EN.520.612 Machine Learning for Signal Processing
Course Project

Cervical Spine Fracture Detection with Deep Learning

Progress Report

Team Members: Ruxiao Duan, Qihua Gong, Tingying Lu, Weichen Qi, Yifan Zhou.

TABLE OF CONTENTS

List of Figures …………………………………………………………………………. 4
Abstract ……………………………………………………………………………... 5

1 Introduction ………………………………………………………………………. 6
1.1 Background ………………………………………………………………………. 6
1.2 Project Objectives ……………………………………………………………….... 6
1.3 Report Outline …………………………………………………………………… 7

2 Related Work ……………………………………………………………………... 7
2.1 Medical Image Analysis …………………………………………………………. 7
2.1.1 MRF and CRF …………………………………………………………………. 7
2.1.2 Supervised Learning Techniques ……………………………………………… 8
2.1.3 Deep KNN for Medical Image Classification ………………………………. 8
2.2 Deep Learning for Computer Version …………………………………………… 9
2.2.1 Neural Network ……………………………………………………………….... 9
2.2.1.1 Fully Connected Neural Network ……………………………………………. 9
2.2.1.2 Activation Function ……………………………………..……………..…… 10
2.2.1.3 Multi-class and Multi-label Classification ……………………………….... 10
2.2.2 Convolutional Neural Network ……………………………………………….... 10
2.2.2.1 CNN Architecture ……………………………………………………………. 10
2.2.2.1.1 Convolutional Layer …………………………………………………….. 10
2.2.2.1.2 Pooling Layer ………………………………………………………….. 11
2.2.2.2 CNN Models ………………………………………………………………. 11
2.2.2.2.1 AlexNet ……………………………………………………………….. 11
2.2.2.2.2 VGG ………………………………………………………………….... 11
2.2.2.2.3 ResNet ……………………………………………………………….... 12
2.2.2.2.4 EfficientNet …………………………………………………………….... 13
2.2.3 Model Training and Optimization ……………………………...………………… 13
2.2.3.1 Model Training with Backpropagation ……………………………………. 13
2.2.3.2 Model Optimization ……………………………………………………… 13
2.2.3.3 Learning Rate Scheduling ……………………………………………………… 14
2.2.4 Transfer Learning ………………………………………………………………. 14
2.2.5 Multimodal Learning ………………………………………………………….... 14
2.2.6 Multi-task Learning ………………………………………………………………. 14

3 Methodology ……………………………………………………………………………. 14
3.1 Dataset …………………………………………………………………………… 15
3.1.1 Medical Images of Cervical Spine ……………………………………………….... 15
3.1.2 Fractures of Each Patient ………………………………………………………. 15
3.1.3 Segmented Images with Vertebrae Labels …………………………………….. 16
3.1.4 Fracture Bounding Boxes ……………………………………………………… 16
3.2 Evaluation Metrics …………………………………………………………….. 17
3.3 Project Workflow ……………………………………………………………….... 17
3.3.1 Data Preprocessing …………………………………………………………….. 18

3.3.1.1 Image Augmentation ………………………………………………………………… 18
3.3.1.2 Image Masking and Segmentation ………………………………………………... 19
3.3.2 Vertebrae Detection ………………………………………………………………. 20
3.3.2.1 Vertebrae Detection with Images Only …………………………………………… 20
3.3.2.2 Vertebrae Detection with Auxiliary Attributes …………………………………… 21
3.3.3 Fracture Detection ………………………………………………………………... 22
3.3.3.1 Fracture Detection with Images Only ……………………………………………... 22
3.3.3.2 Fracture Detection with Multimodal Learning ……………………………………. 23
3.3.3.3 Fracture Detection with Multi-task Learning ……………………………………... 25
3.3.4 Fracture Localization …………………………………………………………….. 26
3.3.5 Cervical Spine Visualization ……………………………………………………... 26

4. Current Experiments ………………………………………………………………. 27
4.1 Experiment Setup ……………………………………………………………………... 27
4.2 Experimental Results ………………………………………………………………….. 28

5. Discussion ……………………………………………………………………………. 29
5.1 Difficulties and Limitations …………………………………………………………… 29
5.2 Future Work …………………………………………………………………………... 30
5.2.1 Image Preprocessing ……………………………………………………………….. 30
5.2.2 Vertebrae Detection with Auxiliary Attributes ………………………………...…… 30
5.2.3 Fracture Detection …………………………………………………………………. 30
5.2.4 Fracture Detection with 3D CNN ………………………………………………… 31
5.2.5 Fracture Localization ……………………………………………………………... 31
5.2.6 Result Visualization ……………………………………………………………… 31

6. Conclusion …………………………………………………………………………... 32

References …………………………………………………………………………… 33

List of Figures

Fig. 1. Seven vertebrae on a human cervical spine 6
Fig. 2. Automated lymphocyte detection framework 8
Fig. 3. Deep KNN model architecture 9
Fig. 4. Model architecture of AlexNet 11
Fig. 5. Model architecture of VGG 12
Fig. 6. The residual block of ResNet 12
Fig. 7. Model architecture of EfficientNet 13
Fig. 8. CT scans of a cervical spine 15
Fig. 9. Number of studies with and without fracture on each vertebra and on a patient overall 16
Fig. 10. Slices with a pixel-level vertebrae annotation 16
Fig. 11. Slices with a bound box to locate the fracture 17
Fig. 12. The project workflow diagram 18
Fig. 13. Masked CT images 19
Fig. 14. A segmented image of a road 19
Fig. 15. Vertebrae detection model with images only 20
Fig. 16. Multimodal learning for vertebrae detection 22
Fig. 17. Fracture detection model with images only 22
Fig. 18. Multimodal learning for fracture detection 24
Fig. 19. Multi-task learning for fracture detection 25
Fig. 20. Fast R-CNN for fracture localization 26
Fig. 21. A 3D view of a cervical spine 27
Fig. 22. Vertebrae detection test accuracy and training time of 14 backbone models 28
Fig. 23. Vertebrae detection cross-validation accuracy and average training time of 5 different backbone models 29
Fig. 24. Medical image classification with 3D CNN 31

Abstract

Manual analysis of medical images for disease diagnosis is an arduous task with relatively low efficiency and accuracy. Machine learning techniques promote automation in the process of disease detection by employing various image analysis models. Convolutional neural network is a class of deep learning models designed for vision-related problems and has proven to be exceptionally effective in medical image analysis. This project aims to detect the fractures on human cervical spine to a vertebrae level and identify their exact locations in the 3D coordinate system with deep learning models.

This project consists of multiple phases. Computed tomography images of two thousand patients are firstly collected for model training and evaluation. A vertebrae detection model is then constructed to identify the vertebrae that are visible from each image, and a fracture detection model is built to recognize the fractures in images. The positions of fractures are revealed by a fracture localization model, and finally visualized in the 3D space.

Up to the current stage, the task of vertebrae detection is accomplished. 14 different types of backbone convolutional neural networks are evaluated and compared to each other with respect to their vertebrae prediction accuracy as well as training time, among which InceptionV3 demonstrates the best performance with a test accuracy of approximately 95%.

1 Introduction

This chapter gives a brief overview of the entire project. Some background information is introduced in section 1.1, and the project objectives are summarized in section 1.2. Section 1.3 gives an outline of the remaining chapters.

1.1 Background

Millions of spine fractures occur annually in the US, resulting in numerous spinal cord injuries. Among those spine fractures, the most common injury spot is around the cervical spine, which is the most crucial part of the human spine that controls the signal transition from brain to body.

Some frequently employed techniques to detect spine fractures include computed tomography (CT) scans and magnetic resonance imaging (MRI). However, a set of medical images from a single patient could contain hundreds of pictures, while some tiny fractures may only appear on a few of them and are sometimes not easily recognizable. If doctors select only a subset of images to scrutinize, it is likely that they can miss some fracture and make a mistake in their diagnosis; if the entire collection of images is to be manually examined, the task might become too laborious even for specialists. Hence, automation in the diagnosis process is of great significance, and machine learning is a solution.

An appropriate machine learning model can notably improve the efficiency and accuracy of diagnosis of cervical spine fractures, and this field has been investigated for years. Convolutional neural network (CNN), a class of deep learning models for computer vision, has demonstrated great potential in medical image analysis in previous research [1, 2], thanks to its capability to accurately extract image features and perform analysis without much human intervention [3].

1.2 Project Objectives

The human cervical spine consists of seven vertebrae, which are C1, C2, to C7 from top to bottom (Fig. 1). This project aims to leverage deep learning techniques to identify fractures on the human cervical spine, at both the level of a single vertebra and the entire patient. It means that the model constructed should be able to not only detect the fracture for a patient overall, but also predict which vertebrae the fracture is located at. Furthermore, a fracture localization model will also be constructed to determine the exact fracture position in the 3D coordinate system.

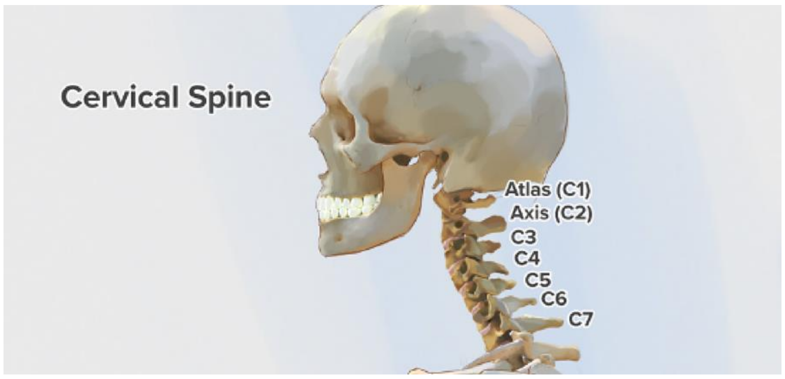


Fig. 1. Seven vertebrae on a human cervical spine.

1.3 Report Outline

This report is separated into six chapters. Chapter 2 delineates the literature review on the topics of medical image analysis as well as deep learning for computer vision. Chapter 3 expounds on the methodology applied in the project to realize the objectives. The current experimental results are illustrated in chapter 4, and some limitations and future work are encapsulated in chapter 5. Finally, a summary of the report content is provided in chapter 6.

2 Related Work

This chapter presents some background information of the project and shows some previous studies relevant to medical image analysis and deep learning.

2.1 Medical Image Analysis

In this chapter, some mature techniques related to medical image analysis are illustrated, and their application scenarios as well as their performance are analyzed. These technologies also serve as the basis and possible directions of improvement for this project.

2.1.1 MRF and CRF

The conditional probability model is a classic model in machine learning, with also excellent performance in medical image analysis. Pathological image analysis is an important procedure in the clinical diagnosis of various diseases. To make the diagnosis more accurate and objective, increasingly more intelligent systems are invented, among which random field models play an integral role in improving the survey performance.

A comprehensive overview of pathological image analysis based on two popular random field models, Markov random fields (MRF) and conditional random fields (CRF), has been presented in a previous study [4]. Some background of the two random fields and pathological images as well as a summary of the basic knowledge of MRFs and CRFs from modeling to optimization have been provided [4]. The authors also give a comprehensive review of recent studies on MRF and CRF for pathological image analysis. Method transfer between CAD domains is also discussed.

In medical image analysis, accurate image segmentation is often required, which has been addressed by previous research in this field. In one of the studies [5], a four-step image segmentation process is employed to classify four categories of teratoma tissues (Fig. 2). In the first step, the segmentation is formulated in the Bayesian framework. Secondly, a set of hidden real-valued random fields designing a given segmentation probability is introduced. In order to produce smooth fields, a Gaussian MRF (GMRF) prior is assigned to reformulate the original segmentation problem. Thirdly, the form of total variation of isotropic vectors is adopted. And finally, the convex optimization problem is solved for MAP inference of hidden fields.

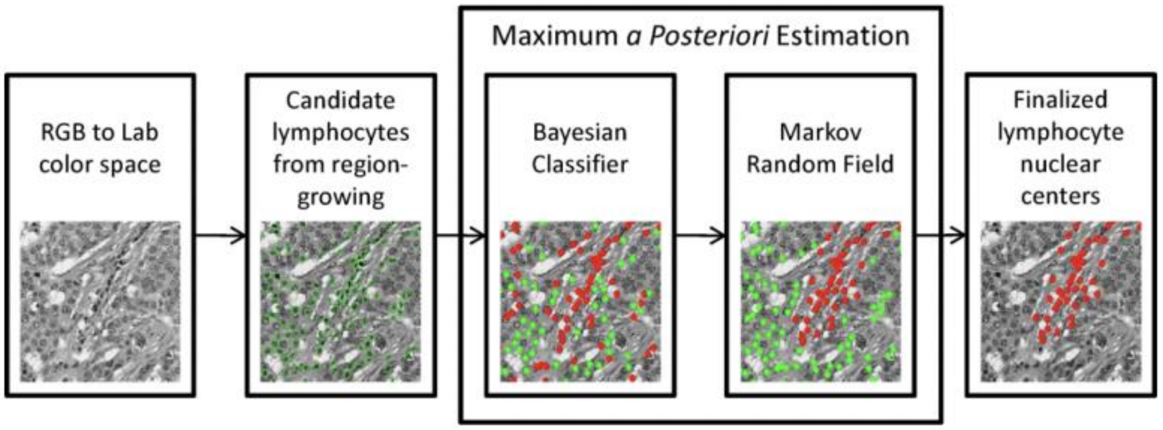

Maximum *a Posteriori* Estimation
RGB to Lab color space
Candidate lymphocytes from region-growing
Bayesian Classifier
Markov Random Field
Finalized lymphocyte nuclear centers

Fig. 2. Automated lymphocyte detection framework [6].

2.1.2 Supervised Learning Techniques

Supervised quantification approaches can not only assist in diagnosis, but also be used to predict the onset or progression of future diseases. Models can be trained on data from longitudinal studies in which the disease status years after the acquisition of the baseline image is known. For example, Erasmus MC [7] showed that hippocampal shape classification in a healthy elderly population is predictive of dementia symptoms up to ten years later. Van Engelen [8] used multivariate sparse Cox regression to take the time to the event into account in the model and found that changes in plaque texture and volume in ultrasound images of the carotid artery could predict future vascular events better than traditional risk factors could.

The main obstacle that currently impedes a wider use of machine learning techniques in medical image analysis is the lack of representative training data. While supervised learning techniques have shown much promise in relatively constrained experiments with standardized imaging protocols, their performance may quickly deteriorate on new images that are acquired under slightly different conditions. These techniques operate under the assumption that both training and test datasets are random samples drawn from the same distribution. Studies such as [9] attempt to cope with these issues by applying transfer learning or domain adaptation techniques. Two approaches mentioned in [9] to make training and test distributions more similar include weighting and feature space transformation techniques. In their study, it was observed that a weight-based transfer learning approach could significantly improve the classification accuracy of MRI segmentation problems when there are insufficient labeled target samples.

2.1.3 Deep KNN for Medical Image Classification

Medical diagnosis usually requires a large-scale dataset, but in general, only some diseases have a sufficient amount of data. The scarce data available restrict the application of most neural networks in medical diagnosis. For small datasets, KNN is a good method. However, in traditional KNN, the classifier and feature extractor are separated, making KNN incapable of extracting targeted features. An improved version of KNN called deep KNN is introduced in [10], which is a combination of a feature extractor and a classifier (Fig. 3).

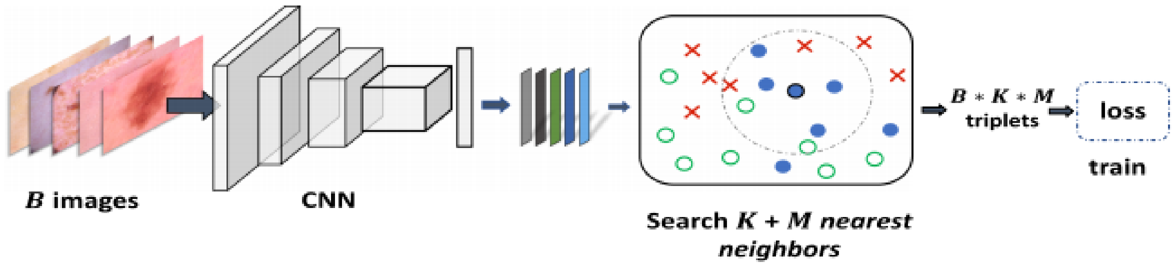


Fig. 3. Deep KNN model architecture.

Similar to DNNs, training can be accelerated with stochastic gradient descent (SGD). But each iteration needs to re-find KNN and MNN of all samples, which is too expensive. The iterator is updated every time a mini-batch of samples is trained, and for each sample, KNN and MNN are only identified in that mini-batch. The traditional idea of KNN is to make the distance between any two points in the same category smaller than the distance between any two points in distinct groups. The idea of Deep KNN is that the distance between any point and its K nearest neighbors of the same classification is less than the distance between the point and its M nearest neighbors of different classifications. The purpose of traditional KNN is to aggregate the distribution of various types of data, and the purpose of Deep KNN is to separate the distributions of different classes, so that the data and its K nearest neighbors belong to the same class, without requiring the data of each class to be tightly clustered. In addition, Deep KNN is also much easier to train.

2.2 Deep Learning for Computer Version

This section focuses on deep learning algorithms for computer vision. Section 2.2.1 and 2.2.2 introduce neural networks and convolutional neural networks respectively, and section 2.2.3 depicts the model training and optimization approaches. Three deep learning techniques, transfer learning, multimodal learning, and multi-task learning, are then delineated in sections 2.2.4, 2.2.5, and 2.2.6.

2.2.1 Neural Network

Neural networks are a class of models that attempts to identify the underlying relationships between inputs and outputs and to find patterns in data. This section includes some basic concepts in neural networks including fully connected neural networks, activation function, and the usage of neural networks for multi-class and multi-label classification.

2.2.1.1 Fully Connected Neural Network

Fully connected neural networks consist of fully connected layers that link each neuron in a layer to each neuron in the adjacent layer. An advantage of fully connected networks is that they are structurally agnostic, and no special assumptions are required on the model input.

2.2.1.2 Activation Function

An activation function can introduce nonlinearity in the output of the neuron. A neural network without an activation function is essentially a linear regression model. The activation function transforms the input nonlinearly, enabling it to learn more complex functions.

2.2.1.3 Multi-class and Multi-label Classification

In general, the input of a neural network is a vector, while the output is also a vector. For classification tasks, the output is typically a probability distribution of the target objects, with each entry specifying the probability of the input belonging to that corresponding category.

In machine learning, multi-class classification is the classification of instances into one of three or more classes. Multi-class classification assumes that each sample is assigned only a single label. Multi-label classification assigns zero or more non-exclusive class labels to each of the model inputs.

The neural network model can be configured for either multi-class or multi-label classification tasks, by setting the activation function before the final output layer. If the multi-class classification is to be performed, it is expected that the output probability vector should contain numbers between 0 and 1 with a sum of 1, specifying the probability of each mutually exclusive class. In this case, sigmoid is a perfect choice of activation function. In multi-label classification, each output entry is between 0 and 1, without the constraint of summing up to 1, thus softmax is the best option in this scenario.

2.2.2 Convolutional Neural Network

In deep learning, convolutional neural network (CNN) is a class of artificial neural network that is often employed to analyze images, with applications in image and video recognition, object detection, image segmentation, etc. This section introduces the basic model architecture of CNN, together with some well-known CNN models.

2.2.2.1 CNN Architecture

The input of CNN models is usually a 2D or 3D matrix representing pixels of some image, from which the features can be extracted by the model to perform various tasks such as classification. The major components of CNN models are the convolution layer and the pooling layer, which are described respectively in the following two subsections.

2.2.2.1.1 Convolutional Layer

The convolutional layer contains a set of filters (or kernels) whose parameters need to be learned. Each filter is convolved with the input to derive a feature map consisting of neurons. The output of a convolutional layer is derived by superimposing the activation maps of all filters along the depth dimension. The local connectivity of the convolutional layer enables the network to learn filters that maximally respond to local regions of the input, thus exploiting the local spatial correlation of the input. Moreover, since the activation map is obtained by the convolution between the filter and the input, the filter parameters at all local locations are shared. Weight sharing reduces the number of parameters and improves the expression efficiency, learning efficiency, and generalization effect.

2.2.2.1.2 Pooling Layer

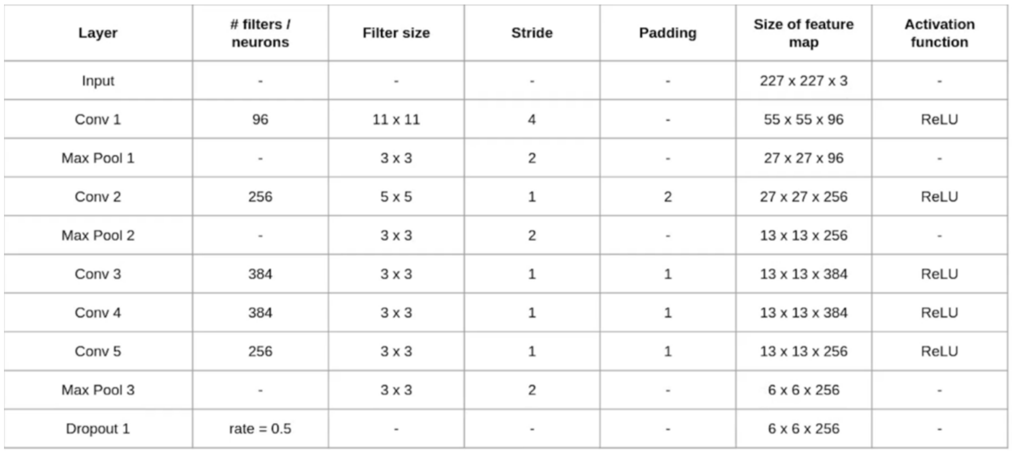
Like the convolutional layer, the pooling operator consists of a fixed-shape window that slides over all regions in the input based on stride length, computing a single output for each position traversed by the fixed-shape window, sometimes called the pooling window. The pooling layer contains no parameter and is deterministic. Typically, the maximum or average value of the elements in the pool window is computed and forwarded to the next layer. These operations are called maximum pooling and average pooling respectively. By downsampling the input, pooling layers reduce the number of parameters of the model while keeping the essential information in the feature map. The efficiency of model training is thus improved without a severe deterioration of model performance.

2.2.2.2 CNN Models

This section presents four off-the-shelf CNN models, which are AlexNet, VGG, ResNet, and EfficientNet.

2.2.2.2.1 AlexNet

AlexNet [11] is a CNN containing eight layers, as illustrated in Fig. 4. With ReLU as the activation function, the training speed is increased by nearly six times. The dropout layer prevents the model from overfitting.

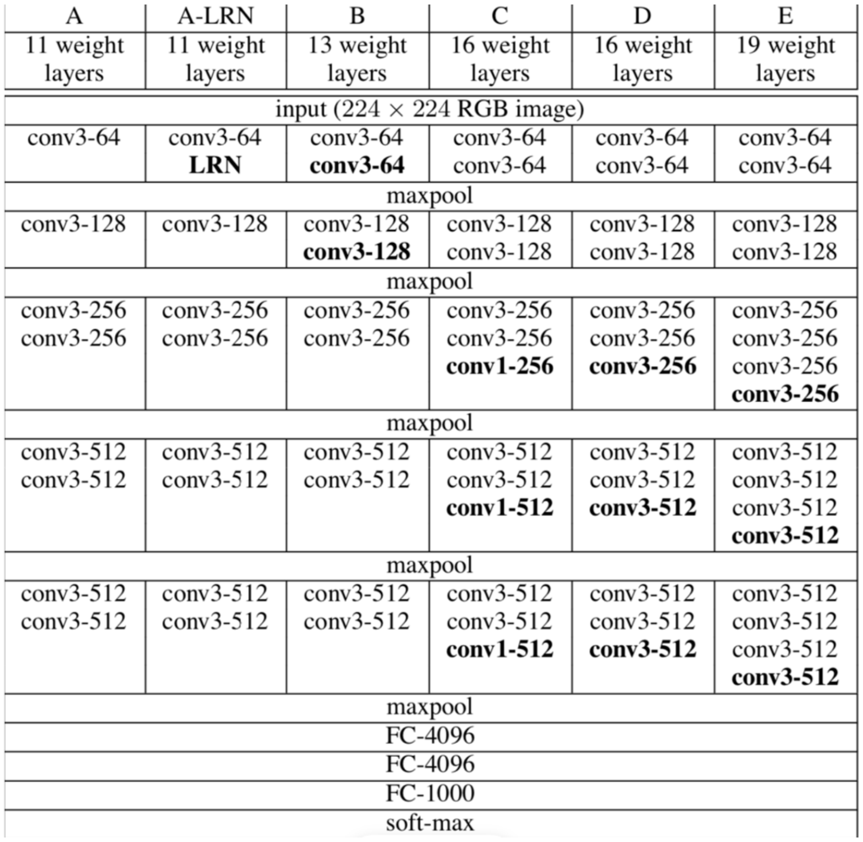


Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

Fig. 4. Model architecture of AlexNet [11].

2.2.2.2.2 VGG

VGG [12] applies 3 × 3 small filters in the convolutional layers, with a fixed stride of 1 (Fig. 5). VGG attempts to increase the depth of the model to 19 and has three fully connected layers to map the image feature vector to the output probability vector. Compared with AlexNet, VGG uses a very small receptive field, and the decision function is more discriminative, because of the more ReLU units adopted in the model.



A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Fig. 5. Model architecture of VGG [12].

2.2.2.2.3 ResNet

Increasing the depth of CNNs often leads to the problem of vanishing gradient and can cause the model performance to degrade. In another class of CNNs named residual neural network (ResNet) [13], a residual block (Fig. 6) is invented to alleviate this issue by a skip connection, and model depth is increased to 152 without a noticeable decrease in prediction accuracy.

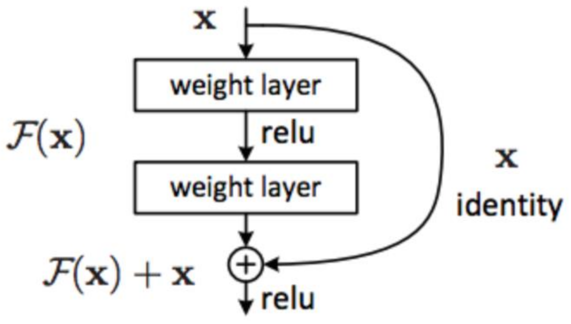
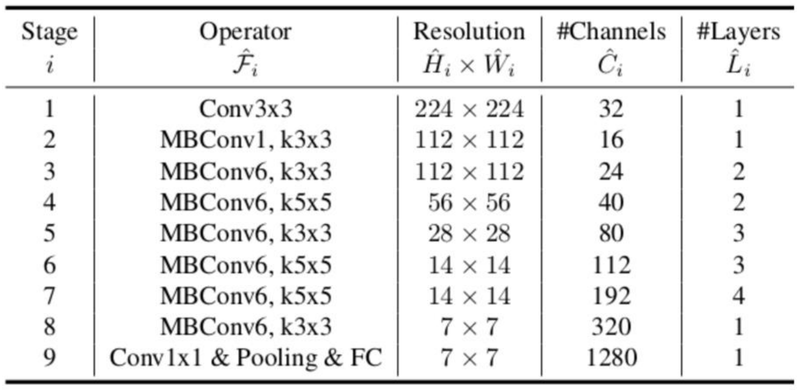


Fig. 6. The residual block of ResNet [13].

2.2.2.2.4 EfficientNet

EfficientNet [14] employs a technique called compound coefficients to scale up the model in a simple and efficient way (Fig. 7). Instead of randomly scaling width, depth, or resolution, compound scaling scales each dimension in a uniform manner with a fixed set of scaling coefficients. Models with seven different scaling dimensions are developed with state-of-the-art prediction accuracy and training efficiency at that time.



Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Fig. 7. Model architecture of EfficientNet [14].

2.2.3 Model Training and Optimization

This section expounds on the deep learning model training and optimization approaches, as well as the technique of learning rate scheduling.

2.2.3.1 Model Training with Backpropagation

Backpropagation is a crucial part of neural network training. It is a practice of fine-tuning the weights of neural networks based on minimizing the errors obtained in the previous stage. By calculating the partial derivate of the error with respect to the weight, the model updates its weights from layers close to the output to those close to the input. Finally, the weights can be adjusted to form an accurate mapping function from the input to the corresponding output.

2.2.3.2 Model Optimization

To optimize the model, the loss has to be minimized. Some frequently adopted optimization methods include gradient descent, stochastic gradient descent, and Adam.

In mathematics, gradient descent [15] is an iterative optimization algorithm that uses the first-order derivative to find the local minima of the objective function. The idea is to repeat the steps at the current point along the opposite direction of the gradient, since this is the direction of the fastest descent.

Stochastic gradient descent (SGD) [15] is an iterative method to optimize the objective function. It can be viewed as a stochastic approximation of gradient descent optimization since it replaces the actual gradient with an estimate of the gradient. Especially in high-dimensional optimization problems, SGD reduces the high computational burden and achieves faster iterations with lower convergence rates.

Adam [16] uses the average of the second moment of the gradient to minimize the loss. The exponential moving average as well as the square of the gradient are calculated, and the decay rate of the moving average is controlled by specified parameters.

2.2.3.3 Learning Rate Scheduling

A learning rate scheduler attempts to adjust the learning rate in the training process by reducing the learning rate according to a predefined schedule. Common learning rate plans include time-based decay, step decay, and exponential decay.

2.2.4 Transfer Learning

Transfer learning [17] refers to the use of previously acquired knowledge and skills in a new learning or problem-solving situation. The models pre-trained on a large-scale dataset can be reused on some similar but smaller datasets to solve new problems. It is a popular technique in deep learning, as it enables deep neural networks to be trained by a relatively small amount of data.

2.2.5 Multimodal Learning

Multimodal learning [18] accepts different types of model inputs, such as images and text. Multiple models are trained simultaneously to extract the embeddings of the inputs. While combining different models or types of information to improve performance may seem trivial, in practice, aggregating different levels of noise and conflict of models is challenging. Besides, models have different quantitative effects on the predicted output. The most common approach is to concatenate high-level embeddings from different inputs and then forward the combined features to some other models, such as a fully connected neural network.

2.2.6 Multi-task Learning

Common machine learning problem has only a single desired output and a single metric. However, the features extracted from the input may sometimes be used to handle other tasks, each of which with its own expected output and loss function. By forwarding the feature to multiple targets and minimizing the combined loss, a multi-task model that can accomplish more than one task at the same time can be obtained. This is far more efficient than training the models separately and can possibly result in a better performance in each individual task [19].

3 Methodology

This chapter expounds on the project methodology. Section 3.1 provides basic information about the dataset available currently. Section 3.2 describes the evaluation metrics adopted in this project for model evaluation purpose. The comprehensive workflow of the project is then elaborated in section 3.3.

3.1 Dataset

The medical image dataset consists of four parts, each of which will be introduced in this section.

3.1.1 Medical Images of Cervical Spine

The medical images applied in this project for model training and testing are retrieved from the public dataset created by Radiological Society of North America (RSNA), American Society of Neuroradiology (ASNR), and American Society of Spine Radiology (ASSR). The images are mostly X-ray images of CT scans, which are collected from 12 sites globally, including approximately 3000 CT studies.

2019 studies in the entire dataset are publicly available (Fig. 8), which contains a total of 711601 CT images. Each study corresponds to a patient (with a unique study instance ID), which includes about 200 to 600 slices of 2D CT scans from the top to the bottom of a patient's cervical spine, covering all seven vertebrae.

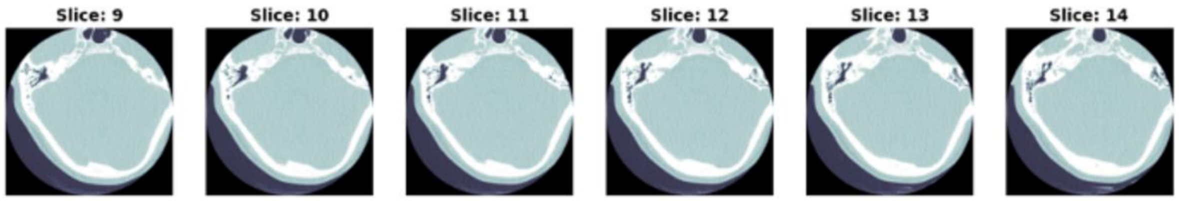


Fig. 8. CT scans of a cervical spine.

Each of the CT scan images is stored in Digital Imaging and Communications in Medicine (DICOM) format. A DICOM file contains not only image pixels, but also some metadata including the slice index, slice thickness, and the physical 3D coordinates of the slice position. Some of the metadata can provide useful insights into the prediction of vertebrae present in a slice, and this will be elaborated in section 3.3.2.2.

3.1.2 Fractures of Each Patient

For each study (patient), the fracture information on the cervical spine is provided in the dataset. Each study is given 8 binary labels, indicating whether the patient has a fracture in each of the 7 vertebrae or not, and has a fracture overall. So long as the patient has a fracture in any one of his vertebrae, the patient is regarded as fractured overall. The comparison between samples of fractured and non-fractured vertebrae as well as patients is demonstrated in Fig. 9.

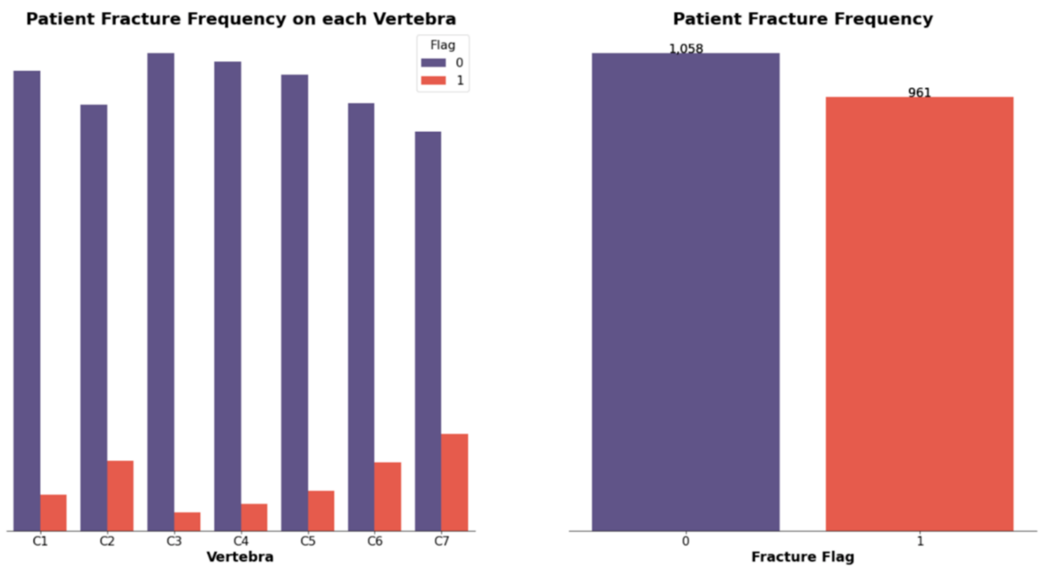


Fig. 9. Number of studies with and without fracture on each vertebra and on a patient overall.

3.1.3 Segmented Images with Vertebrae Labels

87 of the studies have the vertebrae labels given for each slice up to a pixel level (Fig. 10). Each pixel in the image is annotated by either 0, indicating not a vertebra, or 1 to 7, standing for the corresponding vertebra type. Using this information, it can be known what types of vertebrae, if there are any, are being shown by the slice. But to infer the vertebrae indices of all the remaining studies, a vertebrae detection model has to be constructed. This will be described in detail in section 3.3.2.

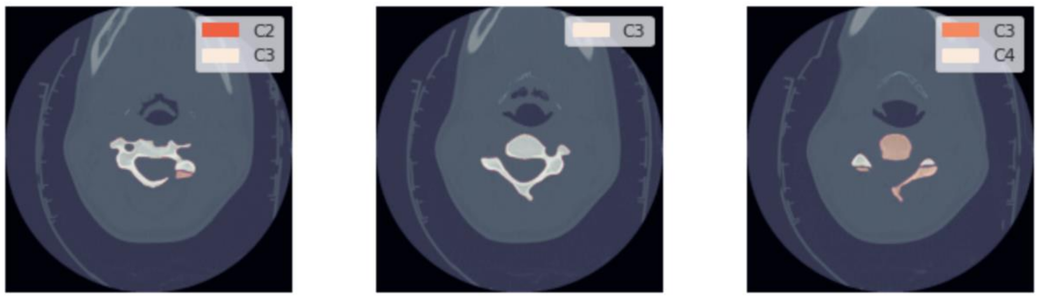


Fig. 10. Slices with a pixel-level vertebrae annotation.

3.1.4 Fracture Bounding Boxes

Bounding boxes denoting the position of fractures are drawn for the slices of 239 studies (Fig. 11). The fracture location information can be employed to build a fracture localization model, and this will be further explained in section 3.3.4.

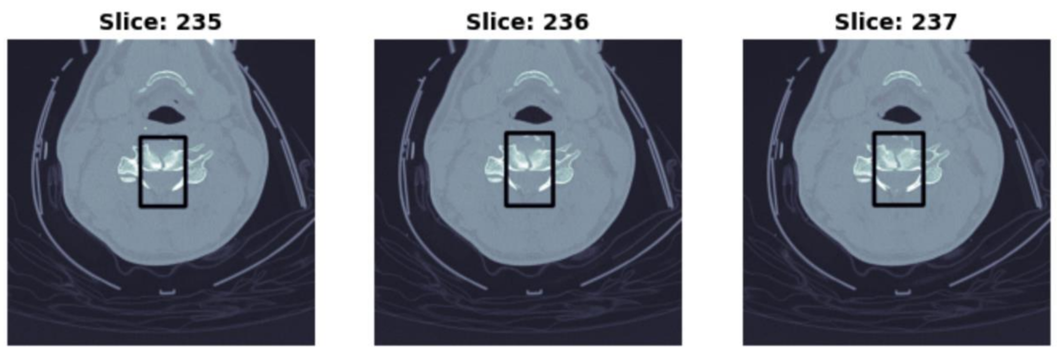


Fig. 11. Slices with a bound box to locate the fracture.

3.2 Evaluation Metrics

The major objective of this project is to detect the existence of fractures on human cervical spines, not only on a patient-level but also on the vertebra-level. Therefore, the two most essential evaluation metrics are the fracture prediction accuracies on the patient-level and the vertebra-level. That is to say, it is desirable to measure the proportion of accurately predicted patients with respect to all the patients, as well as the proportion of accurately predicted vertebrae with respect to all the vertebrae. Since each patient has 7 vertebrae on the cervical spine, the total number of test cases of vertebrae is 7 times that of the patients.

Since the total number of patients with cervical spine fractures is similar to that of patients without any fracture in the dataset, prediction accuracy can be regarded as an appropriate evaluation metric. However, when it comes to fracture detection on the vertebra-level, the problem of class imbalance occurs. A single vertebra individually has only a small proportion of patients having fractures, leading to highly imbalanced prediction classes in this particular task. Hence, f1-score might be a better evaluation metric than accuracy for fracture detection on the vertebra-level, and these two metrics will both be applied to judge the model performance.

Apart from the main task of fracture detection, this project also involves several subtasks which require another few metrics for model performance evaluation purpose. For vertebrae detection, the prediction accuracy of vertebrae indices is adopted. For fracture localization, the Intersection over Union (IoU) is chosen as the metric.

3.3 Project Workflow

The project is roughly divided into five phases (Fig. 12). In the first stage, CT scan images are transformed to improve the model performance in subsequent stages. The second stage focuses on the identification of the vertebrae that are present on each slice. In the third stage, fractures are detected on both the patient-level and the vertebra-level. The fourth stage attempts to detect the exact location of the fracture on a slice that has been identified as fractured. The final stage visualizes the human cervical spine in a 3D space and highlights the locations at which fractures are detected. This project is more concentrated on the fracture detection task, thus the second and the third phases are of the most importance.

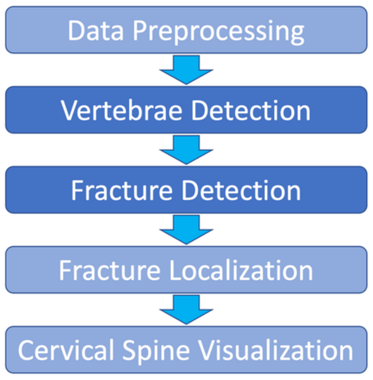


Fig. 12. The project workflow diagram.

Every stage of the project requires its own methodologies to accomplish the goal. The following five subsections elaborate on the detailed procedures adopted in each phase.

3.3.1 Data Preprocessing

CT scans of cervical spines are of great importance to this project, since most tasks are related to image processing and analysis. In particular, project stages 2, 3, and 4 all expect images as input to their models to extract features for further analysis. Considering the insufficiency of training images in some of the experiments and the possible noise that might exist on the slices, it is vital to consider techniques that help to alleviate these issues.

3.3.1.1 Image Augmentation

As investigated previously, among all the 2019 patients in the dataset, only 87 of them have vertebrae indices annotated on the slices, and only 239 of them have bounding boxes drawn to denote the location of fractures. The insufficient training image samples might become a potential problem for vertebrae detection and fracture localization (i.e., project stage 2 and stage 4). Therefore, it can be advantageous to augment the dataset to generate more training samples.

Image augmentation is a technique that can be adopted to produce extra training images based on the current ones. Random rotation, shifting, and flips can be applied to the existing images to create new ones with the same labels. By augmenting the training dataset, the models are more likely to learn informative features from images and tend to give more accurate predictions.

However, some issues of this method might impede its effectiveness. One of the concerns is the increased training time. If only the transformed images are forwarded to the models, leaving the original images out, the actual number of training images does not increase. If both the original and transformed sets of images are fed to the models, model accuracy might be improved but the training time also rises dramatically, which can make this technique too inefficient to carry out.

Another possible problem is that, since the CT scans in the dataset are well aligned, they might be sensitive to certain transformations. For instance, as can be observed from the example slices (Fig. 8), the position of the cervical spine is basically invariant in each slice, and the shape of the vertebrae is not perfectly symmetric. Blindly applying image shifting or flipping might induce negative effects, hence performing image data augmentation requires special care under this circumstance.

3.3.1.2 Image Masking and Segmentation

Due to possible noises in the images, it might be contributive to apply masking or segmentation on the images for noise reduction, which hopefully can improve the model performance in later stages.

Image masking attempts to identify the background part of the images and sets all the pixels in that part to zeros (Fig. 13). This method eliminates the background noises but requires a separate model for background identification.

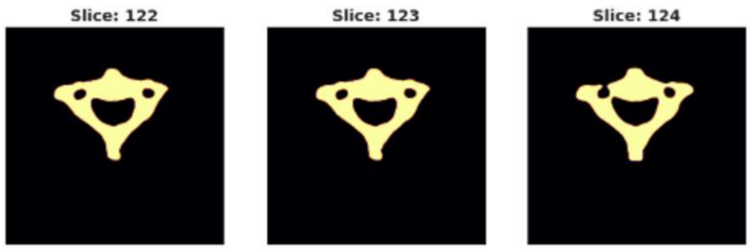


Fig. 13. Masked CT images.

Image segmentation attempts to separate each slice into several sections (Fig. 14), each denoting a specific type of object such as background and vertebrae, or even each individual type of vertebrae. Then, some other processes like averaging can be applied to each identified region to reduce noise.

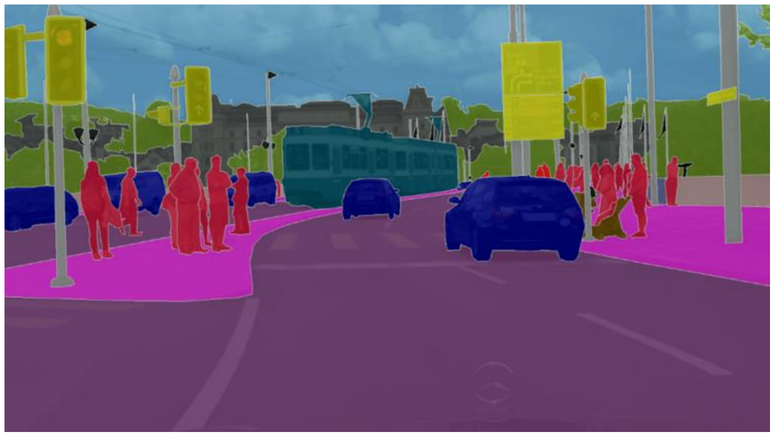


Fig. 14. A segmented image of a road.

Some relatively trivial methods for masking and segmentation include simple thresholding and K-means clustering. Since CT scan images are essentially grayscale images with only a single color channel, K-means clustering is more or less equivalent to finding K-1 splits to separate the pixel values on a real line. Advanced semantic segmentation models involve the usage of a well-annotated segmented dataset, which is available in this project, and also deep learning models such as SegNet, U-Net, DeepLab, Mask-RCNN, etc.

One of the potential problems in simply taking the average of each segmented region is information loss. The variation of pixel values in certain regions might play an important role in future prediction tasks, thus identifying a region and averaging its pixel values may not necessarily be a proper action in this case.

3.3.2 Vertebrae Detection

Each of the 2019 patients has the corresponding hundreds of image slices. However, most of them do not have vertebra index annotated for each slice. Since traditional 2D CNNs only take a single image as input, which means that each slice of a patient is treated independently by the model, even if the model successfully detects the fracture in some slice of a patient, it is hard for the model to tell which vertebra this fracture is located at without the knowing the vertebra index of that slice. Consequently, the vertebra-level fracture detection can hardly be conducted.

To get prepared for the fracture detection task in the next stage, an image dataset with vertebrae labels is needed. Fortunately, 87 patients have their CT scan images annotated with vertebrae indices, which can be used to train a vertebrae detection model that can be applied to infer the vertebrae probability tags of the slices of all the remaining patients, indicating the probability of some vertebra being in some slice.

Two approaches are proposed in the following subsections to predict the vertebrae in each slice.

3.3.2.1 Vertebrae Detection with Images Only

The rationale of the vertebrae detection model with only images as input is straightforward. What is available now is a set of image slices, each with its corresponding vertebrae labels. The model has the input of a 2D image and the output of a 7-dimensional vector, which denotes the probability of each vertebra being present on the input slice. Note that this vertebrae detection task is essentially a multi-label classification instead of a multi-class classification, since multiple vertebrae can be present simultaneously on the same slice. Therefore, each entry of the 7-dimensional output vector, which has to be a number between 0 and 1, independently corresponds to the probability of that vertebra without the constraint of summing up to one. Hence, a model can be designed as illustrated in Fig. 15.

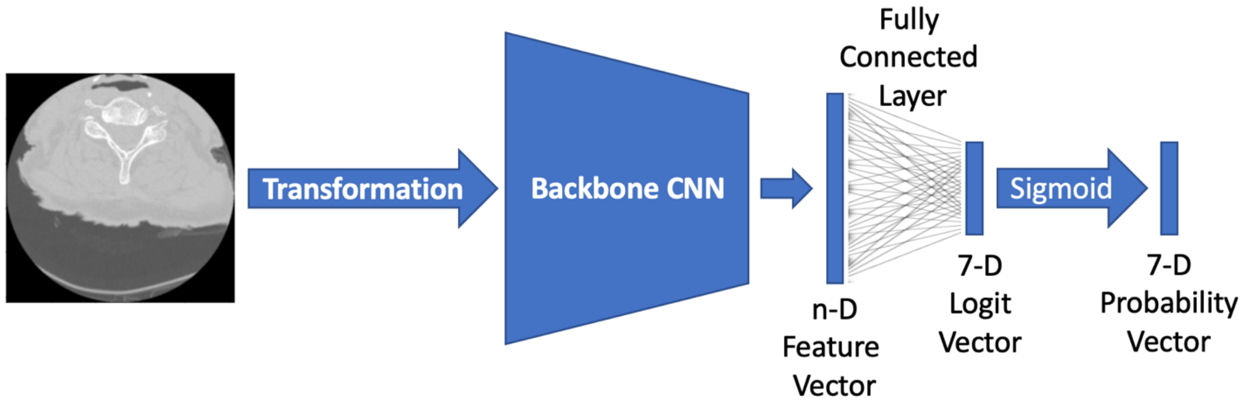


Fig. 15. Vertebrae detection model with images only.

First of all, the transformation has to be conducted on the input images before forwarding them to the CNN. The procedures of this step include pixel value rescaling, image resizing, image augmentation, and noise reduction.

After changing the images into a uniform format, they are ready to be fed to a backbone CNN model. This model serves as a feature extractor, which produces an n-dimensional feature vector of the input image. Employing the technique of transfer learning, off-the-shelf CNN models, whose weights are obtained from pre-training by some large-scale image dataset like ImageNet, can be applied here for the task of vertebrae detection. The options for the backbone CNNs include VGG, ResNet, MobileNet, DenseNet, EfficientNet, Vision Transformer, etc., and their performance remains to be evaluated by experiments. In section 3.3, a backbone CNN refers to any pre-trained CNN model that can be applied for feature extraction.

The resulting n-dimensional feature vector is then forwarded to a fully connected neural network, which generates a 7-dimensional vector with entry values being any real numbers. This vector is treated as the log odds of the presence of the seven vertebrae. By applying a sigmoid activation function of the log odds, the final 7-dimensional vertebrae probability vector with all values between 0 and 1 can be derived. In this model, the loss is defined as the binary cross entropy between the predicted probability vector and the true vertebrae label vector, which is the typical choice of loss function for multi-label classification problems.

Training the model by backpropagation using the image slices of those 87 patients, a model to predict the vertebrae probabilities of any image slice can be obtained. This model is then leveraged to infer the vertebrae in all the remaining images.

3.3.2.2 Vertebrae Detection with Auxiliary Attributes

Images in the dataset are stored in DICOM format, which contains some additional metadata apart from only the image pixel values. The available metadata includes the slice thickness, the slice index (which can be used to calculate the ratio of the current slice in that patient from top to bottom), and the 3D coordinates of the slice position.

Among these auxiliary attributes associated with each image, the slice ratio and the z-coordinate of the slice position are of the most importance to the inference of vertebrae that the current slice is showing. Since the CT images are scanned in order from top to bottom of a patient’s cervical spine, the slice ratio and the z-coordinate are perfect indicators of vertebrae indices. Therefore, a model that can take into consideration of not only the slice image but also those helpful supplementary attributes to make predictions on vertebrae indices is desired. The technique of multimodal learning can be leveraged to integrate these attributes into the model.

As demonstrated in Fig. 16, a separate neural network is incorporated into the original model to extract features from the attributes of the images. Now, the input to this model contains both an image and also the relevant attributes of that image such as the aforementioned ones. Hopefully, by employing these additional features, the model prediction accuracy can be further improved.

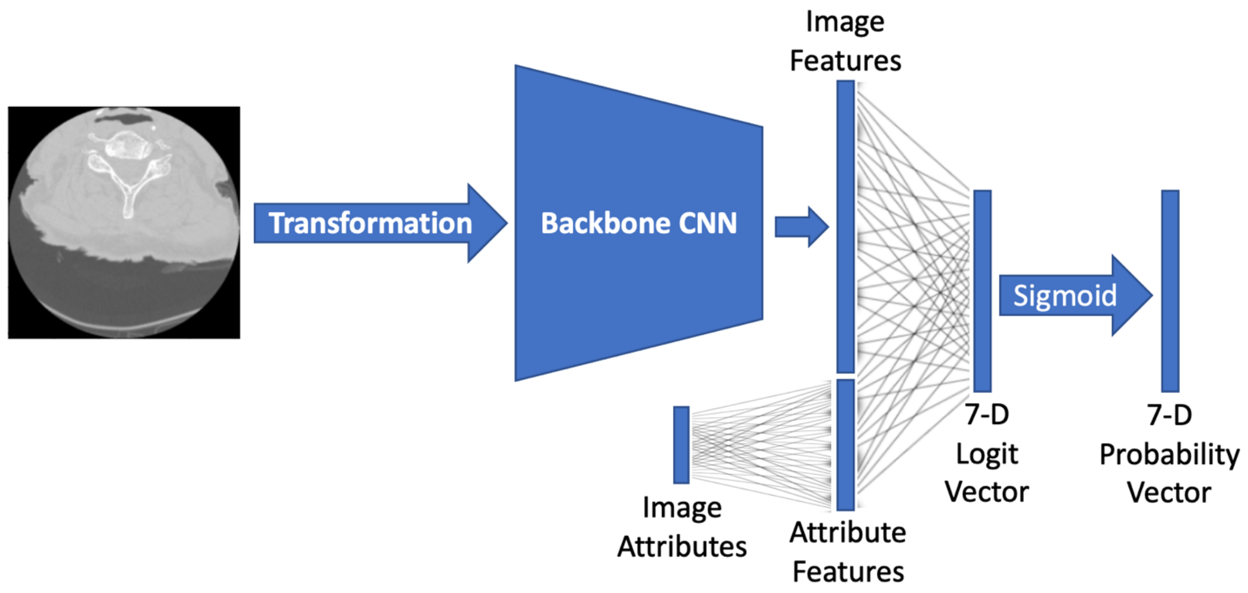


Fig. 16. Multimodal learning for vertebrae detection.

3.3.3 Fracture Detection

Now that each image slice has inferred vertebrae labels attached to it, fracture detection can finally be carried out. Note that these vertebrae labels are not binary, but are the probability of each vertebrae being present on the slice.

Three models are put forward in the subsequent subsections for the fracture detection task.

3.3.3.1 Fracture Detection with Images Only

The most intuitive method to determine whether a patient has fractures on any cervical spine vertebrae is to first design a model to predict the existence of a fracture on a single slice (Fig. 17).

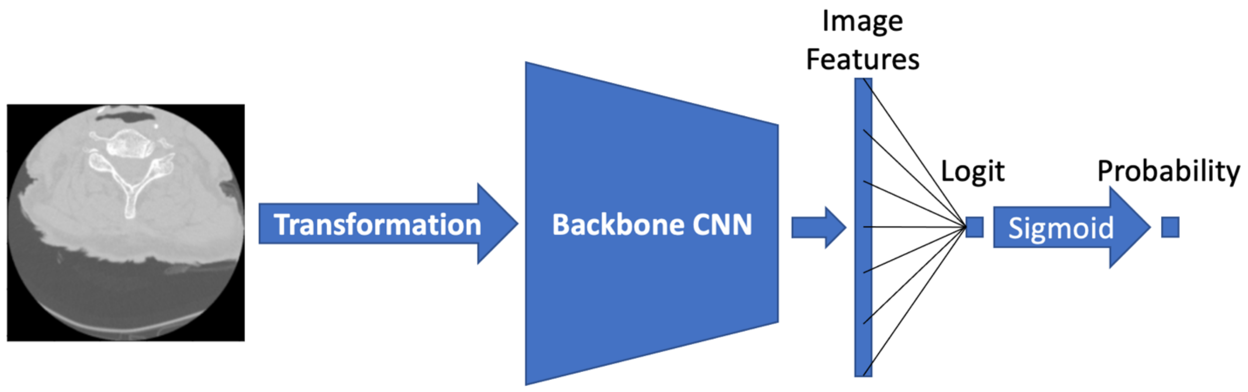


Fig. 17. Fracture detection model with images only.

This model again takes a 2D image slice as the input, but the output is a single number between 0 and 1, indicating the probability of fracture on the input image.

How to determine whether an image actually has a fracture or not? This is the true label for the image and must be well-defined for the model training purpose. The dataset only provides fracture information for the patient overall and up to the vertebra-level, without denoting the existence of fractures on each individual slice. But in the previous stage, all the vertebrae probabilities have already been inferred for the training images of all the 2019 patients. Utilizing the inference results, each slice can be manually labeled as fractured or not by first looking at which vertebrae this slice has (if the probability of some vertebra exceeds some threshold, say 0.5, then the slice is said to have that vertebra), and then checking whether any of those vertebrae have fractures, though the fracture might not necessarily show up on that particular slice.

After assigning a fracture label to each of the training images, the loss can be again defined as the binary cross entropy between the predicted fracture probability and the "true" fracture label. The model can thereby be trained with backpropagation.

Now that the individual probability of fracture on each slice of a patient is available, how can it be determined whether this patient has a fracture on his cervical spine overall or on any one of his vertebrae? Here, the vertebrae probabilities of each slice are to be used once again. For each patient, the predicted probability that vertebra Cj is fractured is calculated as the weighted average of fracture probability of all the slices, i.e.,

$$p_j = \frac{\sum_{i=1}^{n} f_i v_{ij}}{\sum_{i=1}^{n} v_{ij}}, j \in \{1, 2, \dots, 7\}$$

in which f_i is the probability of fracture on the i-th slice of the patient, v_{ij} denotes the probability that vertebra Cj is on slice i. One thing to note here is that, even if some vertebra is fractured, it is not necessarily true that the fracture is visible on every slice containing that vertebra. As a result, the threshold of p_j to determine the existence of fracture might not be 0.5 and needs to be tested.

With the individual probability of fracture on each vertebra, the overall fracture probability of this patient is given by the probability of having fracture on any of the 7 vertebrae, i.e.,

$$p = 1 - \prod_{j=1}^{7}(1 - p_j)$$

assuming the fracture on each vertebra is independent of each other. Though the independence assumption might be too naïve to be correct, this method is still adopted to calculate the overall fracture probability of patients at this time.

3.3.3.2 Fracture Detection with Multimodal Learning

Another model architecture leverages the idea of multimodal learning. This model incorporates the vertebrae probabilities of each image into the input to generate fracture probability predictions (Fig. 18).

Having the vertebrae labels of the image together with the image itself as the input, the model is not only able to judge whether the input image slice has a fracture or not, but also to deduce the probability of fracture on each vertebra of the patient. The output of this model is not the probability of fracture in this single slice, but the individual probability of fracture in each of the seven vertebrae.

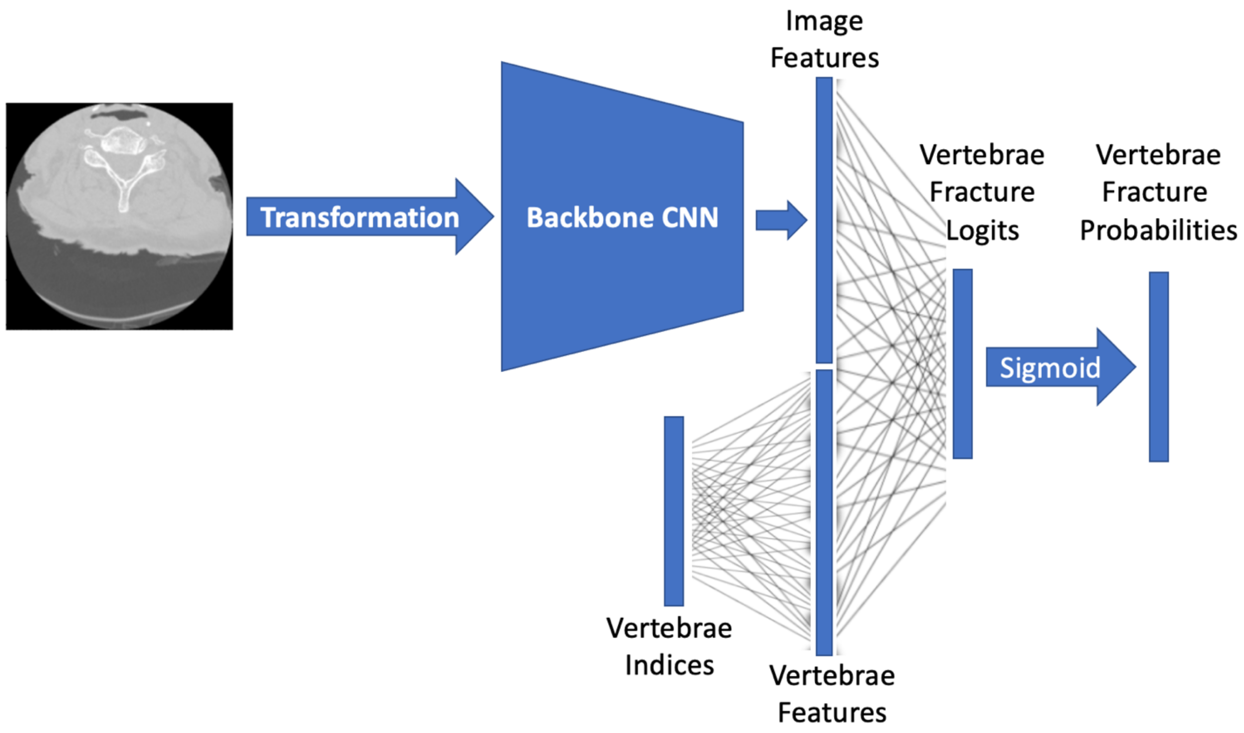


Fig. 18. Multimodal learning for fracture detection.

Although it might sound absurd for a model to give fracture predictions to all the vertebrae by judging from only a single slice image, the vertebrae labels enable the model to do so in some sense, since if the model knows which vertebrae the slice is showing, it can adjust the fracture probabilities of those vertebrae accordingly. Furthermore, even if the model cannot generate accurate fracture predictions on the vertebrae that are not present in the input slice, the final predictions can still give a satisfactory result since they are determined by aggregating all the slices of a patient. The detailed procedures will be elaborated on later in this section.

With this model, there is no need to perform extra computation to generate true output labels as in section 3.3.3.1, since the original dataset fracture labels of each patient can directly be used as the true output of the model.

Again, since this is a multi-label classification problem, the sigmoid function should be used to get the fracture probabilities. This time, weighted binary cross entropy is chosen as the loss function, and the weights are just the vertebrae probabilities, meaning that only the vertebrae visible in the current slice require an accurate prediction of the fracture probability. Inaccurate predictions on other vertebrae that are not in the current slice will not be penalized that much.

By updating the model weights with backpropagation, a model that can produce a 7-dimensional fracture probability vector from each image slice of a patient can be trained.

Finally, the probability that fracture exists on the vertebra Cj of the patient is determined by the weighted average of the vertebra-level fracture probability of all slices, i.e.,

$$p_j = \frac{\sum_{i=1}^{n} f_{ij} v_{ij}}{\sum_{i=1}^{n} v_{ij}}, j \in \{1, 2, \ldots, 7\}$$

in which f_{ij} is the probability that vertebrae Cj has fracture judging from the i-th slice of the patient, v_{ij} denotes the probability that vertebra Cj is on slice i. This method alleviates the potential issue of prediction inaccuracy mentioned previously in this section, since even if the model predicts a high fracture probability on some vertebra (i.e., a high value of f_{ij}) that is not present on the current slice, the low vertebra probability assigned to this slice (i.e., a low value of v_{ij}) can refrain the final weighted average from growing too large for this vertebra Cj.

Similar to the aforementioned approach, under the naïve independence assumption, the overall fracture probability of this patient is given by

$$p = 1 - \prod_{j=1}^{7} (1 - p_j)$$

3.3.3.3 Fracture Detection with Multi-task Learning

Another alternative for fracture detection employs the technique of multi-task learning (Fig. 19). This method is highly similar to the multimodal learning approach mentioned in the previous section, with only a small difference in the model architecture.

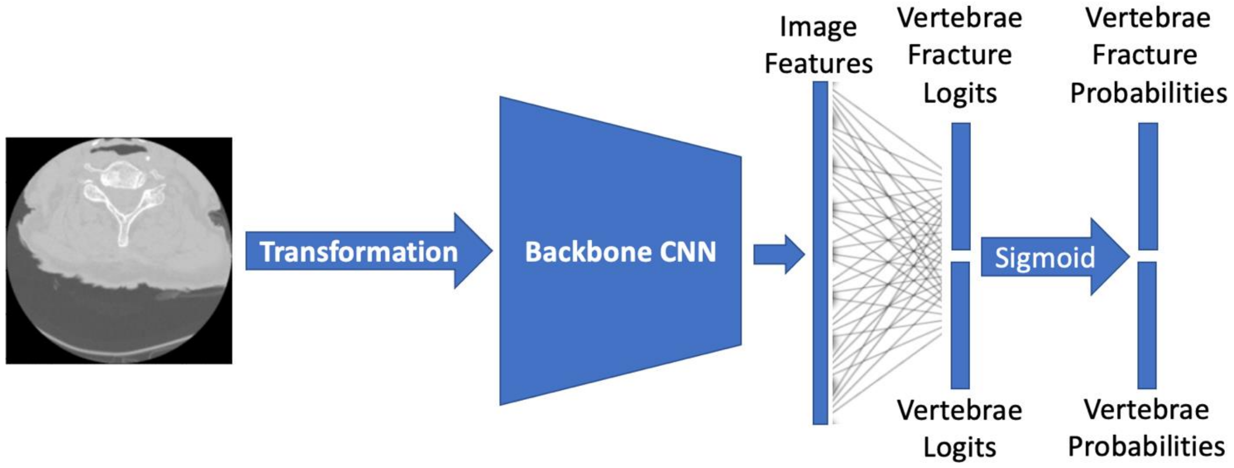


Fig. 19. Multi-task learning for fracture detection.

Instead of treating the vertebrae probabilities as input, the multi-task model considers them as output. So the input to this model is just a single image, while the output contains both the fracture probabilities and the vertebrae existence probabilities (a $7 \times 2 = 14$ dimensional output).

Multi-task learning enables the model to retrieve information from the image to solve multiple tasks. Since the tasks are highly related (e.g., predicting the vertebrae probabilities and the fracture probabilities on the vertebra-level), the model learns more important features from the images for fracture prediction.

The loss function of this model is defined as the summation of two terms. The first one is the same as what has been illustrated in the previous section (i.e., the weighted average binary cross entropy of fracture probabilities), while the second one is the binary cross entropy of the vertebrae predictions. Minimizing the combined loss forces the model to learn from the training images the vertebrae information as well as fracture information simultaneously, thus the resultant output fracture probabilities are likely to be more accurate.

The derivation of the final predictions of the vertebrae fracture probabilities and the overall patient fracture probability is the same as what is introduced in section 3.3.3.2.

3.3.4 Fracture Localization

Among 2019 patients, 239 have bounding boxes in their image slices indicating the position of a fracture. Using these annotated images as training data, a fracture localization model can be obtained. Models such as Faster R-CNN, R-FCN, and SSD can all be applied for fracture localization purpose (Fig. 20).

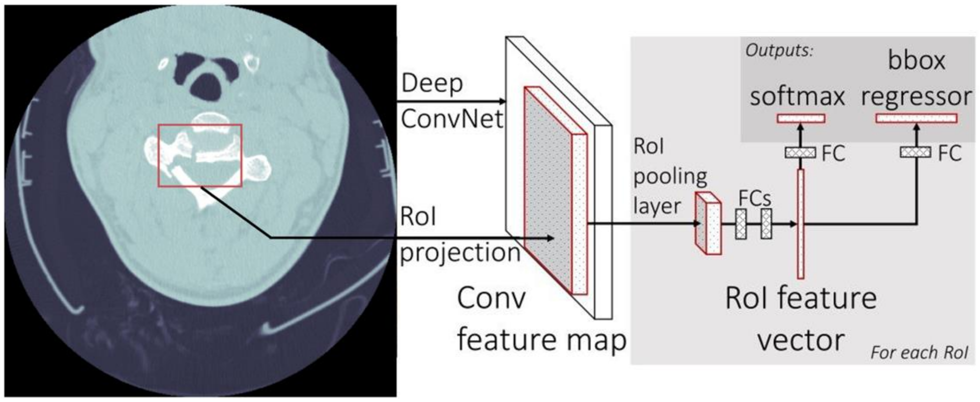


Fig. 20. Fast R-CNN for fracture localization.

The fracture detection model introduced in section 3.3.3.1 can be used to infer whether a single slice has a fracture or not. All the slices that have been identified as fractured can be forwarded to the localization model to determine where the fracture is. By this means, this project goes beyond simply predicting the existence of fractures on a cervical spine. The exact position of fractures in the 3D coordinate system can possibly be disclosed.

3.3.5 Cervical Spine Visualization

In the final stage of the project, it is expected to have the fracture locations displayed in a 3D coordinate system. Since all the slices of a patient share a uniform width and height, and the z-coordinates can also be inferred from the slice index, the 3D cervical spine can be recovered from the slices. The final visualization effect should be similar to Fig. 21. Moreover, the fracture locations given by models in the previous stage can also be identified on the same 3D plot, which hopefully can help to visualize the position of fractures in a 3D space.

Fig. 21. A 3D view of a cervical spine.

4. Current Experiments

Up to now, phase 2 of the project is close to completion. A vertebrae detection model with images only introduced in section 3.3.2.1 was constructed, and vertebrae probabilities were inferred for all the images in the dataset. This section delineates the detailed model training, comparison, and selection process.

4.1 Experiment Setup

Model implementation, training, and evaluation were all realized with a deep learning framework, PyTorch.

The first task was to perform model selection. 14 deep CNNs were chosen as the backbone model, judging from their performance on the classification task with ImageNet. Other off-the-shelf CNN models, though some of which demonstrated an even higher classification accuracy on ImageNet, were mostly too heavy to be trained in a reasonable amount of time. Therefore, most models selected were relatively lightweight and efficient.

- AlexNet
- VGG16 (with batch normalization)
- ResNet50
- ResNeXt50 (32 × 4D)
- GoogLeNet
- InceptionV3
- DenseNet121
- EfficientNetV2 (small)
- MobileNetV3 (large)
- ConvNeXt (small)
- MNASNet (with a depth multiplier of 1.3)
- ShuffleNetV2 (with 2.0 × output channels)
- Swin Transformer (small)
- RegNetY-8GF

A total of 29832 images from 87 patients were tagged with vertebrae labels, and could all be used for model training in this stage. These images were split into two sets, a training set (80%) and a validation set (20%).

For time saving purpose, one epoch training was carried out. The batch size was set to 32, Adam was adopted as the optimizer, and one cycle learning rate policy [20] with a maximum learning rate of 1e-3 was employed for learning rate scheduling. GPU was utilized for training acceleration. Each of the 14 models was trained separately by the training set and then tested on the validation set, with their training time and validation accuracy being recorded.

Then, the 5 models with the highest validation accuracy were selected to be further evaluated by group 5-fold cross-validation, adopting the same training policy. Note that it is inappropriate to apply ordinary K-fold cross-validation here because of the high correlation of slices from the same patient. Randomly splitting the images into K folds is likely to cause some slices of a patient to fall into a fold, and some other slices from the same patient to fall into another fold, leading to information leakage.

The model with the highest average accuracy in cross-validation was regarded as the best model. That particular model, after being trained by the entire set of images (100%), was then employed to infer all the vertebrae probabilities for all the slices of the 2019 patients.

4.2 Experimental Results

The evaluation accuracies of the 14 vertebrae detection models and their training time are illustrated in Fig. 22.

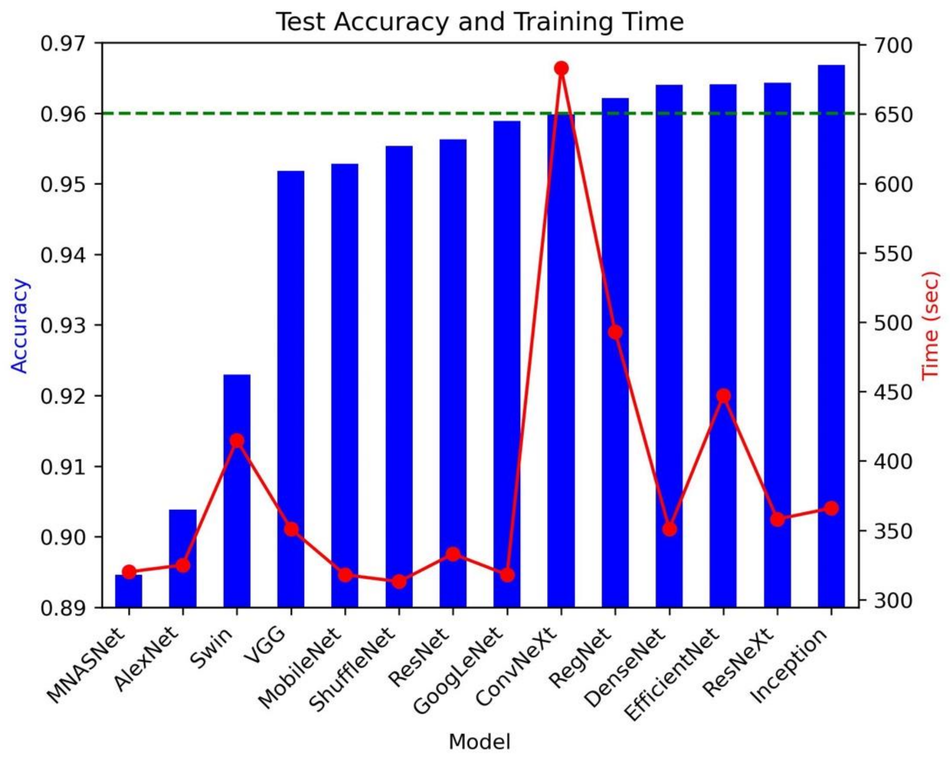


Fig. 22. Vertebrae detection test accuracy and training time of 14 different backbone models.

It could be observed that 5 of these models (RegNet, DenseNet, EfficientNet, ResNeXt, and Inception) reached an accuracy of 96%, which is a remarkable outcome. These 5 models were then tested by cross-validation, with the average validation accuracy and training time measured and demonstrated in Fig. 23.

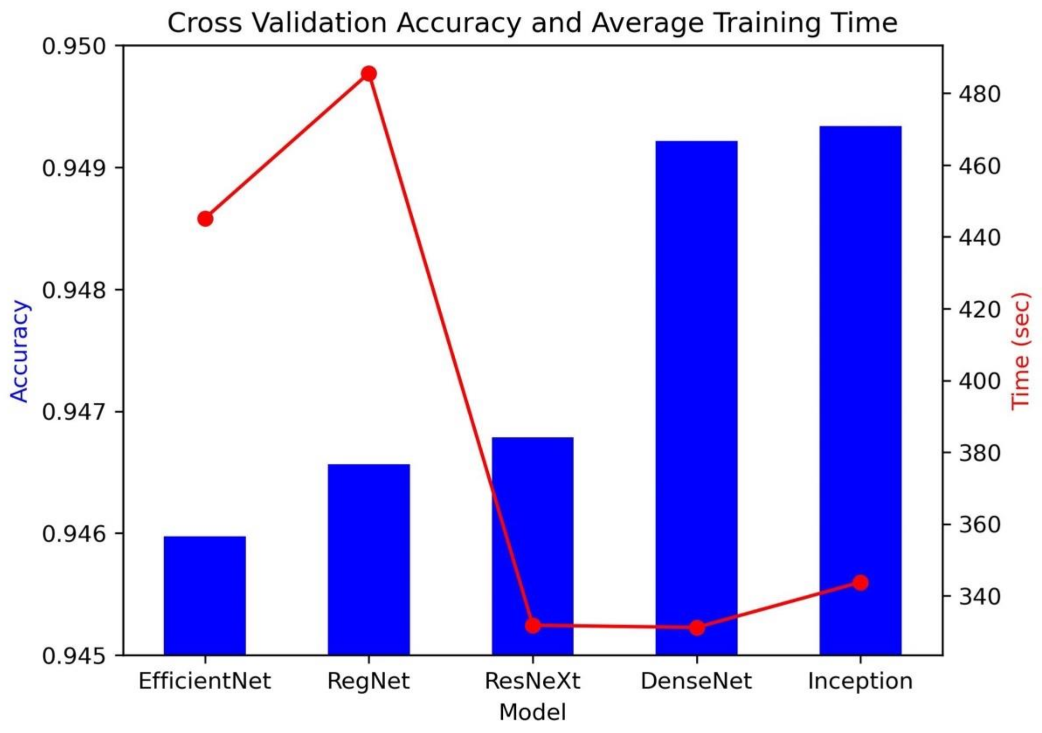


Fig. 23. Vertebrae detection cross-validation accuracy and average training time of 5 backbone models.

InceptionV3 turned out to be the best model among them: with a satisfying training speed, the cross-validation accuracy reached 94.93%, meaning that close to 95% of the time, the model was able to accurately predict the presence of any type of vertebrae on a slice by merely looking at the image.

5. Discussion

This section encapsulates the difficulties and limitations of this project, as well as the future work.

5.1 Difficulties and Limitations

The major difficulty derives from insufficient training time. Training models on a large-scale image dataset can be extremely time-consuming. The time to train a vertebrae detection model for a single epoch with GPU takes about 6 minutes on average. Besides, memory restriction is also another concern. These two problems incur a series of issues.

First, model training for multiple epochs is too expensive to be carried out. Normally speaking, training models for more than one epoch may help to achieve higher accuracy, but even 5 epochs here might be overkill, since more than half an hour of time is required. This is only for vertebrae detection model training, which is only based on image slices from 87 patients. In stage 3 of the project, when training the fracture detection models based on images of 2019 patients, multi-epoch training is almost unrealistic.

Second, model optimization is too costly to carry out. Hyperparameters including the learning rate, optimizer, learning rate scheduling policy, number of hidden layers, and number of neurons in each layer, can all be fine-tuned to reach higher accuracy. For each of the different backbone models, the optimal set of hyperparameters can differ completely. Hence, some models did not show a good performance in the experiments, possibly due to the incorrect hyperparameters chosen for them, but not their incapability to give accurate predictions. However, if a grid search is to be conducted for hyperparameter tuning for each model, the problem of combinatorial explosion occurs, and the training time grows exponentially. Therefore most likely, the training scheme has to stick to some common settings (e.g., adopting a typical learning rate of 1e-3).

Third, due to the GPU memory limit, the performance of some models cannot be evaluated. For instance, many powerful models such as Vision Transformer have shown great success in the field of computer vision, but can hardly be applied in this project for predictions because of their huge sizes. Although the batch size can be decreased to run those models, the time it takes also rockets beyond an acceptable level.

5.2 Future Work

This section summarizes the remaining tasks of the project in 6 different aspects.

5.2.1 Image Preprocessing

The image preprocessing is not put into force up to now. The currently developed vertebrae detection models are based on the original training images, but not the transformed ones. The effect of data augmentation, image masking, and image segmentation will be tested in the future by examining the improvement they can bring to the ultimate prediction accuracy.

5.2.2 Vertebrae Detection with Auxiliary Attributes

The vertebrae detection models have not yet incorporated the extra image features. To make use of the additional valuable information, the models have to be reconstructed, and their performance will be evaluated and compared to the current outcomes.

5.2.3 Fracture Detection

The core stage of this project is fracture detection. Having obtained vertebrae probability labels for each of the training images, models will be constructed following the three approaches mentioned in section 3.3.3. The first model described in section 3.3.3.1 outputs the fracture probability of every single slice, whose results will be forwarded for fracture localization in stage 4. The second and third versions of the model introduced in sections 3.3.3.2 and 3.3.3.3 leverage the techniques of multimodal learning and multi-task learning to further improve the model performance.

5.2.4 Fracture Detection with 3D CNN

An advanced method can be adopted to directly predict the fracture probabilities in each vertebra using all the slices of a patient simultaneously, which is known as 3D CNN (Fig. 24).

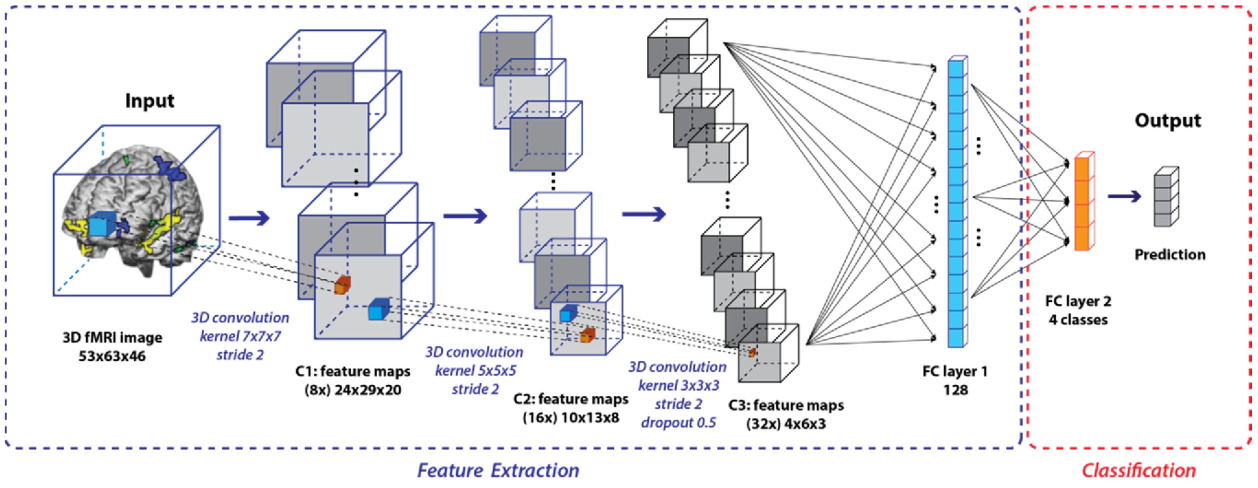


Fig. 24. Medical image classification with 3D CNN.

The rationale of 3D CNN is similar to that of the 2D case, except 3D convolutional kernels are applied to extract the feature maps of some 3D object. Although CT scan images are in 2D form, if all the slices of a patient are stacked together, the z-coordinate can be constructed to recover the original 3D cervical spine. After resizing on the z-axis, a 3D image of the cervical spine with a uniform shape can be generated for each patient, and the model can thus be trained, with the output being the 7-dimensional vertebrae fracture probabilities.

If a 3D CNN model is implemented, this new procedure can replace the original stage 2 and stage 3 of this project, since the 3D model does not require vertebrae labels as training data.

5.2.5 Fracture Localization

With the CT scan images annotated with bounding boxes, fracture localization models such as Faster R-CNN will be trained. Applying this model, the position of fracture in each fractured image can be identified. Images with fractures will be manually selected to test the effect of this model. If the result is satisfying, those predicted bounding boxes will be used in the 3D visualization process.

5.2.6 Result Visualization

The final stage of the project will be cervical spine visualization. As introduced before, a 3D cervical spine will be constructed and visualized in the coordinate system, and the places with a fracture will be highlighted in some special color.

6 Conclusion

The project employs deep convolutional neural networks for cervical spine fracture detection, and CT scans of cervical spines are used for model training and evaluation.

The project is separated into several stages. Before being forwarded to the model, images are processed by a series of transformations including augmentation, masking, and segmentation. Then a vertebrae detection model is trained to identify the vertebrae that are present in each slice. (Currently, a vertebrae detection model with InceptionV3 as the backbone CNN has achieved an accuracy of 95%.) Auxiliary attributes can possibly be incorporated into the model for even higher prediction accuracy. Based on the vertebrae probabilities, fracture detection models can be constructed, with the technique of either multimodal learning or multi-task learning. Another alternative to detect fractures with vertebrae labels is to apply 3D CNN, which takes all the slices of a patient as the model input and directly gives predictions on the fracture probability of each vertebra. A fracture localization model can be adopted to find the position of fractures, so that the cervical spine together with the recognized fractures can then be displayed in a 3D coordinate system.

The major difficulties of this project are the time-consuming training process and the insufficient GPU memory. These problems are an obstacle to multi-epoch training, hyperparameter tuning, and the implementation of heavyweight models. The future tasks include image preprocessing for better model performance, implementation of vertebrae detection model with auxiliary attributes, construction of fracture detection as well as localization model, and 3D visualization of the cervical spine.

References

[1] D. Shen, G. Wu, and H.-I. Suk, “Deep learning in medical image analysis,” *Annu. Rev. Biomed. Eng.*, vol. 19, no. 1, pp. 221–248, 2017.

[2] G. Litjens *et al.*, “A survey on deep learning in medical image analysis,” *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[3] J. Cho, K. Lee, E. Shin, G. Choy, and S. Do, “How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?,” *arXiv [cs.LG]*, 2015.

[4] Y. Li *et al.*, “A comprehensive review for MRF and CRF approaches in pathology image analysis,” *arXiv [cs.CV]*, 2020.

[5] A. N. Basavanhally *et al.*, “Computerized image-based detection and grading of lymphocytic infiltration in HER2+ breast cancer histopathology,” *IEEE Trans. Biomed. Eng.*, vol. 57, no. 3, pp. 642–653, 2010.

[6] J. Bioucas-Dias, F. Condessa, and J. Kovacevic, *Alternating direction optimization for image segmentation using hidden Markov measure field models In: Gurcan MN, Madabhushi A, editors. IS&T/SPIE electronic imaging. International Society for Optics and Photonics*. 2014.

[7] A. Van Engelen K.Á, “Three-dimensional carotid ultrasound plaque texture predicts vascular events,” *Stroke*, vol. 45, no. 9, pp. 2695–2701, 2014.

[8] H. C. Achterberg *et al.*, “Hippocampal shape is predictive for the development of dementia in a normal, elderly population: Hippocampal Shape is Predictive for Dementia,” *Hum. Brain Mapp.*, vol. 35, no. 5, pp. 2359–2371, 2014.

[9] M. de Bruijne, “Machine learning approaches in medical image analysis: From detection to diagnosis,” *Med. Image Anal.*, vol. 33, pp. 94–97, 2016.

[10] J. Zhuang, J. Cai, R. Wang, J. Zhang, and W.-S. Zheng, “Deep kNN for Medical Image Classification,” in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, Cham: Springer International Publishing, 2020, pp. 127–136.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[12] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv [cs.CV]*, 2014.

[13] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *arXiv [cs.CV]*, 2015.

[14] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional Neural Networks,” *arXiv [cs.LG]*, 2019.

[15] S. Ruder, “An overview of gradient descent optimization algorithms,” *arXiv [cs.LG]*, 2016.

[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv [cs.LG]*, 2014.

[17] F. Zhuang *et al.*, “A comprehensive survey on transfer learning,” *arXiv [cs.LG]*, 2019.

[18] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, “Multimodal machine learning: A survey and taxonomy,” *arXiv [cs.LG]*, 2017.

[19] S. Ruder, “An overview of multi-task learning in deep neural networks,” *arXiv [cs.LG]*, 2017.

[20] L. N. Smith and N. Topin, “Super-convergence: Very fast training of neural networks using large learning rates,” *arXiv [cs.LG]*, 2017.