UNIVERSITY OF CALIFORNIA,
IRVINE

Applications of deep learning and cloud computing for the analysis of X-ray data to predict COVID-19 outcome, and for cardiac MRI segmentation

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Biomedical Engineering

by

Manoel Tamraz

Thesis Committee:
Professor Arash Kheradvar, Chair
Professor Hamid Jafarkhani
Associate Professor Liangzhong Xiang

2021

DEDICATION

To

My Family and Friends

in recognition of their love, loyalty, and support

a beginning

Never look down to test the ground before taking
your next step; only he who keeps his eye fixed
on the far horizon will find the right road

Dag Hammarskjold

and a testament

Greater is He that is in you, than he that is in the world

1 John 4:4

TABLE OF CONTENTS

LIST OF FIGURES v

LIST OF TABLES vi

ACKNOWLEDGEMENTS vii

ABSTRACT OF THE THESIS viii

CHAPTER 1: Introduction 1

1.1 Application of deep learning in Medical Imaging 1

1.2 Current deep learning applications for COVID-19 2

1.3 Outline of the thesis 3

CHAPTER 2: Cloud- based approaches for analysis of medical imaging 6

2.1 Background 6

2.2 AI Platform 6

2.3 Methods and processes for segmentation of pediatric heart MRI 7

2.4 Results 10

2.5 Limitations and future work 13

CHAPTER 3: AI-based analysis of medical images for COVID-19 patients 14

3.1 The global impact of COVID-19 on hospitals 14

3.2 Applications of AI and Deep learning in the global COVID-19 effort 15

CHAPTER 4: Data acquisition and analysis to predict COVID-19 outcome 22

4.1 Background 22

4.2 Data sources 22

4.3 Splitting of available data for training purposes 23

CHAPTER 5: A VGG16 model for the prediction of COVID-19 outcome from images and metadata 24

5.1 Background 24

5.2 Computational Platform 28

5.3 Model Structure 28

5.4 Support Vector Machine Utilization 31

5.5 Results 31

CHAPTER 6: Implementation of the CheXNet learning model for the prediction of COVID-19 outcome from X-Ray images 33

6.1 Background 33

6.2 Results 34

CHAPTER 7: Comparison between the VGG16 models and CheXNet 35

7.1 Structural comparison between the model types 35

7.2 Comparison between the results 36

CHAPTER 8: Conclusions 37

8.1 Results 37

8.2 Study limitations and future work 38

References 40

LIST OF FIGURES

		Page
Figure 1	Process flow diagram for a user upload	9
Figure 2	Cloud-based platform “latest upload” page	12
Figure 3	Number of hospital beds available per 100,000 population	14
Figure 4	Number of ICU beds available per 100,000 population	15
Figure 5	Example of a fully connected deep neural network	25
Figure 6	Convolutional layer showing progression of a kernel across a pixel matrix	26
Figure 7	CNN max pooling layer showing how the kernel moves across the convolved matrix	26
Figure 8	Combined CNN structure including convolution, pooling, and a deep neural network	27
Figure 9	VGG model structure visualization	27
Figure 10	Visualization of support vector machine classification task for 2D input data	28
Figure 11	Two VGG16 models for processing only COVID-19 chest x-ray images and for processing both images and patient metadata	30
Figure 12	Visual comparison of VGG16 and CheXNet model structures	35

LIST OF TABLES

		Page
Table 1	An overview of the components in the Django web framework for the website	7
Table 2	Comparison of pre-processing methods between the website and that used for training the segmentation model	11
Table 3	Number of image data from each batch and for each outcome label	23
Table 4	Data split for training and validation	23
Table 5	Data split for testing	23

ACKNOWLEDGEMENTS

For providing me with the tools, resources, finances, knowledge, and, most importantly, support needed to conduct this research, I would like to thank Dr. Arash Kheradvar. Through his patience and experience, I had the time to grow and learn the necessary skills. I would also like to thank Dr. Hamid Jafarkhani and Dr. Liangzhong Xiang for their time and for agreeing to be a part of my thesis committee.

I cannot overstate the importance and value of the help I received from both KLAB members and from the AHA support team. In particular, I would like to thank Saeed Karimi-Bidhendi from the KLAB team for the hours of time and the vast knowledge of python as well as machine learning techniques that shaped my understanding of the principles and methods leveraged in this thesis. From the AHA support team, I would like to thank Laura Stevens for helping me with the implementation of Django on the AHA platform above and beyond any expectation. I am also grateful for Dr. Andrew Cheng's insightful feedback and eagerness to be an early adopter of our MRI segmentation website.

I would like to thank the Department of Biomedical Engineering and the Edwards Lifesciences Center for Advanced Cardiovascular Technology for their support through their state-of-the-art facilities and the abundant resources to build a solid foundation for my research. Thank you, as well, to Dr. Kheradvar and all KLAB members that help secure the funding needed to advance my research and all other research conducted by KLAB.

Finally, I would like to express my gratitude to my family and friends. Through your love, support, and sacrifices, I am able to make achievements that I never imagined I would. I look forward to the doors in front of me that I have yet to open and appreciate the lessons of those that I close behind me.

ABSTRACT OF THE THESIS

Applications of deep learning and cloud computing for the analysis of X-ray data to predict COVID-19 outcome, and for cardiac MRI segmentation

by

Manoel Tamraz

Master of Science in Biomedical Engineering

University of California, Irvine, 2021

Professor Arash Kheradvar, Chair

Current advances in both deep learning techniques and in cloud computing allow the advancement of innovations that work to the benefit of physicians and patients. This dissertation explores leveraging of these advancements to create a cloud-based analysis platforms for physicians to analyze cardiac MRI as well as a four-tier outcome prediction machine learning model for COVID-19 patients based on their chest X-rays and metadata. The MRI analysis website is hosted on the American Heart Association's (AHA) Precision Medicine Platform (PMP) and integrates the cardiac MRI segmentation model by Karimi-Bidhendi, et al.[2] The back-end web framework was created using Python and Django, with MySQL as the database manager. This allowed a flexible and reliable base to build the website on as well as strong support from the AHA. The website includes an automatic end-systolic (ES) and end-diastolic (ED) detection system for each ventricle, which allows physicians to upload patients' MRI DICOMs without the need to manually select files

relating to each cardiac phase for each ventricle. Hundreds of files are processed in seconds and a report of all segmented images relating to the ED and ES phases for each ventricle as well as the associated ventricle volumes would be immediately presented after file processing.

With regards to the COVID-19 outcome prediction model, 6,259 chest X-ray images from 1,771 patients seen at UCI and UCLA Medical Centers were used to train two VGG16 models and a CheXNet model. The first VGG16 model is a convolutional neural network (CNN) that processed only the chest X-ray images and the second is a CNN for the images as well as a separate deep neural network (DNN) for patient metadata including age and BMI and another DNN that processes the combined output of the CNN and the metadata DNN. This combination allows both images and metadata to be factored in when training the model. The CheXNet model is tailored specifically for chest X-ray images and was used to assess the performance of the VGG-16 models. The accuracy of the image-only VGG16 model was 56% on the four-class prediction, compared to 59% for the image and metadata VGG16 model. The CheXNet model resulted in 60% accuracy. This suggests that the metadata did not significantly improve the performance of the model and that the image data was not informative enough beyond 60% accuracy for four-tier predicting of COVID-19 patients' outcomes.

CHAPTER 1: Introduction

1.1 Application of deep learning in Medical Imaging

Medical imaging provides a large wealth of information for physicians' use for patient diagnosis. As more imaging modalities emerged and became increasingly accessible to physicians, the need to process this data in an efficient manner also arose. Deep learning has been increasingly researched as a means to deliver this efficiency to physicians in order to save vital time in the patient diagnosis process.[1] Application of machine learning in medical image diagnostics can be seen in a wide range of settings from feature segmentation[2, 3, 4] to classification for diagnosis[5, 6]. The goal of current work towards deep learning for medical imaging is to achieve an operational accuracy on par with the physicians who will use it to assist in diagnosis and to be able to predict patients' prognosis based on their imaging and other clinical data.

Some current cloud-based tools that are used to analyze medical images include Adioc and Arterys.[8, 9] Adioc is an advanced cloud-based analysis tool for CT scans that can detect and diagnose various health issues such as brain hemorrhaging, spine and rib fractures, and large vessel occlusions using AI and machine learning. The Adioc team claims that their tool has saved over 45 million minutes of turnaround time.[8] Arterys is another cloud-based platform that has a suite of tools for different applications within healthcare, including cardiovascular, lung, chest, and neurological diagnosis. One of which, called CardioAI, can perform advanced volumetric and flow calculations for 2D and 3D cine heart MRI scans. The Arterys team claims that CardioAI saves 25 minutes per patient study and is as accurate as 7 experts.[9]

1.2 Current deep learning applications for COVID-19

Since the beginning of the pandemic, the need to further understand COVID-19 has prompted a large effort to make new discoveries that can help tackle the virus and save lives. Deep learning research is no exception to this. As testing ramped up, AlJame, et al.[10] proposed a Deep Forest model to detect COVID-19 in routine blood tests. When concerns regarding the long-term effects of COVID-19 infection rose, Chen, et al.[11] demonstrated that deep learning can be used to predict whether COVID-19 patients would develop Post-Traumatic Stress Disorder (PTSD) symptoms. Prior to the widespread availability of vaccines, there was also the need to find drugs that would help fight the symptoms. Jha, et al.[12] applied deep learning to discover such drugs. Other studies tested the ability of a learning algorithm to diagnose COVID-19 infection as positive or negative from patient chest X-rays (CXRs) or CT scans.[13, 14, 15, 16] This was further developed by Liu, et al.[17], where CT images of COVID-19 lung infection were segmented by a learning model. Prakash, et al.[18] expanded on this by leveraging CXR segmentation of COVID-19-infected areas to perform a binary classification of COVID-19 viral infection. Given that a sufficient amount of annotated CT image data was not available for training, Zhang, et al.[19] used encoders and decoders to both identify COVID-19 infection features in lung CT scans and segment them. Although these efforts are pivotal to improving the speed, efficiency, and accuracy of COVID-19 patient diagnosis in the long term, Pennisi, et al.[20] sought to deliver a tool for immediate hospital use by combining a deep learning model that categorizes lesions shown in CT scans, a user-friendly interface, and an explainable model. Explainability refers to the ability of a model to describe its reasoning for a decision that is understandable by the user. To expand on the hospital application of deep learning for COVID-19, Perumal, et al.[21] used

a hybrid regression model to combine patient metadata, laboratory data, and CT image data for prediction of COVID-19 criticality score, which categorizes patients as having either early stages of symptoms or having life-threatening symptoms, to make it easier for physicians to prioritize care for at-risk patients.[21] CXRs and CT scans were not the only studied data acquisition methods. La Salvia, et al.[22] and Diaz-Escobar, et al.[23] demonstrated that lung ultrasound data is also an effective input for deep learning models that can detect COVID-19. As treatment of COVID-19 evolved and new methods were introduced, Wang, et al.[24] focused on applying deep learning to predict the recovery time of hospitalized COVID-19 patients. This large surge of proposed deep learning models to address COVID-19 rapidly grew in a short span of time. Sadre, et al.[25] sought to create a method for validating the quality of the emerging deep learning models. On another front, Iloanusi and Ross[26] utilized deep learning along with global weather pattern data to predict the spread of COVID-19 infection as the seasons change. However, like other COVID-19 forecasting attempts, the introduction of new variants complicates those efforts. Rashed and Akimasa[27] studied the effects of variant introduction on deep learning forecasting models to study how drastic they are and how long it takes for the models to readjust.

1.3 Outline of the thesis

A cloud-based platform for segmentation of pediatric cardiac MRI and a learning model for the prediction of COVID-19 outcome to help critical patient care are the two primary goals of this dissertation. In pursuit of the first goal, a Django-based web framework was created on the American Heart Association's Precision Medicine Platform to leverage the model created by Karimi-Bidhendi, et al.[2]. This VGG-16-based model utilizes a fully connected convolutional network with 33 layers to segment the left or right ventricle in a

heart MRI DICOM image. It was integrated into the web framework and an interface was created to upload DICOM files for which the cloud would output the results of segmentation and ventricle volume calculation. In discussion with Dr. Andrew Cheng at The Keck School of Medicine, University of Southern California and Children's Hospital Los Angeles, Los Angeles, it was made clear that an auto-detection feature for the end-diastolic (ED) and end-systolic (ES) phases for each ventricle would save physicians time trying to manually separate the corresponding images. An auto-detection capability was subsequently added to the platform. Users are now able to simply upload all DICOM files for patients and the volumes for the left and right ventricles at ED and ES phases are reported along with the ejection fraction.

In pursuit of the second goal of creating a learning model for the prediction of COVID-19 outcome, a VGG-16 model was selected to perform a classification task. The data was composed of CXR DICOMs along with metadata for each patient. Each patient's outcome was labeled on a four-tier scale, with 0 meaning death, 1 meaning ICU or intensive care, 2 meaning inpatient non-intensive hospital care, and 3 meaning outpatient status. The model consisted of a Convolutional Neural Network (CNN) as well as a Deep Neural Network (DNN) for prediction. We trained the model on the images from four datasets. Two sets of x-rays and metadata were obtained from the UCI medical center and two other sets of x-rays and metadata were obtained from the UCLA medical center. This training led to an accuracy of 56% for classifying the 4 outcomes for the patients, after optimizing the model parameters. Subsequently, we expanded the model to include patient metadata via two methods. The first method was to directly concatenate the metadata into the model right before the DNN, merging it with the CNN output (Figure 11). This method did not improve

the accuracy of the model. The second method was to create a second DNN to train off from the metadata and then concatenate the output into the original model similar to the first method. This approach slightly improved the accuracy to 57%. Then, the application of Support Vector Machines (SVMs) was explored by extracting the output of the CNN and the metadata DNN and making classification predictions using an SVM. After optimizing the SVM, the accuracy slightly improved (58%), but not significantly. Further optimization of the ratio between training and test data brought the VGG16 model with one CNN and two DNNs to a peak accuracy of 59%.

Finally, a new approach was taken to validate our previous approach by exploring the application of the CheXNet model.[5] This model has been applied to the diagnosis of pneumonia by analysis of CXRs, which we found is a good candidate to compare to our VGG16 model. After slight modifications to the CheXNet code to work with the COVID-19 dataset and the prediction labels, as well as optimization of parameters, a peak accuracy of 59% was reached. Thus, the second approach validated the results of our previous VGG16 model.

CHAPTER 2: Cloud- based approaches for analysis of medical imaging

2.1 Background

With the ongoing modernization of medicine, many different cloud-based platforms have been introduced for use by physicians across varying specialties. Some examples include Adioc and Arterys, as mentioned above, as well as Apteryx.[8, 9] Apteryx is a cloud-based software for dental office use. It helps dentists process 2D and 3D X-ray imaging for quick and reliable results during a patient visit.[28] Although software such as Arterys may be helpful for adult cardiovascular patients, there are no current solutions for pediatric cardiovascular patients. The reason for this is due to the differences between pediatric and adult heart anatomy. A child heart is not a scaled-down version of an adult heart, which means that segmentation of pediatric cardiac MRIs requires a new model to be trained specifically for such group of patients. The pediatric ventricle segmentation model created by Karimi-Bidhendi, et al.[2] allows this to be realized. Manual segmentation of cardiovascular MRI images is very labor intensive and is largely dependent on the skills of the operator. Due to this challenge, there are variations in the results even among expert physicians and among institutions. The goal of this online cloud-based platform is to accelerate the process of MRI image analysis by physicians for their pediatric patients.

2.2 AI Platform

The American Heart Association's Precision Medicine Platform (PMP) was leveraged to host a pediatric cardiovascular MRI segmentation and analysis online platform. The PMP is enabled with an Amazon Linux 2 operating system and includes a 16GB NVIDIA Tesla V100 GPU. The reliability and high-quality support behind the PMP made it an ideal choice for this application.

2.3 Methods and processes for segmentation of pediatric heart MRI

Python, Django, and MySQL were used to create a web-based framework that would run on the PMP due to ease of implementation, reliability, robustness of features, and large support base. Below is a table describing the various components of the Django web framework for this website (Table 1).

Component	Function

Settings	• High-level web framework details
	• Auto-generated by Django upon project creation
	• Settings modified for this project:
	  ○ MySQL chosen as database management tool
	  ○ File storage location modified
	  ○ Administrative account established

URLs	• Contains a list of all the pages of the website as well as their linked backend files
Apps	• Multiple applications (Apps) can share the backend framework • This project has one App to handle current functionality • Allows expansion of website capabilities by adding new applications without affecting existing applications • Allows segmentation of website backend to protect different applications during maintenance. • Each App contains its own set of URLs, front-end HTML templates, backend file management and processing of any input data.
Models	• Objects saved within MySQL-managed database • Can include any user-inputted information • Do not directly store files, but links files and information from an upload into a group • This web framework incorporates one model for upload information including upload name and contour type selection and another for the uploaded DICOMS.
Forms	• Translate the structure of the models for front-end user interaction • Links elements of the models to input fields placed on front end • Validates user inputs after form submission • Enables processing of uploaded data in the “views” component • This framework includes one form for each model ○ Form for DICOM model includes a link to the upload information model for grouping and processing

Templates	• HTML templates create the front-end of the website
	• One template for each page
	• CSS used for styling
	• JavaScript used for dynamic content such as populating tables
	• This website uses three templates
	  ○ Upload page template includes the upload form as well as background images and website information
	  ○ Latest upload page template includes upload results of the last user upload after processing. Results include segmented DICOM images and volumes of the left and right ventricles for the ED and ES phases.
	  ○ Upload history page template includes all previous uploads and corresponding results

Views	• Handles all backend processing within an application • All front-end user interaction passes through this file, where input is processed and the user is redirected if necessary

Table 1: An overview of the components in the Django web framework for the website

With the general framework of the website in place, the fully connected network (FCN) machine learning model from Karimi-Bidhendi, Saeed, et al.[2] was implemented into the *views.py* code. Subsequently, the front end of the website was created using HTML templates, CSS styling, and JavaScript.

The backend process of the web framework is as follows. When the user accesses the URL of the upload page, it is passed through the URLs file and into the Views file, where the correct HTML template is selected and shown to the user on the front-end page. Once the user fills out the upload form and selects the DICOM files, they are uploaded and sent to the Views file, which calls on the correct Django form and associates the uploaded information to the correct models. When the forms are validated, the DICOM images begin processing. Since the segmentation model was trained on 128x128 images, they are first downsampled to 128x128. All black margins are also removed to prevent interference with segmentation.

Then, the integrated FCN machine learning model creates a mask for the selected ventricle for each image. The areas of each mask are used in conjunction with the slice thickness, pixel spacing, and relative image plane, all of which are part of the DICOM metadata, to calculate the volume of the selected ventricle using a truncated cone approximation. The calculated volume is added to the upload information of the model and saved. The masks are then overlayed on top of the original DICOMs and saved as well. Once processing is finished, the resulting overlayed images and upload information is passed to the "latest upload" template, redirecting the user to that page so they can view the results.

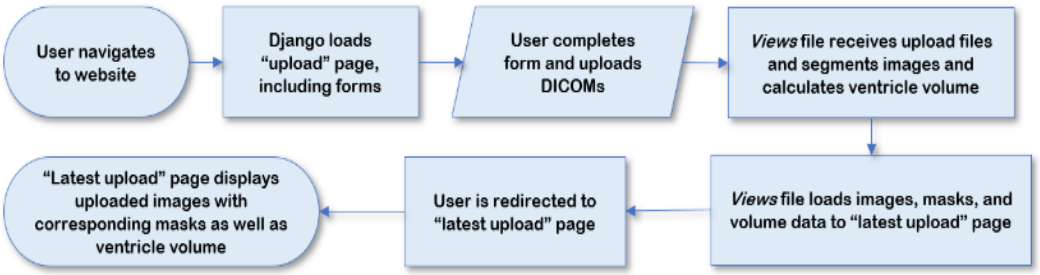

Figure 1: Process flow diagram for a user upload

Upon meeting with Dr. Andrew Cheng (pediatric cardiologist collaborator), it was pointed out that manual selection of patient images corresponding to the ED and ES phases for each ventricle is a labor-intensive task that significantly raised the difficulty barrier for using the website. Thus, a feature for the automatic detection of the ED and ES phases for each ventricle was created. The process flow for this functionality is as follows. When the user creates an upload, they select "auto-detect" in the "contour type" dropdown. When the files are uploaded, they are separated into numbered phases and slices of each phase as labeled in the DICOM metadata. Then the volume for each ventricle is calculated by first segmenting each image in a phase via the FCN model from Karimi-Bidhendi, et al.[2] and then taking a truncated cone volume approximation using the pixel spacing, distance between

slices, and the orientation of the slices. The phase with the largest volume for the left ventricle is selected as the ED phase (LVED) and the phase with the smallest volume for the left ventricle is selected as the ES phase (LVES). The same is determined for the right ventricle (RVED and RVES). These volume values and the corresponding segmented images are stored and reported in the "latest upload" page of the website.

2.4 Results

With regards to the performance of the website, the segmentation of each image takes 249.6ms and 246.6ms for the left and right ventricle, respectively. The complete processing time, from the moment the files are uploaded to the moment the user is redirected to the "latest upload" page is 589ms and 560ms for each image in the left and right ventricle, respectively. With regards to the accuracy of the segmentation, the model used to segment the images has a reported accuracy of 91.3% for the LV at the ED phase, 86.7% for the LV at the ES phase, 84.5% for the RV at the ED phase, and 77% for the RV at the ES phase.[2] Figure 2 is a screenshot of the "latest upload" page of the website. Since the model was trained on images that are pre-processed with a different downsampling package than the one used on the website, a test was done to verify that there is no significant change in mask generation accuracy. 298 images were pre-processed using each package, and dice metrics were calculated for the masks generated from those images (Table 2). The overall mean dice coefficient for the LV masks is 0.99 with a standard deviation of 0.02 and for the RV masks is 0.95 with a standard deviation of 0.10. The small difference in masks is due to the scaling done by the packages. The package used for training the model rescales the images to 0 – 255 while the package used on the website scales the images to 0 – 1. However, the

model contains a normalization layer that counteracts these differences, resulting in similar masks.

It is important to note that, due to the more challenging nature of the RV segmentation model, as mentioned above, it is less robust to the downsampling package change than the LV segmentation model. Thus, when this change is introduced on the website, the RV segmentation model performance is slightly less consistent than the LV segmentation model performance. This results in the lower mean dice coefficient as well as the higher standard deviation for the RV masks as compared to the LV masks. This does not imply, however, that the resulting volume calculations are less accurate, as this is only a comparison of the masks generated between the two pre-processing methods and not a comparison to the ground-truth masks used to train the models.

Subject	LV Mean Dice Coefficient	LV Standard Deviation	RV Mean Dice Coefficient	RV Standard Deviation
2	0.98	0.02	0.89	0.08
11	0.98	0.04	0.94	0.09
12	0.996	0.003	0.96	0.06
13	0.996	0.004	0.95	0.07
14	0.99	0.02	0.97	0.05
15	0.99	0.003	0.85	0.21
16	0.99	0.02	0.95	0.14
17	0.99	0.01	0.98	0.02
18	0.99	0.01	0.90	0.09
19	0.98	0.04	0.99	0.01
20	0.99	0.03	0.99	0.01
21	0.99	0.01	0.98	0.02
22	0.99	0.01	0.95	0.08

Table 2: Comparison of pre-processing methods between the website and that used for training the segmentation model

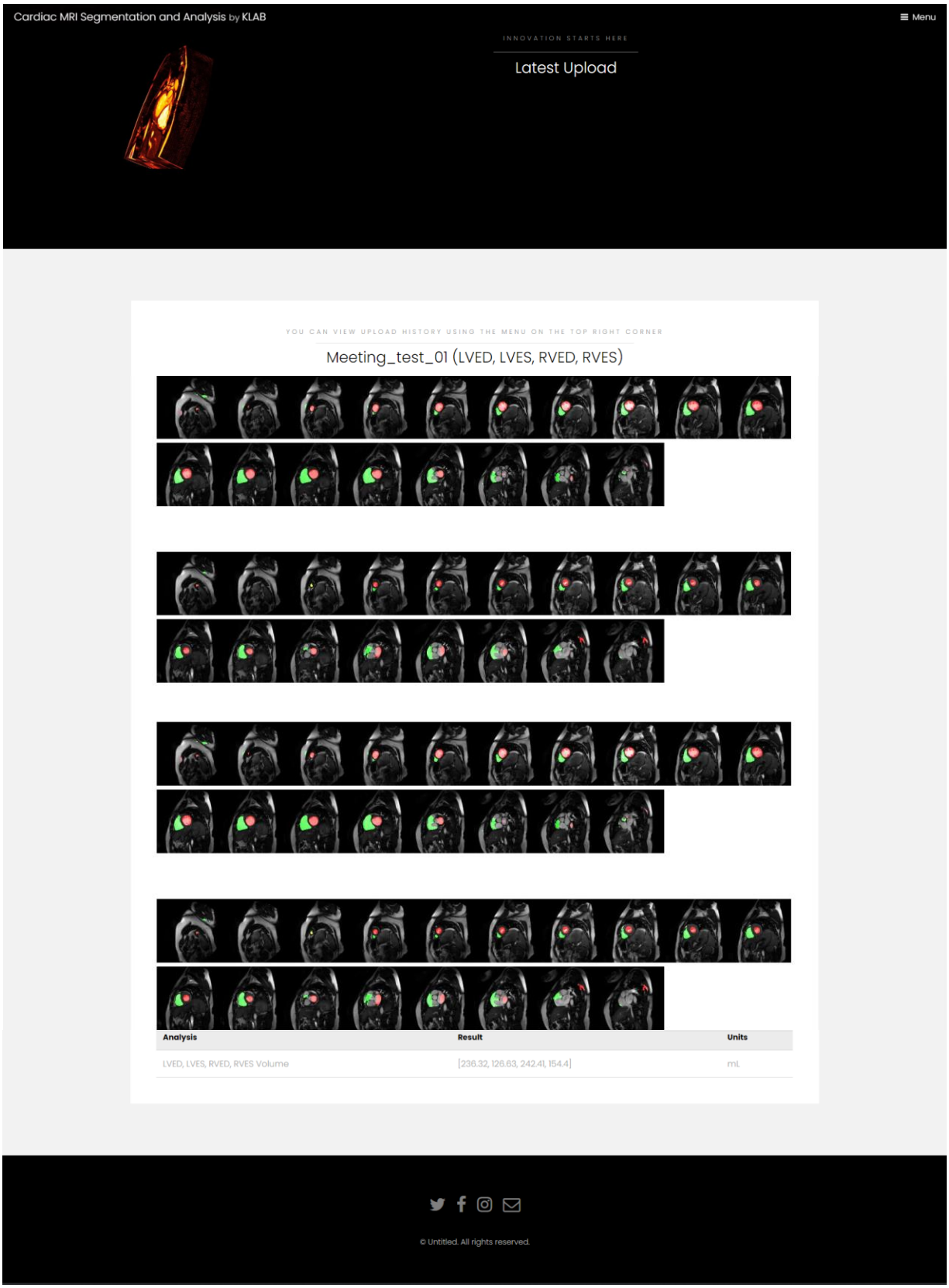

Figure 2: Cloud-based platform “latest upload” page

2.5 Limitations and future work

With regards to future work on the MRI segmentation website, there is room for a plethora of features that can be added as the need arises. One such feature is automatic detection of ventricles in an MRI image slice. The segmentations model used to process the uploaded images was trained on images that all contained the ventricle being segmented. However, when uploading a full patient imaging session, there are images within each cardiac phase that may not contain either ventricle. In these cases, the model attempts to find a ventricle and segment it and does not ignore images with no ventricle. As a result, these images would be incorrectly segmented and contribute to the volume calculation, leading to incorrect results with regards to both phase selection for ED and ES as well as ventricle volume. To fix this issue, a new model must be trained that can identify the existence of a left or right ventricle. This model must then be integrated into the pre-segmentation image processing steps to label or filter out all images that do not contain a ventricle. The flexibility of Django also allows the creation of new apps that can support entirely different purposes on the back-end system. Continuing to work closely with Dr. Cheng and all other future users of the website is vital to making improvements and working on new features or apps.

CHAPTER 3: AI-based analysis of medical images for COVID-19 patients

3.1 The global impact of COVID-19 on hospitals

As the COVID-19 pandemic grew globally, the healthcare systems of every country were tested. A study in the United States examining the period from March 26th, 2020 to June 30th, 2020 found that on April 12th, 2020, among 13 geographically dispersed hospital locations, only three had an ICU capacity under 80% and six had an ICU capacity over 100%.[29] When looking at countries other than the United States, the situation becomes even worse. A study of the availability of hospital beds during COVID-19 found that multiple countries had less than 200 hospital beds per 100,000 people and less than 1 ICU bed per 100,000 people (Figure 3 and 4).[30]

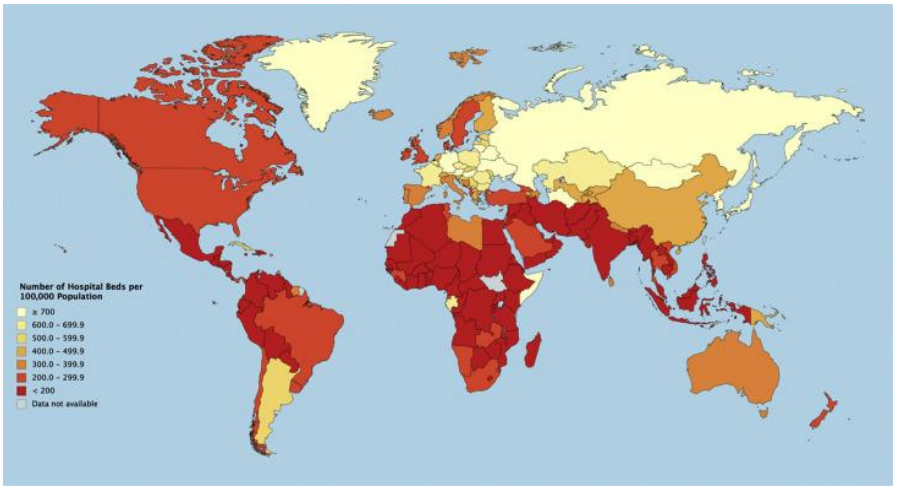

Figure 3: Number of hospital beds available per 100,000 population, from Sen-Crowe, et al.[30]

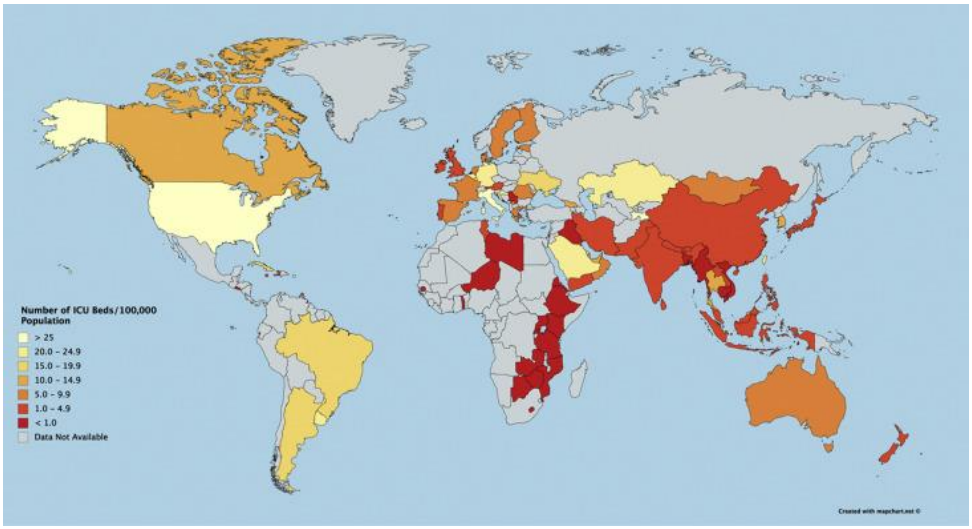

Figure 4: Number of ICU beds available per 100,000 population, from Sen-Crowe, et al.[30]

Due to the high demand for hospital beds and particularly ICU beds, hospitals have to create a prioritization system to ensure that the patients who need ICU beds the most gain access to them. Roy et al. created a prioritization tool based on patient data such as preexisting conditions, body mass index (BMI), and age.[31] Leclerc et al. also created an ICU prioritization tool for the COVID-19 pandemic based on patient data.[32]

3.2 Applications of AI and Deep learning in the global COVID-19 effort

From the moment that the scientific community realized that COVID-19 would turn into a large-scale pandemic, research on the topic, particularly with regards to AI and deep learning, accelerated rapidly. To understand the scale of the COVID-19 pandemic, the need for cost-effective and accurate testing methods quickly arose. A false negative meant that the virus would spread undetected and make contact tracing very difficult. Every extra hour that a test took to determine results meant the potential for another contagious interaction with the individual. As test strips took days for a response, AIJame et al.[10] sought to create a testing solution that was both early and accurate. To accomplish this, a routine blood test was selected as the testing method. This allowed sufficient sample

collection without the need to waste traditional testing swabs, which were in very short supply in the earlier stages of the pandemic. With publicly available datasets, a deep learning model was trained to predict whether or not the patient had been infected. The model was a deep forest (DF) type, where multiple classifiers make a prediction at different layers and a consensus is achieved. This model has a reported accuracy of 99.5%.[10]

As the virus spread further, the need for a remedy was clear. Before the widespread availability of mRNA vaccines, many different existing drugs were recommended and circulated as effective medication for COVID-19 symptoms. However, most of the recommendations required significant testing for physicians to be confident in their recommendation. Jha et al.[12] suggested a deep learning method to work backwards and find an existing drug that best fit the virus. To achieve this, a quantitative structure-activity relationship (QSAR) model was used in combination with linear regression, support-vector machines (SVM), and random forest to construct a deep learning model that analyzed binding affinities and protein interactions and then trained a molecular descriptor dataset to discover existing drugs that would best prevent the multiplication of the COVID-19 virus.[12]

When patients began to stream into hospitals at an increasing rate, the need for a method of quick analysis of symptoms was evident. Liu et al.[17] proposed a deep learning method that allowed the segmentation of the infected region of lungs from CT images. The purpose of doing so was to provide physicians with an analysis tool that could decrease the time required to diagnose patients' COVID-19 symptoms. The deep learning model that segmented the images leveraged attention-aware feature fusion and large receptive fields

to combat low boundary contrast and large variations in infections, two prominent issues related to COVID-19 lung infection segmentation.[17]

Although chest X-rays (CSR) are more simple and more mobile than CT scans, they have been reported to be less informative in the early stages of infection.[18] To address this issue, Prakash et al.[18] proposed leveraging the SqueezeNet classifier model and modifying it for segmentation. The output is then used to classify the input CXR image as COVID-19, viral pneumonia, or normal. With regards to performance, this model, called the COVID-19 Super pixel SqeezeNet (COVID-SSNet), can distinguish between these three different classes with a reported accuracy of 99.79%.[18] Alternatively, Zhang et al.[19] focused on improving segmentation efficiency of COVID-19 lung infection lesions on CT images. A major roadblock to deep learning in the first months of the outbreak was the lack of sufficient manually segmented images for deep learning training. To work past this issue, Zheng et al. implemented two encoders. One encoder was trained on non-COVID-19 patient CT images to detect and segment lung lesions. The other encoder was trained to target COVID-19 lesions. The features extracted by the two encoders were combined for a decoder segment. This method improved the dice similarity coefficient of the segmented lesions by 3%.[19]

Another way to circumvent the lack of segmented images was to not rely on segmentation for classification. Existing models such as ResNet, DenseNet, and VGG-16 have already been in use for various image classification tasks, including chest X-ray and CT scan analysis. As COVID-19 diagnostic image databases grew, the amount of data required to train these large models became sufficient. Many researchers have created new models based on the previously mentioned existing models to show that CXR and CT

images can be used for classification tasks for COVID-19 with accuracies above 90% in most cases.[13, 14, 15, 16] The goal of these efforts has been either binary classification to determine positive or negative COVID-19 status or multi-class classification to differentiate between COVID-19 and other lung infections such as pneumonia.

One of the biggest challenges to bringing deep learning and AI into practice at hospitals is explainability. This refers to the idea that for a physician to make a potentially life-threatening decision, they need to understand the reason that a particular diagnosis/decision is reached. The reason this is a challenge for AI and deep learning is that even though the models are initially created by humans, they become a mystery black box once they have been trained. This means there is no way for any person to understand the reasons behind the decision of a traditional deep learning model. To combat the issue of explainability and bring machine learning tools to the front line of hospitals, Pennisi et al.[20] integrated their two-part deep learning model with a user-friendly interface. One part of their model is a CT scan segmentation model and the other is a lesion categorization model trained to detect COVID-19 lesions. With a categorization accuracy of 84%, this online platform allows physicians to both visually see the segmentation of the images where lesions are identified as well as the decision made by the algorithm based on these lesions. This approach greatly improves the explainability of the models because physicians can confirm that the model is, in fact looking at the right areas and allows a degree of trust towards the platform.[20]

As the pandemic problems moved beyond detection, hospitals found themselves overwhelmed by the number of patients requiring ICU attention. The scarcity of ventilators also contributed to the worsening situation. While companies were hard at work

manufacturing more ventilators, Perumal et al.[21] were applying deep learning to help physicians determine which patients needed ICU attention the most. This was accomplished by leveraging VGG-16 and linear regression to take CT scan data, patient metadata, laboratory results, and any reported symptoms to create a two-level symptom categorization tool. The two levels were early symptoms and severe/late symptoms.[21]

Despite strong advancements in image analysis for both CT scans and CXRs, both are still expensive for developing nations and expose patients to radiation. Lung ultrasound, however, can be more accessible and has no radioactive side-effects. However, a great amount of expertise is required to analyze the data from a lung ultrasound, so it is not commonly used for diagnosis of COVID-19. La Salvia et al.[22] and Diaz-Escobar et al.[23] took on this challenge by using deep learning techniques to analyze the lung ultrasound data. The latter achieved 89% average accuracy in detecting COVID-19. As new treatments for COVID-19 emerged, recovery times based on symptom severity were an important metric to use for comparison. Wang, et al.[24] created a tool to determine these metrics by applying deep learning to predict how long it would take for hospitalized COVID-19 patients to recover. One of the greatest features of this tool is that on top of reporting estimated recovery times, it reports which features or characteristics are associated with accelerated recovery times. The models trained on their dataset reported that treatment schemes, age, symptoms, comorbidities, and biomarkers have the most impact on recovery times.[24]

The emergence of hundreds of deep learning tools with regards to COVID-19 screening from CXR data and the constant demand for more work in this area has made it difficult to stay on top of validation. Sadre et al.[25] have created a method that validates these deep learning models so that their quality is ensured for real-world applications. Namely, their

ROI Hide-and-Seek protocol either hides or emphasizes specific regions of CXR images to test the models' skill at anomaly detection as well as how that relates to radiological signatures. This is a factor that tends to be overlooked in many current deep learning tools.[25]

Deep learning and COVID-19 are not limited to hospital settings. Organizations such as the CDC and WHO as well as global leaders are in need of information regarding the spread and world dynamics considering COVID-19 to make decisions that would affect large populations. The ability to predict the spread of the virus remains a vital task in being able to fight the pandemic. Iloanusi, and Ross[26] leveraged deep learning along with global weather pattern data to create a dynamic tool that can predict the spread of COVID-19 infection as the seasons change. 36 countries from 5 different continents were studied via regression analysis to see which factors are the most important to consider for the random forest and deep learning models. The models predict the COVID-19 case to mortality ratio based on these factors. Temperature was found to be a very important factor for countries not in tropical locations. Being able to leverage these unique factors for each country is what makes the forecasting tool powerful.[26]

However, the introduction of new variants of COVID-19 has complicated forecasting efforts. To study the impact of such variants on forecasting, Rashed, and Akimasa,[27] looked at COVID-19 daily positive cases within Japan since the outbreak started and compared that data to their deep-learning based forecasting model, which was trained on Japan's data. They found that the models had high accuracy for periods before the introduction of the Alpha variant. This variant caused an unexpected increase of the daily positive cases for which the model was not able to predict. They also found that it took the models four

weeks to adjust their predictions and regain high accuracy. These studies are of great importance to the current and future work on forecasting models because they outline the value of incorporating variant introduction and what effects these events may have.[27]

CHAPTER 4: Data acquisition and analysis to predict COVID-19 outcome

4.1 Background

Although there are many deep learning applications for detection of COVID-19 and for segmentation of lung infection lesions as mentioned above, there is still a need for quick and efficient patient outcome prediction methods. Patient outcome prediction is a vital metric for hospitals that are overburdened by COVID-19 patients. A deep learning tool that can inform physicians of the care needs for each patient based on their current lung infection severity will help hospitals save more lives by providing ICU care to the patients that need it the most.

4.2 Data sources

Four batches of both images and patient data were acquired for training a model to predict COVID-19 patient outcome. Two batches of anonymized data were sent from UCI's medical center containing 715 CXR images and 313 CXR images, respectively, and the other two batches were sent from UCLA's medical center containing 3,278 CXR images and 1,998 CXR images, respectively, for a total of 6,259 images. Each patient is labeled with a four-tier outcome status where label 3 indicates outpatient/healthy, label 2 indicates inpatient/non-ICU, label 1 indicates ICU necessary, and label 0 indicates death. A table detailing the number of images per label and per batch is below (Table 3). With regards to metadata, the patients' age, race, gender, BMI, and comorbidities including diabetes, CKD/ESRD, Asthma/COPD, and obesity were provided.

Batch	Label 0	Label 1	Label 2	Label 3	Total
1	109	269	131	206	715
2	610	997	620	1051	3278
3	3	2	107	201	313
4	151	96	1279	472	1998
Total	873	1364	2137	1930	

Table 3: Number of image data from each batch and for each outcome label

4.3 Splitting of available data for training purposes

For training, the available data was split into a roughly 80:20 ratio of training images to test images. This split was made per label for each batch to maintain equal splitting for each label. The training data was further split by a ratio of 80:20 for training and validation, respectively. The two tables below show the actual number of images per label and per batch for training (including validation) and for testing, respectively (Tables 4 and 5).

Batch	Label 0	Label 1	Label 2	Label 3	Total
1	79	219	102	152	552
2	483	783	503	816	2585
3	2	2	90	105	199
4	119	78	1045	400	1642
Total:	683	1082	1740	1473	

Table 4: Data split for training and validation

Batch	Label 0	Label 1	Label 2	Label 3	Total
1	30	50	29	54	163
2	127	214	117	235	693
3	1	0	17	96	114
4	32	18	234	72	356
Total:	190	282	397	457	

Table 5: Data split for testing

CHAPTER 5: A VGG16 model for the prediction of COVID-19 outcome from images and metadata

5.1 Background

Machine learning is a method where a computer tries to improve its ability to predict an outcome based on a set of training inputs. With every iteration, where the computer processes the inputs and makes a prediction, it uses a separate set of validation inputs to calculate its prediction error. Then, it automatically adjusts its equation to reduce this error. After repeating this process enough times, the model has been trained and has reached its best accuracy. Deep learning is a subset of machine learning, where "hidden" processing layers between the input and output layers are inserted to increase complexity of the model. To evaluate the model, it is provided a test input that is not part of the training or validation data and makes a prediction. The accuracy of this prediction is said to be the accuracy of the model.

There are multiple types of models that can be trained, and each has a different application or advantage based on the type, amount, and size of the data. A deep neural network (DNN) consists of a series of interconnected layers that contain nodes through which the data passes. As each data point passes through a node, it is processed, and an output is sent to every node in the next layer. The number of nodes in each layer and the number of layers can vary and are parameters that can be optimized. Once the flow of data makes it through all the internal layers, or "hidden" layers, it arrives at the final prediction/output layer. The number of nodes in this layer corresponds to the number of choices the model has. Each node in this layer will report a probability of being the correct

prediction, and the node with the highest probability is selected as the final prediction (see Figure 5 below).

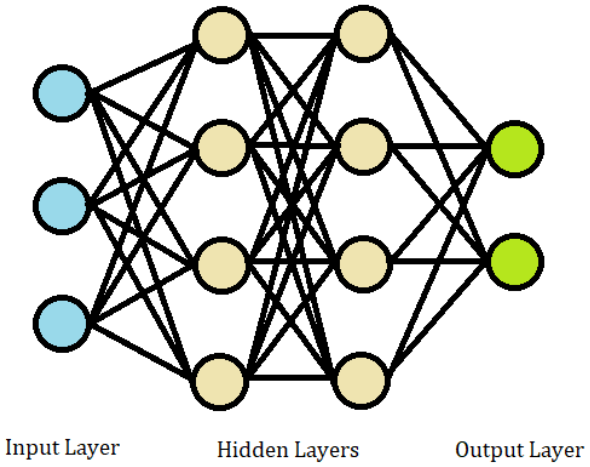

Figure 5: Example of a fully connected deep neural network, original image

A convolutional neural network (CNN) seeks to efficiently dissect an image and extract features from that image before inserting those features into a DNN. To do this, the model first creates a random matrix of 0s, 1s and -1s, called a kernel, that it passes over the pixels of an image. Starting from the top row and moving to the right, the kernel extracts the total value of the pixels within the area covered by the kernel after each pixel is multiplied by the corresponding value in the kernel. It then saves this value to the top left cell of a new matrix and the kernel moves one step (stride) to the right to repeat the process (see Figure 6 below). After the row is finished, the kernel begins again on the left side of the second row and so on until the entire image is processed. This is repeated multiple times with different random kernels. This step is called the convolutional step. Then, another kernel size is selected and placed over the top left portion of one of the new matrices. However, unlike the convolutional step, the cell within the kernel space that has the largest value is selected, and that value is stored in a new matrix. The kernel then moves over to the next

set of values and moves along as before for each matrix created by the convolutional layer (see Figure 7 below). This step is called the max pooling step and reduces the size of the matrices while extracting the most dominant features. Multiple convolutional and max pooling layers may be added to further process and reduce the complexity of the input images. After the final max pooling layer, the output is "flattened" by connecting all the rows of each matrix into a single list of values and is then fed into a DNN for prediction (see Figure 8 below).

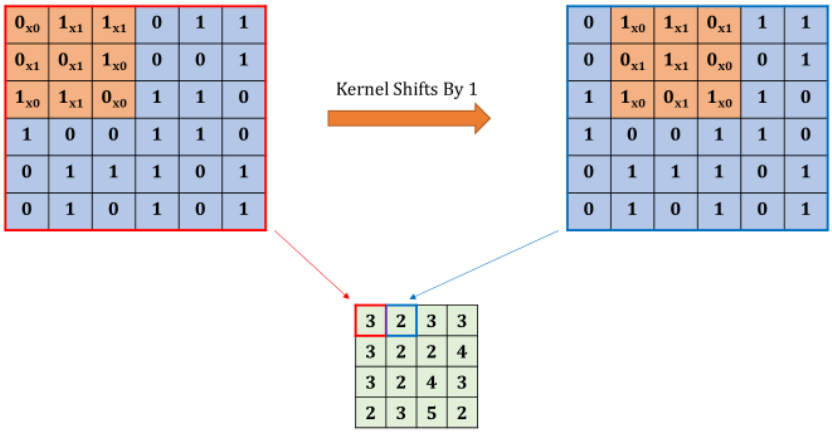

Figure 6: Convolutional layer showing progression of a kernel across a pixel matrix, altered from https://towardsdatascience.com[33]

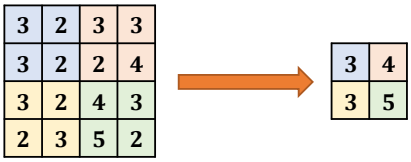

Figure 7: CNN max pooling layer showing how the kernel moves across the convolved matrix, altered from https://cs231n.github.io[34]

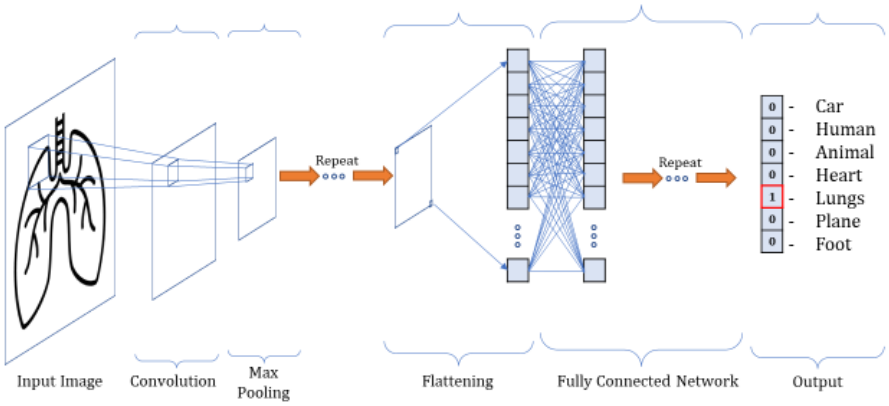

Figure 8: Combined CNN structure including convolution, pooling, and a deep neural network, altered from https://towardsdatascience.com[33]

The VGG16 CNN model was chosen as a baseline for the COVID-19 dataset training due to its proven top 5 accuracy of 92.7% on ImageNet, a very large image database with 1000 classification types.[35] This model features 13 convolutional layers, 5 pooling layers, a convolutional kernel size of 3x3 with a stride length of 1, a pooling kernel size of 2x2 with a stride length of 2, and two final dense layers (see Figure 9 below).

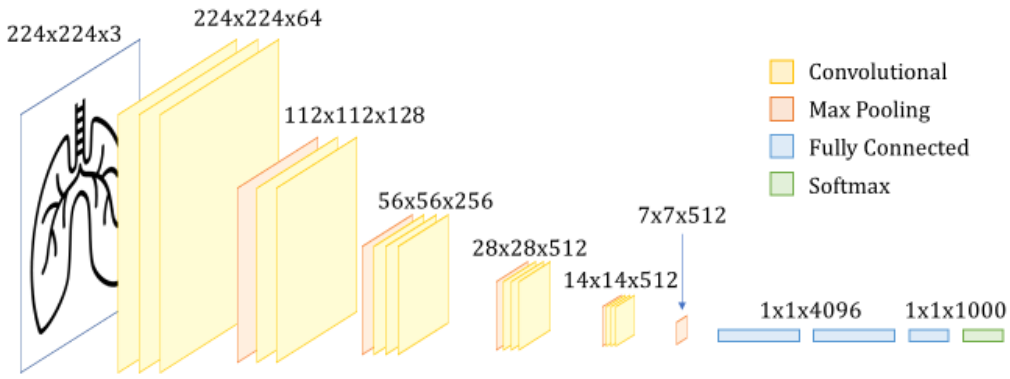

Figure 9: VGG model structure visualization, altered from https://neurohive.io[36]

Another prediction method that will be used below is the support vector machine (SVM). This method involves fitting an equation to a set of data that allows it to best be split by slightly adjusting the coefficients within the equation in each iteration to decrease the error in prediction. The complexity of the equation is one of the variable parameters used to optimize the SVM. Although an SVM is simpler than a DNN, it can be more effective at prediction than a DNN depending on the nature of the data.[38]

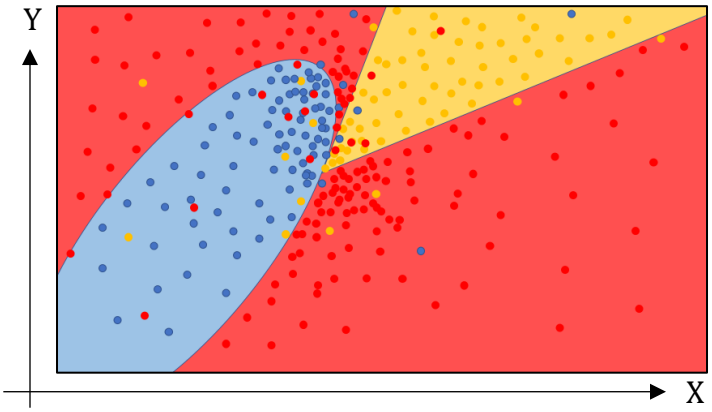

Figure 10: Visualization of support vector machine classification task for 2D input data, altered from https://scikit-learn.org[37]

5.2 Computational Platform

All training was completed on an in-house server at University of California, Irvine. The server contains four GPUs that were used for training including one NVIDIA RTX A6000 GPU with 48GB of VRAM and three NVIDIA GeForce GTX 980 GPUs with 6BG of VRAM each. This allowed four models to train at the same time, while focusing the heavier models on the RTX A6000.

5.3 Model Structure

Two models were chosen for comparison. The first model consisted of the VGG16 framework described above, with slight variations. A mean-variance standardization is performed on multiple layers within the model. Also, dropout layers, where some of the

data is randomly removed, are used throughout the model. The number of kernel filters as well as the number of nodes within the DNN layers is also different (see Figure 11 below). The input for this model is the images from the training dataset described above. To factor in patient metadata, a second model was created that adds two additional DNNs to the VGG16 model. The first additional DNN (DNN1) separately processes the patient metadata and the second additional DNN (DNN2) processes the combined output from the original VGG16 model and DNN1 (see Figure 11 below). This model allows training with both images and patient metadata to explore benefits from metadata utilization. The comparison of these two models provides insight into the impact of patient metadata on the prediction results.

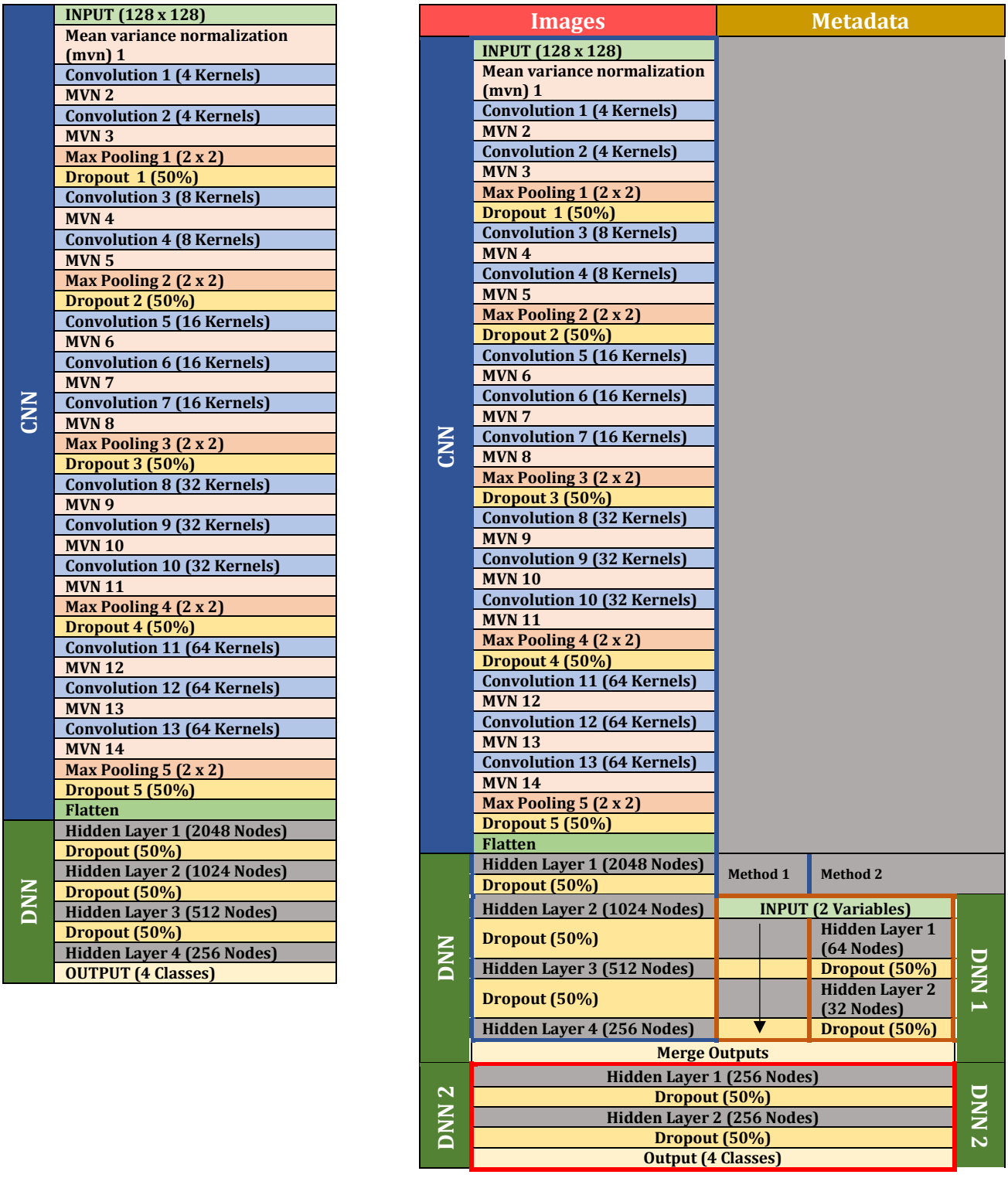

Figure 11: Two VGG16 models for processing only COVID-19 chest x-ray images (left) and for processing both images and patient metadata (right)

5.4 Support Vector Machine Utilization

To implement SVMs, the outputs of the CNN flatten layer and the raw metadata were merged and used as input. An algorithm was used to automatically loop through all the parameter options of the SVM as available through the Scikit-learn SVM package.[38]

5.5 Results

The first training goal was to get a baseline accuracy drawn from only images and the VGG16 model. Each image was extracted, downsampled to 128x128, labeled with a patient outcome, and fed into the CNN. The models were also tested with larger images of size 256x256. However, since this did not improve the accuracy of any of the models described above, the 128x128 images were preferable as they greatly reduced the training time. After repeating the training multiple times to optimize the parameters, including learning rate, batch size, and the number of nodes in the DNN segment, the highest accuracy achieved was 56%. As there were four different outcome classes, the accuracy of random prediction would be 25%. Thus, 56% accuracy is over two times random prediction. The next step was to repeat this with the model that also factors in the metadata and includes one CNN and two DNNs (see Figure 12 above). This model involved more parameters to adjust including the number of nodes and number of hidden layers in DNN1 and DNN2, all the parameters from the first model, and what size output to merge with the CNN. As a result, the optimization of these parameters involved a lengthy testing process where each was adjusted independently to see what the best value would be. The resulting accuracy after optimization improved to 59%. It was also determined, by selecting different metadata to input, that patient age and BMI were the most effective categories. When any other category was included, the accuracy was lower (ranging from 56% to 58%). This may be

due to the binary nature of the other data categories such as comorbidity existence (see Conclusions section below). These results meant that the addition of metadata slightly improved the accuracy of the model, but not significantly as there was only a 3% improvement compared to the image-only model.

With regards to the SVM implementation, the DNN1 output and the CNN output were selected as the data source for SVM training. To optimize the SVM, an algorithm was set up that automatically loops through all parameter options for the SVM, as available through Scikit Learn SVM tools.[38] These parameters included changing the kernel function type between linear, polynomial, radial base function (RBF), and sigmoid. Each kernel function type included different parameters, such as degree for polynomials, tolerance, gamma, and shrinking. The parameter settings with the highest accuracy (57%) were as follows: a third-degree polynomial with a gamma set to "auto", shrinking set to true, and tolerance set to 0.001. Gamma refers to a coefficient that is equal to 1/n_features. Shrinking refers to a process where input variables that do not change with subsequent iterations are removed to "shrink" the number of features trained and decrease training time.

CHAPTER 6: Implementation of the CheXNet learning model for the prediction of COVID-19 outcome from X-Ray images

6.1 Background

CheXNet is a machine learning model that was created by Pranav et al.[5] to diagnose 14 different diseases from x-ray images with a better F1 metric than average radiologist performance. The model has been converted to a Python-Keras format that is available on GitHub.[39] It is a 121-layer CNN that accepts an input of 224x224x3 images and outputs a 14-class prediction.

The CheXNet model was a good fit to validate our results using the VGG16 model training—as described earlier—given that its original intended application was chest X-rays. To apply the CheXNet model to our dataset and compare its output with the VGG16 model's, we had to slightly modify the CheXNet model. The first functional modification to the Python-Keras CheXNet code was to integrate the same image pre-processing steps and patient data splitting as the VGG16 models. This ensured that the input was identical for both the VGG16 models and the CheXNet model. The second functional modification was the input type and output type. For input, the images were converted from single-channel to three-channel as per the requirements of the CheXNet model. Since the images in the COVID-19 dataset were single-channel, they were copied to channels two and three, creating a three-channel image with identical channels. For the output, the original CheXNet code was created to diagnose a chest x-ray among 14 disease options. This was modified so that the output layer predicted four different classes correlating to the four patient outcome labels, mirroring the output of the VGG16 models.

6.2 Results

The CheXNet models were trained and optimized using the exact same image data as the VGG-16 models and with the same split of training and testing sets. The data included 6259 images from four different datasets sent from both the UCI and UCLA medical centers. The images were split in an 80:20 ratio for training and testing (Tables 4 and 5). The CheXNet model was also trained on the same hardware as the VGG-16 models. The main difference was in the training and testing batch sizes. Since CheXNet is a much larger model and requires more memory, a batch size of 10 was selected for optimal processing time. CheXNet model variations were also parameter-based. To keep as close to the original CheXNet model optimization, only the initial learning rate, learning rate reduction factor, number of epochs, and batch size were optimized. The highest accuracy of 60% was achieved with 2,000 epochs, batch size of 10, learning rate of 0.01, and learning rate reduction factor of 0.1.

CHAPTER 7: Comparison between the VGG16 models and CheXNet

7.1 Structural comparison between the model types

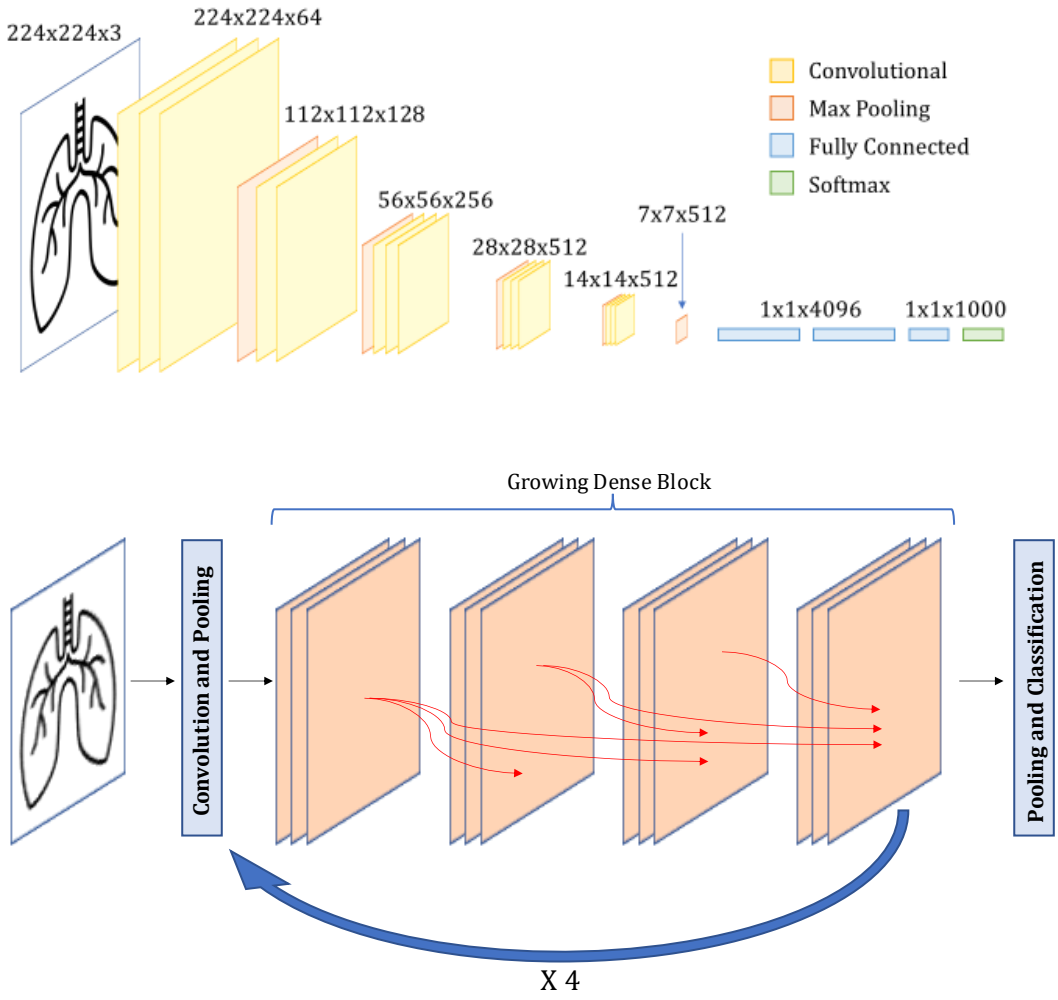

Figure 12: Visual comparison of VGG16 (top) and CheXNet (bottom) model structures, altered from https://neurohive.io[36] and Huang, Gao et al.[41], respectively

The purpose of implementing the CheXNet model was to compare its performance to the VGG-16 model. Since the CheXNet model has already been proven effective with and is designed for CXR image analysis, it is a good model to compare the custom VGG-16 model to. If the accuracy of the CheXNet model greatly differs from that of the VGG-16 model, it

would indicate that either an error was made in the design on the VGG-16 model, or that the VGG-16 model was not a good choice for this usage.

With regards to model structure, the VGG16 model is simpler than the CheXNet model with fewer layers and fewer parameters for training. On the other hand, the CheXNet model leverages the Keras DenseNet121 model, which includes 121 layers. It includes four dense growing blocks with a convolutional and pooling layer between each. Dense growing blocks are sets of dense networks that are connected across the layers as well as between (see figure 12 above). This counteracts a common issue among large CNN models, which is the "washing away" of relevant data as more layers are added. This occurs due to the large amount of data manipulation inherent in a CNN. To address this, each layer in a dense growing block receives the output of every preceding layer in that block as an input. Thus, the fourth layer receives the output of the three preceding layers. This keeps the information from the first layer relevant after multiple layers of manipulation and prevents it from "washing away."[41] However, the shorter nature of the VGG16 model also prevents this from occurring due to its relative simplicity.

7.2 Comparison between the results

Both the VGG16 models and the CheXNet model were able to achieve a similar highest accuracy of 59-60%. This similarity indicates that the accuracy of the VGG16 models is not limited due to the model's structure or to any flaws during construction, but rather it is limited due to the images' lack of sufficient riches for accurate prediction of COVID-19 outcome. Since the CheXNet model was specifically designed for analysis of chest x-rays with high accuracy, it should have performed better leading to an improved accuracy, if there were enough information within the CXR images.

CHAPTER 8: Conclusions

8.1 Results

The first goal of this thesis was to create a cloud-based website that allows physicians to quickly analyze pediatric heart MRIs for diagnosis. The American Heart Association's Precision Medicine Platform was used to host a Python and Django based web framework that leveraged the trained machine learning model from Karimi-Bidhendi, et al.[2] to segment ventricles of pediatric cardiac MRI and calculate the corresponding ventricle volumes for the end-systolic (ES) and end-diastolic phases (ED). The website is also able to automatically differentiate between the ED and ES phases for the left and right ventricles, which allows physicians to upload all the DICOM files for a patient's MRI session and get the resulting volumetric reports within seconds. The total time for image processing and volume calculation is 589ms and 560ms per image for the left and right ventricle, respectively. For just the segmentation task, it takes 249.6ms and 246.6ms per image for the left and right ventricle, respectively.

For the second goal of creating a machine learning model to predict COVID-19 patient outcome, two VGG16 models were optimized for the prediction of COVID-19 outcome. The first model focused on modifying and optimizing the standard VGG16 model for the COVID-19 datasets obtained from UCI and UCLA medical centers. This model was able to achieve an accuracy of 56%.

The second model extended the original VGG16 model to incorporate a DNN (DNN1) that processed patient metadata and another DNN (DNN2) that processed both the output of the image CNN and the metadata DNN. This model, after optimization, achieved an accuracy of 59%. To investigate the reason behind the lower-than-expected accuracy of the

model, the CheXNet model was adapted for use with the COVID-19 dataset and prediction outcomes. After optimizing some of the parameters of the CheXNet model, the highest accuracy achieved was 60%. This meant that the accuracy of the VGG16 models was limited by the informativity of the data as opposed to the quality of the models. Furthermore, the 14 available categories of metadata were not all effective at improving the accuracy of the VGG16 models. Only the age and BMI of the patient were beneficial. However, the impact of metadata on the accuracy of the models was also small (only 3% increase). This indicates that the patient metadata had a low correlation with the outcomes. It also indicates that factors including gender, race, and comorbidities did not help the model predict the outcome of the patients.

8.2 Study limitations and future work

The data acquired for the COVID-19 patient outcome prediction models were received in four separate batches between the summer of 2020 and the spring of 2021. During this time, hospitals and other medical facilities learned how to provide better care for COVID-19 patients.[40] As new methods and practices emerged and were implemented, the outcomes for many patients changed. A patient that might have required ICU attention or could have died due to deteriorating health in June of 2020 may have been an outpatient with mild symptoms in February of 2021 after healthcare procedures improved. This can cause complications with prediction since these factors cannot be considered during model training. Traditional methods to combat this type of issue, such as separate training for each data batch or only training on the latest data, have drawbacks. Most importantly, there was not enough data available to separate trainings or to only train on the latest data. Furthermore, each batch of patients was recorded over an extended period as opposed to a

short time span. This further complicates any attempt to segregate the data as even the more recent batches would get split, reducing the available data again.

References

1. Chan, Heang-Ping et al. “Deep Learning in Medical Image Analysis.” Advances in experimental medicine and biology vol. 1213 (2020): 3-21. doi:10.1007/978-3-030-33128-3_1
2. Karimi-Bidhendi, Saeed, et al. "Fully-automated deep-learning segmentation of pediatric cardiovascular magnetic resonance of patients with complex congenital heart diseases." *Journal of Cardiovascular Magnetic Resonance* 22.1 (2020): 1-24.
3. Avendi, M. R., Arash Kheradvar, and Hamid Jafarkhani. "A combined deep-learning and deformable-model approach to fully automatic segmentation of the left ventricle in cardiac MRI." *Medical image analysis* 30 (2016): 108-119.
4. Avendi, Michael R et al. “Automatic segmentation of the right ventricle from cardiac MRI using a learning-based approach.” *Magnetic resonance in medicine* vol. 78,6 (2017): 2439-2448. doi:10.1002/mrm.26631
5. Rajpurkar, Pranav, et al. "CheXNet: Radiologist-level pneumonia detection on chest x-rays with deep learning." *arXiv preprint arXiv*:1711.05225 (2017).
6. Esteva, Andre, et al. "Dermatologist-level classification of skin cancer with deep neural networks." *nature* 542.7639 (2017): 115-118.
7. Saba, Luca, et al. "The present and future of deep learning in radiology." *European journal of radiology* 114 (2019): 14-24.
8. *Proven Radiology AI.* Adioc. (2021, July 19). https://www.aidoc.com/
9. *Medical imaging cloud ai*. Arterys. (n.d.). https://arterys.com/clinicalApp/ cardioapp.

10. AlJame, Maryam et al. “Deep forest model for diagnosing COVID-19 from routine blood tests.” *Scientific reports* vol. 11,1 16682. 17 Aug. 2021, doi:10.1038/s41598-021-95957-w
11. Chen, Zhiyi et al. “Neural connectome prospectively encodes the risk of post-traumatic stress disorder (PTSD) symptom during the COVID-19 pandemic.” *Neurobiology of stress* vol. 15 100378. 8 Aug. 2021, doi:10.1016/j.ynstr.2021.100378
12. Jha, Nishant et al. “Deep Learning Approach for Discovery of In Silico Drugs for Combating COVID-19.” *Journal of healthcare engineering* vol. 2021 6668985. 20 Jul. 2021, doi:10.1155/2021/6668985
13. Maharjan, Jenish et al. “Application of deep learning to identify COVID-19 infection in posteroanterior chest X-rays.” *Clinical imaging*, vol. 80 268-273. 24 Jul. 2021, doi:10.1016/j.clinimag.2021.07.004
14. Dilshad, Sara et al. “Automated image classification of chest X-rays of COVID-19 using deep transfer learning.” *Results in physics* vol. 28 (2021): 104529. doi:10.1016/j.rinp.2021.104529
15. Nayak, Soumya Ranjan et al. “An Automated Lightweight Deep Neural Network for Diagnosis of COVID-19 from Chest X-ray Images.” *Arabian journal for science and engineering*, 1-18. 9 Aug. 2021, doi:10.1007/s13369-021-05956-2
16. Alshazly, Hammam et al. “COVID-Nets: deep CNN architectures for detecting COVID-19 using chest CT scans.” PeerJ. *Computer science* vol. 7 e655. 29 Jul. 2021, doi:10.7717/peerj-cs.655

17. Liu, Jiannan et al. “COVID-19 lung infection segmentation with a novel two-stage cross-domain transfer learning framework.” *Medical image analysis*, vol. 74 102205. 6 Aug. 2021, doi:10.1016/j.media.2021.102205
18. Prakash, N B et al. “Deep transfer learning for COVID-19 detection and infection localization with superpixel based segmentation.” *Sustainable cities and society*, 103252. 16 Aug. 2021, doi:10.1016/j.scs.2021.103252
19. Zhang, Yichi et al. “Exploiting Shared Knowledge from Non-COVID Lesions for Annotation-Efficient COVID-19 CT Lung Infection Segmentation.” *IEEE journal of biomedical and health informatics*, vol. PP 10.1109/JBHI.2021.3106341. 20 Aug. 2021, doi:10.1109/JBHI.2021.3106341
20. Pennisi, Matteo et al. “An explainable AI system for automated COVID-19 assessment and lesion categorization from CT-scans.” *Artificial intelligence in medicine* vol. 118 (2021): 102114. doi:10.1016/j.artmed.2021.102114
21. Perumal, Varalakshmi et al. “Prediction of COVID Criticality Score with Laboratory, Clinical and CT Images using Hybrid Regression Models.” *Computer methods and programs in biomedicine*, vol. 209 106336. 10 Aug. 2021, doi:10.1016/j.cmpb.2021.106336
22. La Salvia, Marco et al. “Deep learning and lung ultrasound for Covid-19 pneumonia detection and severity classification.” *Computers in biology and medicine*, vol. 136 104742. 8 Aug. 2021, doi:10.1016/j.compbiomed.2021.104742
23. Diaz-Escobar, Julia et al. “Deep-learning based detection of COVID-19 using lung ultrasound imagery.” *PloS one* vol. 16,8 e0255886. 13 Aug. 2021, doi:10.1371/journal.pone.0255886

24. Wang, Jun et al. “iCOVID: interpretable deep learning framework for early recovery-time prediction of COVID-19 patients.” *NPJ digital medicine* vol. 4,1 124. 16 Aug. 2021, doi:10.1038/s41746-021-00496-3

25. Sadre, Robbie et al. “Validating deep learning inference during chest X-ray classification for COVID-19 screening.” *Scientific reports* vol. 11,1 16075. 9 Aug. 2021, doi:10.1038/s41598-021-95561-y

26. Iloanusi, Ogechukwu N, and Arun Ross. “Leveraging Weather Data for Forecasting Cases-to-Mortality Rates Due to COVID-19.” *Chaos, solitons, and fractals*, 111340. 18 Aug. 2021, doi:10.1016/j.chaos.2021.111340

27. Rashed, Essam A, and Akimasa Hirata. “Infectivity Upsurge by COVID-19 Viral Variants in Japan: Evidence from Deep Learning Modeling.” *International journal of environmental research and public health* vol. 18,15 7799. 22 Jul. 2021, doi:10.3390/ijerph18157799

28. *Best-In-Class Cloud Dental Software*. Apteryx. (n.d.). https://info.planetdds.com/dental-imaging-gsa?gclid=CjwKCAjw9uKIBhA8EiwAYPUS3OSGhIjQGbXFnhr6YxOyTHEDTf-cm8XqY7lxSQynKUf6nHY8Gzih3xoCC3AQAvD_BwE

29. Douin, David J., et al. "ICU Bed Utilization During the Coronavirus Disease 2019 Pandemic in a Multistate Analysis—March to June 2020." *Critical care explorations* 3.3 (2021).

30. Sen-Crowe, Brendon, et al. "A closer look into global hospital beds capacity and resource shortages during the COVID-19 pandemic." *Journal of surgical research* 260 (2021): 56-63.

31. Roy, Melyssa, et al. "Rapid Development of a Tool for Prioritizing Patients with Coronavirus Disease 2019 for Intensive Care." *Critical care explorations* 3.3 (2021).
32. Leclerc, Thomas, et al. "Prioritisation of ICU treatments for critically ill patients in a COVID-19 pandemic with scarce resources." *Anaesthesia Critical Care & Pain Medicine* 39.3 (2020): 333-339.
33. Saha, S. (2018, December 17). *A comprehensive guide to convolutional neural networks - the eli5 way.* Towards Data Science. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.
34. Stanford Learning Lab. (n.d.). *Convolutional Neural Networks (CNNs / ConvNets)*. https://cs231n.github.io/convolutional-networks/.
35. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv*:1409.1556 (2014).
36. Neurohive. (2018, November 20). *Vgg16 - convolutional network for classification and detection*. https://neurohive.io/en/popular-networks/vgg16/.
37. Scikit Learn. (n.d.). *1.4. support Vector Machines.* https://scikit-learn.org/stable/modules/svm.html.
38. Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *The Journal of machine Learning research* 12 (2011): 2825-2830.
39. Chou, Bruce, and Lee, Michael. *CheXNet-Keras*. GitHub. https://github.com/brucechou1983/CheXNet-Keras.
40. Buonaguro, Franco M., et al. "Covid-19: time for a paradigm change." *Reviews in Medical Virology* (2020).

41. Huang, Gao, et al. "Densely connected convolutional networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.