61%	99.21%
Model 7	99.69%	99.19%	0.9928	99.78%	99.37%

Table 12: Industry-standard Models Results With Test Data

We have performed a Two-sample Student's t-test on Xception and Model 7, our best performing model.

	Accuracy	Precision	F1-Score	Specificity	Sensitivity
VGG16	60.00%	61.29%	0.6756	40.00%	76.00%
VGG19	62.22%	63.33%	0.6909	45.00%	76.00%
ResNet	64.44%	84.62%	0.5790	65.00%	44.00%
Xception	64.44%	69.57%	0.6666	65.00%	64.00%
Model 7	75.56%	76.92%	0.7843	70.00%	80.00%

Table 13: Industry-standard Models Results With Unseen Data

- **H_0:** Model 7 and Xception's difference in accuracy IS NOT statistically significant.
- **H_1:** Model 7 and Xception's difference in accuracy IS statistically significant.

Assuming a confidence interval of 95%, the p-value calculated is $p < 0.000$. As the p-value calculated is lower than the threshold value of 0.05, we have enough evidence to reject the null hypothesis and state that the difference in accuracy between Model 7 and Xception IS statistically significant.

Having said this, it was likely that this would happen. These industry-standard models were built to find complex features and classify images into 1000 different classes. Therefore, when applied to our project, they may be looking for really complex shapes in the CT scans, which only hurts their performance. The difference in performance could be because Model 7 is built to find simpler shapes or features like calculi in a CT scan.

6 Discussion

This section covers the ethical aspects of developing our model and determines whether it is ready to be used as a medical application in a real-world scenario.

6.1 ALTAI Ethical Assessment

ALTAI stands for Assessment List for Trustworthy Artificial Intelligence. It is a tool used to evaluate whether an AI system protects the fundamental rights of human beings. It has seven requirements, each raising questions that provoke

thoughtful reflection while developing and maintaining a Trustworthy AI system. Fulfilling the criteria marked in the ALTAI is critical in developing XAI. [7]

6.1.1 Human Agency

Human agency and oversight require AI systems to support the idea of human autonomy. That is, the AI system should help the user's agency. The system should also defend its fundamental rights whilst being supported by human oversight. We will focus on human oversight for our project as human agency and autonomy do not apply.

Human oversight helps assess whether the system should have specific human measures to avoid any wrong decisions by the AI. In our case, it would be necessary for the system to be overseen by a Human-in-Command. This supervisor is to manage the overall activity of the system and overrule any decision made by the AI system if they deem it to be incorrect. For example, when our model returns a prediction, it returns two values: the healthy class's probability and the unhealthy class's probability. If the prediction returned for either class is near 0.5, that is, close to chance, the human-in-command could step in and assess the decision made by the AI system. This governor should be a professional in the medical field with the ability to identify renal calculi from CT scans.

6.1.2 Technical Robustness

Technical robustness and safety are critical requirements to achieve trustworthy artificial intelligence. The AI system should be reliable and have a fall-back system, either human or AI.

The objective of an AI system is to learn features from training data that it can later apply to unseen data and produce a reliable prediction. As mentioned earlier, we have used specific preprocessing techniques to ensure that the data is as representative as possible of real-world data. Of course, there are certain factors that we could not reproduce for this project, like, artefacts in CT scans, images produced from many different models, and the radiation dose used to achieve the scans. However, there are some tests that we could perform to improve the

model's accuracy. For example, we could generate heatmaps of the features that the model extracts from the images to assess whether they are correct. It could be looking for too complex characteristics for the problem at hand, harming the model's performance. This is an example of how we could improve the model in the future.

Now, we will discuss the consequences of the model predicting a false negative. Of course, in this case, a false negative is the worst outcome we could get for a prediction followed by a false positive. A false negative means that the model has incorrectly predicted that the subject does not suffer from renal calculi. Therefore, the subject may continue with their day-to-day life without considering that they have renal calculi. So, they may not take any medication to treat the problem allowing the calculus to grow. This would lead to a more complex and painful treatment than if the system had correctly predicted the CT scan of the subject as unhealthy.

6.1.3 Privacy And Data Governance

AI systems should be built to prevent any harm to the subjects' privacy. Therefore, our model should respect each person's right to physical and mental integrity. This can be achieved through data governance. The only personal data used in our project are CT scans. However, steps have been taken to ensure that this data is anonymous. For example, the patient's information and the images' metadata were removed before the model saw any of this data. In future cases, the same process would be applied to any new data used by the model, either for training or predicting purposes.

6.1.4 Transparency

Transparency is a crucial requirement to be fulfilled to develop Trustworthy AI. At the moment, our model is not explainable. This feature is yet to be implemented. We could include explainability in our model by using heat maps that showcase what features or lack thereof are responsible for the prediction scores outputted by the model. Therefore, in a hypothetical final system, we show the user the classi-

fication, whether renal calculi were detected or not, the confidence or score of this classification and, finally, an image of the CT scan used as the input highlighting the features or characteristics that have caused that decision.

6.2 Is Model 7 Ready To Be A Medical Application?

No, it is not. In general, AI systems need to be very accurate. However, a model like the ones we have built needs to be highly accurate and reliable in the medical world. Model 7 may have achieved an accuracy of 99.69% during testing, but when tested with completely unseen data, its accuracy dropped to 75.56%. Furthermore, we do not know what would happen to Model 7's accuracy if we tested against more unseen data. Would its performance increase or decrease? This performance and uncertainty do not allow us to consider it as a ready medical application. The risks of misdiagnosis are too high.

Other threats to validity exist. For example, there is not enough data nor variety in the dataset used to train the model. Most of the data were likely captured on the same machine model making all the images very similar despite involving different patients. In future, we would need more data captured on different machine makes and models, diverse ranges of contrast and data from other countries.

The final model needs to be explainable. We should be able to identify why the model classifies an image as healthy or unhealthy. This can be achieved by generating heatmaps of the features extracted by the model allowing us to visualise what characteristics the model thinks are most important for classification.

Finally, throughout the project, we have realised that CT scans are not the preferred method for diagnosing renal calculi. Computed tomography exposes the patient to large amounts of radiation throughout the whole body. Most medical professionals opt for ultrasound over a CT scan because it is equally effective without exposing the patient to radiation. Therefore, perhaps our choice of input data may not be the most common for renal calculi detection. Having said that, CT scans are used when the calculus may not be visualised correctly using an ultrasound. In short, our model may not be built for the most appropriate diagnostic tool, but if it were reliable, it would still be helpful.

7 Conclusion

7.1 Summary

We began with four models and performed an iterative process to make these models more robust to real-world data. Our initial models could accurately predict whether a CT scan had renal calculi or not. However, it was only accurate on transverse cut CT scans. Therefore, we retrained and built more models that could accurately predict using transverse and coronal cut CT scans.

Next, we tested our best performing model on unseen data collected from the Internet and observed a significant drop in accuracy. So, we analysed the incorrectly classified cases of the unseen data. Based on our findings, we applied specific preprocessing techniques and modified the training of the models to achieve higher accuracy.

Finally, we tested and compared our best performing model, Model 7, with VGG16, VGG19, ResNet and Xception. Using a Student's t-test, we concluded that our model outperformed the industry-standard models in this particular problem.

7.2 Future Improvements

Some changes could increase the model's performance with the previously mentioned enhancements, like introducing heatmaps and improving the dataset.

Firstly, we discovered too late that we did not normalise the image data before feeding it to the model. We trained the model with pixels whose value could be [0, 255] instead of [0, 1]. Had we normalised the data, the training would have been quicker and the models more efficient.

Next, in the latter stages of the project, we discovered a preprocessing technique that may have significantly improved the performance of the models, gamma correction. This technique is a nonlinear operation used to encode and decode luminance in an image. Gamma correction was used in the paper [4], and it is used to combat the different intensities in which images are captured or displayed depending on the device's gamma values. Therefore, gamma correction is used

to visually enhance the images, particularly when those images are captured on different devices or machines. It would be interesting to see the difference in performance that applying gamma correction during preprocessing would have on the performance of the models.

References

[1] Prema T Akkasaligar, Sunanda Biradar and Veena Kumbar. 'Kidney stone detection in computed tomography images'. In: *2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*. IEEE. 2017, pp. 353–356.

[2] Zohair Al-Ameen et al. 'An innovative technique for contrast enhancement of computed tomography images using normalized gamma-corrected contrast-limited adaptive histogram equalization'. In: *EURASIP Journal on Advances in Signal Processing* 2015.1 (2015), pp. 1–12.

[3] Syed Muhammad Anwar et al. 'Medical image analysis using convolutional neural networks: a review'. In: *Journal of medical systems* 42.11 (2018), pp. 1–13.

[4] Priyanka Chak et al. 'Neural network and svm based kidney stone based medical image classification'. In: *International Conference on Computer Vision and Image Processing*. Springer. 2019, pp. 158–173.

[5] François Chollet. 'Xception: Deep learning with depthwise separable convolutions'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1251–1258.

[6] Kaiming He et al. 'Deep residual learning for image recognition'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[7] AI HLEG. *The Assessment List for Trustworthy Artificial Intelligence (ALTAI) for self-assessment, European Commission, Brussels*. 2020.

[8] HSE. *Review of the Clinical Radiology Medical Workforce in Ireland*. `https://www.hse.ie/eng/staff/leadership-education-development/met/plan/specialty-specific-reviews/clinical-radiology-chapter-for-web-2017.pdf`. 2017.

[9] MD NAZMUL ISLAM. *Kaggle CT KIDNEY DATASET: Normal-Cyst-Tumor and Stone*. `https://www.kaggle.com/datasets/nazmul0087/ct-kidney-dataset-normal-cyst-tumor-and-stone`. 2021.

[10] Gary Marcus. 'Deep learning: A critical appraisal'. In: *arXiv preprint arXiv:1801.00631* (2018).

[11] Jürgen Schmidhuber. 'Deep learning in neural networks: An overview'. In: *Neural networks* 61 (2015), pp. 85–117.

[12] Karen Simonyan and Andrew Zisserman. 'Very deep convolutional networks for large-scale image recognition'. In: *arXiv preprint arXiv:1409.1556* (2014).

[13] Jyoti Verma et al. 'Analysis and identification of kidney stone using Kth nearest neighbour (KNN) and support vector machine (SVM) classification techniques'. In: *Pattern Recognition and Image Analysis* 27.3 (2017), pp. 574–580.