ABSTRACT

Title of thesis: JOINT SPACE NARROWING CLASSIFICATION BASED ON HAND X-RAY IMAGE

Author: Fakai Wang
Master of Science, 2020
University of Maryland, College Park

Thesis directed by: Professor Min Wu
Department of Electrical and Computer Engineering
University of Maryland, College Park

Dr. Le Lu
PAII Inc. Bethesda Research Lab

For Rheumatoid Arthritis (RA), Joint Space Narrowing (JSN) is one major symptom that can be read from X-ray images. In this thesis, we investigate the JSN classification for RA diagnosis in terms of methodology, data analysis, neural network models, performance analysis.

We first perform the statistical data analysis and develop algorithms to extract joint patches. We design the baseline model and carry out the prediction analysis. Second, we explore the correlation between joints and develop a fusion model. Sharing information on different joints of the same patient can increase the model performance. We quantitatively compare the performance between unified classifiers and separate classifiers. Third, we design the attention map model for joints with complex contexts, which filters out noise. The resulting neural network models show encouraging JSN prediction for Rheumatoid Arthritis.

JOINT SPACE NARROWING CLASSIFICATION BASED ON HAND X-RAY IMAGE

by

Fakai Wang

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2020

Advisory Committee:

Professor Min Wu, Chair/Advisor
Professor Rama Chellappa
Dr. Le Lu, Co-Advisor

Acknowledgments

For two years, I have been fascinated by the significant changes that computer vision has brought to the society, from autonomous driving to automated manufacturing, to medical diagnosis, to selfie beautification, and to identity verification. Large datasets and computational power have changed the area of computer vision research, bringing a bright future of applying artificial intelligence to fundamental areas. New applications would change our way of living in the future, to improve our life quality and save immense human labor. In this way, human experts could have more time to serve people's individual needs better. Computer-aided medical diagnosis has raised significant interests in both academic study and industry practices. We already have all kinds of powerful radiology machines available such as X-ray, CT, and MRI. Currently, one practical problem is that we do not have enough well-trained radiologists and doctors in many hospitals. It would be expensive for patients to get accurate diagnosis results since it is both time-consuming and labor-intensive to detect and evaluate disease correctly. If we can apply Artificial Intelligence to disease diagnosis, then medical efficiency and medicare procedure would be much different. Many companies are focusing on these directions. During my internship at PAII Inc. Bethesda, I have learned a lot about medical imaging research.

I want to thank all the people who have helped me to complete this thesis. Firstly, I would like to thank my academic advisor Professor Min Wu, who offered lots of encouragement and research advice leading me to a better understanding of

machine learning. Next, I would like to thank Professor Larry S. Davis, who guided me into the area of computer vision two years ago. And I give thanks to Professor Rama Chellappa, for the foundational machine learning and pattern recognition course taught by him through which I have learned a great deal. Thirdly, during my internship at PAII Inc, all colleagues were very supportive. I want to thank Dr. Le Lu, Director of PAII Inc. Bethesda Research Lab, who provided constant encouragement and visionary directions in both research and career development. I want to thank Dr. Shun Miao, who gave me detailed instructions whenever I have questions on research and life. Without his help, this thesis would have been a distant dream. I thank Dr. Kang Zheng, who gave me technical suggestions and wrote many helper functions. Besides, I owe many thanks to other colleagues at PAII Inc, from whom I have deepened the understanding of medical imaging.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

PACS	Picture Archive Communication System
CAD	Computer Aided Diagnosis
MCP	MetaCarpoPhalangeal
PIP	Proximal InterPhalangeal
CMC	CarpoMetaCarpal
JSN	Joint Space Narrowing
CNN	Convolutional Neural Network
DL	Deep Learning
SGD	Stochastic Gradient Descent
LR	Learning Rate
DR	Droup Out
Acc	Accuracy
Val	Validation
GT	Ground Truth
GCN	Graph Convolutional Network
NAS	Neural Architecture Search
CAM	Class Activation Mapping
PNG	Portable Network Graphics
POI	Point Of Interest
ROI	Region Of Interest
DICOM	Digital Imaging and Communications in Medicine

Chapter 1: **Introduction: Rheumatoid Arthritis and Joint Space Narrowing**

1.1 Overview

Rheumatoid Arthritis (RA) is a long-term autoimmune disorder primarily affecting body joints. The immune system would attack the tissues and joints. As symmetrical poly-arthritis, RA in its early stage would affect small joints, causing painful swelling, stiffness, erosion, joint space narrowing (JSN). Although there are many challenges for early diagnosis, it can be achieved through close monitoring RA symptoms [1] [2].

This chapter aims to give necessary background information for the RA disease and JSN task. RA affects a large number of people in the world, and early RA diagnosis would benefit society greatly. RA diagnosis is carried out partially through inspecting X-ray scanning since Joint Space Narrowing (JSN, one major RA symptom) is identified via X-ray images. With the benefits of quantitative imaging, the computer-aided diagnosis would empower doctors with scientific diagnosing methodologies. In this thesis, we study and develop algorithms to classify JSN accurately. To be specific, we would use hand X-ray images to classify the degrees of

joint space narrowing.

We give a brief introduction to the deep convolutional neural network (CNN). With the advent of public datasets and powerful hardware, the convolutional neural network has revolutionized computer vision tasks like image classification, object detection, object segmentation. Convolutional layers work in a hierarchical way to represent shapes and details. Fully connected layers feed on CNN features and make the final prediction.

This chapter summarizes the research steps taken in the thesis. After giving the background information, we list the main topics in each chapter and present chapter focus and conclusions. We also identify reading-related suggestions along the way.

1.2 RA diagnosis and hand joint space narrowing classification

Rheumatoid Arthritis (RA) shows its symptoms gradually. Joint space gradually narrows down, from wide apart to collapsing onto neighboring bones. Doctors have a JSN scoring system to judge the severity of Rheumatoid Arthritis through X-ray images. Hand X-ray images can capture joint attributes such as space, textures, and erosion. Four types of joints (metacarpophalangeal, proximal interphalangeal, carpometacarpal, Scaphoid) are labeled in the range of 0-4.

RA diagnosis models [1] give recommended procedures and steps for RA disease. Early detection and treatment are essential to prevent the disease from deteriorating and affecting more joints. There are many aspects of RA diagnosis [2] [3],

such as morning stiffness, blood test, joint lesion, disease history, and gene inheritance. In this thesis, we only focus on one major symptom, the hand joint space narrowing.

Computer-aided diagnosis (CAD) facilitated by machine learning has drawn considerable attention, with the hope of reducing human workload and providing better medical service. For the JSN task, several steps are involved. First, we need to develop algorithms for hand landmark detection and joint patch extraction. Second, we need to design data pre-processing steps and model architectures. Third, we need to collect results and carry out performance analysis.

1.3 Dataset statistics and data preparation

Joint space narrowing degree as an RA severity indicator is influenced by many factors, such as age, joint moving frequency, nutrition, and other diseases. Four types of hand joints are investigated (MCP, PIP, CMC, Scaphoid). JSN scores range from 0 to 4, with 0 meaning no joint space narrowing, 4 meaning severe joint space narrowing. Expert doctors and radiologists label the JSN score, and we assume no bias in labeled data.

In Chapter 2, we first show JSN statistics and compare JSN distribution in different joints. We do data analysis to understand the JSN task. Then we introduce algorithms to localize joints and to extract joint patches. We use the Curve-GCN [4] as the joint landmark detection algorithm to obtain joint key points. After that, we give details about the joint patch extraction algorithm, such as affine transformation

and coordinates mapping. At the end of Chapter 2, we discuss data augmentation to see the benefits and limitations. The reader can skip Chapter 2 if you are more interested in model designing and performance analysis.

1.4 Convolutional neural network and deep learning models

Neural networks use the backpropagation algorithm [5] to change internal parameters that are used to compute the representation. Convolutional Neural Network (CNN) [6] deals with the variability of 2D shapes and becomes the mainstream method for computer vision tasks nowadays. Deep learning (DL) [7] has shown excellent performance improvement in nearly every machine learning area in recent years. We want to take advantage of CNN and DL advancements for the JSN classification.

ImageNet competition [8] has brought about many Deep CNN models [9], such as Alexnet [10], Clarifia [11], VGG-net [12], ResNet [13] and GoogLeNet [14]. Convolutional neural network models have already surpassed human performance in visual classification tasks, given sufficient training data. The CNN success has also be introduced into medical imaging area [15]. In this thesis, we are particularly interested in applying DL CNN models to joint space narrowing classification. Our models are based on Squeezenet [16] and ResNet [13], with the consideration of RA JSN characteristics [1][17].

In Chapter 3, we analyze the baseline model that is based on Squeezenet. In Chapter 4, we introduce the fusion model, which provides information-sharing

mechanisms among joints. In Chapter 5, we introduce the attention model for joints with complex contexts. In each chapter, we would show the design logic, the training/validation process, experiment results, and performance analysis. For the convenience of naming, we would call a model without attention module a patch model, otherwise an attention model. These three chapters would use similar data preparation, so the readers are suggested to read Chapter 2 if you want to know more about pre-processing details.

1.5 Data augmentation and training techniques

Along with the flourishing abundance of CNN models come neural network training techniques. Data augmentation [18], dropout [19], optimizer [20][21], weight decay [22] are widely used in deep learning models.

It is called overfitting when a trained model could not generalize to the testing set. The lack of labeled data is the leading cause of the overfitting problem. Data augmentation [18] can be used to expend the training data space. For computer vision tasks, there are many conventional augmentation techniques, such as random cropping, translation, flipping, shearing, and rotation. We explain data augmentation with details in Chapter 3.

For all the models, we integrate the data augmentation into the experiments. Other training techniques such as learning rate, stochastic gradient descent (SGD), dropout, weight decay are set similarly. These techniques produce consistent performance-boosting effects on different models. We choose a set of hyperparameters that work

for all models.

1.6 Performance analysis and result visualization

In Chapters 3, 4, and 5, we summarize the experiment results after giving design details. Then we would conduct performance analysis. The comparisons include network backbone, classifier mode, data augmentation, and joint type. For the Scaphoid joints, patch models and attention models are compared for the performance analysis.

In Chapter 2, we inspect the correct prediction on joint patches, which would provide the reader with more intuition on model performance. The visual inspection sheds light on JSN task challenges. In Chapter 3, We use Class Activation Mapping (CAM) [11] to visualize the learned feature representation. In Chapters 3, 4, and 5, we use the confusion matrix to visualize the performance. The thesis conclusion is given at the end of Chapter 5.

Chapter 2: **Joint Space Narrowing Dataset and Data Preparation**

2.1 Overview

Joint space narrowing (JSN) is a major RA manifestation and can be verified from the X-ray image examination. Hand X-ray images are readily available in hospitals and can be integrated into computer-aided diagnosis working flow. Assuming we have RA patients' hand X-ray images, machine learning models can incorporate doctor/radiologist medical knowledge and make sensible predictions. In this thesis, we are interested in applying deep learning models to JSN score classification based on hand X-ray images.

We first conduct a statistical analysis of JSN distribution from labeled data. Precise JSN prediction requires that the model be trained on precisely labeled data. Doctors and radiologists should have sufficient working experience on patient disease diagnosis. Data labels should have consent from all labeling professionals. The training dataset should introduce no selection bias towards lesion distribution. So ideally, the datasets must contain all possible lesion appearances, and the lesion distributions in the dataset should be the same as in real life.

Secondly, we show algorithms for detecting joint key points. Our landmark detection algorithm is based on the graph convolutional neural network. We show the affine transformation process to extract joint patches. We also do visual inspections on joint patches to understand JSN scores better.

In the end, we discuss the strategy for data augmentation. Data augmentation increases the training data variations and alleviates the overfitting problem. It is used across all the models in our experiments.

This chapter is the domain foundation for the following chapters since data preprocessing and augmentation are used for all machine classification models. However, the reader still can skip this chapter if he/she is more interested in the network model and performance analysis.

2.2 Obtain, store and transfer medical images

Picture archiving and communication system (PACS) [23] provides the economical storage and efficient retrial of medical images. The image format used in PACS for storage and communication is DICOM (Digital Imaging and Communications in Medicine). DICOM is the global medical imaging standard designed to ensure the interoperability of systems. DICOM format is used to produce, store, display, process, send, and retrieve medical images. Original X-ray images are in DICOM format. It contains the scanning conditions and geometry information, which could be used to do the affine transformation. Hand X-ray images cover all fingers and the whole wrist.

Table 2.1: JSN Ground Truth score distribution

Joint Type	patients	X-rays	joints	JSN=0	JSN=1	JSN=2	JSN=3	JSN=4
MCP	130	239	1195	420	508	246	84	47
PIP	130	257	1285	420	529	201	86	69
CMC	130	258	774	387	160	141	59	36
Scaphoid	130	258	774	327	169	102	60	125

All the images should be obtained in a standard way. The X-ray picturing systems have the same lighting, field of angle, and resolution. In this chapter, it is assumed that we have integral and precise data. All the DICOM images in our experiments are cleaned to eliminate any private information. Files are named with dates and numbers. Only imaging technology-related information remains. There is no record for patient name, age, gender, or other personal information.

2.3 Overall data distribution

Currently, we have labeled data for 130 patients. Only for model verification purpose, we assume the data is representative and sufficient. Each patient has two hands' X-ray images. Four types of joints, namely MCP (Meta Carpo Phalangeal), PIP (Proximal Inter Phalangeal), CMC (Carpo Meta Carpal), Scaphoid, are investigated for the JSN task. Each joint has a JSN score, ranging from 0 to 4. The statistics of these joints is in Table 2.1. It should be noted that the data in this thesis does not represent the real JSN distribution. There are many ways bias could be introduced, but how to collect medical data scientifically is out of this thesis scope.

From Table 2.1, several points can be made concerning JSN score distribution. MCP and PIP have the same number of joints and similar JSN distribution. CMC and Scaphoid have the same number of joints and a similar number of JSN=0, 1. There are substantially more percentages of JSN=4 entry for Scaphoid compared to other joints.

2.3.1 JSN score correlation among joints

First, we analyze the dependency between joints. We use the dependency matrix to visualize the correlation between a pair of joints. In Figure 2.1, we show MCP1 and MCP2 JSN score correlation on the left, MCP2 and MCP3 correlation on the right. More entries on the diagonal means that the two joints are more correlated in JSN distribution. There are obvious JSN correlations within each pair of joints. More than 70% entries are either on diagonal or next to diagonal in the confusion matrix. This means JSN scores are highly correlated. Similar correlations can be seen for pairs such as MCP3-MCP4, PIP2-PIP3, PIP3-PIP4, and CMC3-CMC4.

2.3.2 JSN score correlation difference for left and right hands

Now we move on to analyze the correlation differences between left and right hands. In Figure 2.2, the left is two MCP joints of the left hand dependency matrix. While the right is for the right hand. The overall patterns are similar in two matrices, but the entry value at (0, 0) has a discrepancy, which indicates that JSN distribution is not the same in the left and right hands. Given enough data, it is better to use

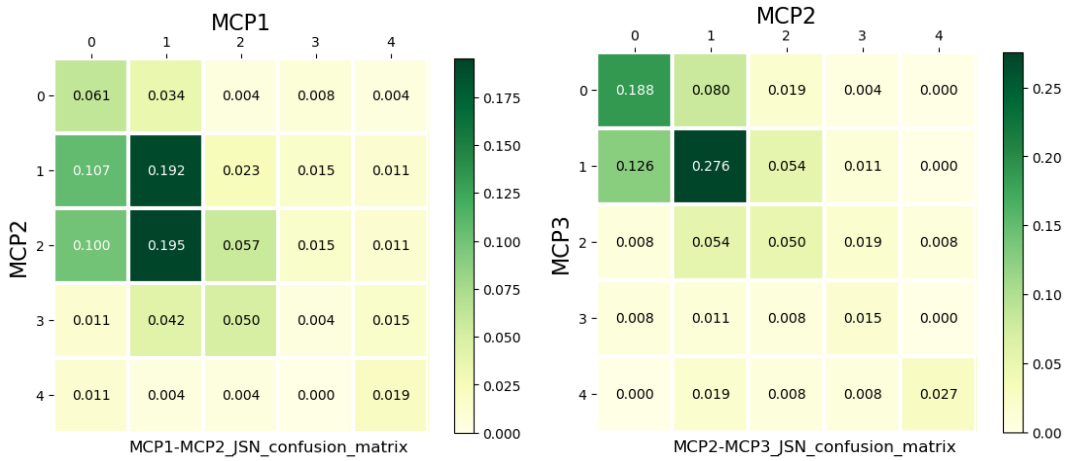

Figure 2.1: *MCP JSN score dependency of neighboring joints*

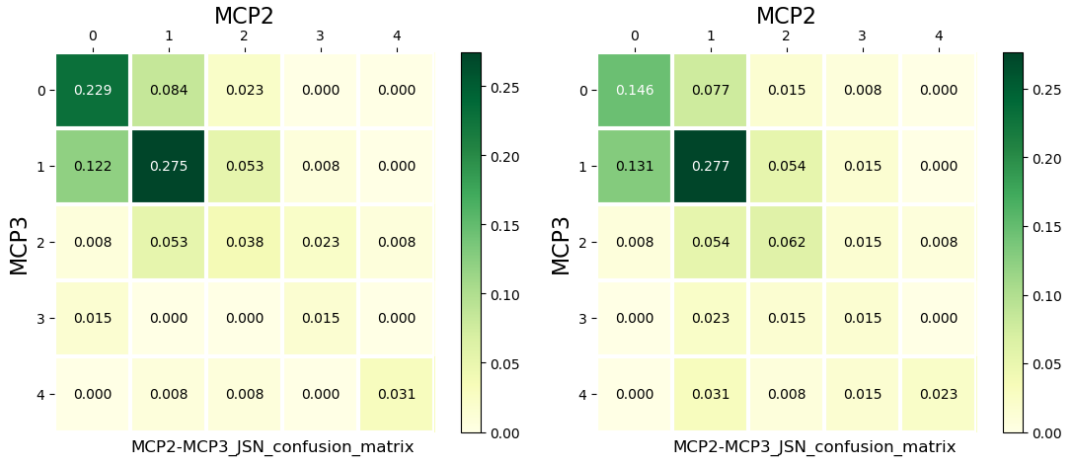

Figure 2.2: *Left and right hand MCP joint JSN dependency*

two separate models for the left and right hands.

2.4 Visual inspection of JSN score on X-ray image

To better understand the JSN labeling, we show different JSN scores on the input images. We visualize the correct predictions on Figure 2.3 2.4 2.5 2.6. Predictions are from baseline model in Chapter 2. These visualizations facilitate model analysis and help gaining insights for improvement.

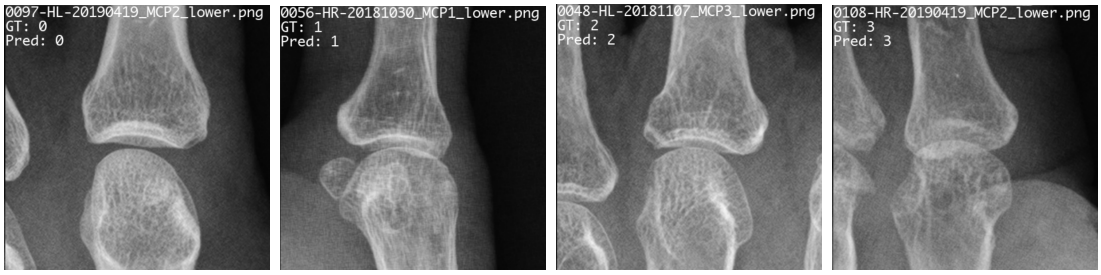

Figure 2.3: *Correct predictions for MCP JSN*

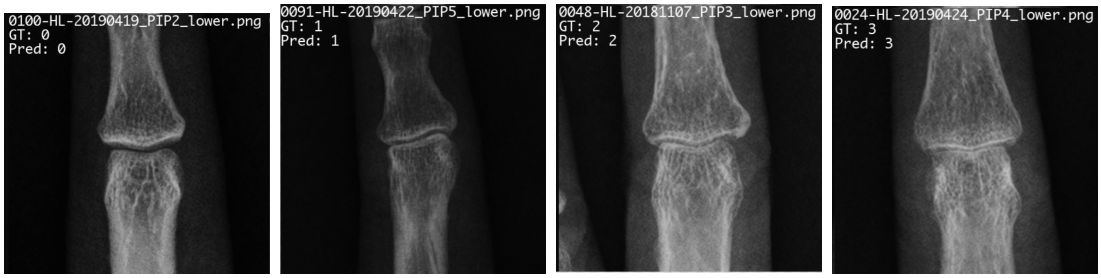

Figure 2.4: *Correct predictions for PIP JSN*

Now we look at JSN scoring standards. From the correct predictions 2.3 2.4 2.5 2.6, we can get an intuitive sense of different JSN levels. The ground truth labels come from professional doctors/radiologists. Clear differences between JSN levels can be spotted. A healthy joint means clear margins between joint bones, little whitening on the joint bone surfaces, and normal bone aligning directions. From the figures, GT=0 usually has these characteristics. For GT=3, most healthy joint characteristics are lost. GT=1 and GT=2 appearances lie in between GT=0 and GT=3.

The Figure 2.4 depicts different JSN levels of PIP joint. The PIP JSN score patterns here are very similar to MCP's. JSN score depends on joint space size and bone surface quality. But the joint shape in PIP is generally more tight along the bone margin lines than MCP's. Figure 2.5 shows that joint space size and

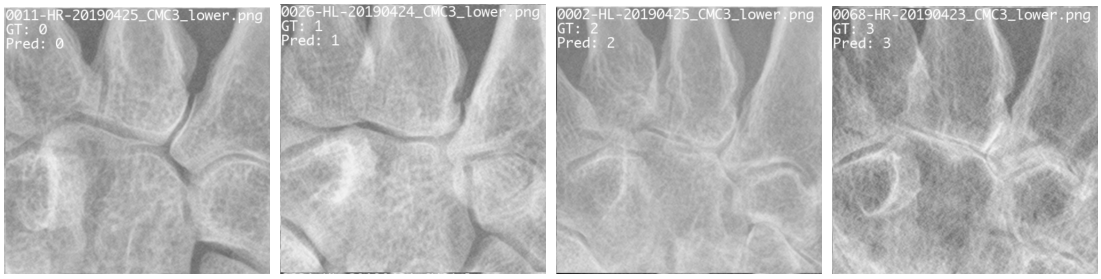

Figure 2.5: *Correct predictions for CMC3 JSN*

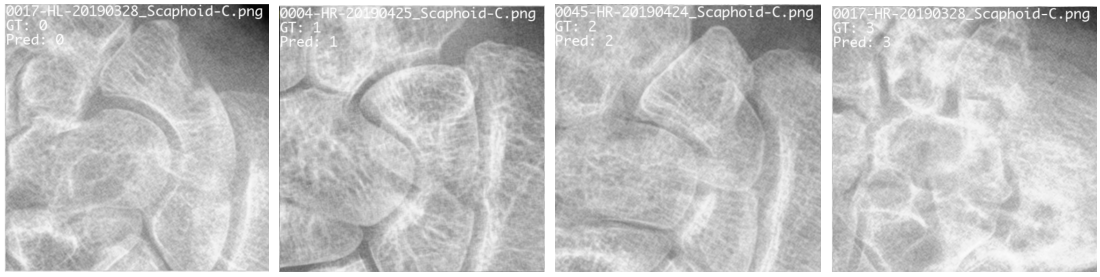

Figure 2.6: *Correct predictions for Scaphoid-C, Scaphoid-Tm, Scaphoid-R*

whitening degree determine the JSN severity level for CMC joints. But the CMC and Scaphoid have more complex surrounding contexts compared to MCP and PIP. These contexts may hinder the JSN prediction since only the joint area is needed. Figure 2.6 show different ground truth levels for Scaphoid-C. Because of the complex surrounding, Scaphoid joints images get more blurred. This can be seen from Figure 2.6. GT=3 indicates it's hard to identify the joint region. For Scaphoid joints, a simple patch model may not perform well because the joints have too much blend with the neighboring context.

2.5 Hand landmark detection

We need to develop algorithms to get accurate joint locations. Joint locations are necessary during JSN labeling. Doctors or radiologists need to know the exact

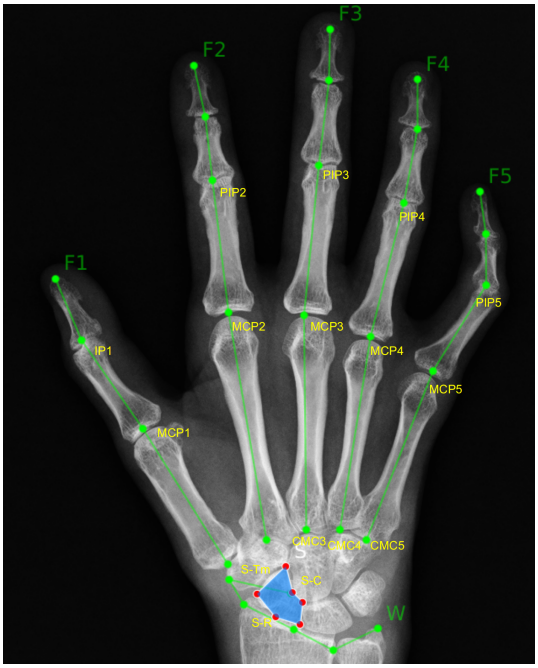

Figure 2.7: *Hand X-ray image joint key points with labels*

joint location, which should be provided to the labeling tool. Another usage of joint location is shown during joint patch extraction. We need to extract patches for all joints since the CNN model feeds on the individual joint patch. In this section, we develop algorithms to localize joints within the X-ray image. Joint locations are shown in Figure 2.7. During the labeling process, the tool shows the exact location for each joint. The user would label joint attributes one by one, following the navigation provided by the tool. We use the joint landmark detection algorithm to compute the joint locations before the labeling process.

Now we take a look at the landmark detection algorithm. While the CNN models have been applied to many tasks in recent years, it is not well suitable for geometric problems [24]. Although CNN based methods like Deeplab [25] can segment objects or generate object contours pretty well, the accurate key point is

different from object contour. We need to find accurate joint keypoint locations in our scenario. And we should take advantage of the hand X-ray image characteristics. Inferring keypoint location involves non-Euclidean space calculation, which can be better represented in the graph. Graph Convolutional Neural network (GCN) can capture the graph structure, node interactions, and object similarities [24]. So GCN can be used to refine joint keypoint locations. We use graph convolutional neural networks to regress and update graph nodes.

2.5.1 Curve-GCN working flow

Our landmark detection algorithm is based on Curve-GCN [4]. Curve-GCN uses cascaded GCNs to regress landmark locations in multiple steps. It first encodes each input image into a feature map. Then each landmarks feature is extracted at the coordinate on feature map by bi-linear interpolation. During each step of regression, it aims to predict coordinate shifts to be applied to landmarks locations. We do this by feeding landmark coordinates as well as their corresponding features into a GCN. Through multiple layers of graph convolutions, GCN outputs two scalar values, which are the predictions for the current step. After multiple steps of the same operation, landmarks gradually move to their target locations. Above process is depicted in Figure 2.8.

Landmark detection steps for Curve-GCN:

Step 1: label the joint key points on all X-ray images

Step 2: normalize the hand outline contour and get the overall shape mean

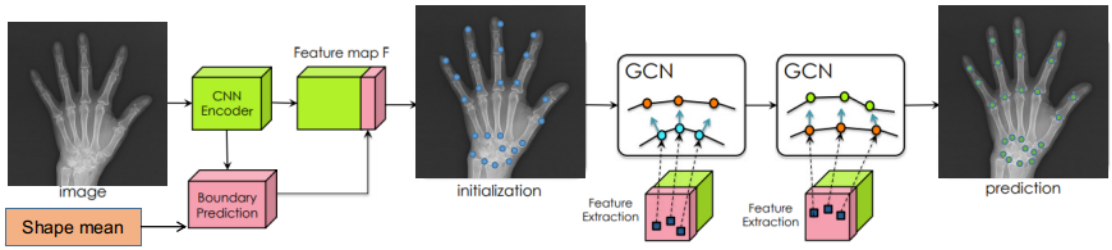

Figure 2.8: *Curve-GCN working flow. Shape mean is the average location of joints in all the data under the same boundary condition. Once we have the predicted boundary we would initialize joint locations using shape mean, and extract initial joint features. Then we use GCN to repeatedly update joint locations, joint features.*

Step 3: use CNN to generate convolutional features for the hand, then predict boundary and contour.

Step 4: normalize hand shape and coordinates using predicted contour in Step 3, then initialize the joint location with overall shape mean in Step 2.

Step 5: extract joint convolutional features with initial joint location

Step 6: feed joint location (from Step 4 or Step 7) and features (from Step 5 or Step 7) into graph convolutional neural network, and apply GCN operation to regress the location

Step 7: get new predicted joint location, then use this new location to get new corresponding feature.

Step 8: repeat 6 and 7 twice.

2.5.2 Curve-GCN model mechanism

Multi-layer GCN is used in our experiment. The graph propagation step for a node $\mathbf{cp}_i$ at layer l is expressed as:

$$f_i^{l+1} = w_0^l f_i^l + \sum_{cp_j \in N(cp_i)} w_1^l f_j^l \tag{2.1}$$

where $N(\mathbf{cp}_i)$ denotes the nodes that are connected to $\mathbf{cp}_i$ in the graph, and w_0^l, w_1^l are the weight matrices. The propagation step at layer l takes the following form:

$$r_i^l = ReLU(w_0^l f_i^l + \sum_{cp_j \in N(cp_i)} w_1^l f_j^l) \tag{2.2}$$

$$r_i^{l+1} = \tilde{w}_0^l r_i^l + \sum_{cp_j \in N(cp_i)} \tilde{w}_1^l r_j^l \tag{2.3}$$

$$f_i^{l+1} = ReLU(r_i^{l+1} f_i^l) \tag{2.4}$$

where w_0,w_1,$\tilde{w}_0$,$\tilde{w}_0$ are weight matrices for the residual. On top of the last GCN layer, the fully connected layer is applied to take the output feature and predict a relative location shift, $(\Delta x_i, \Delta y_i)$, for each node.

New node location becomes

$$[x_i^{'}, y_i^{'}] = [x_i + \Delta x_i, y_i + \Delta y_i] \tag{2.5}$$

Then new locations are used to re-extract features for the nodes, and another GCN predicts a new set of offsets using these features.

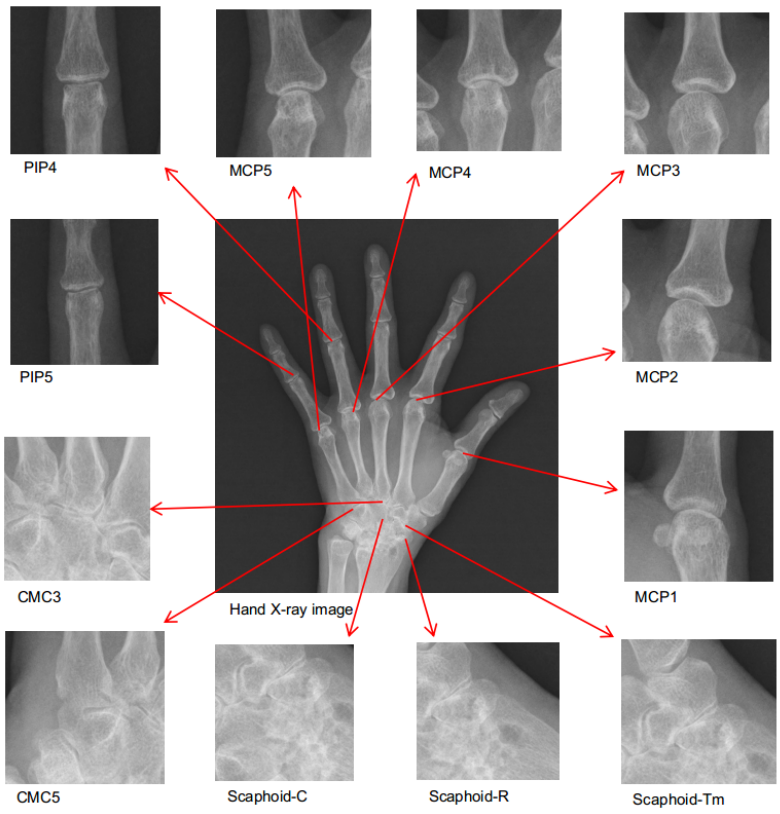

Figure 2.9: *Joint patches to extract*

2.6 Joint patch extraction

When the correct joint location is available from the landmark detection, we can extract the image patch for each joint. The original hand X-ray image is stored in the DICOM file, with both pixel and millimeter coordinates. The target patch for each joint is also stored in DICOM format, with both pixel and millimeter coordinates.

2.6.1 Extraction steps

After we get all the joint locations on the X-ray image, we use affine transformations to extract joint patches. Extracting patch for each joint involves coordinate transformations between local coordinate and global coordinate system. We get the mapping between target patch coordinate and source coordinate. The desired patch coordinates would be used to crop and interpolate from the original image. After patch image mapping, we create ground truth files. The ground truth includes the patch image path, ground-truth label.

The extraction process involves following steps:

Step 1: fill the target patch pixel coordinates with plain increasing x, y values

Step 2: transform the patch pixel coordinate into global millimeter coordinate, using DICOM image affine transformation matrix (associated with the original hand X-ray image)

Step 3: align the joint patch center with original hand image origin, translate to joint keypoint

Step 4: rotate the joint patch according to the joint orientation

Step 5: transform the joint patch coordinates from global millimeter coordinate back to pixel image (now the new coordinates correspond to joint pixel location inside the original hand image)

Step 6: fill joint patch image by interpolating from the original image in its pixel

coordinates

Through the above steps, we can get joint patches. There are multiple transformation steps doing matrix operation, which can be combined into one transforming matrix. The overall transformation matrix is calculated before mapping between the source patch and the target patch to save time.

2.6.2 Joint patch format

As the models work on individual joint patches, we need to define the patch shape. To contain enough context, the individual joint patch needs to cover an area bigger than its actual shape. The cropping size is (448, 448) in pixel and (35 mm, 35 mm) in millimeter coordinate. The joint is placed in the center of the patch image. Affine transformation in the global coordinate system use millimeter as the unit. The affine transformation between the global millimeter coordinate system and the local pixel coordinate system takes into consideration of mm unit to pixel unit ratio.

After patch extraction, we create DICOM files for 5 MCP joints, 5 PIP joints, 3 CMC joints, and 3 Scaphoid joints. These patches can be visualized as in Figure 2.9. DICOM images are then converted into PNG files, which could be fed to the neural network. Image names contain X-ray image id, joint name, joint index. We create the ground-truth information file for each type so that the program could find image patches accordingly.

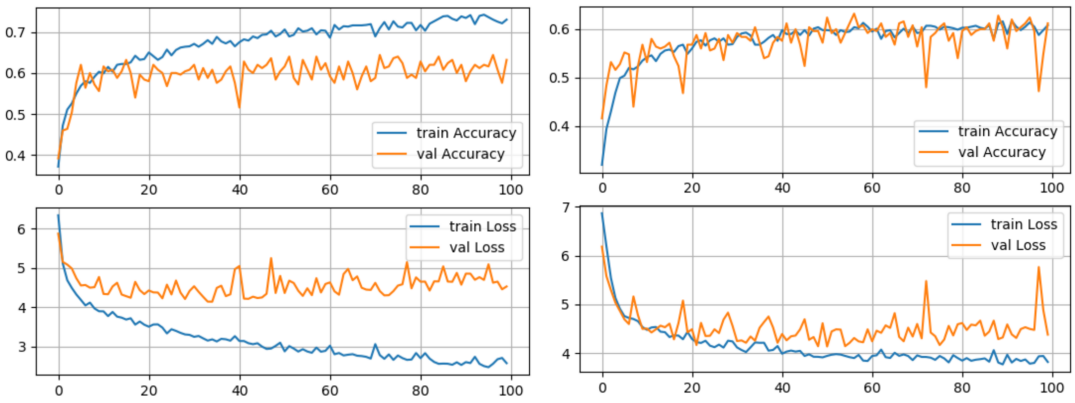

Figure 2.10: *Data augmentation effect on the training & validation curve for PIP patch model. Left figure exert small augmentation, while right one exert larger augmentation.*

2.7 Data augmentation effects

Data augmentation is widely used in convolutional neural network training. It is generally effective to expend data space. In most cases, data augmentation relieves the overfitting problem. Data augmentation also increases prediction accuracy sometimes. Discussion in this section is based on experiments in Chapter 4. However, the conclusions are applicable to all experiments in this thesis. The training curve in Figure 2.10 illustrates the data augmentation effect on the overfitting problem. The left figure is the model curve with less data augmentation, whose training accuracy keeps growing after 20 epochs while validation accuracy stays the same. The right curve has more data augmentation, and the training curves stay close to validation curves. It is clear that data augmentation mostly solves the overfitting problem.

Experiment results in Table 2.2 shows the performance-boosting effect of data augmentation. It shows the experiment results on Scaphoid patch models, using a

Table 2.2: Augmentation effect on the baseline model (Scaphoid joints)

Model type	Scaphoid Patch Squeezenet UniCls					
Model capacity	high		medium		low	
Augment level	low	high	low	high	low	high
parameter set 1	0.4896	**0.5301**	0.4915	**0.5368**	0.5112	**0.5258**
parameter set 2	0.4990	**0.5348**	0.5030	**0.5279**	0.5105	**0.5159**
parameter set 3	0.5012	**0.5117**	0.5181	**0.5341**	0.4971	**0.5239**
parameter set 4	0.5129	**0.5268**	0.5189	**0.5264**	0.5125	**0.5167**

unified classifier with Squeezenet backbone. For different model capacities and different hyper-parameter sets, higher-level augmentation would always help increasing model accuracy. When the augmentation level is higher, it performs better in all hyper-parameter sets, no matter what the model capacity is. So the proper data augmentation should be performed whenever possible. All experiments in the following chapters apply a similar amount of data augmentation.

2.8 Chapter summary

In this chapter, we first analyze the JSN score distribution. We see statistical connections among joints. Secondly, we inspect the joint appearances under different JSN scores. Thirdly we introduce the landmark detection algorithm. Then we explain how to extract joint patches. Lastly, we discussed the data augmentation strategy in this thesis.

Chapter 3: **Baseline Model**

3.1 Overview

In this chapter, we review convolutional neural networks and develop the baseline model. We investigate CNN backbones and training techniques. We also visualize the learned convolutional features to get more intuition.

In chapter 2, we have seen the data preparation steps. We learn how to locate joint key points, how to extract joint patches, how to make ground truth records. In this chapter, we focus on the CNN baseline model. The baseline model works on four types of joints, MCP, PIP, CMC, Scaphoid. To make the analysis more accessible, we visualize the prediction results on joint images. Through visualization, we see the model can make informed decisions when input joint image is not severely blurred.

We also give details about data augmentation operations to expend training space. Given more labeling data, the network could be improved further. The baseline model produces reasonably good predictions.

3.2 Convolutional neural network

The convolutional neural network (CNN) has been widely applied in computer vision tasks. CNN is the mainstream model for general image classification, such as ImageNet 1000 objects, MNIST 10 digits, CIFAR 10/100, and CUB 200 birds. The convolutional neural network is also the backbone of object detection models, such as Faster-RCNN [26]. Convolutional layers can hierarchically capture shape combinations. And many neural network designing techniques have ensured CNN capabilities.

When it comes to medical imaging tasks, the convolutional neural network is also widely adopted. For medical datasets, the number of classes is usually smaller compared to generic computer vision tasks. And the inter-class variability is not apparent compared to generic computer vision tasks. What's more, medical imaging tasks require the models to be explainable. So we are interested in knowing the function of each model layer. So the learned convolutional features should be meaningful.

The deep learning era has seen great numbers of convolutional neural network models. In recent years, Alexnet [10], VGG [12], GoogLeNet [14], ResNet [13] have revolutionized image classification tasks. These models share many similar functional components.

Now we look at the model components and working steps. There are two types of layers in a CNN model based on parameterization status. The parameterized layers include convolutional layers, fully connected layers. Non-parameterized

layers include activation layers, pooling layers, softmax layers, dropout layers, and reshaping layers. For classification tasks, the model input is images with one or three channels. Then data pass through convolutional layers. The representation at different granularity is generated along the way. Initial layers produce basic features, while final layers produce high-level features. The feature representation learned from convolutional layers is fed to the final classifier. And the fully connected layer converts features into predictions.

3.3 Data augmentation and affine transformation

One reason for the success of deep learning is the availability of large amounts of data. These datasets give the CNN models enough training materials in the learning space. However, in many medical tasks, we do not have enough labeled data. So it is essential to apply data augmentation to increase the quantity and diversity of training data.

Now we discuss the augmentation strategy. Data augmentation techniques include random cropping, horizontal flipping, intensity changing, color shifting, and affine transformations. Affine transformation covers rule-based pixel movements, which include translation, scaling, rotation, and shearing. Different datasets have different augmentation policies. The AutoAugment [27] uses reinforcement learning to find the optimal image transformation policy from the data itself. They argue that each data set has its characteristics, which results in a unique optimal augmentation policy. In this thesis, we experiment with several combinations of augmentations to

find the proper policy.

Although data augmentation operation is included in the Pytorch platform, we want to dig deep into the details. For the attention map model in Chapter 5, we need to keep track of the joint location changes during augmentation, because the same changes must be made on the attention maps. Knowing which transformation has been applied to the input image is necessary if we need to apply the same operation on the key points. We use affine matrix transformation for translation, rotation, scaling, which could keep records of augmentation details. Cropping and flipping can be achieved by matrix copying.

Affine transformations involve matrix operations. In the following paragraphs, we give examples of the affine transformation. Now we look at the notations. X represents the input image, and Y represents the output image. Y has the same shape as X. The transformation matrix from X to Y is A. Our purpose is to fill Y with the distorted X in our desired way, so we need to trace each pixel in Y back into X. So the goal here is to find the corresponding coordinate from Y to X. An affine transformation matrix can represent the correspondence relationship.

First, we look at scaling. Suppose we want to up-scale the 2D image by $scale\_x$ time. The original image would expend, and many parts in the transformed output is interpolated from the input image. The transformation matrix for scaling operation is:

$$M_{scale} = \begin{bmatrix} scale\_x & 0 & 0 \\ 0 & scale\_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

Second, we want to add linear translation $(trans\_x, trans\_y)$ on the original image. The transformation matrix for translation operation is:

$$M_{translate} = \begin{bmatrix} 1 & 0 & trans\_x \\ 0 & 1 & trans\_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

Thirdly, we want to rotate the original input by θ degree. The transformation matrix for rotation operation is:

$$M_{rotate} = \begin{bmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Above affine transformations can be combined into one: $M_{all} = M_{translate} * M_{scale} * M_{rotate},\ M_{all} * X = Y$.

Last, we apply the affine matrix on input data. For affine transformation, we don't operate on the image pixels directly. Instead, we use input image, output image pixel locations:

$$X = \begin{bmatrix} x_0 & x_1 & x_2 & x_3 & ... \\ y_0 & y_1 & y_2 & y_3 & ... \end{bmatrix},\ Y = \begin{bmatrix} x_0 & x_1 & x_2 & x_3 & ... \\ y_0 & y_1 & y_2 & y_3 & ... \end{bmatrix} \tag{3.4}$$

As target patch coordinates, Y is a known matrix (Y is coordinates of all pixels in the target image). Y's corresponding coordinate is in X. So we would use

$X = Inverse(M_{all}) * Y$ to get the locations. If the resulting X has pixel location out of the boundary, we would linearly interpolate these missing values. In this way, we get all the pixel correspondences. The image is augmented after applying the pixel mapping.

3.4 Network architecture

Convolutional Neural Network can capture the shape characteristics of different classes. There are many novel CNN architectures, such as Alexnet, VGG net, Resnet, and Squeezenet. For the baseline model, we have tried Squeezenet and Resnet18. Squeezenet could reach Alexnet-level performance on the ImageNet classification task while only uses a tiny amount of resources. Resnet is the state of the art backbone for CNN. But Resnet18 has obvious overfitting problems due to the lack of training data. So Squeezenet is the better option.

The convolutional neural network can recognize object shape patterns. For the joint space narrowing, the desired distinguishing patterns lie along the joint region. Convolutional layers are supposed to recognize the joint area pattern and extract features. After the features are generated, the fully connected layer can classify them into the JSN scores.

For the baseline model, we use Squeezenet [16] as the backbone. SqueezeNet is much lighter than Resnet18 and consumes less energy for training. SqueezeNet architecture achieves AlexNet-level accuracy on ImageNet with 50x fewer parameters. Since limited labeled data causes overfitting problem in large CNN models,

Squeezenet is the better choice.

Figure 3.1 shows the working flow of Squeezenet and the Fire module. There are three variants of the baseline model, distinguished by the number of convolutional layers (model capacity). The model input is the joint image patch. The Fire module [16] is comprised of a squeeze layer and an expand convolution layer. The inputs to the Fire module first pass through 1x1 filters, then go into an expand layer that has a mix of 1x1 and 3x3 convolution layers.

Squeezenet has a simple bypass mode and complex bypass mode. The bypass mode works similarly as Resnet short connection. Once the data scale goes up, we would use complex backbones (Resnet18) to replace vanilla Squeezenet. So we do not need to experiment with Squeezenet bypass mode for now.

3.5 Training and validation

Joint patches are extracted from the original X-ray image using algorithms in Chapter 2. Each joint has its dataset repository containing joint patches and ground truth. We experiment baseline model on four types of joints, MCP, PIP, CMC, Scaphoid.

Throughout the experiment, we use cross-validation. We split all available joint patches into 5 partitions. We use 5 folds to create 5 different "training set + validation set" combinations. We train separate models for each fold, and use averaged accuracy as the model performance.

During training and validation, the overfitting problem occurs frequently.

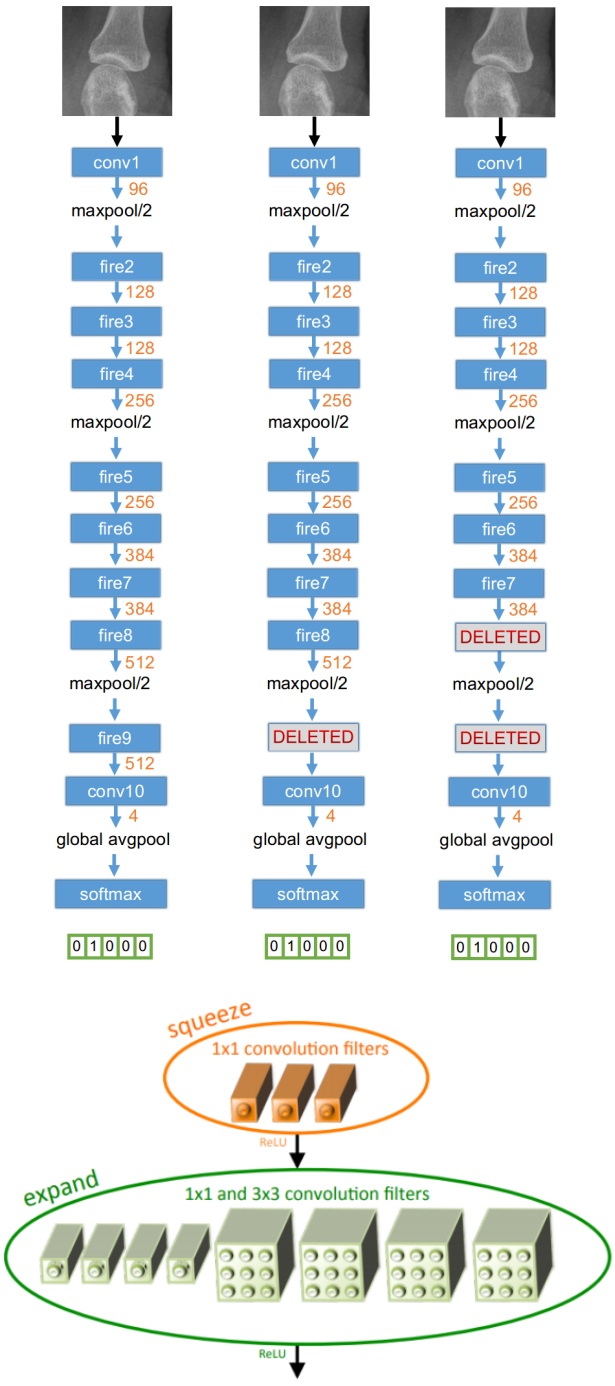

Figure 3.1: *Baseline patch model on the top, Squeezenet Fire module on the bottom. We list three baseline models in different model capacities.*

Overfitting problem can be circumvented or relieved in two ways. The first one is data augmentation, which can increase the training space. Although data augmentation could solve the overfitting problem, it does not necessarily mean high performance as we can see this from Figure 2.10. So a proper amount of augmentation would be enough.

The second way is to decrease model capacity (reduce trainable parameters). When the amount of parameters is more than necessary for learning invariance in the training data, the model tends to remember the training data. Mechanically remembering would result in poor learning processes and overfitting problems. Reducing the model capacity could force the model to use available parameters to fit the main patterns in data.

There are several ways to reduce model capacity, such as decreasing convolutional kernel size, reducing the convolutional layer channel dimension, and removing particular layers. We want to keep the model the same as the original Squeezenet to the most extent. So we remove last fire modules to reduce model capacity. In this way, all layers before removing location stay the same, which can load pre-trained weights directly. The baseline and its variations are in Figure 3.1. The leftmost model is original Squeezenet architecture, the middle one removes one Fire module from original, and the right one removes two Fire modules. We would train and test all three models on all the joints.

We train each patch model for 100 epochs, with a learning rate of 0.001. The models can achieve a decent performance before 50 epochs in most experiments. Sometimes the training would reach its top performance around 70 epochs, and

Table 3.1: Baseline model performance

Joint Type	Model Capacity	Baseline
MCP	high	0.618
MCP	medium	0.613
MCP	low	0.621
PIP	high	0.575
PIP	medium	0.568
PIP	low	0.555
CMC	high	0.538
CMC	medium	0.535
CMC	low	0.546

further training only leads to overfitting. The evaluation of models is based on the average accuracy of the last 25 epoch performance on the validation data set. We would average them as the fold model performance.

3.6 Model performance analysis

Model accuracy is calculated according to Equation 3.5.

$$Model\ accuracy = \frac{1}{5} \sum_{fold_i, i \in [1-5]} \frac{1}{25} \sum_{j=76}^{100} Epoch_j\ Validation\ accuracy \tag{3.5}$$

Baseline performance on MCP, PIP, CMC, Scaphoid is summarized in Table 3.1. For baseline experiments on MCP, model capacity does not have a certain effect. But for PIP, higher model capacity results in better accuracy. The reason is that PIP joints have more variations than MCP joints, so PIP models need more parameters. However, for the CMC joints, low model capacity brings the best

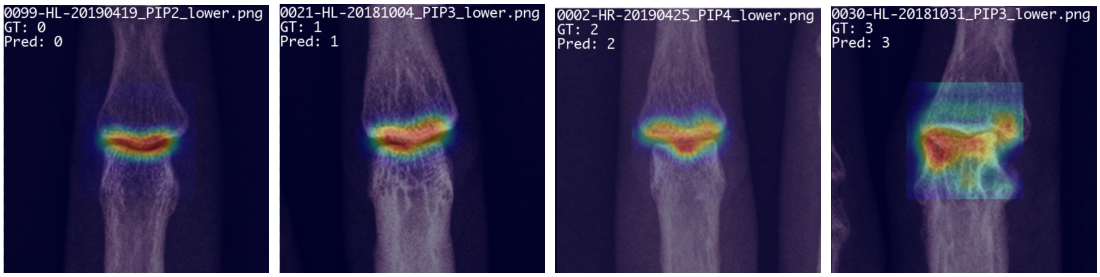

Figure 3.2: *Deep features of patch model, correct predictions for PIP JSN*

performance. This may occur because CMC joints only have $\frac{3}{5}$ the training samples compared to MCP or PIP joints. Less training data makes the model more prone to overfitting problems.

3.7 Visualization for CNN feature and prediction

Class Activation Mapping (CAM) [28] can produce generic localizable deep features. We use CAM visualization to understand the CNN discriminating basis.

First we look at the heat map for PIP JSN prediction in Figure 3.2. CAM visualizes the deep features after the last convolutional layer. We project the heat map on the input image, and we can see the model learn to focus on the joint region. The model learns to capture key parts for prediction during training.

We also check the heat map for the wrong predictions in Figure 3.3. Even for wrong prediction the feature are mapped on joint region.

3.8 Chapter summary

We develop the baseline model for JSN score prediction. The baseline model is based on Squeezenet architecture, and cross-validation is used during training.

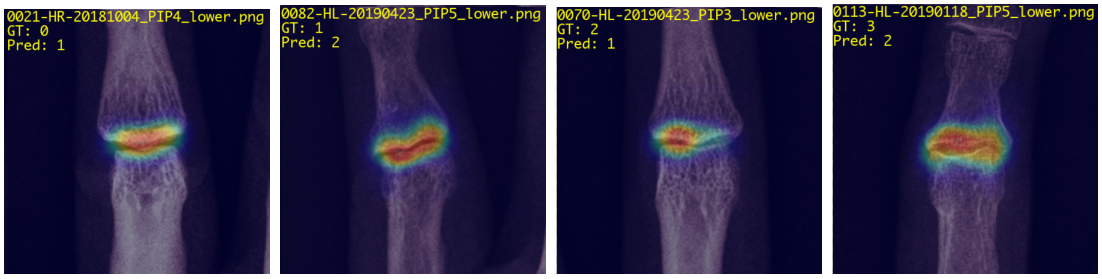

Figure 3.3: *Deep features of patch model, wrong predictions for PIP JSN*

We carry out the performance analysis for different types of joints. Feature visualization is done through Class Activation Mapping. Through visual inspection, we conclude that the baseline model successfully learns deep features. It could provide a reasonably good JSN prediction on MCP, PIP, CMC joints.

Chapter 4: **Fusion Model**

4.1 Overview

When it comes to RA joint space narrowing, hand joints on the same level (such as MCP2 and MCP3) tend to have similar scores. Joints of the same type or acting the same functions would have similar lesion condition because of the autoimmune disease nature. Based on the JSN score similarity observed in the data, aggregating information from the same type joints would help to predict the JSN score.

In this chapter, we introduce the fusion model, which provides the information-sharing mechanism for the same type of joints. We investigate the performance influence of the fusion module capacity. We also compare the performances between baseline and fusion models.

We visualized the results of the prediction using the confusion matrix. We analyze the correct prediction as well as the failure cases. The fusion model can give better predictions than the baseline model.

4.2 Statistical foundation for fusion model

From the biological point of view, the RA disease would have similar effects on neighboring joints. It is an autoimmune body disease where the immune messengers are carried in the blood. Once the chemical substances reach a certain level, all neighboring joints would suffer to a similar extent. RA symptom differences on joints could be attributed to joint size, moving frequencies, and joint shape. Smaller joints would show symptoms first because of their size.

From a statistics point of view, we make several observations. First, for the same type joints, JSN scores are highly correlated. The confusion matrix for same type joints correlation is in Figure 2.1. When there are high percent entries on the confusion matrix diagonals or near the diagonal, we say the two comparing items are highly correlated. From JSN dependency of neighboring MCP joints, we can see a clear correlation between MCP1 and MCP2, between MCP2 and MCP3. Similar patterns can be seen between PIP3 and PIP4, between CMC4 and CMC5.

Next, we give more explanation for the fusion model. For example, if we already know 3 MCP joints have severe JSN symptoms, then the remaining two MCP joints are more likely to have a high JSN score. In clinical diagnosis, doctors would use a similar correlation to make prudent judgments. Incorporating the information-sharing mechanism in the network ensures the potential to reach or surpass human-level judgment. The fusion model derives from the baseline model but combines information of the same type joints together. The fusion module would share information and facilitate the individual JSN prediction.

4.3 Fusion model design

Taskonomy [29] apply multitask learning and prove that there are structure bonds among different tasks. Sharing training experience would improve every model in the end. In our experiments, classifying individual joint is a single task, and the fusion module provides the sharing vehicle for common structure. We could use less training data to get a better model.

We show the fusion model architecture in Figure 4.1. The feature extraction part is the same as the baseline model. The fusion model has an additional fusion module. The input is five joints of the same type, and they pass through feature extraction layers individually. Five features are then fused. And another convolutional layer is used for further feature integration. In the end, the integrated fusion feature is concatenated to each of the five features as classification evidence. These fused features share one same fully connected layer in Figure 4.1. But separate classifiers can be used to exploit different joint characteristics further.

4.4 Experiments

Different models prefer different blends of hyper-parameters. Neural network search [30] shows the benefits of good hyperparameters. Neural architecture search is categorized into three dimensions, search space, search strategy, and performance evaluation metric. For our task, the search space is limited to model capacity, learning rate, dropout, weight decay. While typical NAS methods such as EfficientNet

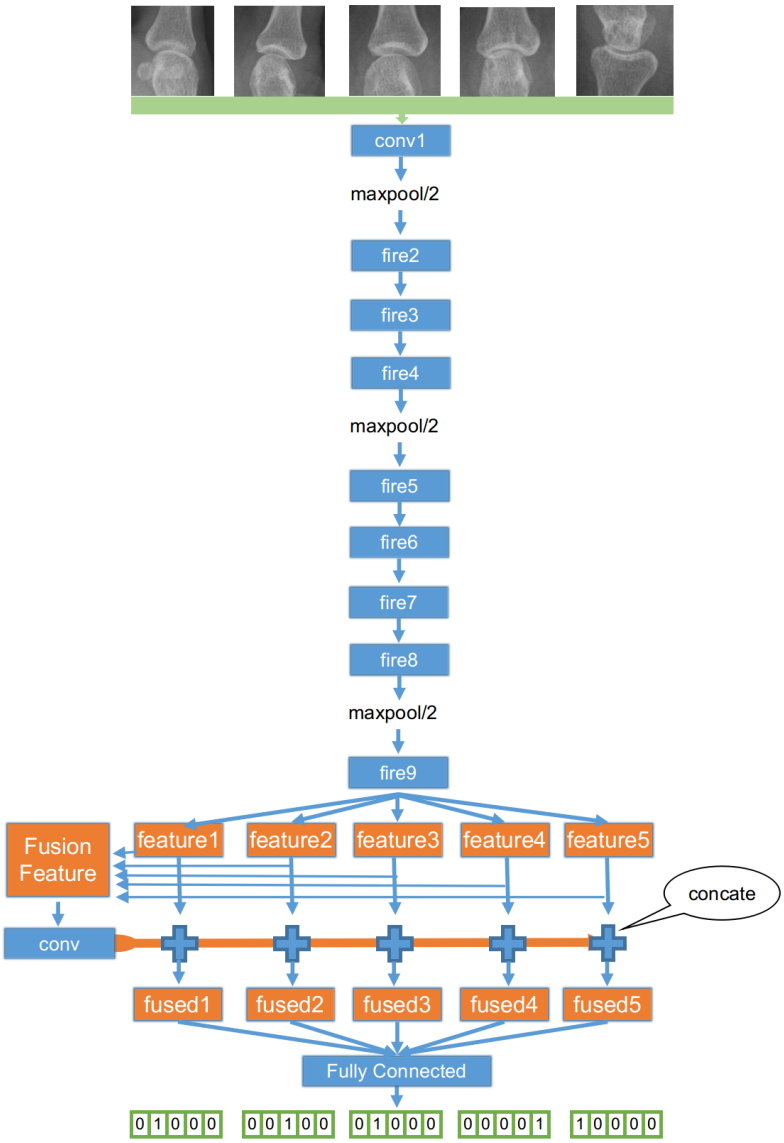

Figure 4.1: *Fusion model based on Squeezenet*

[31] optimize for both accuracy and FLOPS, medical imaging applications care more about model accuracy.

CMC and Scaphoid joints have a more complex context. Their inter-joint co-relationships are different from MCP and PIP. We use the attention map model to work on Scaphoid joints in Chapter 5. In this chapter, the fusion model experiments on MCP, PIP. The fusion model increase about 0.7% on MCP joints, and 0.2% on PIP joints.

$$Model\ accuracy = \frac{1}{5} \sum_{fold_i, i \in [1-5]} \frac{1}{25} \sum_{j=76}^{100} Epoch_j\ Validation\ accuracy \tag{4.1}$$

Cross-validation is applied throughout experiments. Patients data is split into 5 different partitions. There are five folds. Each fold contains 4 partitions as the training set and 1 partition as the validation set. We use the average validation accuracy of the last 25 epoch as the single-fold model performance. Then we average 5 single-fold model performance as the final model performance. The calculation process is illustrated in Equation 4.1.

4.5 Performance analysis

We visualize the model prediction accuracy in each fold in Figure 4.2. Each line connects all performance in 5 folds as well as the final averaged performance. The baseline model is named net0, while the normal fusion model is named net1, and the heavier fusion model is named net2. RM1 means removing one Fire module, RM2 means removing 2 Fire modules, while RM0 keeps original capacity. Aug2

Table 4.1: Patch model performance, baseline versus fusion net.

Joint Type	Model Capacity	baseline	fusion 1	fusion 2
MCP	high	0.618	**0.620**	0.608
MCP	medium	0.613	**0.621**	0.611
MCP	low	0.621	**0.628**	0.617
PIP	high	0.575	**0.578**	0.575
PIP	medium	0.568	0.568	**0.575**
PIP	low	0.555	0.549	**0.560**
CMC	high	0.538	0.527	**0.542**
CMC	medium	0.535	**0.542**	0.535
CMC	low	**0.546**	0.545	0.535

means a regular amount of data augmentation. WD0.05 means weight decay is 0.05. Drop0.5 means dropout is 0.5. For each fold, the top model accuracy is dotted in color.

Figure 4.2 shows that fusion model outperforms baseline model in both MCP and PIP tasks. Although the margin is not impressive, a larger amount of training data could make a difference. Lack of labeled data hinders the full power of the fusion model. In both figures, we can see big performance variations among different folds. This is due to a lack of training data. All the models currently suffer from the overfitting problem.

Table 4.1 lists the comparison between the baseline model and fusion models. Fusion 1 has a light fusion module, while fusion 2 has a heavier one. Model capacity corresponds to the number of convolutional layers. The high capacity model keeps all Squeezenet Fire module, medium-capacity removes the last Fire module, while the low capacity model removes the last two Fire modules.

From Table 4.1, we can make several conclusions. For MCP joints, normal

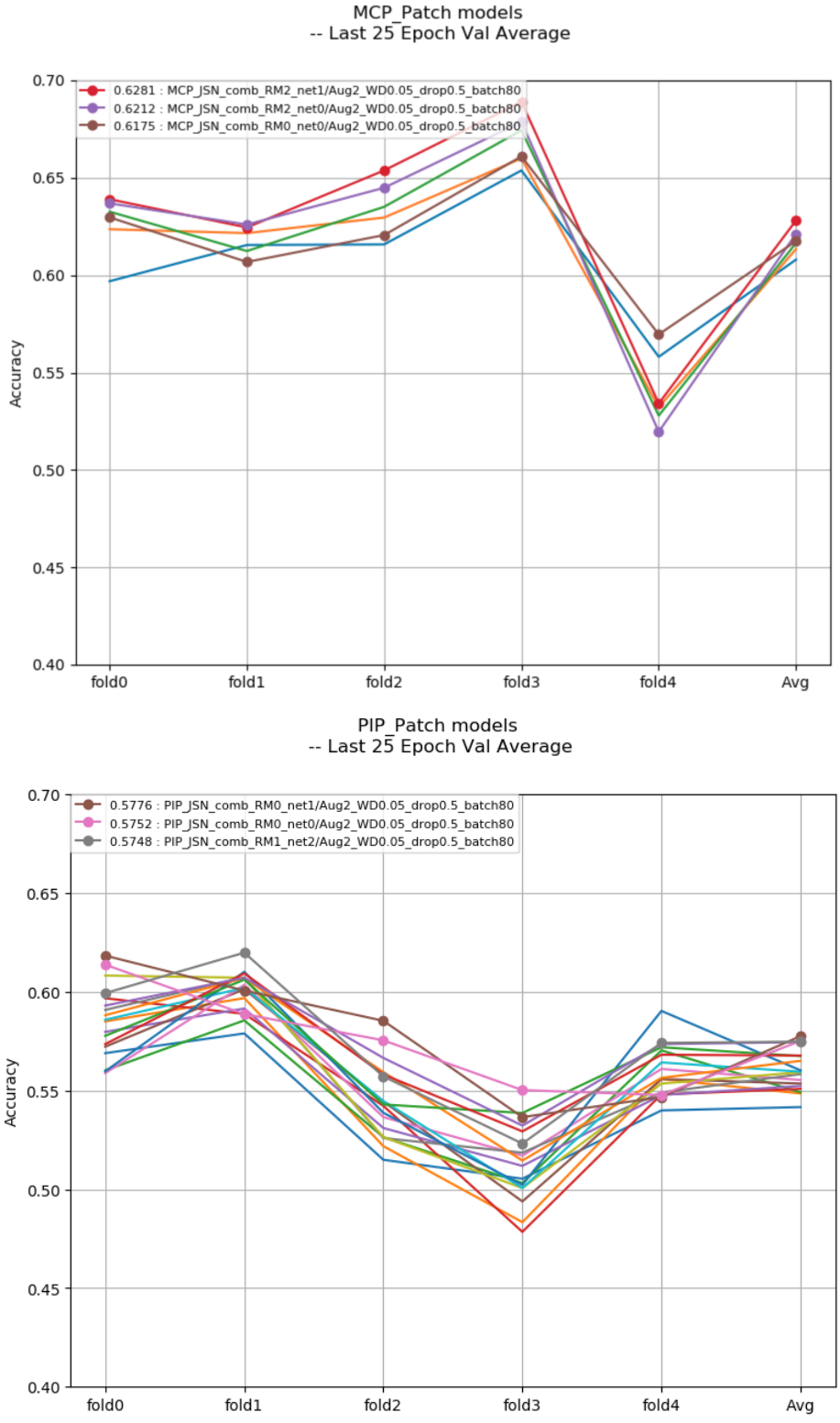

Figure 4.2: *This figure shows patch model performance on MCP and PIP joints.*

fusion (fusion 1) works best. For the PIP JSN task, more fusion (fusion 2) produces better results. For CMC patch models, fusion architecture does not do better. This may result from the complex surroundings of CMC joints. CMC joint areas are closely connected, thus reducing the power of information sharing. Further improvement on joints of complex context is made in the attention map model.

4.6 Comparison between prediction and ground truth

We experiment with the fusion model on four types of joints. Now we inspect the prediction results. These four confusion matrix in Figure 4.3 and Figure 4.4 represent the average performance of fusion models on JSN tasks. We split the dataset into 5 folds, and we combine the statistics of validation data in each fold to form the confusion matrix. As there are few GT=3 scores, we combine the result 3 and 4 as one. Each column of the confusion matrix represents one ground truth JSN score, and prediction results scatter in the column. So each column adds up to 1.0.

4.6.1 Prediction on MCP and PIP joints

For the MCP confusion matrix on the left of Figure 4.3, more than half entries fall along the diagonal. This means the patch model performs well on all JSN levels. Particularly, for score 0 and score 3, there are 73.8% and 71% correct entries. For those not on the diagonal, shorter distance to diagonal indicates being closer to correct prediction. For MCP ground truth label 2, 50.2% are fully correct. 37.2%

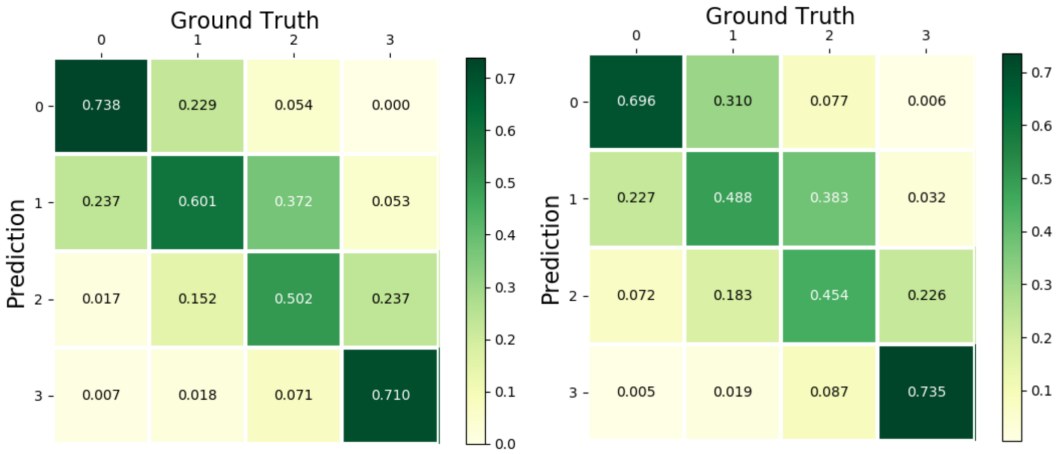

Figure 4.3: *Confusion matrix, left: MCP, right: PIP*

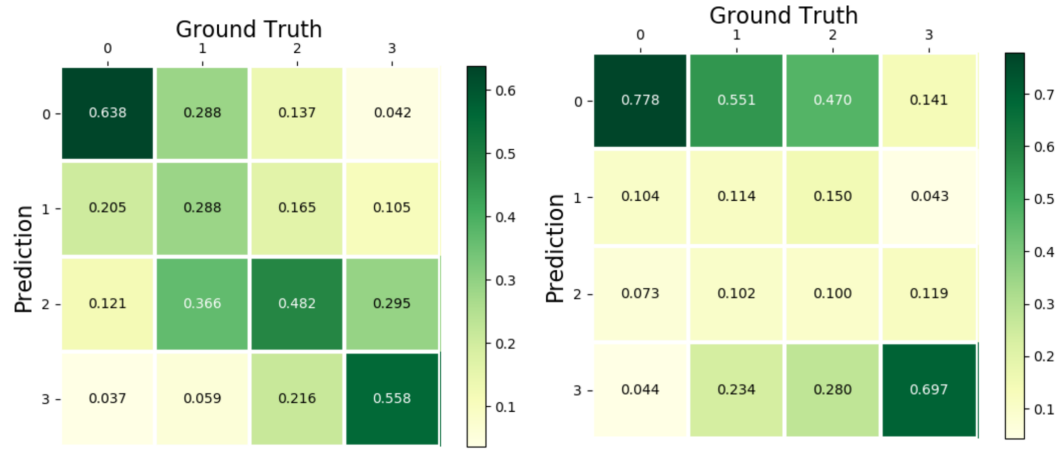

Figure 4.4: *Confusion matrix, left: CMC, right: Scaphoid*

predicting result is label 1, which is not far away from the ground truth. Considering there is no gold standard for the JSN score, near-diagonal entries may not deviate too far away from the true disease condition.

For the PIP confusion matrix, the patterns are similar to MCP's. The difference is that PIP is less intensive on the diagonal. But overall, most entries do not deviate too far away from the diagonal.

4.6.2 Prediction on CMC and Scaphoid joints

For the CMC task, model performance on inputs of GT label 0 and inputs of GT label 3 is acceptable. But when ground truth is 1 or 2, the predictions scatter so much, which indicates little model predicting ability. This poor performance may result from the ambiguous labeling and complex context with CMC joints.

For the Scaphoid patch model, the confusion matrix pattern is different from all the other three. The model tries to output two extremes, 0 and 3, as results for most samples. For GT=0 and GT=3, this scheme works well. But for GT=1, GT=2, the predictions are irrelevant. This may happen when the benefit of distinguishing GT=1, GT=2, is too small during training, and the model is inclined to securing GT=0, GT=3. This assumption finds clues in Scaphoid JSN score distribution. There are less Scaphoid scores ((169+102)/774 = 48%) concentrating at GT=1 and GT=2. By comparison, MCP JSN has ((508+246)/1195 = 63%) at GT=1 and GT=2.

The fusion matrix 4.4 provides insight for improvement direction. Although the model training process already considers the unbalance among different scores by involving weighed loss updating mechanism, naively predicting extremes still happens. For GT=3, more prediction goes to score 0 (14.1%) than score 2 (11.9%). Most of the time, GT=3 means severe RA symptoms, and it is supposed to be labeled as some level of abnormality. However, for GT=3 inputs, the model only outputs 3 when the lesion for input image looks extremely bad. Otherwise, it would predict more score 0 than score 2. This observation informs us how to tune our

model in the future. We should design the model to avoid extreme binary outputs.

4.7 Classifier mode: unified versus separate

Now let's look at the influence of classifier modes in Figure 4.5. There are two classifier modes, separate classifiers, and unified classifier. Theoretically speaking, given enough labeled data, the separate mode would learn to fit pattern details better. But for patch model on Scaphoid joints, unified classifier outperforms separate classifier models by a large margin. One cause is the lack of training data. The unified classifier has three times of training exposure. This would happen as long as different Scaphoid joints have some similarities which could regularize the model.

4.8 Chapter summary

In this chapter, we have discussed the statistical connection within the same type joints and introduced the fusion model. The information-sharing module provides a communication channel between joint features. We gain insights from the prediction confusion matrix. We also analyze the performances using different backbones, classifier modes. The fusion model outperforms the baseline patch model in Chapter 2 on MCP and PIP JSN tasks.

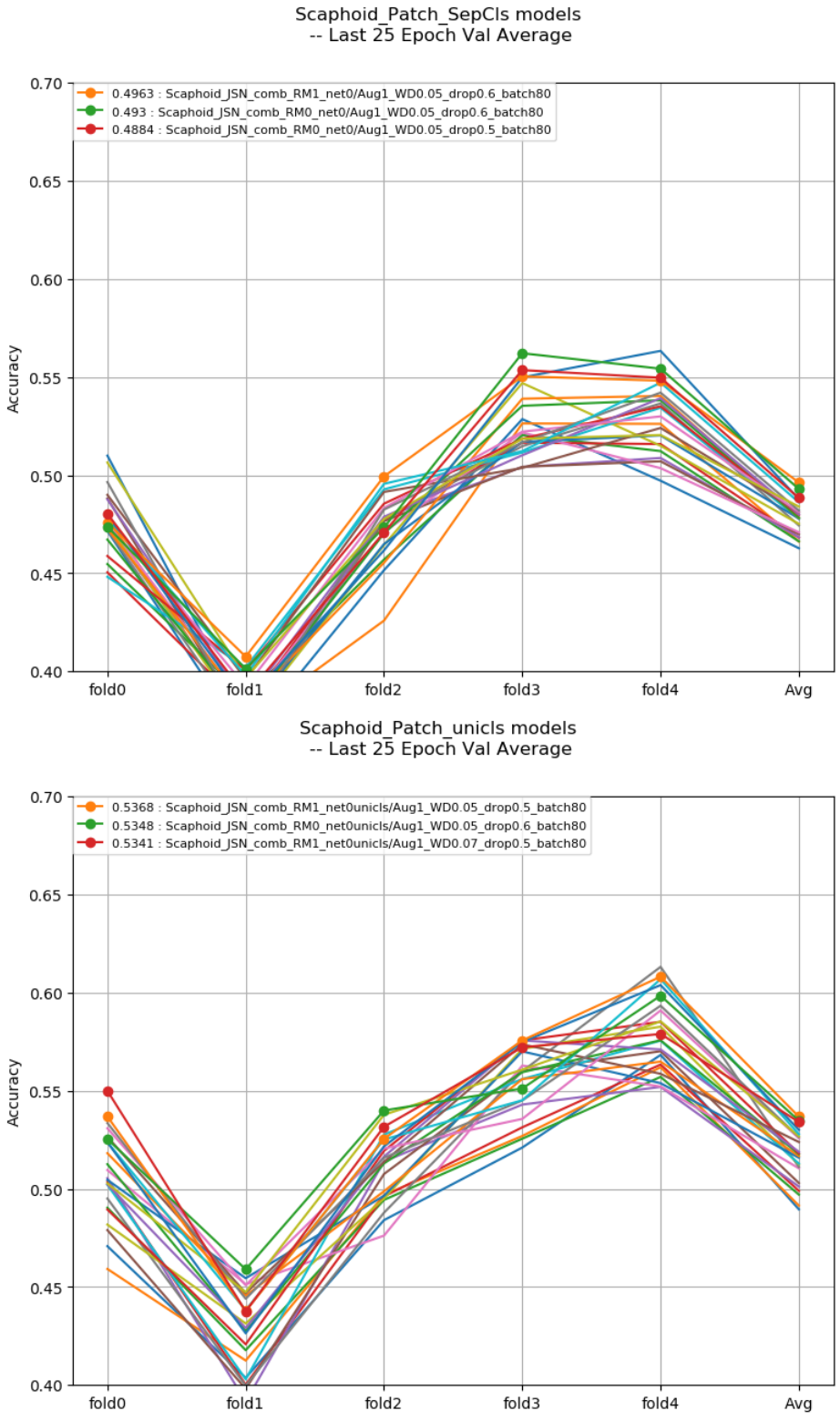

Figure 4.5: *Separate classifier models performance on the left, unified classifier models performance on the right. For patch models, unified classifier is about 5% better than separate classifier.*

Chapter 5: **Attention Model**

5.1 Overview

For CMC and Scaphoid joints, their complex contexts introduce too much noise. One way to separate critical information from irrelevant surroundings is to use the attention mechanism. Attention mechanism has been successfully applied to neural machine translation [32] and image captioning [33]. It directs the network to focus on the most critical information. SCA-CNN [34] incorporates both spatial and channel-wise attention to facilitate image caption.

After getting joint landmarks from Chapter 2, there are two ways to apply attention. First, the Point of Interest (POI) model extracts feature vectors for each key point. Second, the attention map can operate on the intermediate CNN tensors to filter out noise. The attention map works on one layer and keeps other model parts unchanged.

For the attention map model, we experiment with two different backbones. Resnet18 would outperform Squeezenet in most cases. We investigate the effects of the classifier modes. The classifier mode has different effects on Squeezenet and Resnet18. Then we show the attention map shape influence on performance. A good attention map shape not only lets in enough key information but also shuts

off unnecessary context.

We carry out extensive attention map model experiments on Scaphoid joints. We conclude that the attention map method outperforms patch models on complex joints. As this is the last chapter, we provide several future working directions at the end.

5.2 Attention creation

We first look at how to create attention for each joint. Joint is an area, so it is desired to represent a joint by a line of points instead of one central point. So we need to know the boundary points for each joint. Although we have hand landmarks from Chapter 2, they are the central points, and they are not enough to specify the joint area.

For CMC joints, the landmark detection gives us central locations of CMC3, CMC4, and CMC5. CMC3, CMC4, CMC5 would use their landmarks as anchors. Since the CMC joints distribute in a linear form, we interpolate linearly from CMC3, CMC4, CMC5 to get other intermediates and boundary points.

For Scaphoid joints, there is no crucial point from the landmark detection phase. The current landmark detection algorithm is not trained on S-Tm, S-R, S-C joints. We get the key points location for three Scaphoid joints (Scaphoid-Tm, Scaphoid-C, Scaphoid-R) manually. Each Scaphoid joint has 3 marks, and nine key points are labeled totally.

The first way of attention mechanism is to extract features directly based on

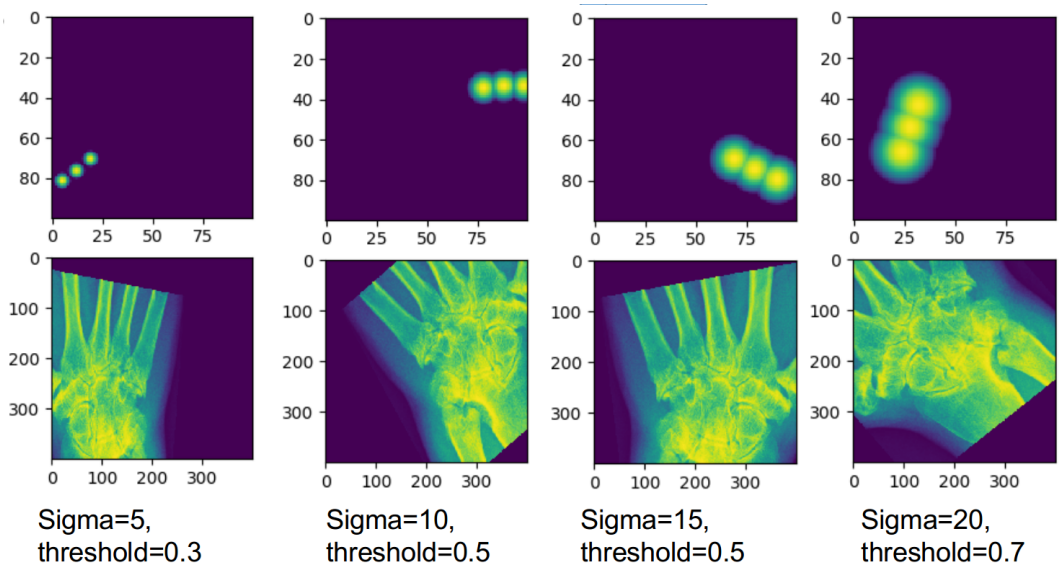

Figure 5.1: *Attention map with Gaussian parameters.*

the anchor points of interest (POI). POI is similar to Region Of Interest feature extraction in Faster-RCNN [26], but we get feature vector for only one point instead of an area. Point Of Interest assumes that the individual feature vector contains the necessary classification information.

The second way is to use the attention map to filter out irrelevant regions far from the joint area. Attention maps of different Gaussian parameters are visualized in Figure 5.1. The attention map is created by 2D Gaussian distribution, and we can change sigma value for a different shape. The threshold adjusts weights for locations away from the joint area. Gaussian distribution would be wider when Sigma is bigger. Smaller threshold lets through less context.

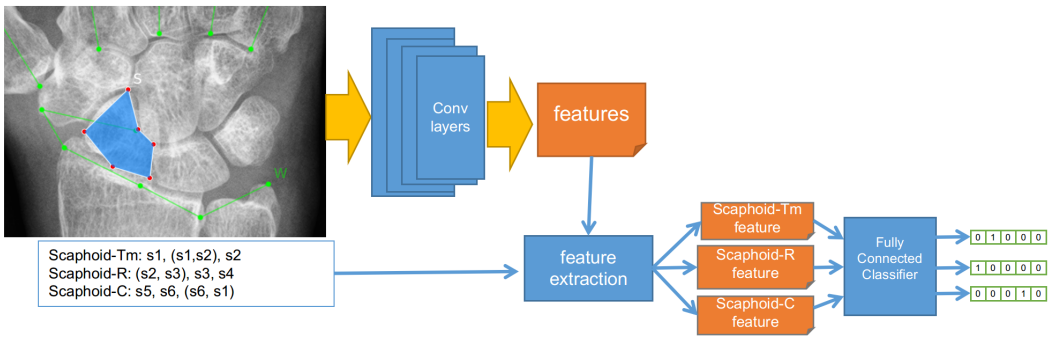

Figure 5.2: *Point of interest extraction model working flow*

5.3 Point of Interest attention model

Figure 5.2 demonstrates Point Of Interest feature extraction working flow. It uses extracted feature vector as classification evidence. It assumes that the receptive field of these points covers the joint area. Its input is the wrist image, with a resolution of 600×600. The input goes through convolutional layers to get the feature map and stops at the extraction location. The tensor size at the extraction location is $N \times 256 \times 100 \times 100$. For Scaphoid joints, nine feature vectors are extracted for three joints.

The extraction layer is where we extract the features. Before the extraction layer, it has gone through enough convolution operations to aggregate useful information. The receptive field covers all the relevant context. The spacial pixels on the extraction layer should be distinguishable for original regions. If the field of view for the extraction point is too large, then different areas get mixed and lost identity in the feature map.

The feature extraction module gets nine feature vectors, corresponding to nine

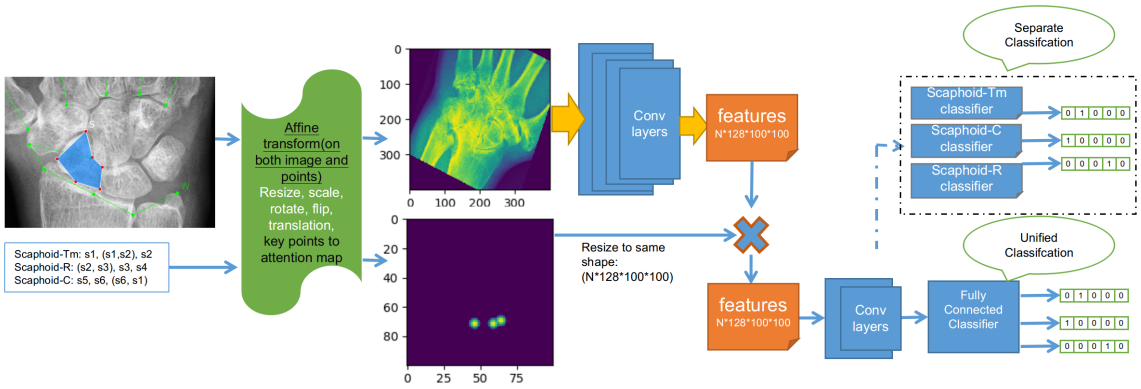

Figure 5.3: *Attention map model working flow*

attention points 3 for the Scaphoid-R joint, 3 for the Scaphoid-Tm joint, and 3 for the Scaphoid-C joint. Then the classifier concatenates these vectors as joint features and outputs the predictions. After several POI experiments, we conclude that the POI method could not be effectively trained. The training accuracy gets stuck around 42%, while validation accuracy gets stuck around 38%. The remaining chapter is for the attention map model.

5.4 Attention map model

The attention map model in Figure 5.3 depicts its working flow. The inputs are wrist image and attention points. The attention map is used as weights during the filtering process. The input resolution is also 600×600, and it goes through convolutional layers to get the feature map. The tensor size at the filtering location is $N \times 256 \times 100 \times 100$.

We do element-wise multiplication at the filtering convolutional layer to apply the attention map. The attention map is first expended to the same shape as

the tensor feature. After attention filtering, the tensor continues going through the remaining convolutional layers. There are two types of classifiers, separate classifier, and unified classifier. The advantages and disadvantages are discussed in the performance section.

There are two backbones. Squeezenet [16] provides simplified CNN architecture consuming less computational power. Resnet [13] provides residual connections between convolutional blocks and can capture complex shapes. Resnet allows for further improvement, given more training data.

5.5 Experiments

For the point of interest (POI) feature extraction results, the model couldn't learn useful information. The training accuracy starts from below 15% and goes up during the first 20 epochs. Then it gets stuck around 42%. The model gets the local minimum and could not jump out. The local minimum would occur when the model's best guessing is better than the best possible learning result. The joint feature vector may not contain all the necessary information for predicting the JSN score.

For the attention map model, we carry out experiments comparing different settings. We try to search for the best model setting. The search process gets some inspiration from the neural architecture search (NAS) [31]. The NAS lists setting dimensions of network width (or channels), depth, resolution. Our search for the best attention model setting is a little different. The network depth is altered

by removing convolutional layers at the end of the network. The network width adjustment is to change the fusion width discussed in Chapter 4. The resolution is adjusted at the input entrance. Another adjustable component is the attention map shape. We compare different Gaussian map shapes and make a detailed analysis.

Here we give some basic experiment settings. We carry out experiments for Scaphoid joints. Scaphoid joints are more complex than CMC joints. Two types of network backbones are Squeezenet and Resnet18. Two classifier modes are unified and separate. We stick to the principle of cross-validation and use 5 folds to train/validate the model. The fold accuracy is averaged from the last 25 epoch validation accuracy. The best model selects highest among 5 folds for all experiments, while the best average select 5-fold average as model selection standard.

5.6 Performance comparison and analysis

As the attention model can be designed in varied ways, we experiment with different combinations of Gaussian sigma value, map background threshold, backbones, and classifier modes. For the Gaussian sigma, we can either make the Gaussian distribution concentrate on key points by using small sigma values or disperse the attention in a wider region by using high sigma values. For the background threshold, we can either let in more context by setting the threshold to be a large value or prevent neighboring information by setting the threshold to be a small value. For the convolutional backbone, we can either use a light one (Squeezenet), or a heavy one with residual blocks (Resnet18). For classifier modes, we can choose

from the unified classifier and separate classifier. We summarize the performances in Table 5.1.

5.6.1 Backbone influence on performance

In Table 5.1, the Resnet backboned models outperform Squeesenet backboned models. For Squeezenet models, unified classifier outperforms separate classifier by a big margin, while it's not the case for Resnet18 models.

To get the agreement between prediction and ground truth labels, we calculate the Cohen-Kappa score [35]. The Cohen-Kappa score is commonly used in radiographic interpretations since the diagnosis often involves subjective interpretations by observers. Two radiologists or annotators give two sets of results, and we use the Cohen-Kappa score to see how consistent they are. Cohen-Kappa's score considers that two annotators may agree or disagree simply by chance. Kappa score of 1 indicates perfect agreement, while Kappa score of 0 means equivalent to chance. Kappa score is calculated from Equation 5.1, where p_o is the observed agreement ratio, and p_e is the expected agreement when both annotators assign labels randomly. p_e is estimated using a per-annotator empirical prior over the class labels.

$$K = \frac{p_o - p_e}{1 - p_e} \tag{5.1}$$

Table 5.1: Attention map model performance on scaphoid JSN

Attention model type	best model	best average	best kappa
SqueezenetSepCls	0.607	0.543	0.356
SqueezenetUniCls	0.647	0.565	0.374
Resnet18SepCls	0.65	0.58	0.398
Resnet18UniCls	0.64	0.575	0.386

5.6.2 Classifier mode influence on performance

There are two classifier modes, unified classifier, and separate classifier. The unified classifier assumes that features for different joints have a similar distribution. It has more training data. While the separate classifier assumes feature vectors for different joints have distinct distribution. And this bias could not be eliminated by the fully connected layers. The number of separate classifiers depends on the number of joints. For the Scaphoid task, the separate classifier only has one third training sample compared to the unified classifier. For limited training data, the unified classifier would outperform separate classifiers if the bias is not big enough to counter the benefit of tripling the training data. If we have an unlimited amount of labeled data, it is reasonable to see better performance in separate classifiers.

Now we take a closer look at Squeezenet model performance under different classifier modes. Figure 5.4 list experiment accuracy for Scaphoid attention model using Squeezenet as backbone. The upper one is for separate classifiers, while the lower one is for unified classifiers. The unified classifier outperforms the separate classifier. Particularly, their accuracy distributions on different folds are different. This means the model using limited training data is sensitive to data pattern

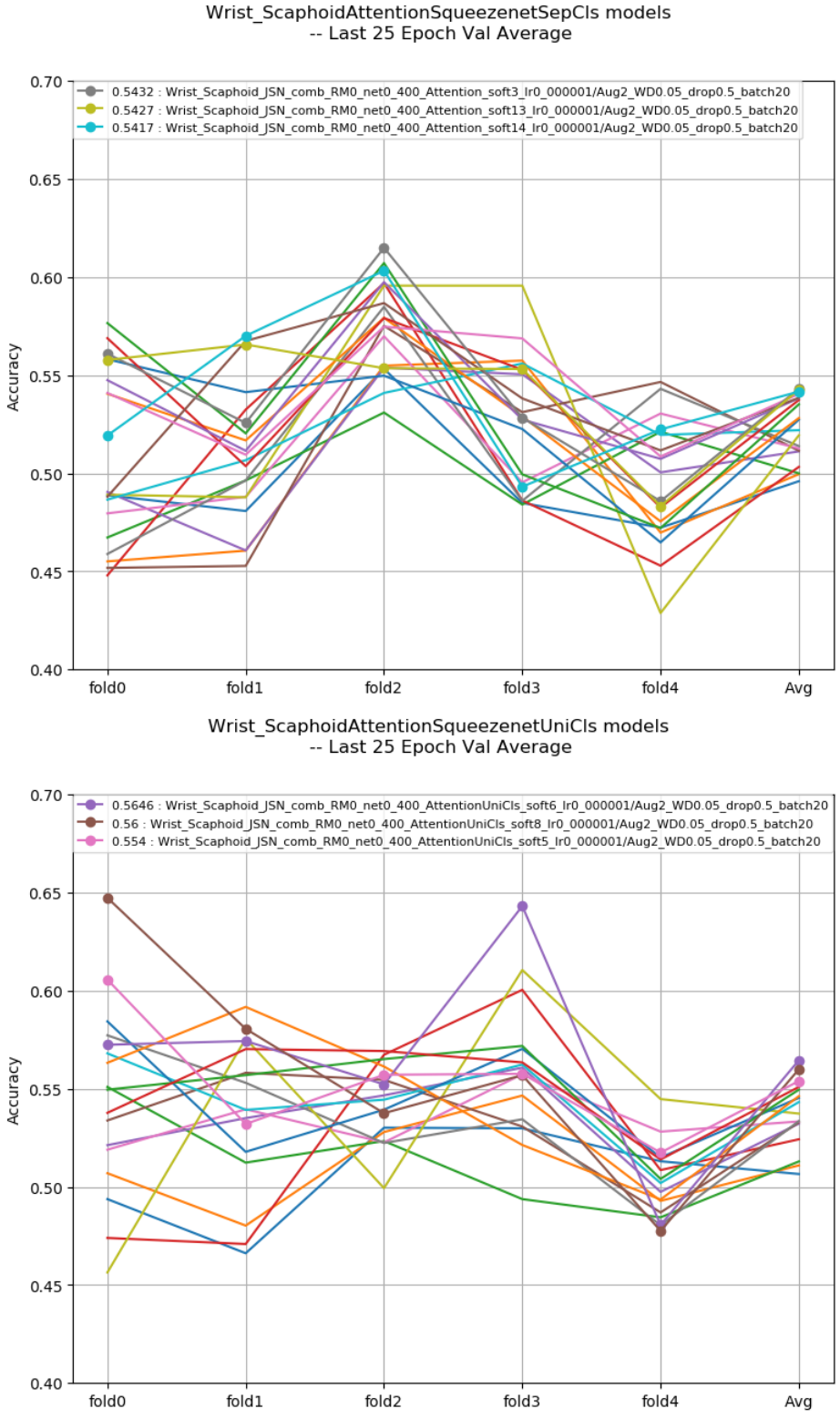

Figure 5.4: *Squeezenet backbone, separate classifier and unified classifier comparison*

changes.

5.6.3 Attention map shape influence on performance

We experiment with different combinations of Gaussian sigma value, map background threshold, backbones, classifier modes. We conduct experiments with variation from above factors, and summarize results in Table 5.2. In the vertical direction, we use soft0, soft1,..., soft20 to represent different combinations of Gaussian sigma and attention map threshold values. In horizontal direction are the network architecture variations. We highlight the top three performances in each network architecture.

As we can see from Table 5.2, Resnet18 Unified Classifier attention map model perform best on Scaphoid joints. Resnet18 backboned models consistently outperform Squeezenet backboned models. To see sigma influence on model performance, we aggregate different thresholds together and compare sigma values in Table 5.3. It summarizes statistics along with different sigma values, getting averaged performance under all threshold values. From the table, we can conclude, sigma=10 would perform the best.

To see the threshold influence on model performance, we aggregate different sigma together and compare threshold value in Table 5.4. It summarizes statistics along with different thresholds, getting averaged performance under all sigma values. From the table, we can conclude that threshold=0 or threshold=1 gives poor performance generally. For Resnet18 separate models, threshold=0.3 gives the best

Table 5.2: Attention map shape influence on performance

Attention index	sigma	threshold	Squeezenet UniCls	Squeezenet SepCls	Resnet18 UniCls	Resnet18 SepCls
soft0	5	1	0.5243	0.4996	0.5415	0.5182
soft1	5	0.7	0.5433	0.5383	0.5468	0.5322
soft2	10	0.7	0.5463	0.5194	**0.5746**	0.5491
soft3	15	0.7	0.5454	**0.5432**	0.5741	0.5393
soft4	20	0.7	0.5374	0.5385	**0.5754**	0.5347
soft5	5	0.5	**0.554**	0.5374	0.5706	0.5451
soft6	10	0.5	**0.5646**	0.5274	0.5701	0.5569
soft7	15	0.5	0.551	0.5112	0.5723	0.5385
soft8	20	0.5	**0.56**	0.5352	0.5737	0.5472
soft9	5	0.3	0.5496	0.5405	0.5745	**0.5801**
soft10	10	0.3	0.5329	0.5383	0.5591	**0.5751**
soft11	15	0.3	0.511	0.5282	0.5456	0.5523
soft12	20	0.3	0.5336	0.5127	0.5585	0.5459
soft13	5	0.1	0.5334	**0.5427**	0.5715	0.569
soft14	10	0.1	0.5227	**0.5417**	**0.586**	**0.5728**
soft15	15	0.1	0.5323	0.522	0.5402	0.5551
soft16	20	0.1	0.5481	0.5115	0.552	0.5454
soft17	5	0	0.513	0.496	0.5236	0.5526
soft18	10	0	0.5225	0.5139	0.5673	0.5502
soft19	15	0	0.5066	0.5	0.5523	0.529
soft20	20	0	0.5245	0.5033	0.5666	0.5406
Best	[5-20]	[0-1]	0.5646	0.5432	0.586	0.5801
Avg	[5-20]	[0-1]	0.536	0.5239	0.5617	0.549

Table 5.3: Attention map sigma influence on performance

Attention index	sigma	threshold	Squeezenet UniCls	Squeezenet SepCls	Resnet18 UniCls	Resnet18 SepCls
sigma	5	[0-1]	0.5363	0.5258	0.5547	0.5495
sigma	10	[0-1]	0.5378	0.5281	0.5714	0.5608
sigma	15	[0-1]	0.5293	0.5209	0.5569	0.5428
sigma	20	[0-1]	0.5407	0.5202	0.5652	0.5428

Table 5.4: Attention map threshold influence on performance

Attention index	sigma	threshold	Squeezenet UniCls	Squeezenet SepCls	Resnet18 UniCls	Resnet18 SepCls
threshold	[5-20]	0	0.5167	0.5033	0.5524	0.5431
threshold	[5-20]	0.1	0.5341	0.5295	0.5624	0.5606
threshold	[5-20]	0.3	0.5318	0.5299	0.5594	0.5634
threshold	[5-20]	0.7	0.5431	0.5349	0.5677	0.5388
threshold	[5-20]	1	0.5243	0.4996	0.5415	0.5182

performance. For all other models, threshold=0.7 gives the best performance.

5.7 Conclusion

In this chapter, we first discuss the complex context for CMC and Scaphoid joints. Then we introduce the attention models. We give details about how to create attention, where to apply attention mechanism, how to keep attention map geometrically consistent. We investigate the performance influencing factors, such as Gaussian sigma value, map background threshold, backbones, classifier modes. We compare different backbone performances. We discuss Gaussian shape effects on attention map models. We conclude that the attention model could make good predictions on joints with a complex context.

At this point, we have given all the procedures and algorithms for the JSN task. For joints with simple context such as MCP and PIP, the fusion patch models could give relatively reliable results. For joints with complex contexts such as CMC and Scaphoid, we have designed the attention model. We conduct performance analysis and visualize prediction results. Although we have a working pipeline for

the hand X-ray joint space narrowing task, the predicting ability is far from the doctor's diagnosis.

There are several working directions in the future. First, we can enlarge the fusion scope by incorporating more joints in the X-ray image of the same patient. JSN correlations exist not only in the same type of joints but also in neighboring joints. For example, MCP3 and PIP3 have a strong correlation since they are next to each other. Second, we can do a fine-grained classification on the joint. The wrong predictions usually have subtle characteristics that can be identified by a human doctor with care. We need to incorporate the subtle shape details in our predicting pipeline. Third, we need more labeled data. Current models suffer from obvious overfitting problems. More training data would improve performance. So we will continue the JSN task by exploring these three directions.

Bibliography

[1] Henk Visser, Saskia le Cessie, Koen Vos, Ferdinand C. Breedveld, and Johanna M. W. Hazes. How to diagnose rheumatoid arthritis early: A prediction model for persistent (erosive) arthritis. *Arthritis & Rheumatism*, 46(2):357–365, 2002.

[2] Mark A. Quinn, Michael J. Green, Helena Marzo-Ortega, Susanna Proudman, Zunaid Karim, Richard J. Wakefield, Philip G. Conaghan, and Paul Emery. Prognostic factors in a large cohort of patients with early undifferentiated inflammatory arthritis after application of a structured management protocol. *Arthritis & Rheumatism*, 48(11):3039–3045, 2003.

[3] Yasser El Miedany, Sally Youssef, Annie N. Mehanna, and Maha El Gaafary. Development of a scoring system for assessment of outcome of early undifferentiated inflammatory synovitis. *Joint Bone Spine*, 75(2):155–162, 2008.

[4] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast Interactive Object Annotation with Curve-GCN. In *CVPR*, 2019.

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*, pages 318–362. MIT Press, Cambridge, MA, USA, 1986.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[7] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. *Nature*, 521(7553):436–444, 2015.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[9] Md. Zahangir Alom, Tarek M. Taha, Christopher Yakopcic, Stefan Westberg, Mahmudul Hasan, Brian C. Van Esesn, Abdul A. S. Awwal, and Vijayan K. Asari. The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches. *ArXiv*, abs/1803.01164, 2018.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[11] Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. *ArXiv*, abs/1311.2901, 2013.

[12] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2014.

[15] G. Litjens, T. Kooi, and B.E. Bejnordi. A survey on deep learning in medical image analysis. *Med Image Anal*, 42:6088, 2017.

[16] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1MB model size. *arXiv*, abs/1602.07360, 2016.

[17] Rachel C. Jeffery. Clinical features of rheumatoid arthritis. *Medicine (United Kingdom)*, 42(5):231–236, 2014.

[18] Connor Shorten and Taghi M. Khoshgoftaar. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6:1–48, 2019.

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[20] Sebastian Ruder. An overview of gradient descent optimization algorithms. *ArXiv*, abs/1609.04747, 2016.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *ArXiv*, abs/1412.6980, 2014.

[22] Anders Krogh and John A. Hertz. A Simple Weight Decay Can Improve Generalization. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 950–957. Morgan-Kaufmann, 1992.

[23] R H Choplin, J M Boehme, and C D Maynard. Picture archiving and communication systems: an overview. *RadioGraphics*, 12(1):127–129, 1992. PMID: 1734458.

[24] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34:18–42, 2017.

[25] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:834–848, 2016.

[26] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015.

[27] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. AutoAugment: Learning Augmentation Policies from Data. *ArXiv*, abs/1805.09501, 2018.

[28] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Learning Deep Features for Discriminative Localization. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2015.

[29] Amir Roshan Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling Task Transfer Learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.

[30] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural Architecture Search: A Survey. *J. Mach. Learn. Res.*, 20:55:1–55:21, 2018.

[31] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *ICML*, 2019.

[32] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *ArXiv*, abs/1409.0473, 2014.

[33] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 2048–2057, 2015.

[34] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T. Chua. SCA-CNN: Spatial and Channel-Wise Attention in Convolutional Networks for Image Captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6298–6306, July 2017.

[35] Julius Sim and Chris C Wright. The Kappa Statistic in Reliability Studies: Use, Interpretation, and Sample Size Requirements. *Physical Therapy*, 85(3):257–268, 03 2005.