Deep Learning for Microbiome-Based Disease Prediction and Rheumatoid Arthritis Hand Joint Detection

by

Daryl Fung

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Computer Science

University of Manitoba

Winnipeg, MB, Canada

November 2022

Thesis advisors **Author**

Dr. Pingzhao Hu & Dr. Carson K. Leung Daryl Lerh Xing Fung

Deep learning for microbiome-based disease prediction and rheumatoid arthritis hand joint detection

Abstract

The presence of gut microbiome can have a significant impact on a person's health and diseases. Gut microbiome that are collected from different locations tend to have different measurement even though they are the same samples due to the difference in equipment or handling creating batch effects. Rheumatoid arthritis is an autoimmune disease that affects multiple joints especially the finger leading to joint damage. A highly trained medical professional is often required to monitor the development of joint damage in a resource limited area which can reduce the efficiency to review the joint damage. In this MSc thesis, we show how using deep learning was able to classify longitudinal gut microbiome with missing data and ways to resolve missing data using imputation methods or padding methods. We also develop a deep learning for joint detection on rheumatoid arthritis patients, and that YOLOv5l6 is able to predict the bounding boxes of joints of rheumatoid arthritis patients with high performance even though YOLOv5l6 was trained on healthy joints. Pre-training with COCO dataset with YOLOv5l6 before training on the healthy joints was able to improve the performance of joint detection. Moreover, we also propose a deep learning autoencoder and extended LassoNet to classify disease status and remove batch effect in a single forward step through the analysis of the oral microbiome.

Acknowledgements

I am extremely grateful to people that have helped and given me the never-ending support throughout my journey. I want to strongly express my gratitude to my supervisor, Dr. Pingzhao Hu for his perseverance in providing me support and amazing guidance for my MSc degree at the University of Manitoba. Throughout my MSc degree journey, he has provided me with valuable support and creative solutions to the obstacles that I have encountered in my research as well as providing interesting ideas for my projects. No words can describe how grateful I am for Dr. Hu. I want to extend my gratitude to my co-supervisor, Dr. Carson Leung, who have given me immense feedbacks, guidance and support. Dr. Carson Leung has always been ensuring I am going into the right direction with my progress. I will not be able to publish my first few papers without the help of Dr. Carson Leung. Furthermore, I want to thank Dr. Yang Wang and Dr. Olivier Tremblay-Savard for being my advisory committee team and their stimulating comments and suggestions.

Thank you, Dr. Carol Hitchon and Dr. Liam O'Neil, for their consistent feedbacks on the rheumatoid arthritis project and their help on retrieving rheumatoid arthritis x-ray images data for our research. Their feedbacks and their help have made the rheumatoid arthritis research possible. Qian Liu, Judah Zammit, Saqib Islam, Leann Lac, Wasif Khan, Yan Sun, Hamid Hadipour, who are my colleagues and friends in Hu Lab have given me amazing support and help throughout my MSc degree.

Finally, I want to thank my family for giving me all the support, guidance, love, and the amazing opportunity for being able to extend by education in Canada. I would not have gotten where I am right now without their love and support.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviations	Description
AI	Artificial intelligence
AUC	Area under curve
CBRG	Craniofacial Biology Research Group
CF	Caries free
CLR	Centered log-ratio
CNN	Convolutional neural network
COCO	Common Objects in Context
CPAR	Classification based on predictive associative rules
CS	Corticosteroid
CSP	cross stage partial network
DNA	Deoxyribonucleic acid
EAC	Early Arthritis Cohort
ECC	early childhood caries
FD	First distillation
FN	False negative
FP	False positive
HOMD	Human Oral Microbiome Database
IBD	Inflammatory bowel disease
IOU	Intersection over union
IP	Interphalangeal

IV	Intravenous
JSN	Joint space narrowing
KL Divergence	Kullback-Leibler divergence
LASSO	Least absolute shrinkage and selection operator
LODO	Leave one dataset out
LSTM	Long short term memory
MCP	Metacarpophalangeal
MIL	Multi-Instance Learning
MITRE	Microbiome Interpretable Temporal Rule Engine
mRNR	Minimum Redundancy Maximum Relevance
MSE	Mean square error
MTP	Metatarsal pharyngeal
NMS	Non-maximum suppression
OTUs	Operational taxonomic units
PANet	path aggregation network
PCA	Principal component analysis
PCR	Polymerase chain reaction
PETS	Peri/Postnatal Epigenetic Twins Study
PIP	Proximal internalangeal
PROTECT	Predicting Response to Standardized Pediatric Colitis Therapy
PS-KD	Progressive self-knowledge distillation
RA	Rheumatoid Arthritis
RF	Random forest

RNFL	Retinal nerve fiber level
RNN	Recurrent neural network
ROC	Receiver operating characteristics
rRNA	Ribosomal ribonucleic acid
scRNA	small conditional ribonucleic acid
SD	Second distillation
Stdev	Standard deviation
SvH	Sharp/van der Heijde
SVM	Support vector machine
TCC	Tag count comparison
TP	True positive
YOLOv5l6	You only look once version 5l6

Chapter 1 Introduction and Background

1.1 Metagenomics

Genomics is the study of the genome of a single organism and the interaction of the genome with its environment including the genes of other organisms. In the past, cultivated clonal cultures were needed to undergo genome sequencing. Such method causes a vast majority of the diversity of the ecology of microorganism to be missed [1]. An estimated of more than 99% of the microorganism are not observed in cultivated culture [2].

Instead of using cultivated clonal cultures, metagenomics studies the genetic materials by directly sampling the environment. Metagenomics is powerful as it has the ability to analyse and expose a wide diversity of microorganisms in an environment, providing a huge potential in understanding the biological world [3].

In order to analyse genomic data or to determine the taxa of the microbiome of an environment, sequencing is needed. There are several sequencing methods that can be used but we will talk about the two widely used sequencing methods: *16S Sequencing* and *shotgun metagenomic sequencing.*

1.1.1 16S rRNA Gene Sequencing

16S rRNA gene sequencing is a type of sequencing that reads the region of the 16S rRNA gene which is found in bacteria, archaea and eukaryote. 16S rRNA gene is highly conserved and does not change much, making it a highly suitable region for identification. PCR is first used to amplify

the 16S rRNA gene. The amplified regions are then processed and sequenced. The sequenced data are analysed with a 16S reference database to identify the type of microorganism. The amplified 16s rRNA gene sequencing is limited to these bacteria, archaea, and eukaryote as 16s rRNA gene sequencing can only identify these microorganisms. The identification of viruses would require metagenomic sequencing as viruses do not contain 16S gene.

1.1.2 Shotgun Metagenomic Sequencing

Unlike 16S sequencing which reads only the 16S rRNA gene, shotgun metagenomic sequencing sequences the entire genome in organisms. This helps researchers evaluate the abundance and diversity of microbes in the environment. Shotgun metagenomic sequencing works by obtaining all the genomes in a given sample. Then, the DNAs are broken down into smaller pieces of fragments which are made possible to be analysed and sequenced independently. The pieces of fragments are compared with each other and overlapping reads are assembled together to form a continuous sequence. A reference database can then be used to compare against the sequences for identification. As shotgun metagenomics sequencing targets all the genetic material in a sample, the information can be used for additional analysis including but not limited to metagenomic assembly and binning, antibiotic resistance gene profiling, and metabolic function profiling.

Moreover, the advancements in technology have enabled cheaper sequencing that increased the amounts of metagenomic sequencing data publicly available. This has enabled the availability of large amount of metagenomic sequencing paired with disease information from patients [4].

1.2 Microbiome

Microorganisms exist everywhere inside and outside our bodies. They inhabit the skin, nose, eyes, gut, and even oral cavity. The human body makes up of at least a trillion microorganisms, around 10 times more than the human cells excluding red blood cells [5], [6]. [120] gave a general statement that there exists between $10^{12} - 10^{14}$cells in the human body. An estimation of human total cells calculated on a variety of cell types and organs were found to be 3.72×10^{13} [121]. If we do include red blood cells, the total count for both human cells and the microorganism will come close to each other resulting in around 3.0×10^{13} for human cells and 3.8×10^{13} [6]. However, due to their small size, they only make up around 3 percent of a human bodyweight. A person with 100 pounds of weight will consist of at most 3 pounds of weight from microorganism. Depending on the type of microorganisms, they can cause sickness to human bodies. Often times, they aid with our survivability, essentially providing important functions for humans.

1.2.1 Gut Microbiome

Gut microbiome is closely related and plays an important role in the human health [7], [8]. Many microorganisms especially in the gut intestine tract provides a lot of benefits for humans. They help with absorption of ions, vitamin production, protection against harmful bacteria, improve immune system, etc. [15]–[19]. Researchers also stated that the gut microorganism can have an effect on human's mental state [20], nervous system [21], Parkinson's disease [22], seizures [23], Schizophrenia [24], hypertension [25], atherosclerosis [26]. Exercise and diet can also change the composition of the gut microbiome suggesting that treatment on the gut microbiome can affect the daily life of a person [9]–[11]. The microorganisms in the gut produces metabolites that can

regulate inflammation and create an immune process for diseases which shows an availability in creating treatment through metabolites supplements [12]–[14].

1.2.2 Oral Microbiome

The way food passes through the oral cavity can have a big impact on the digestion of the food as food are typically chewed up into smaller particles before entering the stomach to increase the surface area. One of the databases that contains oral microbiome, sequenced with a 16S rRNA identification tool, can be found in the Human Oral Microbiome Database (HOMD) [27]. Similar to gut microbiome, there is a large diversity of microorganisms in the oral cavity [28]. There exist more than 700 species of bacteria in the oral cavity which are important for the nourishment of microorganisms living in the mouth [29]. Microorganisms in the oral cavity can have a high chance of spreading to nearby location including the nose and the ears. They have been shown to cause several oral infections --- gum disease, tooth decay, root canal infections, and tonsillitis [27]. Additionally, they could also have an effect on stroke [30], diabetes [31], pneumonia [32], and cardiovascular disease [33], [34]. Metagenomics and microbiomics have emerged as tools to identify the different types of microbes in the body [35]. Instead of relying on cultivated culture to sequence genomes, metagenomics can sequence the genomes directly from samples obtained from environments [36]. The oral microbiome is one of a well-studied human microbiome to date due to the ease of obtaining samples from the oral cavity [37].

1.3 Rheumatoid Arthritis

Rheumatoid arthritis (RA) is an autoimmune disease that affects multiple small and large joints leading to joint damage. The earliest findings for rheumatoid arthritis are usually found in the

proximal interphalangeal (PIP) joints and metacarpophalangeal (MCP) of the hands, the wrists, and the metatarsal pharyngeal (MTP) joints of the feet [38]. Rheumatologists rely on a variety of clinical clues to diagnose RA and to determine optimal antirheumatic therapy with a goal of preventing joint damage [39]. Standard radiographs are usually used to identify and monitor for the development of joint damage which is manifested as joint space narrowing and erosion. This often requires a highly-trained medical professional to review the radiographs and in resource limited regions, there can be limited timely access to these professionals. [40]–[44]

1.4 Machine Learning

Artificial intelligence are computer systems that are able to perform tasks that require human intelligence. To further advance the technology of artificial intelligence, machine learning was introduced. Machine learning models are able to learn from data and predict outcomes without being explicitly programmed to do so. Machine learning models carry out this process by learning and recognizing general patterns through data which consists of inputs with labels. For instance, there is an image of a dog and a text, “dog”. When we feed the image of a dog to the machine learning model, the machine learning model will learn to classify that image as “dog”. Machine learning models however tend to deteriorate when the distribution needed to learn is very complex. There are several ways to train machine learning. Two of the ways are supervised learning and unsupervised learning. Supervised learning is an approach where the dataset consists of inputs and labels. The labels are used to train the machine learning models to classify or predict the outcome based on the inputs. Unsupervised learning is an approach where the dataset used to train the machine learning models only consists of inputs. There are no labels in the dataset. Unsupervised learning approach helps machine learning models to discover hidden patterns in the dataset by carrying out clustering, association, and dimensionality reduction.

1.4.1 Deep Learning

When it comes to modeling complex distributions, deep learning surpasses machine learning as deep learning models can generalize complex distributions much better. Deep learning consists of building blocks of neural networks stacked on top of each other, creating deep layers, hence the word “deep”. Deep learning does however require more samples to learn from to model more complex distributions and to achieve a higher performance than machine learning models. Otherwise deep learning models would overfit/memorize the dataset instead of generalizing/understanding the data itself. Deep learning has been substantially applied to a diverse field including computer vision [45]–[53], natural language processing [54]–[59], speech processing [60]–[64], and planning [65]–[70].

Moreover, deep learning has been applied in omics data including microbiome data. Deep learning is suitable for these data as these data can be highly dimensional and have a complex distribution. As the distributions in the microbiome data can be used to find associations with disease status, disease severity, and disease classification, deep learning can be used to find the corresponding disease through the microbiome data by learning and extracting complex patterns.

Chapter 2 Motivation and Research Objectives

2.1 Motivations

2.1.1 Gut Microbiome

Since gut microbiome is dynamic in nature, there contains more disease-specific information in longitudinal gut microbiome than a single time point in gut microbiome. Machine learning models may achieve a high performance in predicting disease severity with a low amount of data (e.g. low dimensional data) but will not be able to model data with complex distributions (e.g. high dimensional data) as well as deep learning. We propose to use deep learning to model the complex distribution of longitudinal gut microbiome profiles to determine disease severity of patients. There exists also missing data at specific time points in patients as gathering data for longitudinal gut microbiome from patients can be difficult. We will experiment with several methods including data imputation to solve the missing timepoints. As there is a limited amount of data available for longitudinal gut microbiome, this would cause issues on deep learning models as deep learning models require a good amount of data to perform well. We will use self-knowledge distillation to solve this issue to improve the performance of the deep learning models. We will also run sensitivity analysis on the different number of timepoints used to classify the disease severity of longitudinal gut microbiome to determine the cutoff point of timepoints needed to classify disease severity through the gut microbiome before the performance is affected.

2.1.2 Batch Effect in Oral Microbiome

Batch effects are scenarios where factors that are not related biologically cause change of data in an experiment. This can happen when experiments are carried out with a difference in location, time, tools, or equipment. Batch effects can skew data and cause classification to be incorrect due to the unrelated factors. It is important for us to include batch effects into experiments and to remove them so that any classification done is more accurate. However, it can be complicated to extract and remove batch effects from studies as the variation in batch effects is not very straightforward. We will experiment using an end to end deep learning model to classify disease status of oral microbiome using several studies with batch effects simultaneously. The oral microbiome studies that we will use are *Agnello 2017* [71], *Gomez 2017* [72], *Teng 2015* [73], *Vivianne 2020* [74], *Kalpana 2020* [75].

2.1.3 Rheumatoid Arthritis

To the best of our knowledge, there is no publicly available rheumatoid arthritis dataset on joint detection that we could use to train our deep learning models on. We will use one of the state-of-the-art object detections on rheumatoid arthritis patients. We will show that even without a rheumatoid arthritis dataset for joint detection, we were still able to achieve a good performance on detecting the hand joints of rheumatoid arthritis patients. Being able to detect joints of rheumatoid arthritis patients will open up a wide variety of applications including but not limited to determining the severity score of rheumatoid arthritis through individual joints using artificial intelligence.

2.2 Research Objectives

This thesis covers three research objectives:

- **Aim 1**: Use deep learning network with self-knowledge distillation to analyze multiple time points of longitudinal gut microbiome to classify disease severity and the presence of allergies in subjects with missing timepoints. (Chapter 3)
- **Aim 2:** Classify disease severity of subjects through oral microbiome with batch effects taken into account using autoencoder and LassoNet with robust loss, batch loss and ranger optimizer. (Chapter 4)
- **Aim 3:** Detect joints of rheumatoid arthritis of patients without being trained on rheumatoid arthritis patients' joints as there is a limited amount of data available. (Chapter 5)

Chapter 3 A self-knowledge distillation-driven CNN-LSTM model for predicting disease outcomes using longitudinal microbiome data

3.1 Introduction

Our human body contains a highly dynamic microbial environment called the gut microbiome [76]. The gut microbiome changes over time due to the presence of microbes, interacting with the environment and between microbes and the host. The gut microbiome can be influenced by infections or medical interventions over time resulting in a different microbial composition [77]. The recent advancements in technology have enabled cheaper sequencing that increased the amounts of metagenomic sequencing data publicly available. This has enabled the availability of a large amount of metagenomic sequencing paired with disease phenotypes from patients [4]. Due to the large amount of publicly available data of metagenomic sequencing, researchers have been motivated to use machine learning models and deep learning models to extract the features of the metagenomic sequencing data to predict their disease outcomes and phenotypes.

LaPierre *et al.* reviewed a variety of machine learning models and feature extraction methods on an analysis of Type 2 Diabetes [78]. In metagenome-based disease prediction, the most commonly used machine learning models are support vector machine (SVM) [79] and Random Forest (RF) [80]. They are able to output the most informative features that include the microbes or the functional elements which contributes the most to the disease prediction in the area of metagenome-context disease prediction. The informative features can be used to provide insight into the relationship between the microbiome and the disease. Additionally, deep learning has also

been used to predict disease through metagenomic sequencing data. Deep learning can learn more complex functions with more layers. However, the interpretability of deep learning models can be reduced as more complex functions are produced.

Rahman and Rangwala proposed to use a new Multi-Instance Learning (MIL) to determine the clinical phenotype from metagenomic sequencing data [81]. They tested their model on Liver Cirrhosis study [82] and Inflammatory Bowel Disease (IBD) study [83]. They showed that their model was able to outperform other comparative MIL [84]–[86] and non-MIL methods [83], [87]. Multi Instance Learning is a weakly supervised approach that contains bags of instances with labels. In a MIL approach, rather than having each instance containing a label, there will be a bag of instances where the bag will contain the label. In a binary case, bags that contain at least one positive instance will be assigned with a label of 1. Bags that contain all negative instances will be assigned a label of 0. They were able to achieve an area under the receiver operating characteristic curve (ROC-AUC) score of 0.8442 for IBD dataset and an AUC score of 0.9272 for Liver Cirrhosis dataset. However, the methods showed only to extract features from a single time-points, missing valuable information of the temporal changes of the gut microbiome. As gut microbiome is highly dynamic in nature, the temporal information can contain rich information about the individual's microbe environment and how they would affect disease.

Sharma and Xu proposed a novel deep learning framework called "phyLoLSTM" that uses convolutional neural network to extract the features and LSTM to analyse the temporal dependency in microbiome sequencing data with the host's factors to determine the disease [88]. They also proposed a novel data pre-processing that is able to handle the variable timepoints in each subject, and weight balancing to handle the imbalance classes in the dataset. They tested their

model on simulated dataset and real dataset. They tested their models on two real studies, DIABIMMUNE [89] and DiGiulio [90]. The DIABIMMUNE study includes three country cohorts that determined food allergy outcomes of subjects to milks, peanuts, eggs and overall. DiGiulio study contains the preterm delivery as outcome based on the microbial taxa from vagina, saliva, distal gut, and gum. They showed that their method was able to achieve 0.897 AUC on simulated dataset, 0.713 AUC and 0.762 AUC on the DIABIMMUNE and the DiGiulio datasets, respectively.

Chen *et al.* stated that single-point prediction cannot capture the dynamic patterns of the temporal changes between gut microbiome and host status [91]. They proposed to use a deep learning network, specifically Gated Recurrent Unit (GRU) [92], that uses longitudinal microbiome sequential data to feed into the deep learning network to predict the human host status. They tested their methods on both semi-synthetic and real datasets [25], [89], [93]–[99]. The semi-synthetic dataset is from [100] and the same setting and parameters based on a Microbiome Interpretable Temporal Rule Engine (MITRE) [101] were used. A pipeline of data pre-processing was undergone to further improve the performance of the deep learning networks. They were able to achieve high performance compared to other baseline classifiers and improve the evaluation time taken to classify the subjects.

Metwally *et al.* investigated using LSTM to predict food allergies through the use of the features from longitudinal gut microbiome profiles [102]. They compared their model against machine learning models including Hidden Markov Model [103], Multi-Layer Perceptron Neural Network [104], Support Vector Machine [79], least absolute shrinkage and selection operator (LASSO) Regression [105], and Random Forest [106]. They also reduced the microbial features by using

sparse autoencoder [107] to extract the latent representations of the microbial features in addition to using Minimum Redundancy Maximum Relevance (mRMR) and ranking based on variance to select standard feature before the training of the LSTM network. The latent representations of sparse autoencoder achieved the highest performance compared to using mRMR and ranking based on variance achieving an ROC-AUC of 0.67. The study showed that the learning of LSTM can be useful to be used on another longitudinal microbiome data dataset as well in predicting subject's clinical outcome.

García-Jiménez *et al.* integrated deep learning techniques to condense the microbial composition into a deep latent space representations with less but rich features from plant microbiome data [108]. They showed that using deep learning techniques, they are able to predict the plant microbiome composition through the use of environmental factors including temperature, plant age, and precipitation. They showed that transfer learning was able to improve the performance further. They experimented with different orders to transfer the features over from Maarastawi dataset [109] to Walters *et al* dataset [110]. Using Pearson correlation and Bray-Curtis dissimilarity, and the best transfer features is the phylum order achieving 0.9451 Pearson correlation and 0.1833 Bray-Curtis dissimilarity.

As deep learning networks require a huge amount of data to prevent overfitting, deep learning networks are more prone to overfitting when trained on gut microbiome datasets because of the lack of a huge data availability. Zhang *et al.* proposed a general training called self-distillation which improves the performance of deep learning networks [111]. Self-distillation is similar to knowledge distillation where a student network's distribution is encouraged to share similar distributions or weight features with the teacher network's weight features. However, instead of

learning from a teacher network, self-distillation learns from its network itself. The network is separated into separate shallow sections and the learned knowledge of the deeper parts of the network is shared with the shallow sections of the network. They showed that this technique was able to improve the accuracy at an average of 2.65%, having a variation between 0.61% to 4.07% using ResNeXt [112] and VGG19 [113] respectively.

Kim *et al.* proposed a simple but effective regularization method by using a self-distillation called progressive self-knowledge distillation (PS-KD) [114]. It does self-distillation by progressively regularizing the training using the model's previous epoch weights. The model's previous epoch weights act as the teacher model to distil the knowledge into the student model (itself). PS-KD is applicable to any kinds of supervised learning approach with hard targets, and can be used to further generalize the performance through combining with existing regularization methods. Their evaluation showed that PS-KD achieved high accuracy and high-quality confidence estimation as measured through ordinal ranking and calibration. They experimented their method on multiple tasks including object detection, image classification, and machine translation, and showed that their method was able to improve the performance of models in all three tasks.

The major challenge with longitudinal data of the gut microbiome is that there exists missing information along the timeline of different subjects, causing uneven distribution. To mitigate the problem of missing information, we use padding in sequence, where the missing information is padded. To address the multiple operational taxonomic units (OTUs) in the microbiome profiles and small number of subjects collected at each time points, we develop an efficient hybrid deep learning architecture CNN-LSTM (convolutional neural network - Long Short-Term Memory), combined with self-knowledge distillation to create high accurate models to predict disease

outcomes from gut microbiome. The CNN is used to extract informative features and the LSTM is used to understand the temporal dynamic of the gut microbiome.

3.2 Materials and Methods

3.2.1 Data sets and data processing

3.2.1.1 Longitudinal microbiome data

The datasets that we use to evaluate our models are PROTECT (Predicting Response to Standardized Pediatric Colitis Therapy) study [115] and DIABIMMUNE dataset [89].

PROTECT study consists of 428 subjects with new-onset paediatric ulcerative colitis disease from USA and Canada, and was monitored over the course of a year. Patients did not receive any treatments at week 0, and were assigned into one of two treatments – either 5-aminosalicylic acid (mesalamine) or oral / intravenous (IV) corticosteroid (CS) followed by 5-aminosalicylic acid (mesalamine). The disease progression was monitored with the treatment progression. Within the 428 subjects, there are 405 subjects where stool and rectal samples were collected for microbiome sequencing. The samples were collected on week 0, week 4, week 12, and week 52. Clinical and metadata were collected throughout the year, including gender, ethnicity, age, PUCAI, treatment, disease progression, stool consistency, fecal calprotectin, and the extent of the disease involvement. Patients are between the age of 4 to 17 with 48% of them being females and 52% being males. The data set includes 1015 OTUs.

DIABIMMUNE is a study that aims to determine the autoimmune diseases and allergies through biological mechanisms. The dataset contains families from three different countries: Finland, Estonia, and Russia. Each subject is an early infant before the first 6 months of age and they were followed until the age of 3. There are 74 infants as subjects in each country. Three years of monthly stool samples, laboratory assays and questionnaires that include breastfeeding, diet, allergies, family history, infections, clinical examinations and use of drugs were collected. A total of 1584 stool samples were sequenced using 16s rRNA amplicon sequencing and 785 stool samples using whole-genome shotgun sequencing. The data set includes 282 OTUs.

3.2.1.2 Data pre-processing

As the microbiome sequencing data contains invariant timepoints caused by missing information from the subjects at some time points, we propose to use several methods to solve the issue.

The first method that we used is to pad the data in sequence. For instance, in the case of the PROTECT study, if a subject contains week 0 and week 52 but misses week 4 and week 12, the subject will be padded with 0s on week 4 and week 12, maintaining the sequence and the order of the longitudinal data of the subject. The second method that we used is to pad and mask the padded redundant data from the subject. An example of the padding techniques is shown in **Table 3-** 1 .

In addition, we also passed the microbiome sequencing data into a dimension reduction algorithm. Here we used principal component analysis (PCA) to determine if there is an improvement on performance through selecting the top principal components of the microbiome data. We set the number of principal components to be 300.

Table 3- 1 An example of our padding techniques. Both padding techniques are done on the same subject that contains missing data in week 4 and week 12.

Pad in Sequence					
	Ruminococcaceae	Peptostreptococcaceae	Alcaligenaceae	…	Porphyromonadaceae
Week 0	0.00029747	0.00000381	0.0000114	…	0.00188396
Week 4	0	0	0	0	0
Week 12	0	0	0	0	0
Week 52	0.0000168	0.00000839	0. 00000839	…	0.0050269
Pad and mask					
	Ruminococcaceae	Peptostreptococcaceae	Alcaligenaceae	…	Porphyromonadaceae
Week 0	0.00029747	0.00000381	0.0000114		0.00188396
Week 52	0.0000168	0.00000839	0.00000839	…	0.0050269
0	0	0	0	0	0
0	0	0	0	0	0

3.2.2 Methods

3.2.2.1 Self-knowledge distillation modelling framework

Long-Short Term Memory (LSTM) model

The LSTM network learns temporal information of the data. It consists of three different gates: input gate, output gate, and the forget gate. The gates control how much of the information needed

to be passed into the next time step. Sigmoid is used as a gating mechanism to control the input to be between the value of 0 and 1. The forget gate is defined as follows:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.1}$$

W_f is the weights for the forget gate to transform the input at the current time step, x_t is the input at the current time step, U_f is the weights for the forget gate to transform the hidden state from the previous time step, h_{t-1} is the hidden state from the previous time step, b_f is the bias and σ is a sigmoid function that squashes the input to value between 0 and 1.

The input gate is defined as follows:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3.2}$$

W_i is the weights for the input gate to transform the input at the current time step, x_t is the input at the current time step, U_i is the weights for the input gate to transform the hidden state from the previous time step, h_{t-1} is the hidden state from the previous time step, b_i is the bias and σ is a sigmoid function.

The output gate is defined as follows:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3.3}$$

W_o is the weights for the output gate to transform the output at the current time step, x_t is the input at the current time step, U_o is the weights for the output gate to transform the hidden state from the previous time step, h_{t-1} is the hidden state from the previous time step, b_o is the bias and σ is a sigmoid function.

The cell state at the current time step is calculated as:

$$c_t = f_t * c_{t-1} + i_t * tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3.4)$$

The cell state helps to control how much to forget from the previous time step and how much to include the current time step information into the cell state. c_{t-1} is the cell state from the previous timestep. If c_t is at time $t = 0$, then . c_{t-1} is 0. W_c is the weight for the cell state, x_t is the input at the current time step, U_c is the weight for the cell state to transform the hidden state from the previous time step, h_{t-1} is the hidden state from the previous time step, b_c is the bias and $tanh$ is a Tanh function that squashes values between -1 and 1.

The hidden state of the current time step is then calculated as:

$$h_t = o_t \circ \sigma(c_t) \quad (3.5)$$

The last output of the LSTM network is then flattened into a vector and fed into a multi-layer perceptron with a sigmoid activation to classify the disease task.

$$y = sigmoid(W_y h_t + b_y) \quad (3.6)$$

Convolutional Neural Network- based Long-Short Term Memory (CNN-LSTM) model

The CNN-LSTM model uses a combination of CNN and LSTM to classify the disease task. The model first uses convolutional neural network to extract the features of the gut microbiome data at each time point and pass the extracted features as input to the LSTM network. The CNN uses a kernel to extract features based on a group of input features. The equation for the extraction of the input using CNN is:

$$O_{d,e} = \sum_i^k \sum_j^k w_{i,j}^l * i_{d+i,e+j}^{l-1} + b_{d,e}^l \quad (3.7)$$

$w_{i,j}^l$ is the weights in layer l at row i and column j of the kernel, $i_{d+i,e+j}^{l-1}$ is the output from the previous layer at row $d+i$ and column $e+j$, $b_{d,e}^l$ is the bias in layer l for the output at row d and column e.

CNN-LSTM contains two pipelines. First, the gut microbiome data is fed into the convolutional neural network (CNN) [25] to extract features. The extracted features from the CNN are then fed into the LSTM [26] to learn the temporal dynamics of the gut microbiome. We also evaluated with and without a dimension reduction algorithm, principal component analysis (PCA) [27], to determine if PCA could improve the performance of the deep learning models. We used the first 300 principal components to feed into the deep learning models and compared the results against each other.

Self-distillation Model

We implemented self-distillation on the LSTM and CNN-LSTM models. We implemented multiple shallow classifiers and branched them out from the layers before the main classifier in the LSTM network. The multiple shallow classifiers classify the disease task just like the main classifier. We integrated additional loss functions to the shallow classifier to improve the features learned by the network: (1) Cross entropy loss function is used to improve the shallow classifier predictions to the ground truth labels. (2) KL divergence is used to minimize the distribution between the prediction of the shallow classifier and the main classifier. (3) L2 loss is incurred on the shallow classifier layer right before the output layer with the layer right before the output layer of the main classifier.

The cross-entropy loss function is defined as:

$$Cross\ entropy\ loss = -\frac{1}{m}\sum_{i=1}^{m} y_i * \log(\hat{y}_i) \tag{3.8}$$

The KL divergence loss is:

$$KL\ Divergence\ loss = -\sum_{i=1}^{m} y_i * \log\left(\frac{y_i}{\hat{y}_i}\right) \tag{3.9}$$

y_i is the ground truth label and $\hat{y}_i$ is the prediction. m is the total number of samples. The distribution between the prediction of the shallow classifier and the main classifier is minimized to make the shallow classifiers' distributions as similar as possible to the main classifier's distribution.

The L2 loss function is defined as follows:

$$L2\ loss = \frac{1}{n}\sum_{i=1}^{n}(W_i^m - W_i^s)^2 \tag{3.10}$$

W_i^m is the weight of the layer before the main classifier and W_i^s is the weight of the layer before the shallow classifier. **Figure 3-1a** shows the architecture of the first self-distillation method on a CNNL-STM network.

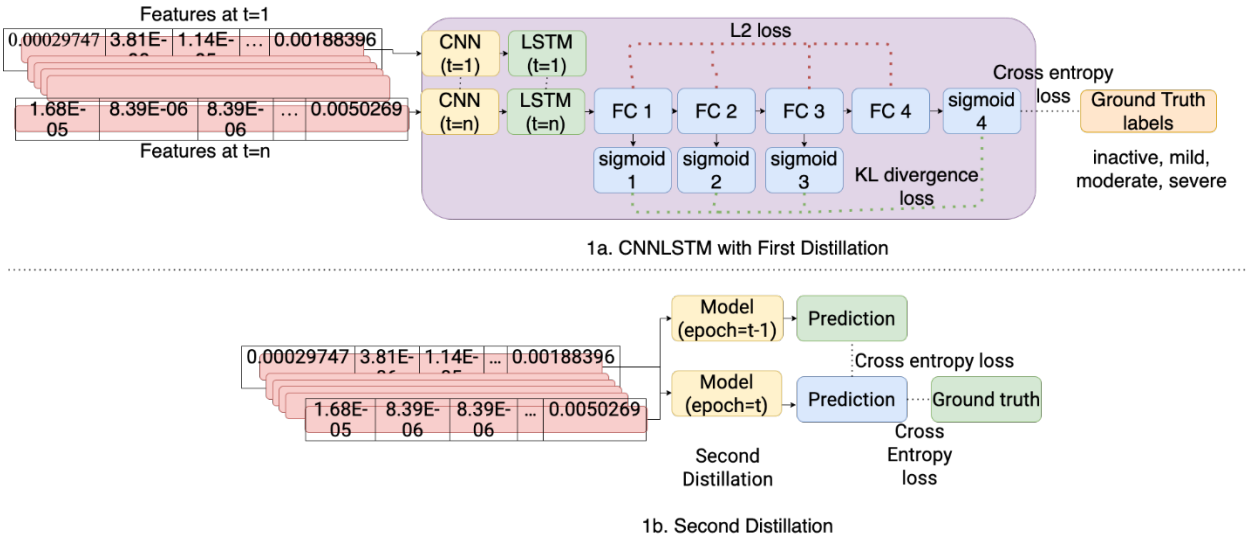


Figure 3-1 Architecture of CNN-LSTM. a. the first self-distillation method consists of several sub-classifier to predict the labels of the ground truth and the prediction of the main classifier; b. the second self-distillation method uses the model's previous epoch prediction as a regularization.

Moreover, we implemented progressive self-distillation that uses its past prediction as a teacher model to have more informative training. The total loss function will become:

$$Loss = (1-\alpha) * CrossEntropy\left(y, \hat{y_i}^t\right) + (\alpha) * CrossEntropy(\hat{y_i}^{t-1}, \hat{y_i}^t) \tag{3.11}$$

$\hat{y}_i{}^t$ is the current prediction and $\hat{y}_i{}^{t-1}$ is the prediction using the weights of the model at the previous step. The α parameter determines how much of an emphasis we want to get from the loss of using the past prediction against the loss with the hard targets. As the teacher model does not have a reliable knowledge in the beginning, we start out with a low value of α and gradually increase α. We set α at the *t*-epoch as follows:

$$\alpha_t = \alpha_T \times \frac{t}{T} \tag{3.12}$$

T is the total epoch to train the model. **Figure 3-1** shows the architecture of the second self-distillation approach.

3.2.2.2 Transfer learning

For the two data sets we collected in the study, we will undergo transfer learning from the dataset with better performance to the one with worse performance. We will run the learning of the features of the data set with better performance first and then use the learned weights to transfer to learn the dataset with worse performance. In order to undergo transfer learning, we need to have the same features in both datasets. We first categorize the bacteria by the taxonomy class. We found that the overlaps between each taxonomy class between PROTECT study and DIABIMMUNE is as follows: kingdom: 2/2, phylum: 10/12, class: 17/24, order: 27/48, family: 41/85, and genus: 71/153. The denominator is the total number of different groups in a taxonomy class. The nominator is the total amount of overlap of the different groups in the taxonomy class between the PROTECT study and the DIABIMMUNE study. For phylum: 10/12, this shows that

there are 10 groups that overlap out of 12 groups in the phylum order. Only the kingdom order has all the overlap of the taxonomy order. However, the kingdom order only contains 2 features which can be insufficient to learn rich information. Instead, we use the order of phylum. We only keep the overlapping features between the two datasets and remove the features that do not overlap. The overlapping features are *Synergistetes, Bacteroidetes, Verrucomicrobia, Actinobacteria, Fusobacteria, Tenericutes, Proteobacteria, Euryarchaeota, Cyanobacteria, Firmicutes.* The non-overlapping features are *TM7, Lentisphaerae.*

We removed the features that are non-overlapping and kept the overlapping features to train on PROTECT study. Once the network is trained on the PROTECT study, we transfer the weights of the network to train on the DIABIMMUNE dataset. During the transfer of the learned weights, we used several methods to fine-tune the weights on the DIABIMMUNE dataset - discriminative fine-tuning, gradual unfreezing, and concat pooling. Discriminative fine-tuning uses a different learning rate for different layers in the network where the learning rate is higher in the last layer and gradually gets lesser towards the first layer.

Gradual unfreezing froze all layers except the last layer and adjusted the weights of the last layer only on the first epoch. On the second epoch, the second last layer is unfrozen together with the last layer and adjusted during training. This goes on for the following epochs for all other layers.

Concat pooling concatenates the mean of all the hidden states, the max of all the hidden states and the last hidden state and feeds them into the next feed forward network.

3.2.3 Model training

During training, we split the data into 10-fold cross-validation and averaged the validation result between all the folds. We used a batch size of 64 and had a total epoch of 100. We ran the experiment with LSTM and CNNLSTM along with self-distillation and transfer learning.

3.2.4 Comparison with baseline models

We also compare our model against an existing deep learning network that focuses on longitudinal gut microbiome [102] and a recurrent neural network (RNN) [118] by replacing LSTM to RNN in our model. Since microbial profiles contains a lot of features, they decided to extract meaningful features before feeding the features into an LSTM network. Their method consists of two modules, the first module is a feature extraction module that extracts the features from the input into a compressed latent representation. The second module is an LSTM network that receives the compressed latent representation at each time steps to learn the temporal dynamics of the gut microbiome.

[102] used an autoencoder for their feature extraction module. Autoencoder is a neural network that carries out unsupervised learning by reconstructing the input data. It is very similar to a deep neural network, the main difference is that the middle layer of the autoencoder is much smaller than the rest of the network and is used to represent the latent space. An autoencoder hidden layer has the following equation:

$$x_l = ReLU(W_l x_{l-1} + b_l) \tag{3.13}$$

Where x_l is the lth hidden layer, W_l is the weights for the previous hidden state's layer, x_{l-1} is the previous hidden layer, when l is 1, $x_{l-1} = x_0 = X$, is the input. The last layer of the autoencoder is the reconstructed input $\hat{X}$. b_l is the bias term at the current hidden layer. To prevent overfitting, they added an L2 regularization on the weights. They also enforced sparsity in the autoencoder by incorporating Kullback-Leibler (KL) divergence [119] to force only a small fraction of the neurons to be activated. The total loss function for the autoencoder is:

$$Autoencoder\ loss = \frac{1}{n}\sum_{i=1}^{n}\left|\left| X_i - \hat{X}_i \right|\right|^2 + \lambda \sum_{j=1}^{m} \left|\left|W_j\right|\right|^2 + \beta \sum_{j=1}^{k} KL(p||p'_j) \quad (3.14)$$

The first term is the reconstruction loss, n is the number of samples, X_i is the input, $\hat{X}_i$ is the reconstructed input. The second term is the L2 regularization term. λ is the parameter that emphasize on how much to regularize, W_l is the weights at the l layer and m is the number of layers. The third term is the KL divergence where β is the parameter to emphasize on the KL divergence loss, k is the number of neurons in the compressed latent representation, p is the sparsity parameter, p'_j is the average activation of jth neuron in the compress latent representation. Having a lower sparsity parameter will cause the network to be more sparse.

As for the RNN network, we replace LSTM with RNN in our model to compare as one of the baseline model. Instead of having 3 gates like the LSTM, RNN has a simpler equation to learn the temporal dynamics:

$$a_t = W_x X_t + W_h h_{t-1} + b_l \quad (3.15)$$

$$h_t = tanh(a_t) \quad (3.16)$$

W_x is the weights of the input, X_t is the input at the current time step, W_h is the weights of the previous hidden state, h_{t-1} is the previous hidden state, b_l is the bias term for the current time step, and a_t is the pre-activation of the current hidden state. a_t is then fed into a tanh to convert into values between [-1, 1] before being an input to the hidden state in the next time step.

3.2.5 Model performance evaluation

After training the models, it is important to evaluate the performance of the models with each other. We measured the models' performance by using ROC-AUC and F1 score. AUC is calculated through using a ROC curve which measures the performance of models with different thresholds. The AUC is obtained by calculating the area under the ROC curve. The higher the AUC, the better the model's performance is.

$$F1\ score = 2 \times \frac{precision \times recall}{precision + recall} = \frac{tp}{tp + \frac{1}{2}(fp + fn)} \tag{3.17}$$

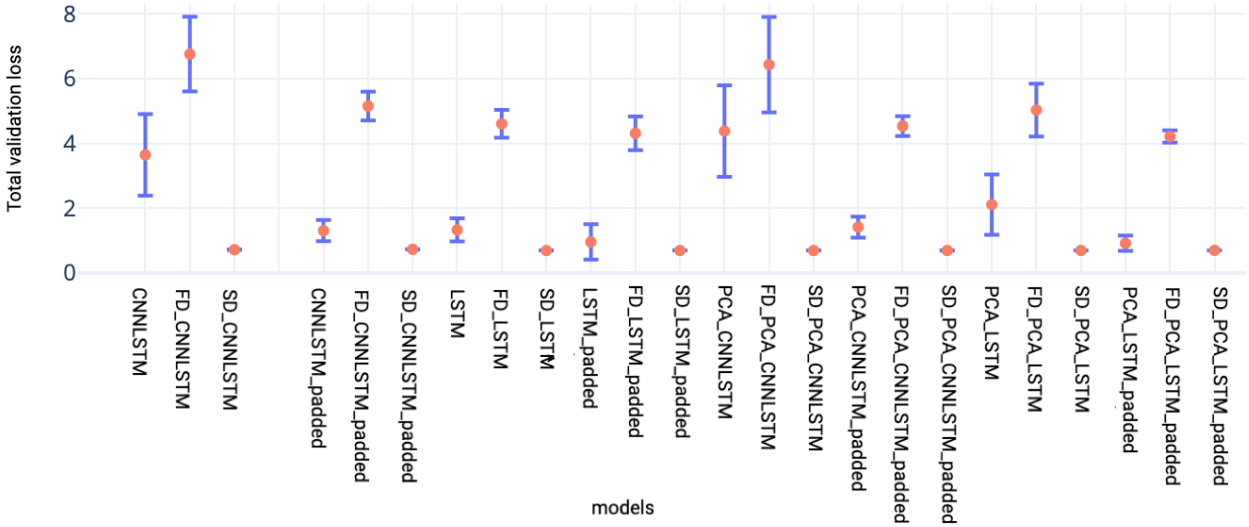
Here, *tp* refers to true positive, *fp* refers to false positive, *fn* refers to false negative. True positive are labels that are correctly classified by the model, false positive are positive labels that are falsely classified by the model, and false negative are negative labels that are falsely classified by the model.

3.3 Results and Discussions

3.3.1 Model training results

The averaged validation loss of the models' training on the PROTECT study and DIABIMMUNE study can be found in **Figure 3-2.**

a. **PROTECT Study**



b. **DIABIMMUNE Study**

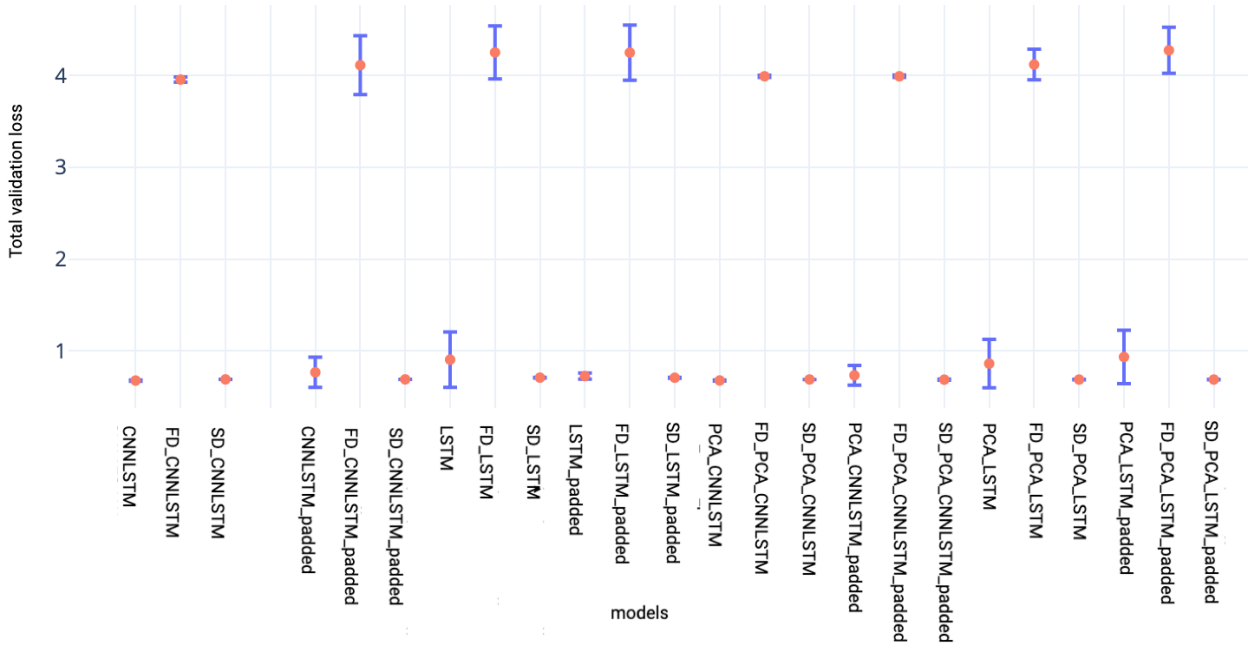


Figure 3-2 Results of different models of 10-fold cross-validation. SD: second distillation, FD: first distillation.

We can see that in both PROTECT study and DIABIMMUNE study, the validation loss for the first distillation loss tends to be higher. This is due to the multiple sub-classifiers that try to learn the main classifier prediction in addition with the feature loss and the KL divergence. The validation loss for the second distillation loss is lower due to the fact that the model is getting a percentage of the total loss from its previous epoch prediction as a regularization where the model can be already good at performing the prediction. Although the FD distillation has a higher loss than its variants, the FD distillation has a better AUC score. This is because AUC and binary cross entropy loss are different metrics, AUC measures if the predicted class is the correct class while binary cross entropy loss measures how far off a prediction is compared to the true class. With the right threshold, the models will be able to perform at a higher performance.

3.3.2 Prediction performance based on the PROTECT study

In this analysis we only obtained the OTU features and removed all other information about the patients. Each patient contains either stool samples or biopsy samples at week 0, week 4, week 12, and week 52. A large proportion of the patients have missing samples at different time points/weeks. We solved this by using either one of the three methods that we mentioned in the Data pre-processing section. We used the subject's disease severity as the disease outcomes for prediction using the networks based on the OTUs features. The disease severity includes three labels: *inactive, mild, moderate, severe*.

We split the dataset into 10-fold cross-validation to evaluate our models. We also evaluated our models on the different data pre-processing method and determined which one shows the best performance (***Table 3- 2***).

Table 3- 2 shows the prediction performance of the 10-fold cross-validation on the PROTECT Study. CNN-LSTM with padding in sequence achieves the best performance with AUC score 0.889 when compared to the other models.

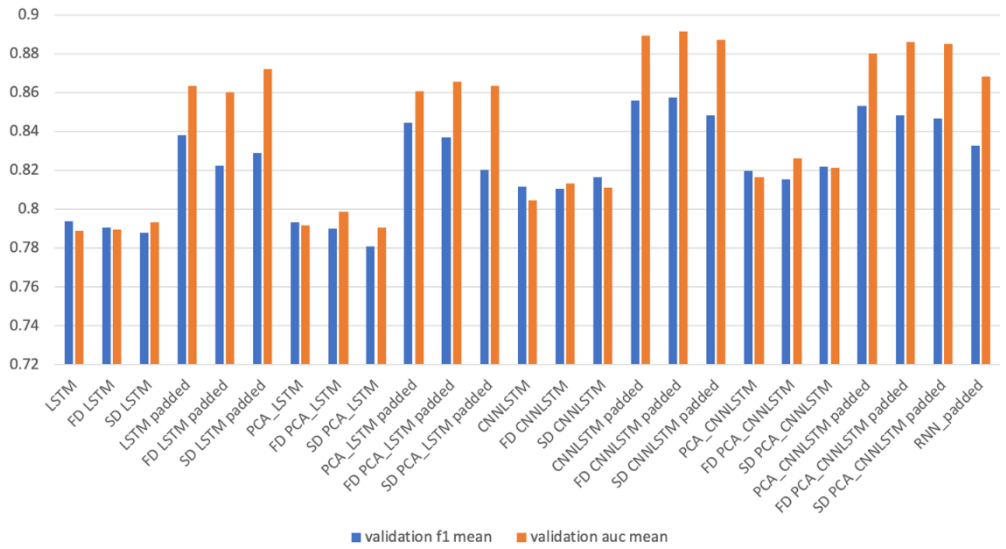
Table 3- 2 Prediction performance of the 10-fold cross-validation on the PROTECT Study without self-distillation.

Model	**PCA**	**Preprocessing**	AUC mean	AUC stdev	F1 mean	F1 stdev
LSTM	TRUE	Pad in sequence	0.861	0.041	0.69	0.059
		Pad at end	0.792	0.041	0.59	0.042
	FALSE	Pad in sequence	0.864	0.054	0.685	0.064

		Pad at end	0.789	0.038	0.58	0.036
CNN-LSTM	TRUE	Pad in sequence	0.877	0.045	**0.717**	0.057
		Pad at end	0.814	0.024	0.612	0.025
	FALSE	Pad in sequence	**0.889**	0.042	0.716	0.057
		Pad at end	0.802	0.03	0.598	0.033

We evaluated self-distillation on the PROTECT study. We can see the self-distillation techniques improve the baseline performance. The first distillation improved most of the baseline model performance. The best performing model is the CNN-LSTM padded in sequence with first distillation achieving 0.89 AUC and 0.86 F1 score (**Figure 3-3a**).

a. **PROTECT Study**



a. DIABIMMUNE study

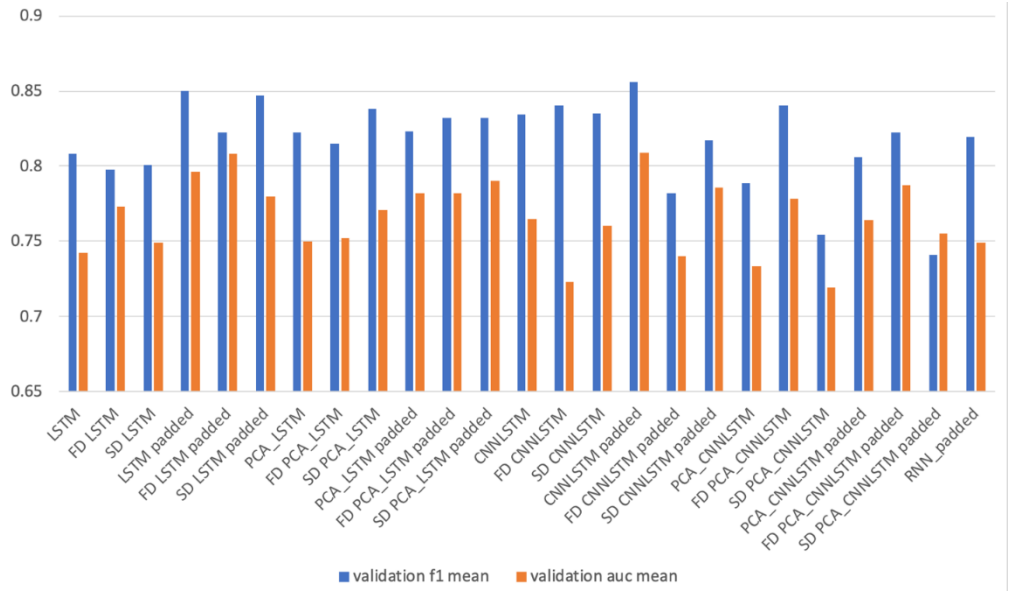


Figure 3-3 Prediction performance of the 10-fold cross-validation. a. PROTECT Study. As our goal is to evaluate the disease status of patients on week 52 (the last time points of each subject), we removed any patients that does not have week 52. Models with padded use padding in sequence method while models that does not have padded use padding and masking method. **b. DIABIMMUNE study**. Models

with padded were evaluated on the max time points (max time points is the last time points available among all the subjects). Since our goal is to evaluate the allergic reaction on the last time point, subjects which do not have the last time points were removed. Models with padded use padding in sequence method while models that does not have padded use padding and masking method. FD is First Distillation. SD is Second Distillation.

3.3.3 Prediction based on the DIABIMMUNE study

In this analysis we used 282 OTU features as input to the models. There are 203 subjects and 3 different allergies: *milk, egg, and peanut.* 53 of them have milk allergies, 40 of them have egg allergies, 9 of them have peanut allergies and the rest does not have any allergies. Each subject has a total of 38-time steps.

We split the dataset into 10-fold cross-validation and evaluated our models based on the split. The results can be seen in **Table 3-** 3 . In the DIABIMMUNE study, the best performing model is the CNN-LSTM model with no PCA, achieving 0.798 AUC score and 0.508 F1 score.

Table 3- 3 **The 10-fold cross-validation on the DIABIMMUNE study without self-distillation.**

Model	**PCA**	**Pre-processing**	AUC mean	AUC stdev	F1 mean	F1 stdev
LSTM	TRUE	Pad in sequence	0.783	0.068	0.456	0.109
		Pad at end	0.751	0.079	0.437	0.133
	FALSE	Pad in sequence	0.795	0.068	0.481	0.134
		Pad at end	0.742	0.073	0.428	0.117
CNN-LSTM	TRUE	Pad in sequence	0.754	0.084	0.443	0.17
		Pad at end	0.723	0.071	0.416	0.139
	FALSE	Pad in sequence	**0.798**	0.764	**0.508**	0.111
		Pad at end	0.764	0.764	0.464	0.116

We also evaluated self-distillation on the DIABIMMUNE study and the results can be seen in **Figure 3-3b**. We can see that the first self-distillation method was able to improve most of the baseline methods. The result is consistent with that of the PROTECT study. The best performing model in the DIABIMMUNE study is the LSTM padded in sequence with first distillation and CNNLSTM padded in sequence achieving 0.81 AUC, 0.82 F1 score and 0.81 AUC, 0.86 F1 score respectively (**Figure 3-3b**).

3.3.4 Comparison with another longitudinal deep learning model

Moreover, we compared our best performing models with an existing deep learning network that uses LSTM to predict allergies longitudinal microbiome taxonomic profiles [102]. To prevent confusion, we will call this existing deep learning network U-LSTM. We trained their model on the PROTECT and DIAIBIMMUNE dataset and compared the result against our models. We also created a recurrent neural network (RNN) by replacing LSTM to RNN in our network to compare against our models. The comparison result can be seen in **Table 3-4.** Our model was able to outperform U-LSTM and RNN in AUC and F1 score.

Table 3-4 Comparison of prediction performance of our best model on 10-fold cross-validation in each dataset with other baseline models.

PROTECT				
Model	AUC mean	AUC stdev	F1 mean	F1 stdev
CNNLSTM_padded + FD	**0.89**	0.04	0.86	0.037
U-LSTM	0.74	0.04	0.74	0.03
RNN	0.87	0.04	0.83	0.03
DIABIMMUNE				
Model	AUC mean	AUC stdev	F1 mean	F1 stdev

LSTM_padded + FD	**0.81**	0.07	0.82	0.06
U-LSTM	0.62	0.1	0.58	0.13
RNN	0.75	0.07	0.82	0.07

Additionally, we ran transfer learning to determine if there is any performance improvement by transferring the features learned from the PROTECT study to the DIABIMMUNE study. As the PROTECT study achieves AUC higher than the DIABIMMUNE study, we decided to use the features learned from the PROTECT study to transfer to the DIABIMMUNE study to evaluate if performance can be further improved. The transfer learning did not show significant improvement on the DIABIMMUNE study when transferring the learned features from the PROTECT study (results are not shown).

We ran sensitivity analysis on whether how much of the time in the past is relevant in predicting the outcome of the disease for both the DIABIMMUNE and PROTECT dataset. The results can be seen in **Table 3-** 5 .

Table 3- 5 **Sensitivity analysis on the number of time points to the performance of predicting the outcome of the model using our best model.**

PROTECT				
Data points	AUC mean	AUC stdev	F1 mean	F1 stdev
5	0.89	0.04	0.85	0.03
4	0.88	0.05	0.85	0.03
3	0.89	0.04	0.86	0.05
DIABIMMUNE				
months	AUC mean	AUC stdev	F1 mean	F1 stdev

13	0.77	0.07	0.83	0.07
12	0.77	0.07	0.83	0.08
11	0.78	0.07	0.81	0.1
10	0.77	0.07	0.81	0.1
9	0.77	0.07	0.81	0.1
8	0.78	0.07	0.81	0.1
7	0.77	0.06	0.81	0.1
6	0.78	0.07	0.81	0.1
5	0.78	0.07	0.81	0.1
4	0.78	0.07	0.8	0.1
3	0.74	0.05	0.78	0.1
2	0.74	0.06	0.76	0.1

For the sensitivity analysis, we used CNNLSTM_padded + FD on the PROTECT study and LSTM_padded + FD on the DIABIMMUNE study. The months in the DIABIMMUNE study are the number of past months used to predict the outcome of the disease. We can see that the performance does not deteriorate that much when the number of past months is above 3. When the number of months is 3 or less, the performance reduced by a visible amount. In the PROTECT study, the time points are (0, 4, 12, 52) weeks on which biopsy samples are only included in week 0 and week 52 while stool samples are included in all time points (6 data points in total). There is no visible difference in reduction in performance with different time points used.

3.4 Summary

Deep learning is able to classify disease severity and allergy reaction based on the longitudinal gut microbiome as shown in our experiments. Using temporal dynamics of the gut microbiome, we are able to capture more information about the longitudinal gut microbiome to gain a richer feature

representation of the gut microbiome to classify the disease severity in the PROTECT study and the allergy reaction in the DIABIMMUNE study.

	FALSE	Pad in sequence	0.795	0.068	0.481	0.134
		Pad at end	0.742	0.073	0.428	0.117
CNN-LSTM	TRUE	Pad in sequence	0.754	0.084	0.443	0.17
		Pad at end	0.723	0.071	0.416	0.139
	FALSE	Pad in sequence	**0.798**	0.764	**0.508**	0.111

Using CNN in combination with LSTM helps to improve the performance in achieving higher AUC score as evaluated in both the PROTECT study and the DIABIMMUNE study. For the PROTECT study, the CNN performs better than LSTM if we look at both the AUC and the F1 score. However, if we are looking in terms of speed and computation efficiency, the LSTM model would be a better choice than the CNN model as there is not a drastic improvement in performance. Our experiments showed that self-distillation is able to improve the performance of the LSTM in both the PROTECT study and the DIABIMMUNE study. Using data imputation does not necessarily improve the deep learning model's performance. In the PROTECT study and the DIABIMMUNE study, the best performing models are the ones without any data imputation. We also compared our models against other existing deep learning model for longitudinal gut microbiome studies and RNN and were able to surpass them.

The LSTM model that we used treat the timepoints equally spaced although the timepoints in the PROTECT study and the DIABIMMUNE study are not spaced equally. It would be interesting to see how the LSTM or CNN LSTM models would perform if we incorporate a time encoding into them.

In summary, we showed that the CNN combined with the LSTM can achieve better generalizing models to classify disease severity and allergy reaction of subjects. With the addition of self-distillation, it is able to achieve the highest performing models in both the PROTECT study and the DIABIMMUNE study for the LSTM model. As for the CNNLSTM model, it was able to improve the performance in the PROTECT study, the performance for the DIABIMMUNE study remains no significant changes when self-distillation was incorporated.

Chapter 4 An Extended LassoNet for Disease Classification using Microbiome Profiles from Multiple Cohorts

4.1 Introduction

4.1.1 Machine Learning in Oral Microbiome

Machine learning has also been applied to oral microbiome to predict the outcome or progression of various diseases. As microbiome is one of the factors that affects several neurodegenerative diseases, [122] used multinomial logistic regression and associate mining rule with machine learning to determine microbiome biomarker that is related to the severity of glaucoma by investigating the features of oral microbiome of patients with glaucoma. They obtained the taxonomic composition of the oral microbiome by using a 16S rRNA gene sequencing. Then, they identify the glaucoma biomarkers by using random forest [80] to identify taxon level and the significant taxa is done by using Tag Count Comparison (TCC) [123]. A multinomial logistic regression is used to analyze the significance of the biomarker found in patients with glaucoma based on several factors: age, baseline IOP level, and retinal nerve fiber level (RNFL) thickness. The glaucoma severity is divided based on the RNFL thickness. A general complete ruleset is created by using a classification based on predictive associative rules (CPAR) [124]. Random forest was able to achieve an accuracy between 57% to 74% on genus level identification and Lactococcus was found to have the highest Gini index indicating the significance of Lactococcus in glaucoma patients. [125] created G2S which predicts the taxa of the human fecal microbiome based on the oral microbiome of that individual. They used deep convolutional neural network

(CNN) [50] which was trained on 768 paired samples, oral and fecal samples from 384 individuals. Spearman correlation was used as a metric to measure the correlations between the predicted human fecal microbiome and the actual human fecal microbiome. 52 percent of the prediction achieved spearman correlation of more than 0.8 which were deemed excellent, and 29 percent of the prediction is between 0.71 and 0.8 which were considered good.

4.1.2 Batch Effects

As samples tend to be collected with a difference in location, time taken, handlers, equipment, batch effects are bound to happen due to a large variation in similar samples. Batch effects can be complex, making it difficult to align their distributions while preserving their important biological factors. In order to address these differences, there have been several tools and algorithms that were introduced. [126] proposed using Bayes frameworks to adjust data to remove batch effect which can be robust to outliers and performs well in large dataset. Additionally, [127] described the different normalizations that can be used to adjust batch effect which arises from variation in technology. However, these batch effect removal algorithms suffer from dropout events in gene expressions. Dropout events are when non-zeros values are recorded as zeros due to the low expression of RNA. An approach to resolve this issue is proposed by [128], they identify the cell mappings between datasets and recreate the data distribution in a shared space. The algorithm first identifies the mutual nearest neighbors between the two datasets to establish a relationship. Then, the pair of connections are used to align the dataset in a shared space. This method however requires a lot of computation power for calculating the list of relationships in a high dimensional space. FastMNN was introduced to speed up and reduce the computation power required by

applying MNN in a principal component analysis space. Additionally, [129] also carried out MNN in a reduced space similar to FastMNN to undergo batch effect removal.

The need to carry out batch effect removal first and then running classification is a two steps approach. Our aim is to remove two steps approach and carry them out on a one step approach to improve the efficiency of the process. We will use LassoNet [130], a deep learning algorithm, in combination with batch loss to classify the disease outcome and incorporate batch effect into the network to predict the disease status. LassoNet is also able to determine the importance of each features and how each feature contributes to its prediction. This technique can be applied to gut microbiome in predicting disease severity when combining different gut microbiome datasets as the different types of gut microbiome datasets can also have different batch effect variations. To the best of our knowledge, there has not been a method that combines batch effect and deep learning on oral microbiome data. Additionally, we experimented with autoencoder and several optimizers and loss functions to predict disease status based on the oral microbiome. Our contributions in this chapter are as follows:

- We incorporated LassoNet with batch loss that is able to remove batch effect and classify disease outcome simultaneously
- We evaluated our combined approach on oral microbiome data
- We experimented with autoencoder (combined with attention, robust loss, ranger optimizer, and batch loss) to see if we can obtain high performance on the oral microbiome data

4.2 Materials and Methods

4.2.1 Datasets

We used *Teng 2015* [73], *Agnello 2017* [71], *Gomez 2017* [72], *Vivianne 2020* [74], *Kalpana 2020* [75] for our analysis and classification of severe early childhood caries (ECC) and caries-free (CF) based on the oral microbiome. In Teng 2015, there are 27 samples from children who are caries free and 13 samples who have early child caries. In Agnello 2017, there are 50 samples of which 30 samples are ECC and 20 samples are CF. Only children who were less than 6 years old (72 months) and is a Canadian First Nation were included in Agnello 2017 study. The children with ECC were recruited from Misericordia Health Centre in Winnipeg, Manitoba, Canada and the children with CF were recruited from the community and assessed to ensure that they are caries free. The plaque samples were collected from each child using a sterile interdental brush to swap their tooth surfaces and frozen immediately until further analysis in a glycerol solution at -80 Celsius. In Gomez 2017, the plaque samples were collected from participants in University of Adelaide Craniofacial Biology Research Group (CBRG) and Murdoch Children's Research Institute (MCRI)'s Peri/Postnatal Epigenetic Twins Study (PETS). Participants were refrained from brushing their teeth from the evening before until they visited the clinic the next morning. We included 32 samples into our analysis. In Vivianne 2017, there are 80 total subjects of which 40 (15 males, 25 females) are ECC and 40 (19 males, 21 females) are CF. All of which are less than 6 years old (72 months). In Kalpana 2020, they obtained samples from 55 samples but sequencing was performed on only 30 samples because of low quality DNA.

Fasterq-dump (NCBI SRA-toolkit which can be found at https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=software) was used to download the data

from NCBI repositories. Additionally, some raw sequences were obtained directly from the authors. Qiime2 pipeline was used to analyze and obtain the OTU table at species and genus level. Several combinations of right and left trim were used in DADA2. The performance of DADA2 was manually checked. HOMD 15.22 was used a reference database.

4.2.2 LassoNet

Lasso (least absolute shrinkage & selection operator) is a regression method that uses shrinkage to shrink data points towards a center point. It is very similar to ridge regression; the only difference is its regularization technique. Lasso regression uses L1 regularization while ridge regression uses L2 regularization. The equation for Lasso regression is:

$$\sum_{i=1}^{N}(Y_i - W_l x) + \alpha \sum_{j=1}^{p} |W_l| \quad (4.1)$$

Where Y_i is the ground truth label, N is the total number of samples, W_l is the weights of the Lasso regression, x is the OTUs features, α is the hyperparameter to control the regularization and p is the number of weights. The first term minimizes the distance between the predicted label and the ground truth label while the second term is the regularization that prevents the weights from fitting the training data too well. The regularization prevents overfitting on the training data.

LassoNet consists of a residual connection that connects directly from the input features to the prediction output. This helps to select important features and to prevent features that are redundant to participate in the prediction of the network. The LassoNet objective function is as follows:

$$Objective\ function = L(\theta, W) + \lambda||\theta||_1 \tag{4.2}$$

such that $$\left|\left|W_j^{\,1}\right|\right|_\infty \leq M|\theta_j|$$

and $$L(\theta, W) = \frac{1}{N}\sum_{i=1}^{N} Y_i \log \widehat{Y}_i - (1 - Y_i)\log\left(1 - \widehat{Y}_i\right) \tag{4.3}$$

where $W_j^{\,1}$ is the jth weight of the first layer of the network, λ is the hyperparameter emphasize on the regularization of the residual weights, higher λ will create a sparser network. θ_j is the jth weight of the residual layer, N is the total number of samples, Y_i is the ground truth label for the ith sample, $\widehat{Y}_i$ is the predicted label and M is the hyperparameter to control the strength between using Lasso regression and a standard feed forward neural network. When M is 0, all the hidden units of the neural network becomes inactive and becomes Lasso. When we let M to $+\infty$, it becomes a standard feed forward neural network.

4.2.3 Extension of LassoNet for integrating multiple cohorts

Although LassoNet is a powerful supervised classification tool for high dimensional data set, when it is applied to process high dimensional data from multiple cohorts, which may include strong batch effect, the algorithm may be ineffective. Batch effects occur when samples are collected with a difference in time taken, location, equipment, or handlers. The difference caused in the samples are factors that are not relevant to the experiments which causes inaccurate results. Batch effects can be removed by using several tools and algorithms including normalization technique. However, we want to do a one-step approach in disease classification with batch effect present.

Hence, we design a network architecture that extends the standard LassoNet for classifying disease status using microbiome profiles from multiple cohorts. The network diagram for LassoNet with batch loss can be seen in **Figure *4-1***.

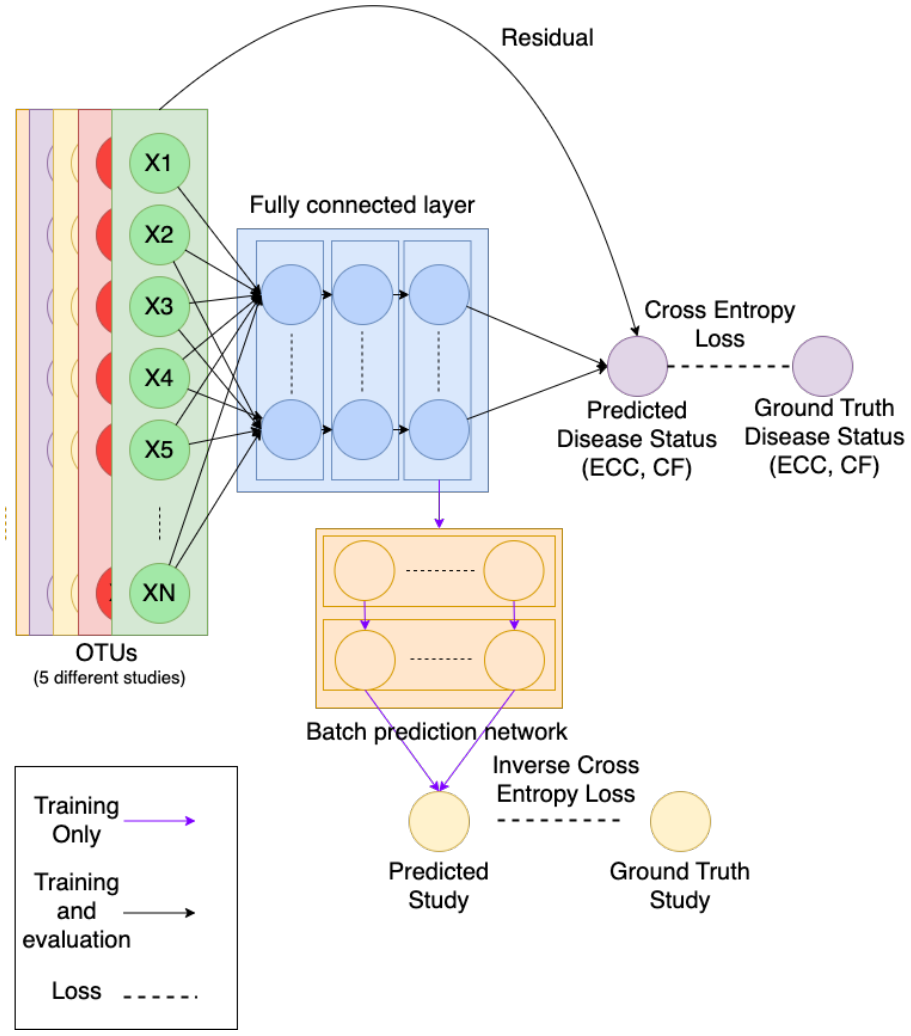


Figure 4-1 LassoNet with batch loss architecture.

In **Figure *4-1*,** the green section is the OTUs of the oral microbiome. The OTUs of the oral microbiome is fed into LassoNet to create the disease status prediction after passing them into fully connected layers with a residual layer connecting the disease status from the OTUs.

Looking back at Equation 4.2, the constraint for setting the weights for the first layer of the fully connected network is less than or equal to $M|\theta_j|$, we can see if we set M = 0, then all the weights for the first layer will be 0, this means that only the residual layer is activated and makes LassoNet becomes the original Lasso. M is used to control the strength between using Lasso and a standard feed forward network.

In the last layer of LassoNet, we added 3 fully connected layer to connect with it using the 3 fully connected layer as a batch prediction network to predict the study category. Instead of doing a normal cross entropy loss that will make the study differentiable to each other, the loss from the study category is inversed to make all the studies as similar as possible such that batch effect can be minimize at best. We do this by maximizing the loss of the prediction from the batch prediction network. The batch prediction network only runs in training, it does not run in evaluation. The normal flow from OTUs of the oral microbiome to the disease status remains the same for evaluation.

4.2.4 Autoencoder

We used an autoencoder network to incorporate batch effect and to classify the disease status simultaneously. We tried several autoencoders and the normal autoencoder performs the best. Our autoencoder network consists of an encoder and a decoder. The encoder creates a latent space that contains information about the data in a low dimensional space through learning the reconstruction of the data and classifying the disease status. Our encoder has two layers including the latent layer. The output of the encoder will be fed into the decoder which consists of 2 layers. The last layer of

the decoder has the same size as the input size of the encoder. The autoencoder will reconstruct the input of the encoder, creating a learned latent space of the input in a lower dimensional space. The loss function of the reconstruction of the input is:

$$recons\ loss = \frac{1}{N}\sum_{n=1}^{N}(\widehat{X}_\iota - X_i)^2 \quad (4.4)$$

Where N is the total number of samples in the current batch, $\widehat{X}_\iota$ is the reconstructed input, X_i is the ground truth.

The latent from the autoencoder is fed into a deep neural network (classification network) with 3 layers to classify the disease status. Before the latent is fed into the classification network, a gated linear unit is used as a gating mechanism to select the latent features to pass into the classification network. Then, the output of the classification network is fed into a self-attention mechanism before classifying the disease status. Additionally, the latent from the autoencoder is also fed into a batch prediction network to carry out an inverse cross entropy loss to make the different study groups as similar as possible by making the network not able to differentiate between the study groups in the latent features. The autoencoder network can be seen in **Figure *4-2*.**

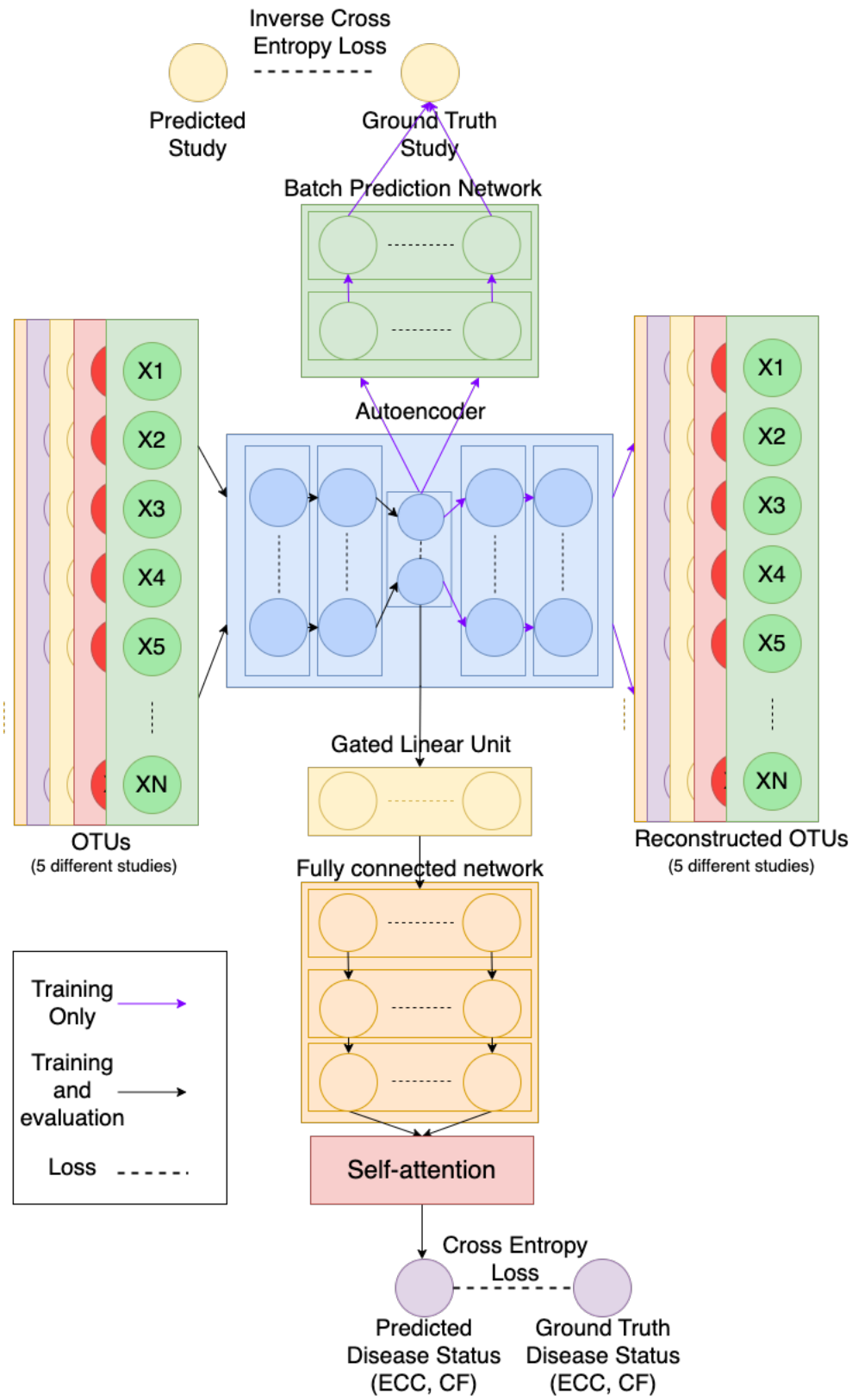


Figure 4-2 Autoencoder with batch loss architecture.

4.2.5 Attention

Attention mechanism has been used to ensure that the model focuses on important features rather than giving the same weights on redundant features. We use a multiheaded attention mechanism as using multiple heads have been shown to learn multiple projections of the features. Each head consists of the following equation:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.5}$$

Where Q is the query, K is the key, V is the value, d_k is the dimension of the kth layer. Q, K, and V are obtained by passing the latent space of the autoencoder to a linear layer:

$$Q = lW_q \tag{4.6}$$

$$K = lW_K \tag{4.7}$$

$$V = lW_V \tag{4.8}$$

Where l is the latent space, W_q is the weights of the query linear layer, W_K is the weights of the key linear layer, and W_v is the weights of the value linear layer. The final multi headed attention mechanisms are the concatenation of the heads:

$$MultiHeadedAttention = Concat(head_1, head_2, \dots, head_3)W_{MH} \tag{4.9}$$

W_{MH} are the weights of the multi headed attention to transform the dimension into a smaller dimension to be fed into the next few linear layers.

4.2.6 Robust Loss

We experimented with robust loss function for the classification network. As the most used loss functions is mean square error (MSE). Mean square error tends to be very sensitive to outliers and will be biased towards the points with the largest errors. The main idea of robust loss is to create a generalized loss function that is very flexible and can take the form of many loss functions. The equation for robust loss is:

$$f\left(x,\alpha,c\right)=\frac{\left|\alpha-2\right|}{\alpha}\left(\left(\frac{\left(x/c\right)^{2}}{\left|\alpha-2\right|}+1\right)^{\alpha/2}-1\right) \tag{4.10}$$

Robust loss can take the form of many loss function and is the superset of many loss functions:

$$\rho\left(x,\alpha,c\right)=\begin{cases}\frac{1}{2}\left(x/c\right)^{2} & \text{if } \alpha=2\\ \log\left(\frac{1}{2}\left(x/c\right)^{2}+1\right) & \text{if } \alpha=0\\ 1-\exp\left(-\frac{1}{2}\left(x/c\right)^{2}\right) & \text{if } \alpha=-\infty\\ \frac{\left|\alpha-2\right|}{\alpha}\left(\left(\frac{\left(x/c\right)^{2}}{\left|\alpha-2\right|}+1\right)^{\alpha/2}-1\right) & \text{otherwise}\end{cases} \tag{4.11}$$

If we look at the equations above, they are l2 loss, l1 loss, cauchy loss, German-McClure loss, Welsch loss respectively depending on the alpha value. C is the value that controls the size of the quadratic bowl near x = 0 for the loss function. The alpha value is adaptive can be learned from backpropagation and using negative log likelihood as not using negative log likelihood will cause

the alpha value to be as small as possible by trying to minimize the cost of outliers. We set x as the L1 loss for the predicted disease status and the ground truth disease status in robust loss:

$$x_i = |\hat{Y}_i - Y_i| \tag{4.12}$$

where x_i is the L1 loss for the ith sample, $\hat{Y}_i$ is the predicted disease status, and Y_i is the ground truth disease status.

4.2.7 Ranger Optimizer

Ranger optimizer combines several components including Adam [131], adaptive gradient clipping [132], gradient centralization [133], Positive-Negative momentum [134], norm loss [135], stable weight decay [136], linear learning rate warm-up [137], explore-exploit learning rate schedule [138], and Lookahead [139].

4.2.8 Inverse Cross Entropy

To incorporate the batch effects, we used an inverse cross entropy loss function on the latent space to remove the difference in distribution between different studies. The inverse cross entropy is as follow:

$$inverse\ CE\ loss = -[\frac{1}{N}\sum_{n=1}^{N} Y_i \log \hat{Y}_i - (1 - Y_i)\log(1 - \hat{Y}_i)] \tag{4.13}$$

However, just having the inverse cross entropy loss would cause the loss to be unstable as it will diverge towards infinity. To resolve this, we clamped the inverse cross entropy loss to -10 at minimum.

$$inverse\ CE\ loss = min(-\left[\frac{1}{N}\sum_{n=1}^{N} Y_i \log \widehat{Y_i} - (1 - Y_i) \log\left(1 - \widehat{Y_i}\right)\right], -10) \quad (4.14)$$

We trained the network simultaneously to create an end to end training. We incorporate the reconstruction loss, the removal of batch effect loss, and the classification loss together. The total loss will become:

$$total\ loss = recons\ loss + classification\ loss + \ inverse\ CE\ loss \quad (4.15)$$

We used Adam optimizer and set the learning rate to 0.001.

4.2.9 Autoencoder Weight Scheduler

As the beginning stage will not contain much information from the latent space, we use a scheduler such that the autoencoder loss is emphasized with a higher weight and the classification loss with a lower weight. The weight of the autoencoder loss is slowly reduced while the classification loss is slowly increased as the network train through each epochs. The weights will focus more on the classification loss towards the end. The weight scheduler for the autoencoder loss can be seen in **Figure *4-3***.

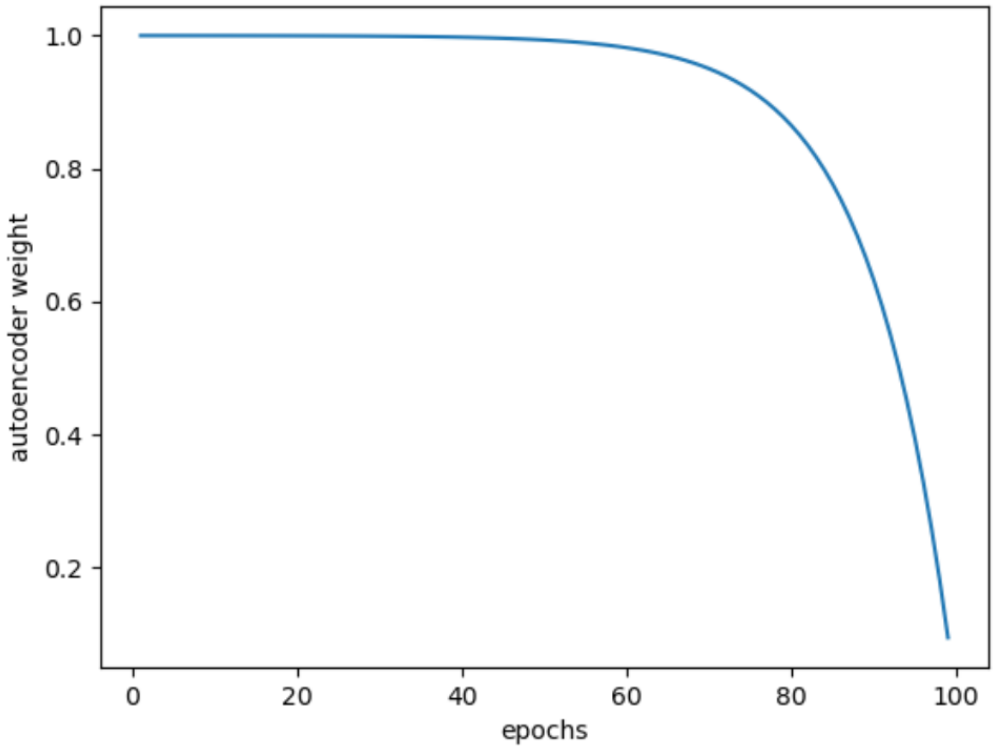


Figure 4-3 Weight decay for autoencoder weights.

With the addition of the scheduler, the total loss can be rewritten as:

$$total\ loss = \alpha * \ recons\ loss + (1 - \alpha) * classify\ loss + \ \gamma * inverse\ CE\ loss \quad (4.16)$$

α starts at 1 and decays to 0 at epoch 100 of which the total loss function will focus solely on the classification loss and the inverse CE loss.

4.2.10 Baseline Models

The baseline models that we will compare against are Lasso Regression and NWCQ [140]. NWCQ is a deep learning framework to remove batch effect while classifying omics data in a single network. Their network consists of 3 modules: calibrator, discriminator, and reconstructor. The calilbrator extracts the latent features and provides the extracted latent features to the discriminator and the reconstructor. The discriminator discriminates between the study groups and the reconstructor reconstructs the OTUs from the extracted latent features. They used two adversarial steps for their training approach. Their approach was shown to improve 10% more than the state-of-the-art methods when evaluated on scRNA-seq dataset and two private metabolomics (MALDI MS) datasets.

Lasso regression is a machine learning model that uses shrinkage to classify data points. We can see the equation and the detailed description of lasso regression at Section **4.2.3**.

4.2.11 Model Evaluation

We used AUC and F1 score to evaluate our models. AUC metric is calculated using receiver operating characteristics (ROC) curve which measures the performance of models at different thresholds by plotting true positive rate against false positive rate. AUC is then obtained by calculating the area under the curve of the ROC curve. The higher the AUC the better.

F1 score is a measurement that takes precision and recall into account when measuring the models' performance. As real-world datasets tend to be imbalanced, using accuracy as measurement can

create a performance metrics that is not accurate. Instead, we use F1 score for the one of the evaluation metrics. The F1 score can be calculated as:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.17}$$

$$Precision = \frac{True\ positive}{(True\ positive + False\ positive)} \tag{4.18}$$

$$Recall = \frac{True\ positive}{True\ positive + False\ negative} \tag{4.19}$$

Where true positives are the correctly predicted positive, false positive are the wrongly predicted positives, false negative are the wrongly predicted negatives.

We carried out the evaluation using leave one study out. There are several studies which use leave one dataset out [141]–[143]. Leave one study out is when we use one dataset as a test set and the rest of the datasets as training sets. Hence, the word "leave one study out".

4.3 Results and Discussion

4.3.1 Strong Batch Effects across Microbiome-based Studies

We carried out our evaluation on the 5 different studies (Agnello 2017, Gomez 2017, Kalpana 2020, Teng 2015, Vivianne 2020) listed in Section 2.1. The plot for the data can be seen in **Figure *4-4***.

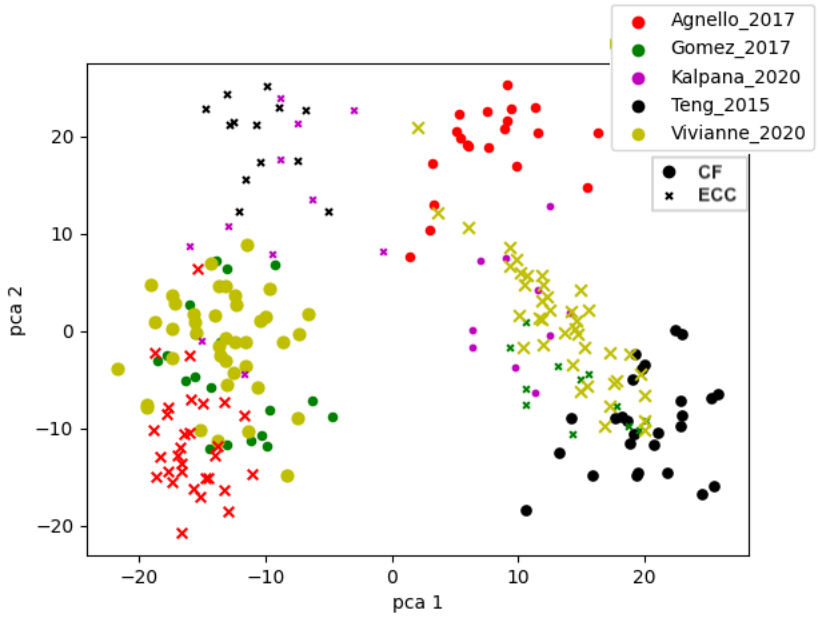


Figure 4-4. Data plot for the 5 studies using PCA.

We can see visible batch effects on the 5 studies. The studies were gone through centered log-ratio (CLR) normalization before being passed into the models. The data plot in figure 3 shown were done after the studies have gone through CLR transformation as CLR transformation have shown to provide meaningful representations to long-term information [144].

4.3.2 Performance Evaluation and Comparison

We ran the evaluation of our autoencoder model and LassoNet on the 5 studies with CLR transformed. The best performing autoencoder model variation is the one with attention, batch loss and robust loss combined together which can be seen in **Table *4- 1***.

Table 4- 1 Performance of our best performing autoencoder and LassoNet on the 5 studies using "leave one dataset out".

	AE with attention + batch loss + robust loss		**LassoNET**	
	AUC	F1	AUC	F1
Agnello 2017	0.85	0.82	0.84	0.86
Gomez 2017	0.67	0.63	0.65	0.6
Kalpana 2020	0.77	0.74	0.76	0.76
Teng 2015	0.43	0.56	0.7	0.59
Vivianne 2020	0.77	0.72	0.88	0.87

We use a leave one dataset out strategy to evaluate our models on the studies. Leave one dataset out refers to training on all 4 datasets and testing on one dataset. In **Table *4- 1***, the performance shown are the datasets that were tested. For example, in Agnello 2017 column, the model is trained on Gomez 2017, Kalpana 2020, Teng 2015, and Vivianne 2020 but tested on Agnello 2017. We used 3 layers for the classification network, 2 layers for the encoder, 2 layers for the decoder, 128 latent size, and 256 hidden size for the classification network. We have tried tuning the hyperparameters for the autoencoder and these hyperparameters are the ones that perform the best.

The result for the LassoNet model can also be seen in **Table *4- 1***. We evaluated LassoNet on the 5 studies using leave one dataset out. The performance of LassoNet outperforms the performance of autoencoder with attention, batch loss, and robust loss on all the studies except for Gomez 2015.

We compared our LassoNet and Autoencoder model with a few baseline models – Lasso Regression and NWCQ [140]. NWCQ is a deep learning framework for removing batch effect and

classifying various omics data in one network. They experimented their approach and showed that their approach was able to remove batch effect effectively and provides high performance in classification results. The comparison of the results between NWCQ, lasso regression and our models can be seen in **Table 4-** 2 .

Table 4- 2 **Comparison of LassoNet (+ batch loss + robust loss) and Autoencoder with the baseline models (NWCQ and Lasso Regression).**

	Autoencoder + attention + batch loss + robust loss		LassoNet + batch loss + robust loss		Lasso Regression		NWCQ	
	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Agnello 2017	0.85	0.72	**0.89**	0.9	0.79	0.8	0.72	0.67
Gomez 2017	0.67	0.56	**0.7**	0.65	0.63	0.4	0.63	0.4
Kalpana 2020	0.77	0.74	**0.86**	0.84	0.56	0.67	0.65	0.76
Teng 2015	0.43	0.63	**0.7**	0.6	0.52	0.22	0.57	0.45
Vivianne 2020	0.77	0.82	**0.86**	0.84	0.83	0.83	0.72	0.69
Average	0.698	0.694	**0.798**	0.766	0.67	0.584	0.658	0.594

We can see that LassoNet with batch loss and robust loss outperforms all the models in 4 of the studies. LassoNet with batch loss also obtained the highest average AUC of 0.798.

4.3.3 Ablation Study

We carried out an ablation study on LassoNet as it is the best performing model. We experimented adding batch loss, robust loss, and ranger optimizer to LassoNet and determine if there is a performance improvement in the network. The ablation study on LassoNet can be seen in **Table 4- 3**.

Table 4- 3 . Ablation study on LassoNet. Results are in AUC.

	Agnello 2017	Gomez 2017	Kalpana 2020	Teng 2015	Vivianne 2020	Average
LassoNet	0.84	0.65	0.76	**0.7**	0.88	0.766
+ Robust loss	0.88	0.68	0.76	0.68	0.85	0.77
+ Ranger	0.75	**0.76**	0.71	0.6	0.8	0.724
+ batch loss	0.86	0.68	0.82	**0.7**	**0.9**	0.792
+ ranger + robust	0.78	0.62	0.76	0.63	0.74	0.706
+ batch loss + robust	**0.89**	0.7	**0.86**	0.68	0.86	**0.798**

On average, LassoNet with batch loss and robust loss performs the best, achieving 0.798 AUC. However, LassoNet with batch loss only was able to achieve the best performance in Teng 2015 and Vivianne 2020. We suspect that due to the LassoNet being able to select the important features to determine the disease status with regards to the studies, the performance of LassoNet vs the best performing LassoNet variation (LassoNet + batch loss + robust) is not very different.

4.3.4 Feature Importance

We extracted the importance of the oral microbiome features as detected by LassoNet which was made possible due to the nature of its architecture. The feature importance can be seen in **Figure *4-5***.

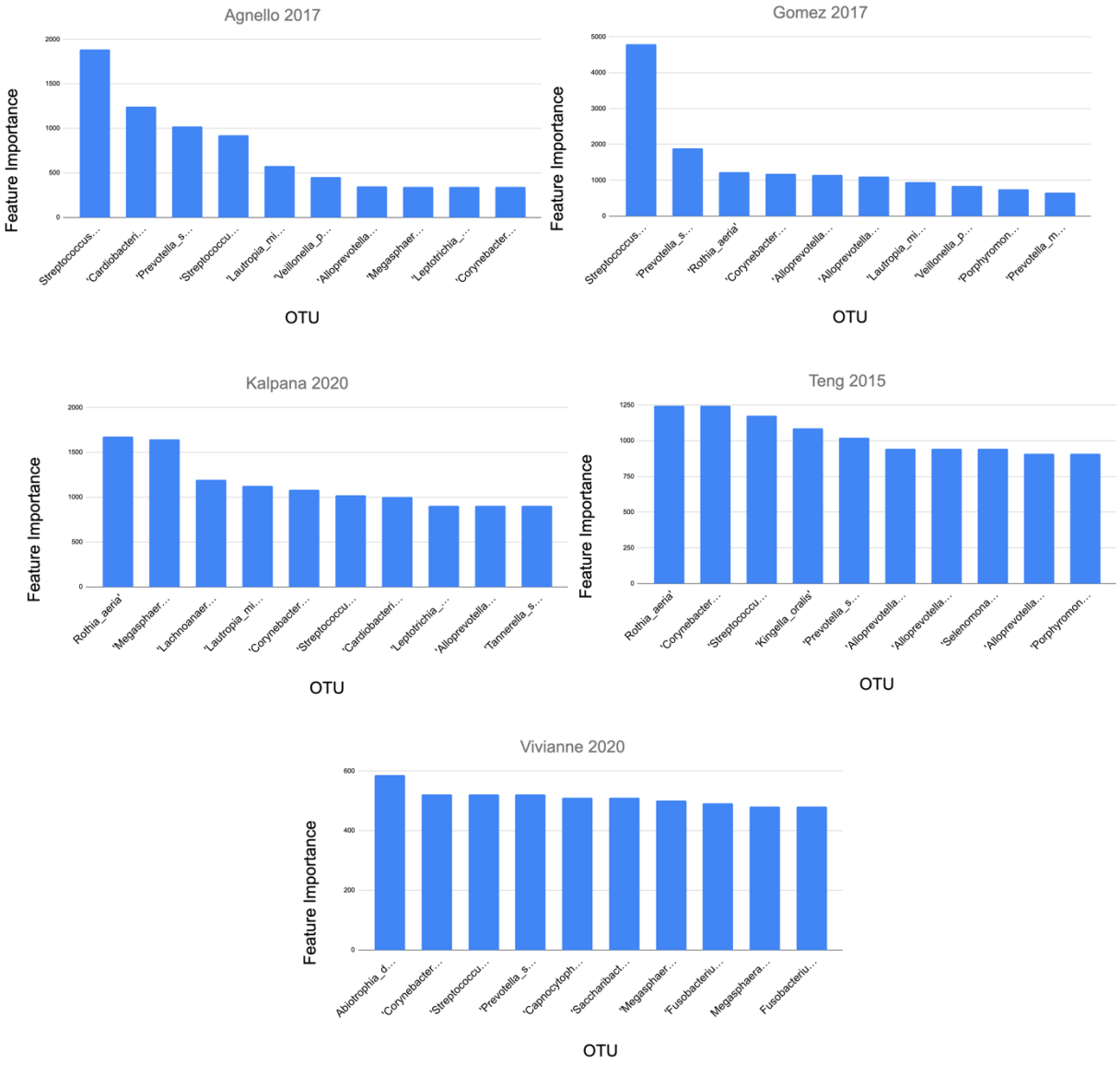


Figure 4-5 Feature importance from LassoNet with batch loss and robust loss on the 5 studies.

We can see that Agnello 2017 and Gomez 2017 has the same top 1 feature, Streptococcus_mutans, Kalpana 2020 and Teng 2015 has the same top 1 feature, Rothia_aeria, and Vivianne 2020 has Abiotrophia_defectiva as its top 1 feature. We analysed and created a table that shows the most common top 5 features among the 5 studies as shown in **Table *4- 4***. It seems that Prevotella_salivae is an OTU that exist in all the studies except for Kalpana 2020 among the top 5 features.

Table 4- 4 Top 5 features occurrence among 5 studies. We looked at the top 5 OTUs of each study and showed the most frequently occurring OTUs.

LassoNet + batch loss + robust loss	
OTUs	Number of occurrence in top 5 features
Prevotella_salivae	4
Streptococcus_mutans	3
Rothia_aeria	3
Lautropia_mirabilis	2
Corynebacterium_durum	2
Corynebacterium_matruchotii	2
LassoNet	
OTUs	Number of occurrence in top 5 features
Streptococcus_mutans	5
Prevotella_salivae	4
Alloprevotella_sp._HMT_912	3
Streptococcus_salivarius	2
Leptotrichia_goodfellowii	2

Based on our evaluation on LassoNet with batch loss, it seems that Streptococcus_mutans and Prevotella_salivae plays an important role in determining the disease status of patients as they are the top 2 features in LassoNet with batch loss, robust loss and LassoNet respectively.

4.4 Summary

We showed that deep learning, specifically using autoencoder and LassoNet, with incorporated batch effect is able to obtain good performance when predicting disease status through oral microbiome on the 5 studies: *Agnello 2017, Gomez 2017, Kalpana 2020, Teng 2015, Vivianne 2020.* We compared our autoencoder and LassoNet extensions with two baseline models: NWCQ and Lasso Regression and showed that our model extensions were able to perform better than the baseline models. When comparing the best performing autoencoder (*attention, batch loss, robust loss*) with LassoNet, LassoNet outperforms autoencoder even without the additional special optimizer and loss functions. We suspected that due to the nature of LassoNet being able to select and choose the important features, LassoNet is able to achieve a higher performance than the best performing autoencoder. We also showed the important features selected by LassoNet and LassoNet with the additional optimizer and loss functions and found that within the top 5 most occurring features among the 5 different study groups, Streptococcus_mutans and Prevotella_salivae appeared the most times indicating that the two features contribute significantly to the classification of the disease status.

Chapter 5 Deep learning-based joint detection in Rheumatoid arthritis hand radiographs

5.1 Introduction

Technology has advanced exponentially as the years have passed. Artificial intelligence (AI) has been used to improve the advancement of a diverse range of fields including but not limited to speech recognition, recommender system (predicts the preference of users), computer vision, self-driving cars, natural language processing, and translation [145]. Deep learning is a subset of AI that can surpass the performance in content creation or classification of rule-based AI or machine learning if there is enough data to train on. Deep learning helps to classify or predict complex solutions including object detection, speech translation, image generation, music generation, etc. In the medical field, deep learning has been used to detect radiographic evidence of pneumonia [146], [147], obtained good representation of gut microbiome data for easier future analysis [148], and cancer diagnosis [149].

Rheumatoid arthritis (RA) is an autoimmune disease where the immune system mistakenly attacks healthy joints causing joint damage. [38][39]There are several scoring methods proposed for evaluating radiographs in rheumatoid arthritis [40]–[44]. The current "gold standard" scoring tool is the Sharp/van der Heijde (SvH) score which assesses joint space narrowing (JSN) and joint erosions in multiple hand and feet joints. Although the SvH method is widely used for clinical research studies, its use in clinical practice is limited as SvH radiograph scoring requires a highly skilled assessor and is time consuming.

Some technological advancements have addressed these challenges by utilizing deep learning to assess the joint damage directly [39], [150]. Maziarz et al proposed using a deep multi-task method [151] that predicts joint space narrowing and erosion scores using a deep convolutional neural network [50]. The network simultaneously performs joint detection, and predicts joint space narrowing and erosion scores. This group was able to achieve a root mean squared error of 0.4075 and 0.4607 for narrowing and erosion respectively in the RA2-DREAM Challenge [152] from 119 patients obtained from two NIH supported clinical studies. Additionally, Hirano et al [153] created a two step method to perform joint score evaluation using deep learning. The steps are 1) joint detection and 2) joint evaluation. The joint detection step uses a machine learning algorithm that uses Haar-like features [154] to detect the joints. The joint evaluation step uses a convolutional neural network to assign scores to the joints. They evaluated their model on 30 patients that with diagnosed with RA based on standard criteria [155]. They were able to achieve detection of PIP, Interphalangeal (IP), and MCP joints with sensitivity of 95.3%. The accuracy for erosion detection was between 70.6% to 74.1% while the accuracy for JSN was 49.3% to 65.4%.

However, there are no known AI methods that show that we could train on normal or healthy joint images and evaluate them on rheumatoid arthritis joint images. We used one of the state-of-the-art object detection algorithms, YOLOv5l6 [156], to detect the finger joints (PIP, and MCP) and wrist joints (wrist, radius, and ulna). We used a pre-trained YOLOv5l6 on Common Object in Context (COCO) dataset and showed that it is transferable and can be used to detect x-ray joints. We showed that the YOLOv5l6 model trained on imaging from the RSNA Pediatric BoneAge challenge [157] with only left-hand x-ray images was able to detect joints of rheumatoid arthritis patients with left hand, right hand, or both hands in the images in our Manitoba dataset.

5.2 Methods

The dataset that we obtained is from RSNA 2017 Pediatric BoneAge Challenge [157]. The dataset contains images of children's hand skeleton with skeletal age between 0 to 216 months. It contains a total of 14,236 hand skeletal images of which 47% of them are female with mean age of 129 months. There was no description whether the children in the dataset are healthy or unhealthy as the challenge is to promote the showcase of machine learning on x-ray hand images. The annotations for the joints were annotated in https://github.com/razorx89/rsna-boneage-ossification-roi-detection from the RSNA Pediatric BoneAge Challenge dataset (**Figure 5-1**). There are 240 training images in total and 89 evaluation images annotated with region of interests of the joints. All of the images consist only x-ray images of the left hand. The region of interests were labeled with DIP, PIP, MCP, Wrist, Radius, and Ulna. We included only PIP, MCP, wrist, radius, and ulna in the training. Each image contains 5 PIP joints, 5 MCP joints, 1 wrist, 1 radius, and 1 ulna joint which can be seen in **Figure 5-1**.

In addition, a test set with serial radiographs collected from participants enrolled in the prospective Manitoba Early Arthritis Cohort (EAC) were used to validate the algorithm/tool. The Manitoba EAC is an inception cohort that enrolls adults with recent onset RA and collects clinical data and radiographic images as part of a study protocol. Participants have been followed for up to 10 years. The cohort and related studies have been approved by the Ethics review board at the University of Manitoba.

5.2.1 Model Architecture

In order to detect each joint, we first used YOLOv5l6 to train on the dataset to predict the joint. The difference between YOLOv5 and YOLOv5l6 is that YOLOv5 has 3 output layers while YOLOv5l6 has 4 output layers. Having 4 different output layers increases the total number of scales the model looks at and can enhance performance. YOLOv5l6 is an object detection algorithm that uses convolutional neural network as one of its building blocks to detect objects. It only requires a single pass to the neural network to detect all the objects in the image. YOLOv5l6 consists of 3 sections – backbone, neck, and head. The backbone section uses a cross stage partial network (CSP) [158] to generate feature maps at multiple levels, similar to that of a feature pyramid network [159]. The lowest level of the backbone is replaced with spatial pyramid pooling to remove the requirement of having a fixed-constraint image size. The output from the multiple level of CSP is passed into a path aggregation network (PANet) [160]. PANet generates feature pyramids that are useful to generalize features in multiple scales. This helps with detecting objects of different sizes. PANet upscales and concatenates the features from the backbone, and then passes the features into a bottom-up augmentation before passing them to the head of the model to be used for prediction. [161] stated that bottom-up path augmentation can capture both global and local features as higher-level layers correspond to the entire object while lower-level layers correspond to local patterns and features. This enhances the ability of the network to classify objects. The architecture of YOLOv5 is shown in **Figure 5-2** which was obtained from an online message group [162].

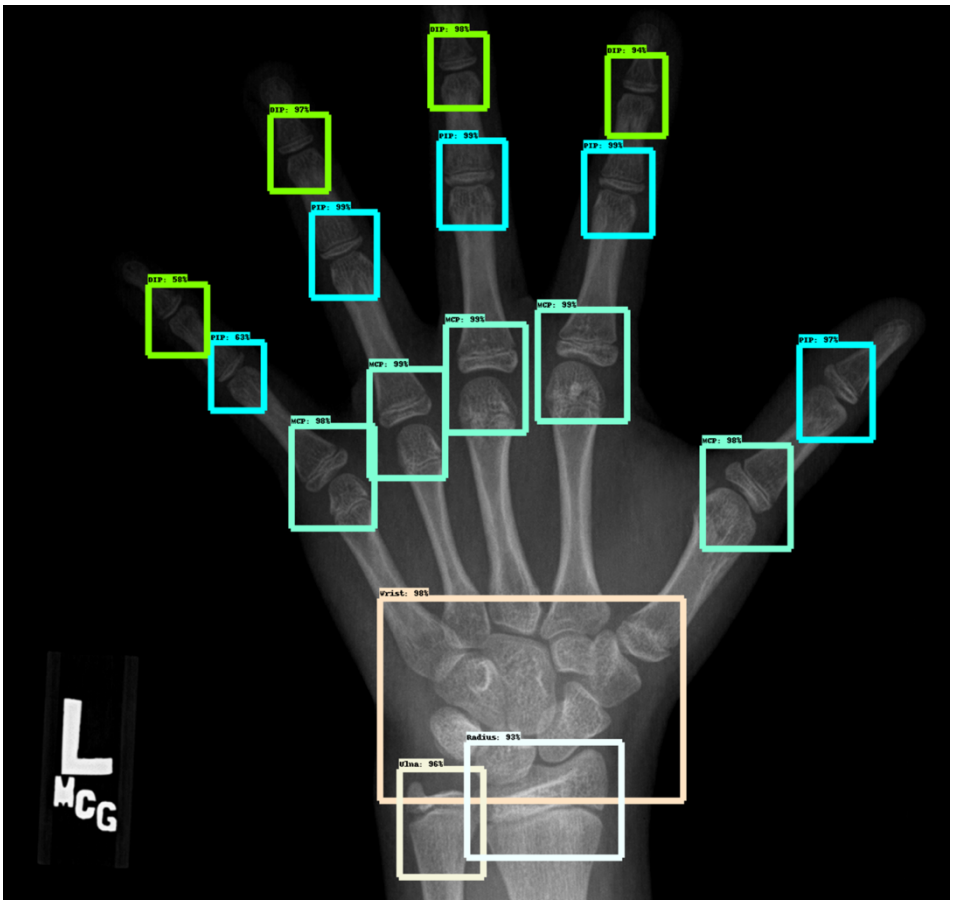


Figure 5-1 An example of an x-ray image from the RSNA Pediatric BoneAge Challenge with annotated joints

The loss functions of YOLOv5l6 consists of 3 parts, object loss, classification loss, and bounding box regression loss.

The object loss is:

$$obj\ loss = -\sum_{i=0}^{S^2}\sum_{j=0}^{B} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1-\hat{C}_i\right)\log(1-C_i)\right] - \sum_{i=0}^{S^2}\sum_{j=0}^{B} I_{ij}^{noobj}\left[\hat{C}_i \log(C_i) + \left(1-\hat{C}_i\right)\log\left(1-C_i\right)\right] \quad (5.1)$$

S is the size of the grid of the images divided by yolo, B is the number of bounding boxes, $\hat{C}_i$ is the prediction of the existence of an object in grid i, and C_i is the ground truth of the existence of an object in grid i. As there are many grid cells that does not contain an object, this will skew the model to predict all 0s (no object in grid cell). To solve this, I_{ij}^{obj} and I_{ij}^{noobj} are used. I_{ij}^{obj} emphasizes more weights on the grids that contain an object while I_{ij}^{noobj} will reduce the weights on the grids that does not contain an object.

The classification loss is:

$$classification\ loss = -\sum_{i=0}^{S^2} \sum_{c \in classes} [\hat{p}_i(c) \log(p_i(c)) + (1 - \hat{p}_i(c)) \log(1 - p_i(c))] \quad (5.2)$$

$\hat{p}_i(c)$ is the confidence of the class predicted and $p_i(c)$ is the ground truth label.

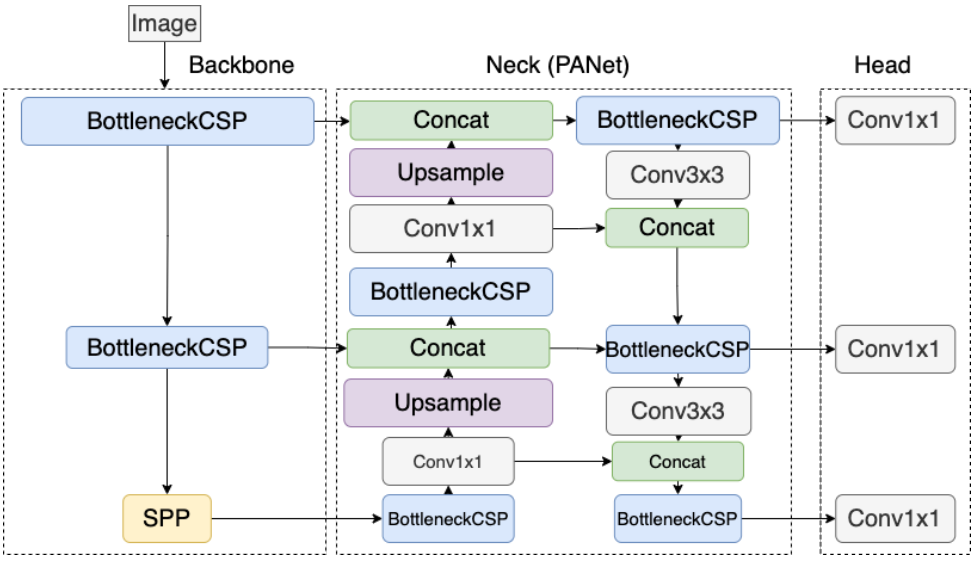


Figure 5-2 The architecture of YOLOv5 model

The bounding box regression loss is:

$$IoU\ loss = 1 - IoU + \frac{p^2(b,b^{gt})}{c^2} + \alpha v \tag{5.3}$$

p^2 is the Euclidean distance and c is the diagonal length of the smallest box covering $b\ and\ b^{gt}$.

5.2.2 Model Training

We used YOLOv5l6 to detect PIP, MCP, Radius, Wrist, and Ulna and removed the DIP labels as they are not important in determining damage for rheumatoid arthritis. We started the training from a pre-trained YOLOv5l6 model on the COCO dataset (**Figure 5-3**). We added several augmentations to the image before feeding them into YOLOv5l6 model including mixup [163] (mixing up features of different images together into one), mosaic [49] (combines 4 training images into 1 image), rotation, translation, scaling, and shearing augmentations.

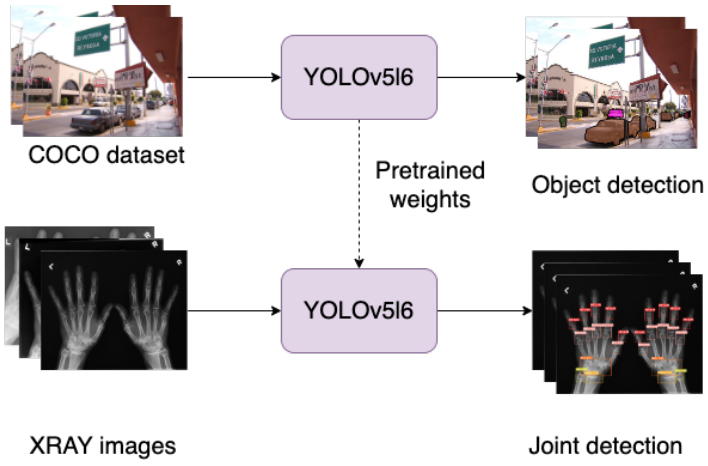


Figure 5-3 Pre-training YOLOv5l6 model on COCO dataset before training on joint detection on x-ray images of hands

5.2.3 Model Evaluation

We used the intersection over union (IOU) metric to evaluate the performance of the joint detection. IOU is a method that looks at the overlapping area of the predicted output and the ground truth mask. The equation for IOU is:

$$IoU = \frac{predicted\ output \cap ground\ truth\ mask}{predicted\ output \cup groud\ truth\ mask} \tag{5.4}$$

Instead of using accuracy to measure the performance of the object detection model, we used F1 score. While accuracy is one of the ways to measure performance, it is not the best metric to measure skewed distributions. We used F1 score because F1 score is a much better evaluation metric that treats both positive classes and negative classes equally. The equation for F1 score is as follows:

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{5.5}$$

where precision is:

$$precision = \frac{TP}{TP+FP} \tag{5.6}$$

and recall is:

$$recall = \frac{TP}{TP+FN} \tag{5.7}$$

TP is true positive, FP is false positive, FN is false negative. An example of this would be if there is a PIP joint and the model predicted it as PIP, then it is considered as true positive. PIP joints that are detected as MCP joints are false negatives. MCP joints which are detected as PIP joints are false positive. The same applies to the other joint types.

We provided a set of IOU threshold and any predicted bounding boxes that have an IOU of greater than the IOU threshold when compared to the ground truth bounding boxes were considered as the prediction.

5.2.4 Results

We set the confidence threshold for our model to be 0.35 with 0.1 non-maximum suppression (NMS) IOU threshold as it seemed to remove wrongly predicted joints when we looked at the detection ourselves. Any joint predictions that did not have a confidence threshold of more than 0.35 was removed. As for the NMS IOU threshold, any boxes that have a lower scoring box but with IOU greater than the NMS IOU threshold with another higher scoring box was removed.

Table *5- 1* shows the average precision on different values of IOU with the ground truth labels. We can see that there is still a high performance with the overlap between the predicted boxes with the ground truth labels when the IOU is 0.3. Even though this is not much, the joints detected were still located at the fundamental joints area indicating that the model could have either predicted the boxes too big or too small but is still able to capture the joints location.

YOLOv5l6 that is pre-trained on COCO is able to achieve a better performance than no pre-training. **Table 5-** 2 shows the F1 score, precision, and recall metrics for the joints with 0.35 confidence threshold and 0.1 NMS IOU threshold. All joint types were able to achieve at least 0.98 F1 score performance when evaluated on the evaluation set.

Table 5-1 Joint detection average precision on different thresholds

Pre-trained on COCO										
Threshold /Average precision	0.1	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.5	0.55
PIP	0.99	0.99	0.99	0.99	0.97	0.87	0.73	0.55	0.38	0.26
MCP	0.99	0.99	0.97	0.89	0.75	0.58	0.42	0.27	0.2	0.16
Wrist	0.99	0.99	0.99	0.99	0.97	0.97	0.94	0.93	0.92	0.88
Radius	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.97	0.93
Ulna	0.99	0.99	0.99	0.99	0.97	0.97	0.96	0.96	0.82	0.7
No Pre-trained										
PIP	0.98	0.98	0.98	0.98	0.97	0.93	0.85	0.69	0.46	0.3
MCP	0.98	0.98	0.95	0.89	0.82	0.71	0.60	0.46	0.35	0.26
Wrist	0.99	0.99	0.98	0.96	0.94	0.94	0.91	0.89	0.89	0.84
Radius	0.98	0.98	0.98	0.98	0.98	0.98	0.97	0.94	0.93	0.86
Ulna	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.65	0.48

Table 5- 2 Joint detection metrics

pre-trained on COCO					
	PIP	MCP	Wrist	Radius	Ulna
F1	0.994	0.991	0.993	0.988	0.987
Precision	1	0.993	1	1	0.997
Recall	0.987	0.989	0.986	0.976	0.978
No pre-trained					
F1	0.928	0.971	0.921	0.977	0.504
Precision	0.883	0.986	0.87	1	1
Recall	0.978	0.957	0.978	0.955	0.337

The mean metrics are shown in **Table 5- 3** . Map0.1 is the mean average precision with IOU threshold set as 0.1, map is the mean average precision from 0.1 IOU threshold to 0.55 IOU threshold. The PIP and MCP joints have the highest number of targets as there exist 5 joints for each of them while only 1 joint each for wrist, radius, and ulna.

Table 5- 3 Joint detection mean metrics

Pre-trained on COCO						
	Seen	Number targets	Mean precision	Mean recall	map0.1	map
All	89	1157	0.998	0.983	0.993	0.854
PIP	89	445	1	0.987	0.994	0.772
MCP	89	445	0.993	0.989	0.994	0.624
Wrist	89	89	1	0.986	0.994	0.958

Radius	89	89	1	0.976	0.994	0.983
Ulna	89	89	0.997	0.978	0.988	0.933
No pre-trained						
All	89	1157	0.948	0.841	0.919	0.81
PIP	89	445	0.883	0.978	0.984	0.814
MCP	89	445	0.986	0.957	0.977	0.7
Wrist	89	89	0.87	0.978	0.987	0.932
Radius	89	89	1	0.955	0.977	0.957
Ulna	89	89	1	0.337	0.669	0.647

Figure 5-4 shows the visualization of the joint prediction by YOLOv5l6 on the Manitoba dataset of RA patients. They were all correctly predicted by YOLOv5l6. YOLOv5l6 was able to predict the joints regardless of the image sizes. It was also able to predict joints correctly on images that contain either left/right hand or both hands even though the training data consists only of left hands.

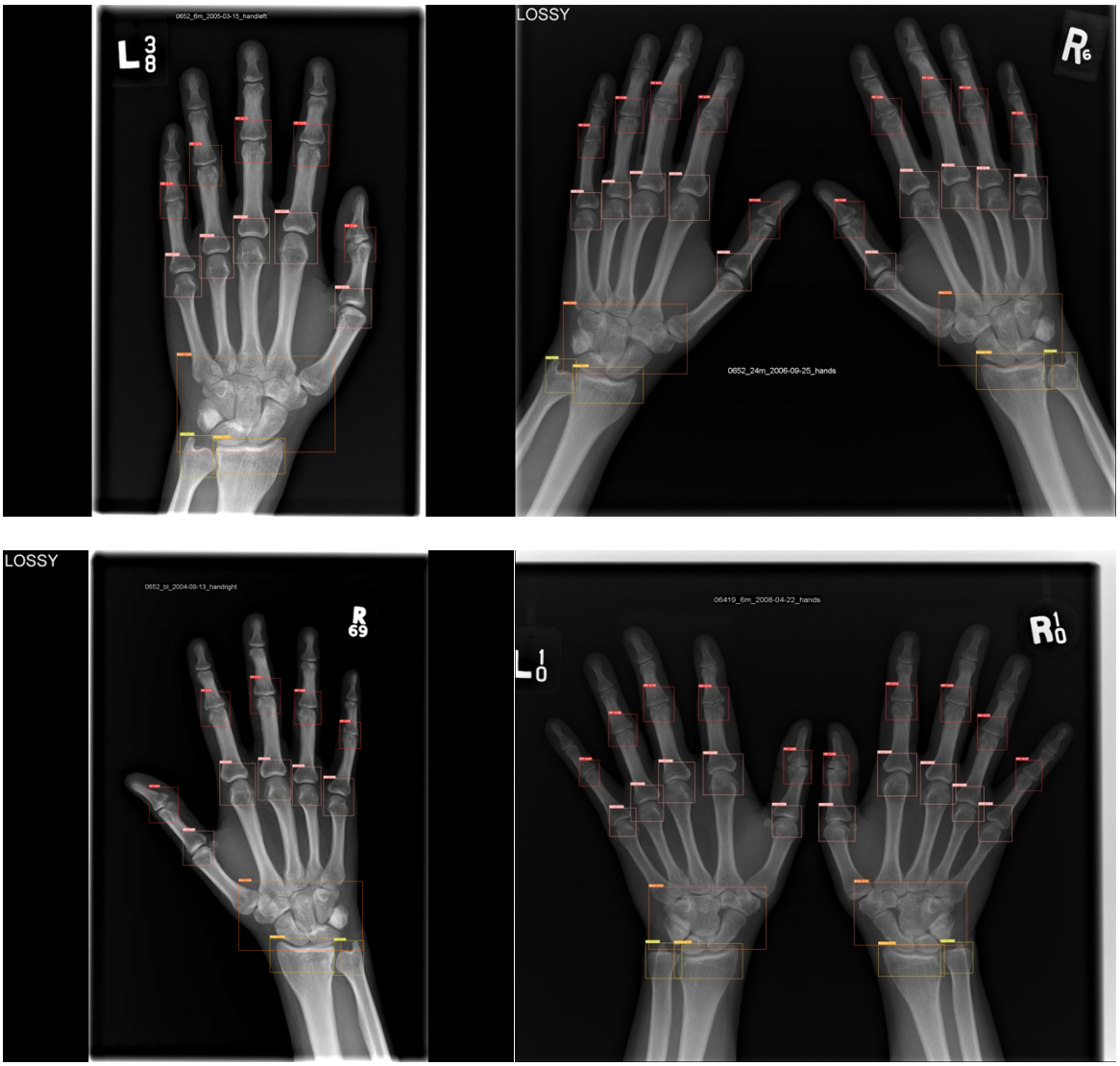


Figure 5-4 Visualization of predicted joints (PIP, MCP, wrist, radius, ulna) of rheumatoid arthritis patients by YOLOv5l6.

We showed an example of the extracted joints for a patient's left hand in **Figure *5-5*.** This image shows a close-up picture of the extracted joints of PIPs, MCPs, wrist, radius and ulna joints.

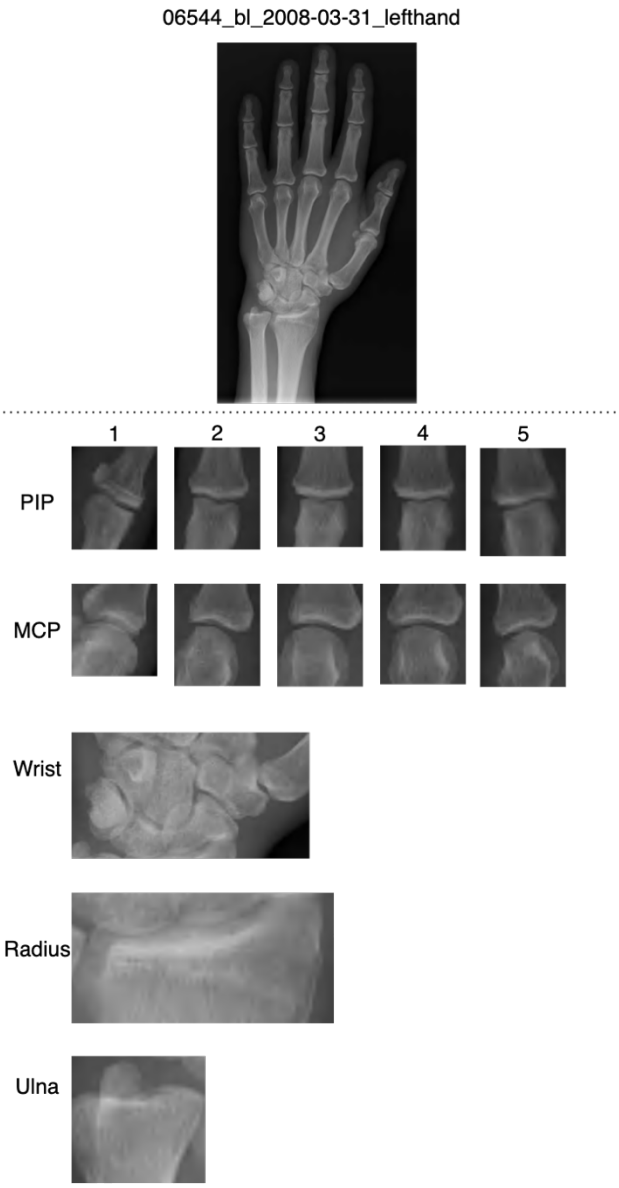


Figure 5-5 Extraction and numbering of joints of hand.

5.3 Summary

In this chapter, we showed YOLOv5l6 is capable of detecting joints (PIP, MCP, wrist, ulna, radius) of rheumatoid arthritis patients even if YOLOv5l6 was trained on left hand joints of healthy patients. YOLOv5l6 was able to achieve a higher performance when the pre-trained weights were started on weights that were trained on the COCO dataset.

Chapter 6 Conclusion and Future Direction

6.1 Conclusion

In conclusion, this thesis shows how deep learning can be applied to gut microbiome, oral microbiome for disease status prediction and joint detection on rheumatoid arthritis datasets without rheumatoid arthritis joints datasets. We could use statistical models to apply to the datasets, statistical models will be able to find, explain relationships and establish significance of relationships between variables much better than deep learning. However, statistical models would not be able to achieve a higher performance than deep learning models as statitical models are not the best with complex distributions.

Even though deep learning models are high performing, they can be difficult to apply to healthcare data as it is hard to trust models that have no explanation with their predictions. It can be also difficult for deep learning to apply knowledge learned from a domain to a totally different domain since healthcare consists of a lot of complicated causal connections that lead to one another.

In **Chapter 3**, we described how deep learning can be used to extract information from the temporal information of gut microbiome to predict disease severity. Due to the nature of gut microbiome, it is very difficult to obtain longitudinal gut microbiome data without missing information. Missing information can impose a challenge for the analysis and classification of data using deep learning. We experimented with data imputation and padding techniques to resolve the missing information in longitudinal gut microbiome. Additionally, deep learning tends to require a large amount of data to achieve a stable performance. However, there tends to be a limited amount of longitudinal gut microbiome data available as it is challenging to obtain them. We

proposed to use self-knowledge distillation to resolve this. Specifically, we used CNN-LSTM with self-knowledge distillation and padding techniques to resolve these challenges. We were able to obtain a high AUC when evaluated the deep learning model on DIABIMMUNE study and PROTECT study.

In **Chapter 4**, we showed how deep learning can be used to incorporate batch effects into predicting disease status through oral microbiome. As samples tend to be collected with a difference in location, time taken, equipment, and handlers, difference in measurement for similar samples are bound to happen creating batch effects in samples. We proposed to use an autoencoder and LassoNet while incorporating batch effects from samples into them to learn the distributions and to predict the disease status (CF, ECC) through the oral microbiome. We carried out a "leave one dataset out" on 5 different studies: *Teng 2015, Agnello 2017, Gomez 2017, Kalpana 2020, and Vivianne 2020* and showed that LassoNet is able to outperform the autoencoder and several baseline methods that we compared against (*NWCQ* and *Lasso regression*).

In **Chapter 5**, we carried out joint detection on rheumatoid arthritis patients using YOLOv5l6. As there is a lack of dataset of rheumatoid arthritis patients, this poses a challenge for deep learning since deep learning requires a large amount of dataset to be able to achieve a high performance. Instead of using data from rheumatoid arthritis patients for joint detection, we used joints from normal patients to train YOLOv5l6 on, then used the trained YOLOv5l6 to detect joints on rheumatoid arthritis patients. Specifically, we predict *PIP, MCP, wrist, radius*, and *ulna* of the hands. We used RSNA 2017 Pediatric BoneAge Challenge for our training dataset of healthy joints and evaluated on the Manitoba dataset of joints with rheumatoid arthritis. We also showed that

starting YOLOv5l6 on a pre-trained COCO dataset is able to improve the performance for joint detection on rheumatoid arthritis patients.

6.2 Future Direction

In the future, we can apply the self-knowledge distillation techniques to other microbiome dataset or to areas where there is a limited amount of dataset for deep learning to be able to achieve a good performance.

Batch effects are carried out by using two-step approach in the past, and the batch effects are removed using some normalization technique followed by a statistic model to carry out disease classification. Instead of using two-step approach, we incorporated batch effects directly into LassoNet to carry out a one-step approach which could save time and run a more efficient approach. For future work, this research can be extensible to other areas of the human body other than the gut and oral microbiome. Other kinds of disease classification can also be extended including but not limited to cancer, tumours, nervous system disease, Parkinson's disease, and Alzheimer's disease.

As for the future work on rheumatoid arthritis experiment, we will use this tool to identify/extract the joints detected on radiographic images from rheumatoid arthritis patients, classify the severity joint damage (joint space narrowing and erosions), and correlate our deep learning algorithm /tool with clinician assessed radiographic damage. Additionally, we will use deep learning to extract joints in combination with serially collected radiographs and longitudinal clinical data to predict changes in rheumatoid joint damage over time. This application may assist clinicians caring for individuals with rheumatoid arthritis by informing RA treatment strategies.

References

[1] P. Hugenholtz, B. M. Goebel, and N. R. Pace, "Impact of culture-independent studies on the emerging phylogenetic view of bacterial diversity," *Journal of Bacteriology*, vol. 180, no. 18. 1998. doi: 10.1128/jb.180.18.4765-4774.1998.

[2] R. I. Amann, W. Ludwig, and K. H. Schleifer, "Phylogenetic identification and in situ detection of individual microbial cells without cultivation," *Microbiological Reviews*, vol. 59, no. 1. 1995. doi: 10.1128/mmbr.59.1.143-169.1995.

[3] J. R. Marchesi, "Metagenomics: Current Innovations and Future Trends," *Future Microbiol*, vol. 7, no. 7, 2012, doi: 10.2217/fmb.12.41.

[4] J. Wang *et al.*, "A metagenome-wide association study of gut microbiota in type 2 diabetes," *Nature*, vol. 490, no. 7418, 2012, doi: 10.1038/nature11450.

[5] T. Reynolds, B. Kuska, and L. Curtis, "NIH Human Microbiome Project defines normal bacterial makeup of the body | National Institutes of Health (NIH)," *National Institutes of Health*, 2012.

[6] R. Sender, S. Fuchs, and R. Milo, "Revised Estimates for the Number of Human and Bacteria Cells in the Body," *PLoS Biol*, vol. 14, no. 8, 2016, doi: 10.1371/journal.pbio.1002533.

[7] S. Vieira-Silva *et al.*, "Statin therapy is associated with lower prevalence of gut microbiota dysbiosis," *Nature*, vol. 581, no. 7808, 2020, doi: 10.1038/s41586-020-2269-x.

[8] D. Rothschild *et al.*, "Environment dominates over host genetics in shaping human gut microbiota," *Nature*, vol. 555, no. 7695, 2018, doi: 10.1038/nature25973.

[9] S. J. Carter *et al.*, "Gut microbiota diversity is associated with cardiorespiratory fitness in post-primary treatment breast cancer survivors," *Exp Physiol*, vol. 104, no. 4, 2019, doi: 10.1113/EP087404.

[10] G. K. Fragiadakis, H. C. Wastyk, J. L. Robinson, E. D. Sonnenburg, J. L. Sonnenburg, and C. D. Gardner, "Long-term dietary intervention reveals resilience of the gut microbiota despite changes in diet and weight," *American Journal of Clinical Nutrition*, vol. 111, no. 6, 2020, doi: 10.1093/ajcn/nqaa046.

[11] H. Liu *et al.*, "Resilience of human gut microbial communities for the long stay with multiple dietary shifts," *Gut*, vol. 68, no. 12. 2019. doi: 10.1136/gutjnl-2018-317298.

[12] E. Zagato *et al.*, "Endogenous murine microbiota member Faecalibaculum rodentium and its human homologue protect from intestinal tumour growth," *Nat Microbiol*, vol. 5, no. 3, 2020, doi: 10.1038/s41564-019-0649-5.

[13] A. Kumar and V. Sperandio, "Indole signaling at the host-microbiota-pathogen interface," *mBio*, vol. 10, no. 3, 2019, doi: 10.1128/mBio.01031-19.

[14] M. Funabashi *et al.*, "A metabolic pathway for bile acid dehydroxylation by the gut microbiome," *Nature*, vol. 582, no. 7813, 2020, doi: 10.1038/s41586-020-2396-4.

[15] F. Guarner and J. R. Malagelada, "Gut flora in health and disease," *Lancet*, vol. 361, no. 9356. 2003. doi: 10.1016/S0140-6736(03)12489-0.

[16] J. K. Nicholson, E. Holmes, and I. D. Wilson, "Gut microorganisms, mammalian metabolism and personalized health care," *Nat Rev Microbiol*, vol. 3, no. 5, 2005, doi: 10.1038/nrmicro1152.

[17] M. Egert, A. A. de Graaf, H. Smidt, W. M. de Vos, and K. Venema, “Beyond diversity: Functional microbiomics of the human colon,” *Trends in Microbiology*, vol. 14, no. 2. 2006. doi: 10.1016/j.tim.2005.12.007.
[18] H. J. Flint, S. H. Duncan, K. P. Scott, and P. Louis, “Interactions and competition within the microbial community of the human colon: Links between diet and health: Minireview,” *Environmental Microbiology*, vol. 9, no. 5. 2007. doi: 10.1111/j.1462-2920.2007.01281.x.
[19] E. M. Bik, “Composition and function of the human-associated microbiota,” in *Nutrition Reviews*, 2009, vol. 67, no. SUPPL. 2. doi: 10.1111/j.1753-4887.2009.00237.x.
[20] N. Heym *et al.*, “The role of microbiota and inflammation in self-judgement and empathy: implications for understanding the brain-gut-microbiome axis in depression,” *Psychopharmacology (Berl)*, vol. 236, no. 5, 2019, doi: 10.1007/s00213-019-05230-2.
[21] J. A. Bravo *et al.*, “Ingestion of Lactobacillus strain regulates emotional behavior and central GABA receptor expression in a mouse via the vagus nerve,” *Proc Natl Acad Sci U S A*, vol. 108, no. 38, 2011, doi: 10.1073/pnas.1102999108.
[22] P. Perez-Pardo *et al.*, “Role of TLR4 in the gut-brain axis in Parkinson’s disease: A translational study from men to mice,” *Gut*, vol. 68, no. 5, 2019, doi: 10.1136/gutjnl-2018-316844.
[23] E. A. Felton and M. C. Cervenka, “Dietary therapy is the best option for refractory nonsurgical epilepsy,” *Epilepsia*, vol. 56, no. 9, 2015, doi: 10.1111/epi.13075.
[24] P. Zheng *et al.*, “The gut microbiome from patients with schizophrenia modulates the glutamate-glutamine-GABA cycle and schizophrenia-relevant behaviors in mice,” *Sci Adv*, vol. 5, no. 2, 2019, doi: 10.1126/sciadv.aau8317.
[25] L. A. David *et al.*, “Diet rapidly and reproducibly alters the human gut microbiome,” *Nature*, vol. 505, no. 7484, 2014, doi: 10.1038/nature12820.
[26] W. H. W. Tang and S. L. Hazen, “The contributory role of gut microbiota in cardiovascular disease,” *Journal of Clinical Investigation*, vol. 124, no. 10. 2014. doi: 10.1172/JCI72331.
[27] F. E. Dewhirst *et al.*, “The human oral microbiome,” *J Bacteriol*, vol. 192, no. 19, 2010, doi: 10.1128/JB.00542-10.
[28] D. T. Pride *et al.*, “Evidence of a robust resident bacteriophage population revealed through analysis of the human salivary virome,” *ISME Journal*, vol. 6, no. 5, 2012, doi: 10.1038/ismej.2011.169.
[29] P. N. Deo and R. Deshmukh, “Oral microbiome: Unveiling the fundamentals,” *Journal of Oral and Maxillofacial Pathology*, vol. 23, no. 1. 2019. doi: 10.4103/jomfp.JOMFP_304_18.
[30] K. J. Joshipura, H. C. Hung, E. B. Rimm, W. C. Willett, and A. Ascherio, “Periodontal disease, tooth loss, and incidence of ischemic stroke,” *Stroke*, vol. 34, no. 1, 2003, doi: 10.1161/01.STR.0000052974.79428.0C.
[31] R. J. Genco, S. G. Grossi, A. Ho, F. Nishimura, and Y. Murayama, “A Proposed Model Linking Inflammation to Obesity, Diabetes, and Periodontal Infections,” *J Periodontol*, vol. 76, no. 11-s, 2005, doi: 10.1902/jop.2005.76.11-s.2075.
[32] S. Awano *et al.*, “Oral health and mortality risk from pneumonia in the elderly,” *J Dent Res*, vol. 87, no. 4, 2008, doi: 10.1177/154405910808700418.
[33] K. J. Joshipura, E. B. Rimm, C. W. Douglass, D. Trichopoulos, A. Ascherio, and W. C. Willett, “Poor oral health and coronary heart disease,” *J Dent Res*, vol. 75, no. 9, 1996, doi: 10.1177/00220345960750090301.

[34] J. D. Beck and S. Offenbacher, “Systemic Effects of Periodontitis: Epidemiology of Periodontal Disease and Cardiovascular Disease,” *J Periodontol*, vol. 76, no. 11-s, 2005, doi: 10.1902/jop.2005.76.11-s.2089.
[35] M. F. Zarco, T. J. Vess, and G. S. Ginsburg, “The oral microbiome in health and disease and the potential impact on personalized dental medicine,” *Oral Diseases*, vol. 18, no. 2. 2012. doi: 10.1111/j.1601-0825.2011.01851.x.
[36] *The new science of metagenomics: Revealing the secrets of our microbial planet*. 2007. doi: 10.17226/11902.
[37] L. P. Shaw, A. M. Smith, and A. P. Roberts, “The oral microbiome,” *Emerg Top Life Sci*, vol. 1, no. 4, pp. 287–296, Nov. 2017, doi: 10.1042/ETLS20170040.
[38] CDC, “Arthritis | CDC,” *Centers for Disease Control and Prevention*. 2020.
[39] J. Fukae *et al.*, “Convolutional neural network for classification of two-dimensional array images generated from clinical information may support diagnosis of rheumatoid arthritis,” *Sci Rep*, vol. 10, no. 1, 2020, doi: 10.1038/s41598-020-62634-3.
[40] J. H. KELLGREN, “Radiological signs of rheumatoid arthritis; a study of observer differences in the reading of hand films.,” *Ann Rheum Dis*, vol. 15, no. 1, 1956, doi: 10.1136/ard.15.1.55.
[41] J. T. Sharp, M. D. Lidsky, L. C. Collins, and J. Moreland, “Methods of scoring the progression of radiologic changes in rheumatoid arthritis. Correlation of radiologic, clinical and laboratory abnormalities,” *Arthritis Rheum*, vol. 14, no. 6, 1971, doi: 10.1002/art.1780140605.
[42] J. T. Sharp *et al.*, “Reproducibility of multiple-observer scoring of radiologic abnormalities in the hands and wrists of patients with rheumatoid arthritis,” *Arthritis Rheum*, vol. 28, no. 1, 1985, doi: 10.1002/art.1780280104.
[43] A. Larsen, K. Dale, and M. Eek, “Radiographic evaluation of rheumatoid arthritis and related conditions by standard reference films,” *Acta Radiologica - Series Diagnosis*, vol. 18, no. 4, 1977, doi: 10.1177/028418517701800415.
[44] H. K. Genant, “Methods of assessing radiographic change in rheumatoid arthritis,” *Am J Med*, vol. 75, no. 6 PART 1, 1983, doi: 10.1016/0002-9343(83)90473-4.
[45] G. R. Machado, E. Silva, and R. R. Goldschmidt, “Adversarial Machine Learning in Image Classification: A Survey Toward the Defender’s Perspective,” *ACM Comput Surv*, vol. 55, no. 1, 2023, doi: 10.1145/3485133.
[46] P. Wang, E. Fan, and P. Wang, “Comparative analysis of image classification algorithms based on traditional machine learning and deep learning,” *Pattern Recognit Lett*, vol. 141, 2021, doi: 10.1016/j.patrec.2020.07.042.
[47] X. Liu, Q. Hu, Y. Cai, and Z. Cai, “Extreme Learning Machine-Based Ensemble Transfer Learning for Hyperspectral Image Classification,” *IEEE J Sel Top Appl Earth Obs Remote Sens*, vol. 13, 2020, doi: 10.1109/JSTARS.2020.3006879.
[48] S. Loussaief and A. Abdelkrim, “Machine Learning framework for image classification,” *Advances in Science, Technology and Engineering Systems*, vol. 3, no. 1, 2018, doi: 10.25046/aj030101.
[49] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” Apr. 2020.
[50] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, vol. 2.

[51] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December. doi: 10.1109/CVPR.2016.90.
[52] Y. Wang, R. Huang, S. Song, Z. Huang, and G. Huang, “Not All Images are Worth 16x16 Words: Dynamic Vision Transformers with Adaptive Sequence Length,” in *NeurIPS*, 2021, no. NeurIPS.
[53] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9351. doi: 10.1007/978-3-319-24574-4_28.
[54] A. Kumar *et al.*, “Ask me anything: Dynamic memory networks for natural language processing,” in *33rd International Conference on Machine Learning, ICML 2016*, 2016, vol. 3.
[55] A. W. Yu *et al.*, “QaNet: Combining local convolution with global self-attention for reading comprehension,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.
[56] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013.
[57] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global vectors for word representation,” in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014. doi: 10.3115/v1/d14-1162.
[58] A. Vaswani *et al.*, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-December.
[59] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019, vol. 1.
[60] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, “Generalized end-to-end loss for speaker verification,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2018, vol. 2018-April. doi: 10.1109/ICASSP.2018.8462665.
[61] S. Kriman *et al.*, “Quartznet: Deep Automatic Speech Recognition with 1D Time-Channel Separable Convolutions,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2020, vol. 2020-May. doi: 10.1109/ICASSP40776.2020.9053889.
[62] J. Luo, J. Wang, N. Cheng, G. Jiang, and J. Xiao, “Multi-Quartznet: Multi-Resolution Convolution for Speech Recognition with Multi-Layer Feature Fusion,” in *2021 IEEE Spoken Language Technology Workshop, SLT 2021 - Proceedings*, 2021. doi: 10.1109/SLT48900.2021.9383532.
[63] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in *Advances in Neural Information Processing Systems*, 2020, vol. 2020-December.
[64] D. Amodei *et al.*, “Deep speech 2: End-to-end speech recognition in English and Mandarin,” in *33rd International Conference on Machine Learning, ICML 2016*, 2016, vol. 1.
[65] “Exploration by random network distillation,” in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[66] L. Espeholt *et al.*, “IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures,” in *35th International Conference on Machine Learning, ICML 2018*, 2018, vol. 4.
[67] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in *35th International Conference on Machine Learning, ICML 2018*, 2018, vol. 5.
[68] M. Hessel *et al.*, “Rainbow: Combining improvements in deep reinforcement learning,” in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, 2018. doi: 10.1609/aaai.v32i1.11796.
[69] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Frcitas, “Dueling Network Architectures for Deep Reinforcement Learning,” in *33rd International Conference on Machine Learning, ICML 2016*, 2016, vol. 4.
[70] Y. Huang, “Playing Atari with Deep Reinforcement Learning,” *Deep Reinforcement Learning: Fundamentals, Research and Applications*, 2020.
[71] M. Agnello *et al.*, “Microbiome Associated with Severe Caries in Canadian First Nations Children,” *J Dent Res*, vol. 96, no. 12, 2017, doi: 10.1177/0022034517718819.
[72] A. Gomez *et al.*, “Host Genetic Control of the Oral Microbiome in Health and Disease,” *Cell Host Microbe*, vol. 22, no. 3, 2017, doi: 10.1016/j.chom.2017.08.013.
[73] F. Teng *et al.*, “Prediction of Early Childhood Caries via Spatial-Temporal Variations of Oral Microbiota,” *Cell Host Microbe*, vol. 18, no. 3, 2015, doi: 10.1016/j.chom.2015.08.005.
[74] V. C. de Jesus *et al.*, “Sex-Based Diverse Plaque Microbiota in Children with Severe Caries,” *J Dent Res*, vol. 99, no. 6, 2020, doi: 10.1177/0022034520908595.
[75] B. Kalpana *et al.*, “Bacterial diversity and functional analysis of severe early childhood caries and recurrence in India,” *Sci Rep*, vol. 10, no. 1, 2020, doi: 10.1038/s41598-020-78057-z.
[76] J. Handelsman, “Metagenomics: Application of Genomics to Uncultured Microorganisms,” *Microbiology and Molecular Biology Reviews*, vol. 68, no. 4, Dec. 2004, doi: 10.1128/MMBR.68.4.669-685.2004.
[77] J. A. Gilbert, M. J. Blaser, J. G. Caporaso, J. K. Jansson, S. v. Lynch, and R. Knight, “Current understanding of the human microbiome,” *Nat Med*, vol. 24, no. 4, 2018, doi: 10.1038/nm.4517.
[78] N. LaPierre, C. J. T. Ju, G. Zhou, and W. Wang, “MetaPheno: A critical evaluation of deep learning and machine learning in metagenome-based disease prediction,” *Methods*, vol. 166, 2019, doi: 10.1016/j.ymeth.2019.03.003.
[79] C. Cortes and V. Vapnik, “Support-Vector Networks,” *Mach Learn*, vol. 20, no. 3, 1995, doi: 10.1023/A:1022627411411.
[80] L. Breiman, “Random forests,” *Mach Learn*, vol. 45, no. 1, 2001, doi: 10.1023/A:1010933404324.
[81] M. A. Rahman and H. Rangwala, “RegMIL: Phenotype Classification from Metagenomic Data,” in *ACM-BCB 2018 - Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, 2018. doi: 10.1145/3233547.3233585.
[82] J. Qin *et al.*, “A human gut microbial gene catalogue established by metagenomic sequencing,” *Nature*, vol. 464, no. 7285, 2010, doi: 10.1038/nature08821.
[83] N. Qin *et al.*, “Alterations of the human gut microbiome in liver cirrhosis.,” *Nature*, vol. 513, no. 7516, 2014, doi: 10.1038/nature13568.

[84] S. Andrews, I. Tsochantaridis, and T. Hofmann, “Support vector machines for multiple-instance learning,” in *Advances in Neural Information Processing Systems*, 2003.
[85] R. C. Bunescu and R. J. Mooney, “Multiple instance learning for sparse positive bags,” in *ACM International Conference Proceeding Series*, 2007, vol. 227. doi: 10.1145/1273496.1273510.
[86] D. Kotzias, M. Denil, N. de Freitas, and P. Smyth, “From group to individual labels using deep features,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, vol. 2015-August. doi: 10.1145/2783258.2783380.
[87] E. Pasolli, D. T. Truong, F. Malik, L. Waldron, and N. Segata, “Machine Learning Meta-analysis of Large Metagenomic Datasets: Tools and Biological Insights,” *PLoS Comput Biol*, vol. 12, no. 7, 2016, doi: 10.1371/journal.pcbi.1004977.
[88] D. Sharma and W. Xu, “phyLoSTM: a novel deep learning model on disease prediction from longitudinal microbiome data,” *Bioinformatics*, 2021, doi: 10.1093/bioinformatics/btab482.
[89] T. Vatanen *et al.*, “Variation in Microbiome LPS Immunogenicity Contributes to Autoimmunity in Humans,” *Cell*, vol. 165, no. 4, 2016, doi: 10.1016/j.cell.2016.04.007.
[90] D. B. DiGiulio *et al.*, “Temporal and spatial variation of the human microbiota during pregnancy,” *Proc Natl Acad Sci U S A*, vol. 112, no. 35, 2015, doi: 10.1073/pnas.1502875112.
[91] X. Chen, L. Liu, W. Zhang, J. Yang, and K.-C. Wong, “Human host status inference from temporal microbiome changes via recurrent neural networks,” *Brief Bioinform*, 2021, doi: 10.1093/bib/bbab223.
[92] K. Cho *et al.*, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014. doi: 10.3115/v1/d14-1179.
[93] E. Pasolli *et al.*, “Accessible, curated metagenomic data through ExperimentHub,” *Nature Methods*, vol. 14, no. 11. 2017. doi: 10.1038/nmeth.4468.
[94] B. Brooks *et al.*, “Strain-resolved analysis of hospital rooms and infants reveals overlap between the human and room microbiome,” *Nat Commun*, vol. 8, no. 1, 2017, doi: 10.1038/s41467-017-02018-w.
[95] A. B. Hall *et al.*, “A novel Ruminococcus gnavus clade enriched in inflammatory bowel disease patients,” *Genome Med*, vol. 9, no. 1, 2017, doi: 10.1186/s13073-017-0490-5.
[96] A. Heintz-Buschart *et al.*, “Integrated multi-omics of the human gut microbiome in a case study of familial type 1 diabetes,” *Nat Microbiol*, vol. 2, no. 1, 2016, doi: 10.1038/nmicrobiol.2016.180.
[97] F. Raymond *et al.*, “The initial state of the human gut microbiome determines its reshaping by antibiotics,” *ISME Journal*, vol. 10, no. 3, 2016, doi: 10.1038/ismej.2015.148.
[98] C. Vincent, M. A. Miller, T. J. Edens, S. Mehrotra, K. Dewar, and A. R. Manges, “Bloom and bust: Intestinal microbiota dynamics in response to hospital exposures and Clostridium difficile colonization or infection,” *Microbiome*, vol. 4, 2016, doi: 10.1186/s40168-016-0156-3.
[99] Y. Shao *et al.*, “Stunted microbiota and opportunistic pathogen colonization in caesarean-section birth,” *Nature*, vol. 574, no. 7776, 2019, doi: 10.1038/s41586-019-1560-1.

[100] N. A. Bokulich *et al.*, “Antibiotics, birth mode, and diet shape microbiome maturation during early life,” *Sci Transl Med*, vol. 8, no. 343, 2016, doi: 10.1126/scitranslmed.aad7121.
[101] E. Bogart, R. Creswell, and G. K. Gerber, “MITRE: Inferring features from microbiota time-series data linked to host status,” *Genome Biol*, vol. 20, no. 1, 2019, doi: 10.1186/s13059-019-1788-y.
[102] A. A. Metwally, P. S. Yu, D. Reiman, Y. Dai, P. W. Finn, and D. L. Perkins, “Utilizing longitudinal microbiome taxonomic profiles to predict food allergy via long short-term memory networks,” *PLoS Comput Biol*, vol. 15, no. 2, 2019, doi: 10.1371/journal.pcbi.1006693.
[103] L. R. Rabiner and B. H. Juang, “An Introduction to Hidden Markov Models,” *IEEE ASSP Magazine*, vol. 3, no. 1, 1986, doi: 10.1109/MASSP.1986.1165342.
[104] R. Tadeusiewicz, “Neural networks: A comprehensive foundation,” *Control Eng Pract*, vol. 3, no. 5, 1995, doi: 10.1016/0967-0661(95)90080-2.
[105] R. Tibshirani, “Regression Shrinkage and Selection Via the Lasso,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, 1996, doi: 10.1111/j.2517-6161.1996.tb02080.x.
[106] T. K. Ho, “Random decision forests,” in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 1995, vol. 1. doi: 10.1109/ICDAR.1995.598994.
[107] A. Ng, “Sparse autoencoder,” *CS294A Lecture notes*, vol. 72, 2011.
[108] B. García-Jiménez, J. Muñoz, S. Cabello, J. Medina, and M. D. Wilkinson, “Predicting microbiomes through a deep latent space,” *Bioinformatics*, vol. 37, no. 10, 2021, doi: 10.1093/bioinformatics/btaa971.
[109] S. A. Maarastawi, K. Frindte, M. Linnartz, and C. Knief, “Crop rotation and straw application impact microbial communities in Italian and Philippine Soils and the rhizosphere of Zea mays,” *Front Microbiol*, vol. 9, no. JUN, 2018, doi: 10.3389/fmicb.2018.01295.
[110] W. A. Walters *et al.*, “Large-scale replicated field study of maize rhizosphere identifies heritable microbes,” *Proc Natl Acad Sci U S A*, vol. 115, no. 28, 2018, doi: 10.1073/pnas.1800918115.
[111] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, “Be your own teacher: Improve the performance of convolutional neural networks via self distillation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, vol. 2019-October. doi: 10.1109/ICCV.2019.00381.
[112] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January. doi: 10.1109/CVPR.2017.634.
[113] Simonyan K and Zisserman A, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *CoRR*, vol. abs/1409.1556, 2015.
[114] K. Kim, B. M. Ji, D. Yoon, and S. Hwang, “Self-Knowledge Distillation: A Simple Way for Better Generalization,” *ArXiv*, 2020.
[115] J. S. Hyams *et al.*, “Factors associated with early outcomes following standardised therapy in children with ulcerative colitis (PROTECT): a multicentre inception cohort study,” *Lancet Gastroenterol Hepatol*, vol. 2, no. 12, 2017, doi: 10.1016/S2468-1253(17)30252-2.
[116] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.

[117] K. Pearson, " LIII. On lines and planes of closest fit to systems of points in space ," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, 1901, doi: 10.1080/14786440109462720.
[118] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," in *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, 2013. doi: 10.1016/B978-1-4832-1446-7.50035-2.
[119] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, 1951, doi: 10.1214/aoms/1177729694.
[120] J. M. W. Slack, "Molecular biology of the cell," in *Principles of Tissue Engineering*, 2020. doi: 10.1016/B978-0-12-818422-6.00005-8.
[121] E. Bianconi *et al.*, "An estimation of the number of cells in the human body," *Ann Hum Biol*, vol. 40, no. 6, 2013, doi: 10.3109/03014460.2013.807878.
[122] B. W. Yoon, S. H. Lim, J. H. Shin, J. W. Lee, Y. Lee, and J. H. Seo, "Analysis of oral microbiome in glaucoma patients using machine learning prediction models," *J Oral Microbiol*, vol. 13, no. 1, 2021, doi: 10.1080/20002297.2021.1962125.
[123] J. Sun, T. Nishiyama, K. Shimizu, and K. Kadota, "TCC: An R package for comparing tag count data with robust normalization strategies," *BMC Bioinformatics*, vol. 14, no. 1, 2013, doi: 10.1186/1471-2105-14-219.
[124] X. Yin and J. Han, "CPAR: Classification based on Predictive Association Rules," 2003. doi: 10.1137/1.9781611972733.40.
[125] S. Rampelli, M. Fabbrini, M. Candela, E. Biagi, P. Brigidi, and S. Turroni, "G2S: A New Deep Learning Tool for Predicting Stool Microbiome Structure From Oral Microbiome Data," *Front Genet*, vol. 12, 2021, doi: 10.3389/fgene.2021.644516.
[126] W. E. Johnson, C. Li, and A. Rabinovic, "Adjusting batch effects in microarray expression data using empirical Bayes methods," *Biostatistics*, vol. 8, no. 1, 2007, doi: 10.1093/biostatistics/kxj037.
[127] G. K. Smyth and T. Speed, "Normalization of cDNA microarray data," *Methods*, vol. 31, no. 4, 2003, doi: 10.1016/S1046-2023(03)00155-5.
[128] L. Haghverdi, A. T. L. Lun, M. D. Morgan, and J. C. Marioni, "Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors," *Nat Biotechnol*, vol. 36, no. 5, 2018, doi: 10.1038/nbt.4091.
[129] K. Polański, M. D. Young, Z. Miao, K. B. Meyer, S. A. Teichmann, and J. E. Park, "BBKNN: Fast batch alignment of single cell transcriptomes," *Bioinformatics*, vol. 36, no. 3, 2020, doi: 10.1093/bioinformatics/btz625.
[130] I. Lemhadri, F. Ruan, L. Abraham, and R. Tibshirani, "Lassonet: A neural network with feature sparsity," *Journal of Machine Learning Research*, vol. 22, 2021.
[131] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.
[132] A. Brock, S. De, S. L. Smith, and K. Simonyan, "High-Performance Large-Scale Image Recognition Without Normalization," Feb. 2021.
[133] H. Yong, J. Huang, X. Hua, and L. Zhang, "Gradient Centralization: A New Optimization Technique for Deep Neural Networks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12346 LNCS. doi: 10.1007/978-3-030-58452-8_37.
[134] Zeke Xie, Li Yuan, Zhanxing Zhu, and Masashi Sugiyama, "Positive-negative momentum: Manipulating stochastic gradient noise to improve generalization.," *ArXiv*, 2021.

[135] T. Georgiou, S. Schmitt, T. Bäck, W. Chen, and M. Lew, “Norm loss: An efficient yet effective regularization method for deep neural networks,” in *Proceedings - International Conference on Pattern Recognition*, 2020. doi: 10.1109/ICPR48806.2021.9412171.
[136] Z. Xie, I. Sato, and M. Sugiyama, “Stable Weight Decay Regularization,” *ArXiv*, no. 2018, 2020.
[137] Jerry Ma and Denis Yarats, “ On the adequacy of untuned warmup for adaptive optimization.,” *ArXiv*, 2019.
[138] Nikhil Iyer, v Thejas, Nipun Kwatra, Ramachandran Ramjee, and Muthian Sivathanu, “Wide-minima density hypothesis and the explore-exploit learning rate schedule.,” *ArXiv*, 2020.
[139] M. R. Zhang, J. Lucas, G. Hinton, and J. Ba, “Lookahead optimizer: k Steps forward, 1 step back,” in *Advances in Neural Information Processing Systems*, 2019, vol. 32.
[140] Q. Wang, J. Niu, W. Xu, D. Wei, and K. Qian, “Deep learning based multi-batch calibration for classification in various omics,” *Res Sq*, 2021.
[141] J. Gardner, I. Simon, E. Manilow, C. Hawthorne, and J. Engel, “MT3: Multi-Task Multitrack Music Transcription,” Nov. 2021.
[142] A. Ashraf, S. Khan, N. Bhagwat, M. Chakravarty, and B. Taati, “Learning to Unlearn: Building Immunity to Dataset Bias in Medical Imaging Studies,” Dec. 2018.
[143] Y. Gao and F. Sun, “Batch Normalization Followed by Merging Is Powerful for Phenotype Prediction Integrating Multiple Heterogeneous Studies,” *BioRXiv*, 2022.
[144] F. K. Muriithi, “Centered log-ratio (clr) transformation and robust principal component analysis of long-term NDVI data reveal vegetation activity linked to climate processes,” *Climate*, vol. 3, no. 1, 2015, doi: 10.3390/cli3010135.
[145] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, May 2015, doi: 10.1038/nature14539.
[146] A. Tilve, S. Nayak, S. Vernekar, D. Turi, P. R. Shetgaonkar, and S. Aswale, “Pneumonia Detection Using Deep Learning Approaches,” in *International Conference on Emerging Trends in Information Technology and Engineering, ic-ETITE 2020*, 2020. doi: 10.1109/ic-ETITE47903.2020.152.
[147] C. Qin, D. Yao, Y. Shi, and Z. Song, “Computer-aided detection in chest radiography based on artificial intelligence: A survey,” *BioMedical Engineering Online*, vol. 17, no. 1. 2018. doi: 10.1186/s12938-018-0544-y.
[148] M. Oh and L. Zhang, “DeepMicro: deep representation learning for disease prediction based on microbiome data,” *Sci Rep*, vol. 10, no. 1, 2020, doi: 10.1038/s41598-020-63159-5.
[149] K. A. Tran, O. Kondrashova, A. Bradley, E. D. Williams, J. v. Pearson, and N. Waddell, “Deep learning in cancer diagnosis, prognosis and treatment selection,” *Genome Medicine*, vol. 13, no. 1. 2021. doi: 10.1186/s13073-021-00968-x.
[150] K. Üreten, H. Erbay, and H. H. Maraş, “Detection of rheumatoid arthritis from hand radiographs using a convolutional neural network,” *Clin Rheumatol*, vol. 39, no. 4, 2020, doi: 10.1007/s10067-019-04487-4.
[151] R. Caruana, “Multitask Learning,” *Mach Learn*, vol. 28, no. 1, 1997, doi: 10.1023/A:1007379606734.
[152] D. Sun *et al.*, “A Crowdsourcing Approach to Develop Machine Learning Models to Quantify Radiographic Joint Damage in Rheumatoid Arthritis,” *SSRN Electronic Journal*, 2022, doi: 10.2139/ssrn.4051459.

[153] T. Hirano *et al.*, “Development and validation of a deep-learning model for scoring of radiographic finger joint destruction in rheumatoid arthritis,” *Rheumatol Adv Pract*, vol. 3, no. 2, 2019, doi: 10.1093/rap/rkz047.
[154] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001, vol. 1. doi: 10.1109/cvpr.2001.990517.
[155] D. Aletaha *et al.*, “2010 Rheumatoid arthritis classification criteria: An American College of Rheumatology/European League Against Rheumatism collaborative initiative,” *Arthritis and Rheumatism*, vol. 62, no. 9. 2010. doi: 10.1002/art.27584.
[156] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December. doi: 10.1109/CVPR.2016.91.
[157] S. S. Halabi *et al.*, “The rSNA pediatric bone age machine learning challenge,” *Radiology*, vol. 290, no. 3, 2019, doi: 10.1148/radiol.2018180736.
[158] C. Y. Wang, H. Y. Mark Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh, and I. H. Yeh, “CSPNet: A new backbone that can enhance learning capability of CNN,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2020, vol. 2020-June. doi: 10.1109/CVPRW50498.2020.00203.
[159] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January. doi: 10.1109/CVPR.2017.106.
[160] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, “Path Aggregation Network for Instance Segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018. doi: 10.1109/CVPR.2018.00913.
[161] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8689 LNCS, no. PART 1. doi: 10.1007/978-3-319-10590-1_53.
[162] “Overview of Model Structure about YOLOv5.” https://github.com/ultralytics/yolov5/issues/280 (accessed Sep. 12, 2022).
[163] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “MixUp: Beyond empirical risk minimization,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.