SSI

2020 Distinguished Projects

Summer STEM Institute

Preface

The Summer STEM Institute (SSI) is an international virtual summer program for top young scientists and leaders. SSI provides academically talented and highly motivated students the opportunity to learn through a data science and research bootcamp, a Masterclass lecture series, research workshops, weekend challenges, and a mentored research project.

Students in the research program pursue a research project under the guidance of SSI mentors and complete the entire research project lifecycle, from identifying a topic of interest to completing a final paper and presentation. This booklet highlights distinguished papers and projects that students completed over the summer.

SSI is proud to be an international program. In 2020, SSI received 2,450+ applications from 78 different countries, and over 40 countries were represented at SSI over the summer. Below are quotes from students highlighting their experiences from the summer:

"I learned SO much and will continue building upon the strong foundation that I acquired through this program... I was able to meet so many interesting students with interests very similar and vastly different from mine, work with absolutely amazing and generous mentors, and I was actually able to speak to a few very helpful academics in my field that I will be keeping in touch with... The topics SSI taught were very varied and extensive, ranging from machine learning to writing research papers to learning how to write an effective cold email, so to say that I learned a great deal is putting it very lightly, and I am endlessly grateful to have learned so much... I will continue to build upon what I have learned here, which includes continuing to conduct research projects on my own... The opportunities, connections, friendships, and knowledge I acquired from SSI are invaluable."

- *Amanda L., student from the United States*

"Having done the final research presentation I feel so emotional. I'm so grateful for this experience. I got to meet such awesome people. And with all the things I learned this summer I feel so proud. This is truly an unforgettable experience. I learned so much and I really, really enjoyed SSI! I think the beauty in SSI is that it is unlike other programs. This one taught me many skills, not just one, and they were all very valuable. Even though the classes were indeed very challenging, I was able to eventually comprehend all of the concepts, and I'm going to go deeper after the program is over. "

- *Ammar A., SSI student from Saudi Arabia*

"My time in SSI this past summer was a great learning experience. I really appreciated the mentorship and learned a lot along the way... I only wished that there were more hours in the day to fit everything in! I am going to continue to pursue my interest in coding and STEM related research. Because of SSI, I am chipping away at another course on Python to continue mastering that programming language...and I am eager to build on my foundational knowledge in Machine Learning, especially topics geared toward the brain!"

- *John C., SSI student from the United States*

"I acknowledge you for enabling me to experience this intensive and fascinating program! Both the academic intensity and excitement were perfectly balanced in my point of view. Within the help of these lectures, I found the confidence to reach KOLs and leading research facilities in my field. A few weeks ago, I applied and became one of the data accessors of the MWA telescope, which is a leading facility in my research topic that hosted the majority of pilot experiments and frontier projects..."

- *Alper T., SSI student from Turkey*

Summer STEM Institute

Acknowledgements

The Summer STEM Institute (SSI) could not have been possible without the dedicated help of almost 100 instructors, mentors, and speakers, as acknowledged below.

Bootcamp Course Instructors

Conducting Data Science Research Course

Dhruvik Parikh	Stanford University
Franklyn Wang	Harvard University
Anne Lee	Stanford University

Programming for Data Science Course

Alex Tsun	Stanford University
Adam Pahlavan	Stanford University
Aleks Jovcic	University of Washington

Masterclass Lecturers

Amol Punjabi	Harvard University
Amy Jin	Harvard University
Dhruvik Parikh	Stanford University
Michael Truell	MIT
Jiwoo Lee	Stanford University
Joey Feffer	Harvard University
Kasha Akrami	Stanford University
Harshal Agrawal	Stanford University
Sahaj Garg	Stanford University
Katherine Du	Stanford University
Jenny Xu	MIT
Benjamin Spector	MIT
Arjun Ramani	Stanford University
Shawn Chao	MIT
David Zhu	Harvard University
Leo Lo	Columbia University
Anne Lee	Stanford University
Marco Lorenzon	Stanford University
Michael Oduoza	Stanford University
Julius Vering	UC Berkeley
Vahid Fezal-Rezai	MIT
Katherine Ho	Stanford University
Anushka Walia	Yale University
Daniel Wu	Stanford University
John Hallman	Princeton University
Adam Pahlavan	Stanford University
Sabrina Wong	Cornell University
Stephen Yoo	Caltech
Michael Ren	MIT
Eric Zhang	Harvard University

Other Staff Members

Joshua Chiang	Harvard University
Matthew Taing	University of Washington
Sabrina Wong	Cornell University
Naveen Durvasula	UC Berkeley

Summer STEM Institute

Acknowledgements (cont).

Research Mentors

Allen Huang	MIT
Muntaha Samad	UC Irvine
Brian Huang	MIT
Karen Ge	Stanford University
Ethan Weber	MIT
Rahul Khanna	USC
Nithin Buduma	MIT
Javier Cuesta	Stanford University
Cannon Lewis	Rice University
Matthew Tan	Stanford University
Anmol Warman	Duke University
Chris Wang	Stanford University
Hari Sadasivan	University of Michigan
Cierra Beck	Cornell University
Nic Rothbacher	MIT
Eric Xia	UC Berkeley
Yatin Chandar	MIT
Lyna Kim	Stanford University
Eric Frankel	Stanford University
Ryan Tolsma	Stanford University
Jordan Wick	MIT
Aansh Shah	Brown University
Gaeun Kim	Stanford University
James Boggs	University of Michigan
Harshal Agarwal	Stanford University
Allison Tam	MIT
Nathan Lam	Princeton University
Steven Okada	MIT
Jonathan Lu	Harvard University
Dhaman Kaur	MIT
Ross Teixeira	Princeton University
Abhijit Mudigonda	MIT
Gheric Speiginer	Georgia Tech
Stephanie Brito	Stanford University
Jesse Stern	University of Chicago
Shrey Gupta	Duke University
Sydney Zink	Brown University
Anurag Sengupta	UC Irvine

Teaching Fellows

Annie Vu	UT Dallas
Trey Gilliland	Columbia University
Rahul Chandra	University of Washington
Felix Tran	MIT
Yeray Lopez	Villanova University
Matthew Kolodner	Stanford University
Jamie Loeber	UC Irvine
Cassia Trusty	Villanova University
Honson Tran	Rutgers University
Brice Wang	MIT

**We would like to especially acknowledge Franklyn Wang for his formatting, without which this document would not have been possible*

Table of Contents

MRI-based Diagnosis of Alzheimer's Disease using Deep Learning with CycleGAN for Data Augmentation

Sunny Wang

Abstract

Alzheimer's disease is a progressive disease causing deterioration of neurons in the brain, leading to dementia and eventually death. Diagnosis of Alzheimer's conventionally consists of a combination of neuropsychological tests and laboratory tests, and clinical diagnosis accuracy lies at around 77%, As Alzheimer's is associated with loss in brain mass, which can be discerned from MRI scans, it is a suitable task for deep learning and computer vision. An accurate and efficient machine learning model could be of great assistance to physicians as it could reinforce their diagnosis. However, deep learning typically requires large amounts of data, and medical data is often scarce. A recent breakthrough in machine learning, the generative adversarial network (GAN), allows for generation of realistic images, providing a potential solution to lack of data. In this study, we construct ResNet50-based convolutional neural networks to perform Alzheimer's disease classification using MRI scans, achieving an F-1 score of 89%. Furthermore, by generating samples using CycleGAN, we demonstrate that GANs can significantly improve classification accuracy when used for data augmentation, achieving an F-1 score of 95%.

1 Introduction

1.1 Background

Alzheimer's disease (AD) is a progressive disease characterized by the loss of cognitive ability and is the sixth leading cause of death in the United States. The progression can be categorized by severity, consisting of mild, moderate, and severe AD. These stages are typically classified and diagnosed based on a variety of factors, including cognitive tests, interviews with family members, and laboratory tests. Early diagnosis of AD is critical, as it can greatly improve patients' quality of life and in some cases halt or slow the rate of progression. According to a study conducted by Beach et. al. in 2012, there was wide variation in AD diagnosis accuracy, but the overall accuracy was 77% [1]. This is far from perfect, as many AD cases are misdiagnosed, with a low true negative rate. This raises the demand for a computer assisted tool to reinforce physicians' diagnosis.

As AD causes the breakdown and death of neurons, these changes in brain mass can be observed through technology such as magnetic resonance imaging (MRI), computerized tomography (CT), and positron emission tomography (PET). These scans are suitable for computer vision and deep learning algorithms, particularly in image classification. Recent advancements in machine learning, such as the convolutional neural network (CNN), have achieved results in classification that even outperform humans in some cases [2]. Diagnosis of AD using machine learning can serve as a powerful tool for physicians, supplying an additional metric for diagnosis.

Convolutional neural networks typically require large datasets to perform effectively. However, medical data is often scarce and limited in size. This is largely due to the high standards of consistency and organization required for medical data, and the cost and time required for data collection. For example, the ADNI dataset, one of the largest datasets created for Alzheimer's disease neuroimaging, only consisted of 800 individuals. As a result, the lack of data in medical imaging is a prominent obstacle preventing more widespread use of machine learning.

This raises a demand for data augmentation techniques to improve medical machine learning models. One recently introduced technique is the generative adversarial network (GAN) [3], one of the most influential milestones in machine learning. By having two neural networks, a generator and a discriminator, compete against each other, GANs achieve promising results in image generation, super resolution,

and data augmentation, among many other applications. The lack of large amounts of medical data leads to significant potential for the use of GANs for data augmentation. Using a relatively small dataset, GANs can generate similar but original images, as opposed to image modifications used in classical data augmentation. In this study, we investigate the potential for using deep learning in Alzheimer's disease classification by creating a convolutional neural network model. We also test the feasibility of using GANs for data augmentation, specifically using the CycleGAN architecture.

1.2 Literature Review

1.2.1 Convolutional Neural Networks.

A convolutional neural network (CNN) is a type of neural network that specializes in dealing with pattern recognition in images. Input usually consists of a three-dimensional array containing width, height, and the pixel values in the image. A diagram of the architecture of a convolutional neural network is shown in Figure 1.

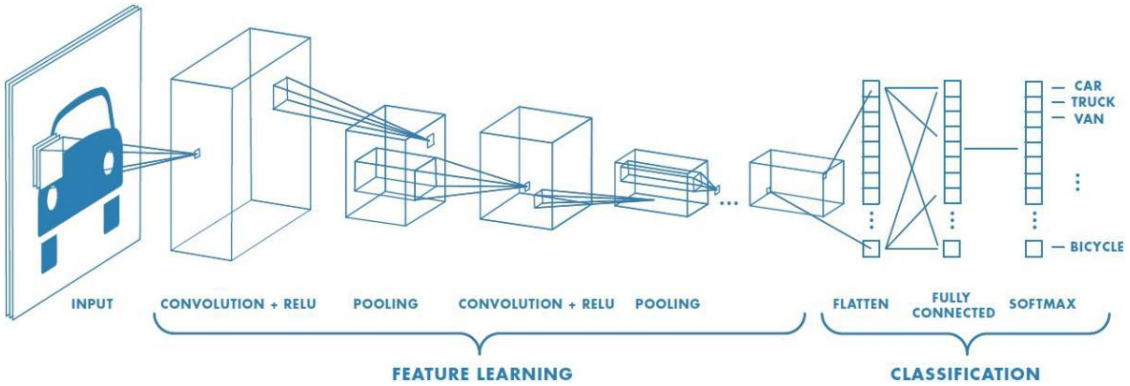


Fig. 1: CNN Architecture. Adapted from [4].

CNNs use layers of convolutions, where a filter or kernel is used to create a feature map, allowing the network to sample context within each frame as neighboring pixels are included. These feature maps allow the model to detect low level features within regions. Pooling layers are also added, which create subsamples of the previous layer. In classification tasks, a standard fully-connected feed forward neural network is commonly applied to the flattened final feature map output by the convolutional layers.

1.2.2 Transfer Learning.

A common method of training image classification networks is through transfer learning, which is the process of using pretrained models to operate on a different task, essentially transferring the knowledge stored in the original network. Common pretrained models include VGG, Inception, or ResNet, all of which have performed well on datasets such as ImageNet [5]. Transfer learning can often allow for quicker training times along with high accuracy, and is also less prone to overfitting.

1.2.3 Generative Adversarial Networks.

Generative adversarial networks were first proposed in 2014 by Ian Goodfellow [3]. The paper suggests the simultaneous training of two adversarial networks in the style of a zero-sum minimax game. The goal of the generator is to create images that trick the discriminator into classifying them as real, and does so by taking in random noise and upsampling it through convolutional layers. In contrast, the goal of the discriminator is to correctly classify what is fake data from the generator and what is real data from the training set. This is also usually done through a convolutional neural network, returning a probability that the image is real.

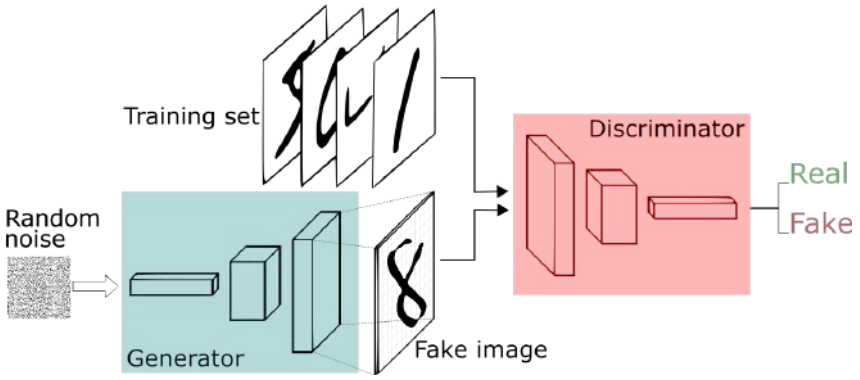


Fig. 2: GAN Architecture. Adapted from [6].

The GAN uses the following loss function, where x represents real samples and z represents generated samples.

$$Loss = E_x[\log(D(x))] + E_z[\log(1 - D(G(z)))]$$

The discriminator attempts to maximize D(x), which is the likelihood of a correct sample being classified correctly. The generator wants D(G(z)), the likelihood of the discriminator classifying its generated image as real, to be as high as possible, and thus wants to minimize 1-D(G(z)). This means that the desired G will be minimized and the desired D will be maximized. During each training iteration, the discriminator is updated through gradient ascent in order to maximize the loss function, and the generator is then updated through gradient descent.

1.2.4 CycleGAN

Zhu et al. [7] proposed a cycle consistent GAN network that allows for unpaired image to image translation.

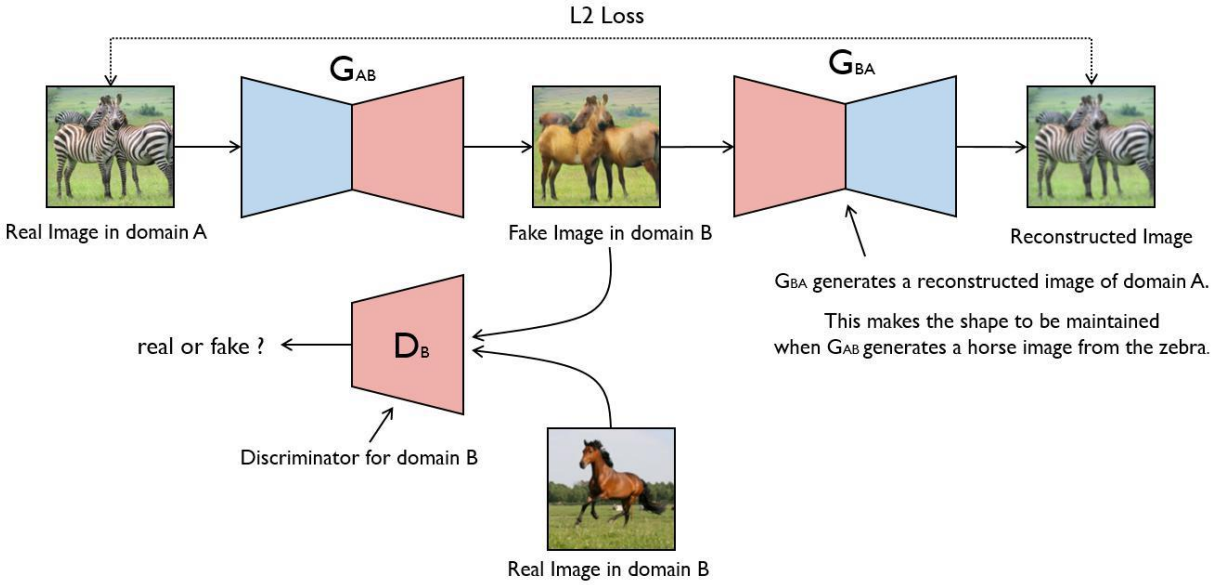


Fig. 3: CycleGAN architecture. Adapted from [8].

Consisting of two generators and two discriminators, CycleGAN allows for discriminatory verification in both directions. Standard GAN loss is used to determine the realisticness of the generated images by comparing the generated image with a real image in the other domain. To ensure that the transformations are cycle consistent, cycle consistency loss is also introduced. When an image that is passed through both generators, the resulting image is compared with the original image, as shown in Figure 4. Using the loss functions together essentially allows the generators to learn a spatial transformation from one class to the other.

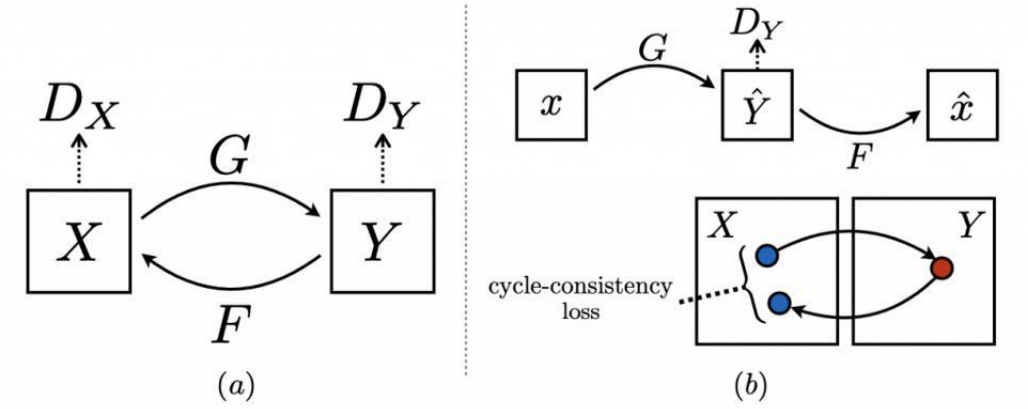


Fig. 4: CycleGAN Loss. Adapted from [9].

The generators G and F learn the spatial mappings from X to Y and vice versa, and are verified by discriminators D_Y and D_X. CycleGAN has been applied to topics such as style transfer and object transfiguration and produces impressive realistic results.

1.2.5 Data Augmentation

As stated before, GANs have great potential in data augmentation. Traditional data warping augmentation consists of techniques such as geometric transformations, filters, and random erasing [10]. This can reduce overfitting and improve accuracy. In contrast, GANs are a method of oversampling, as they are able to extrapolate beyond the training set to generate synthetic data, rather than solely modifying existing data within the training set. This is substantially useful in fields lacking large amounts of data, such as in medicine. One instance of GANs being used in augmentation is a study conducted by Frid-Adar et al. proposing the use of synthetic images in liver lesion classification [11]. GAN generated data was used with a CNN for image classification on a dataset of size 182. Standard DA techniques resulted in 78.6% sensitivity and 88.4% specificity, while the addition of GAN synthesized data yielded 85.7% sensitivity and 92.4%.

1.2.6 Alzheimer's and Machine Learning

As MRI scans are an important aspect of the diagnosis of Alzheimer's, there exists substantial literature regarding machine learning methods. The most common model is a convolutional neural network, as they are ideal for image processing. Farooq et al. [12] used a four way CNN classifier between the following classes: normal cognition (NC), early mild cognitive impairment (MCI), late mild cognitive impairment, and Alzheimer's disease (AD). The model was trained on the Alzheimer's disease Neuroimaging Initiative (ADNI) dataset. The only data augmentation performed was flipping images due to the symmetrical nature of MRIs, and specific slices were selected to exclude those without gray matter information. The proposed network based on GoogLeNet and ResNet outperformed other studies done on the same dataset, with about 98% accuracy. Hosseini-Asl et al. [13] used a deeply adaptive 3D convolutional neural network (DSA 3D-CNN) on the CADDementia dataset, which could then be generalized to the ADNI dataset, which they used for validation. The model achieved a 94.8% accuracy in task specific classification. Glotzman et al.

[14] proposed a network of 2D CNNs, applied to each of three images extracted from each sample in the ADNI dataset. This allowed for the use of two dimensional CNNs on a three dimensional dataset while preserving features. Both MRI scans and PET scans were used, and both two way classification (NC vs AD) and three way classification (NC vs MCI vs AD) were tested. Two way classification with PET-AV 45 scans performed the best, with 83% accuracy.

There have been relatively few applications of generative adversarial networks in classifying AD. A study by Bowles et al. [15] used GANs to model the progression of the disease using MRI data. Using image arithmetic, the model could predict changes in the brain over time and results were comparable to longitudinal examination data. Another study conducted by Pan et al. [16] used GANs to synthesize PET scans from MRI scans in order to fill in gaps in data as many AD patients do not have both due to the high cost of PET scans. A cycle consistent generative adversarial network was used as the first step, which was used in order to learn mappings between the two image domains. The features are then fed into a landmark based multimodal multi-instance learning classifier for diagnosis. Kim et al. [17] conducted a feasibility study on using GANs for slice selective learning on PET scans. Using a BEGAN, they showed that double slices over the posterior cingulate cortex achieves the best performance, and that two slices performed significantly better than using one slice.

It is also important to consider the practical applications of machine learning in AD diagnosis. A survey conducted by R. Bryan [18] on applying machine learning to AD diagnosis notes that machine learning models are biased by the original population, as it is limited to the dataset that it was trained upon. The overall conclusion reached was that although it is unlikely that machine learning models can replace the skills of radiologists, they can serve as useful tools to complement human skills.

The remainder of this paper is organized as follows. Section 2 describes the methodology and pipeline for model construction. Section 3 presents and discusses the results obtained from using different model architectures and displays the significance of using GAN augmentation. Section 4 and 5 report conclusions and future work respectively.

2 Methodology

Figure 5 outlines the model pipeline, consisting of dataset acquisition, preprocessing (including GAN augmentation), and classification using a CNN.

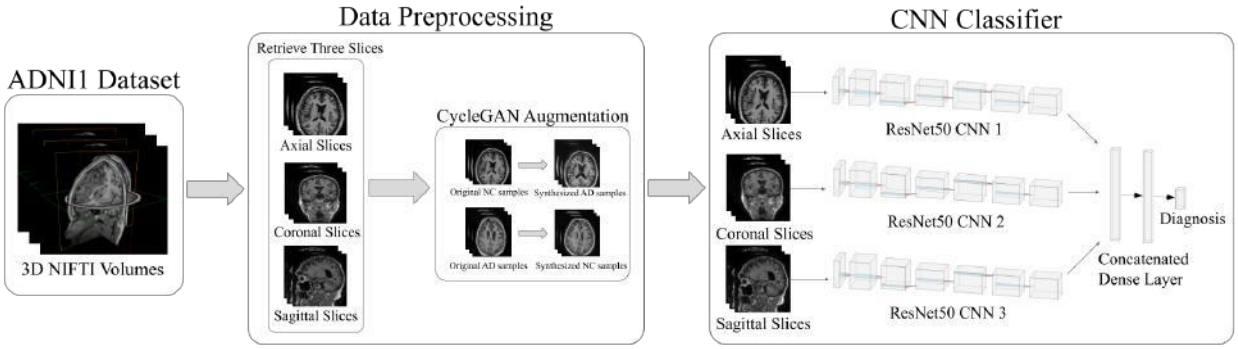


Fig. 5: Overview of the model pipeline.

2.1 Data Acquisition

Data used in the preparation of this article were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). The specific dataset used for training was the ADNI1 standardized MRI dataset, categorized by severity: AD, MCI, and normal cognition (NC). For the

purposes of this study, we trained a network to classify between AD and NC. The dataset contained 705 samples labeled as NC and 476 samples labeled as AD.

2.2 Data Preprocessing

The data was stored in NIFTI files, which were converted into three dimensional numpy arrays using nibabel. A csv file containing information about the scans, patients, and ground truth diagnosis labels was also downloaded from the ADNI database. As the NIFTI image data is three dimensional, slicing was required to prepare samples for training. In order to capture as much information from the original image, we extract three slices, one each from the axial, coronal, and sagittal orientations. The slices are taken by retrieving the midpoint of each axis length, and were resized to 224 x 224.

Two different methods of preprocessing were considered and tested:

1. Skull stripping was applied, which is the process of removing the skull from the MRI images. This isolates the brain tissue, allowing for more consistency among samples. This was done using the Extractor function from the deepbrain library.

2. RAS + ISO transforms and histogram normalization were performed using the TorchIO library [19]. These transforms change the orientation of the MRI image to also improve consistency.

2.3 GAN based Data Augmentation

We constructed the CycleGAN models using the implementation from [7]. The model architecture is shown in Figure 6.

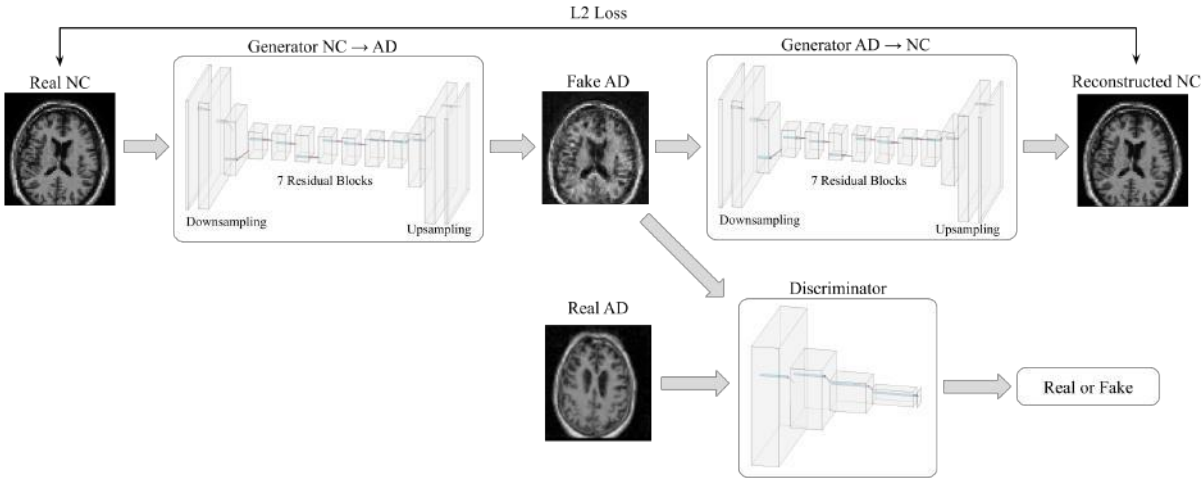


Fig. 6: CycleGAN Architecture.

The model consists of two generators, where one is trained to convert NC samples to AD samples and the other is trained to convert AD samples to NC samples. During training, a real NC image is passed through the first generator, and the resulting fake AD image is compared with an separate real AD image through the discriminator, computing GAN loss. The loss equation, as displayed in equation (1), is the same as the one proposed in the original GAN paper [3]. G represents the generator, D_Y represents the discriminator for AD samples, x represents a real NC image, and y represents a real AD image.

$$\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(x)}[\log(1 - D_Y(G(x)))]$$

This process is repeated, starting with a real AD image, resulting in another GAN loss, represented in equation (2) F represents the generator, D_X represents the discriminator for NC samples, x represents a real NC image, and y represents a real AD image.

$$\mathcal{L}_{GAN}(F, D_X, Y, X) = \mathbb{E}_{x \sim p_{data}(x)}[\log D_X(x)] + \mathbb{E}_{y \sim p_{data}(y)}[\log(1 - D_X(F(y))]$$

The fake images are also passed through the second generator, returning a reconstructed version of the original image. Cycle consistency loss is computed by summing the losses from comparing the original NC image x with its reconstructed image $F(G(x))$ and comparing the original AD image y its reconstructed image $G(F(y))$. This is represented in equation (3), adapted from [7].

$$\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[||F(G(x) - x||_1] + \mathbb{E}_{y \sim p_{data}(y)}[||G(F(y) - y||_1]$$

The overall loss function incorporates both GAN losses and the cycle consistency loss, and is represented in equation (4). λ is a constant representing how much weight is placed on the cycle consistency loss, and $\lambda = 10$ is used as described in the paper [7]. The objective of this loss function is to minimize G and F, which represent loss for the two generators, and maximize D_x and D_y, which are the two discriminators.

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) + \mathcal{L}_{GAN}(F, D_X, Y, X) + \lambda \mathcal{L}_{cyc}(G, F) \tag{4}$$

The generator is based on the ResNet architecture, and consists of downsampling, 9 residual blocks, and upsampling. Instance normalization and reflection padding is used as described in [7]. The tanh activation function is used in its last layer to scale the output image between -1 and 1. The generator architecture is shown in Figure 7.

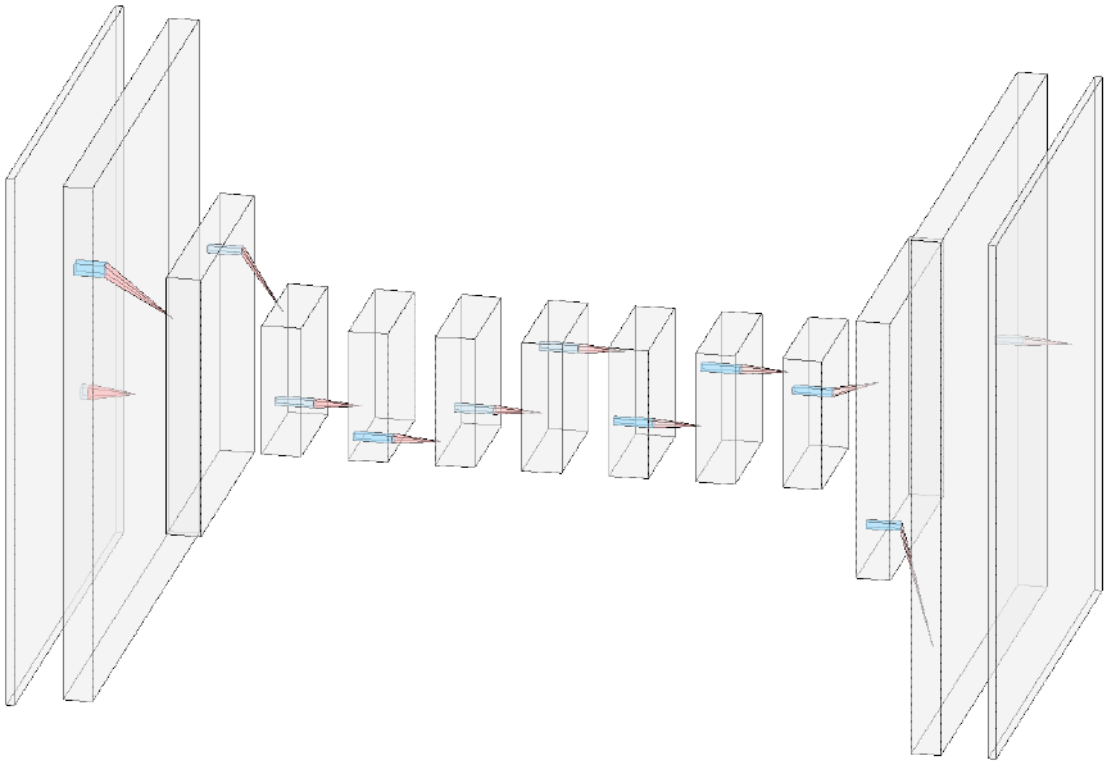


Fig. 7: CycleGAN Generator Architecture.

The discriminator is a CNN using PatchGANs, which classify whether an image is real or fake based on patches. This decreases the amount of parameters needed and is effective for images with high resolutions. The model also uses LeakyReLU as its activation function and utilizes instance normalization. The discriminator architecture is shown in Figure 8.

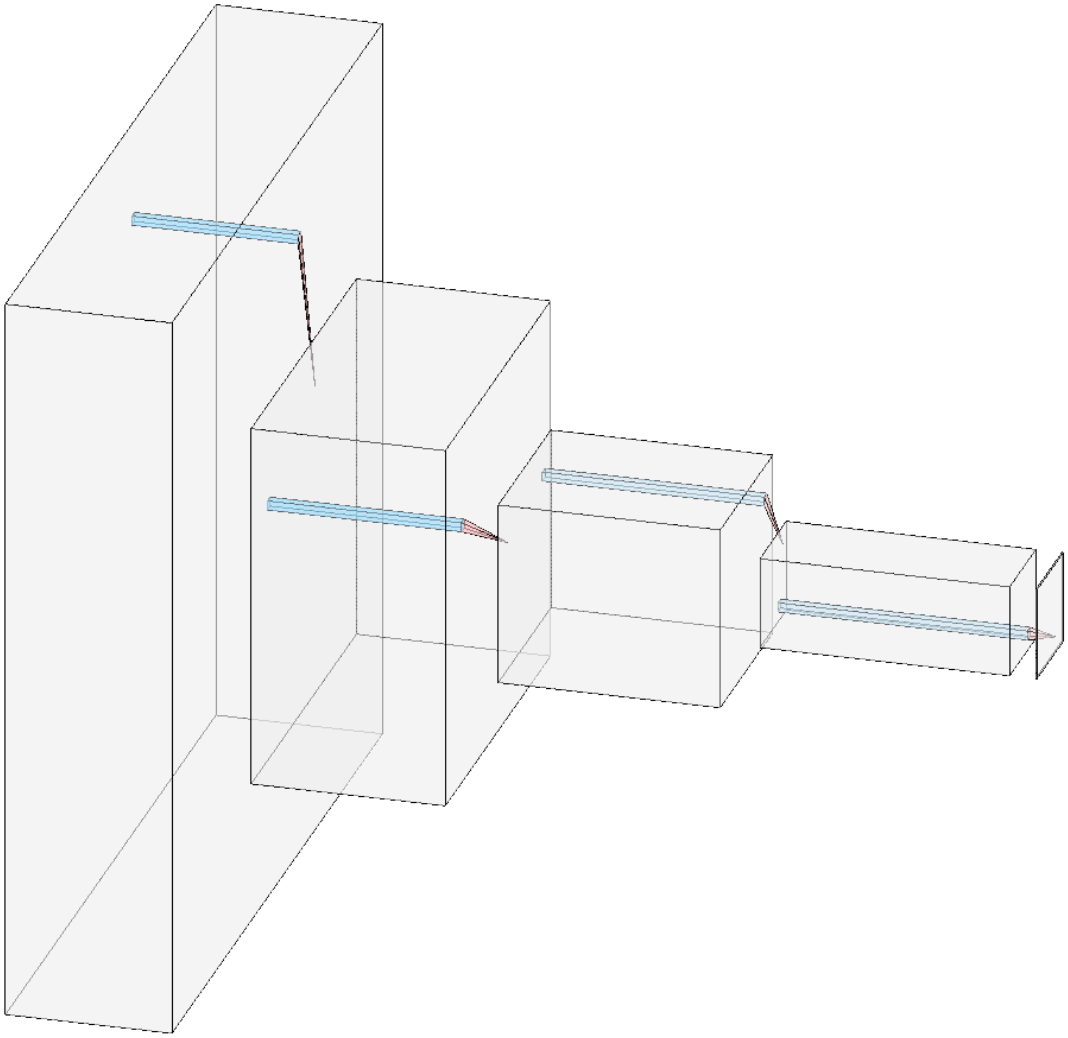


Fig. 8: CycleGAN Discriminator Architecture.

The original dataset was split according to the label and randomly paired. Three individual CycleGAN models were created, where each one was trained on data from a different MRI slice. Each model used the Adam optimizer with a learning rate of 2e-4, and were trained for 100 epochs with a batch size of 1, as specified in the CycleGAN paper.

The trained model was used to generate sufficient samples to create a balanced dataset. An AD version of each NC sample was generated and vice versa. A total of 705 AD samples and 476 NC samples of each orientation were generated, for a total of 1181 images of each class, as shown in Table 1.

	Normal Samples	Alzheimer's Samples	Total
Dataset	705	476	1181
GAN generated	476	705	1181
Total	1181	1181	2362

Tab. 1: Dataset sizes after GAN augmentation.

2.4 Convolutional Neural Network Classifier

We used a transfer learning approach to create the model architecture as it would save training time and is generally effective when datasets are small. We used the ResNet50 convolutional neural network (CNN) as our pretrained model, as shown in Figure 9.

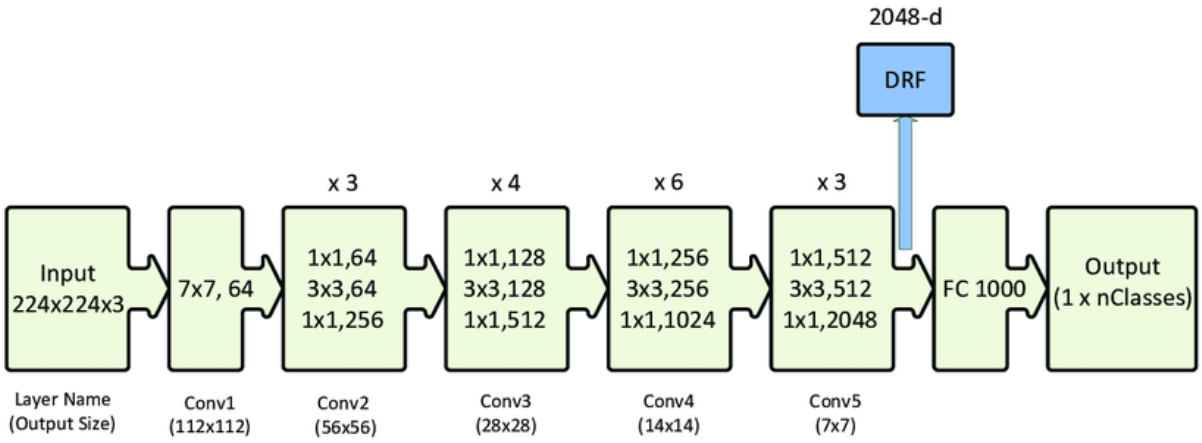


Fig. 9: ResNet50 Architecture. Adapted from [20].

The pretrained ResNet50 architecture takes in 3-channel RGB images while the MRI scans are grayscale. To match the network, the one channel images were transformed to three channels by stacking the tensor three times across dimension 0. The last layer was also modified to become a binary classifier. We used a modified CNN with multiple inputs in order to better encapsulate volumetric data. The model architecture consists of three ResNet50 CNN models, where outputs from each individual CNN are concatenated and passed through fully connected layers, which returns the diagnosis group. The model architecture is shown in Figure 10.

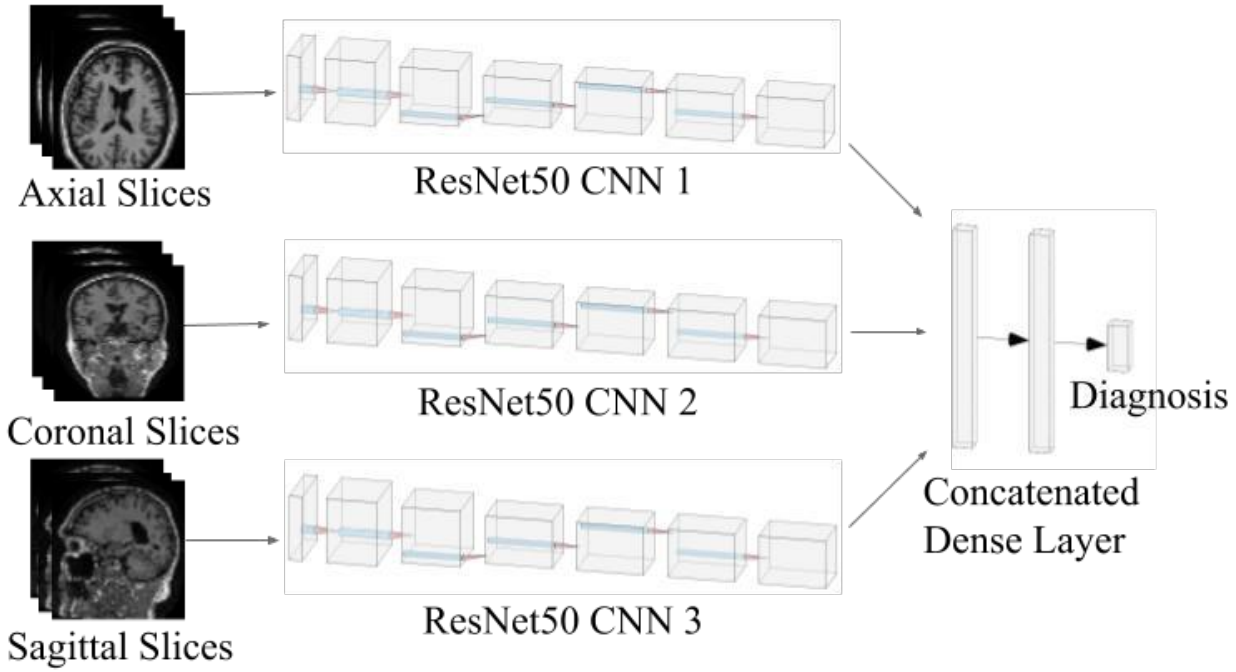


Fig. 10: Proposed Multiple CNN architecture.

The neural network was fine-tuned using the Adam optimizer with a learning rate of 1 e-4 and trained for 50 epochs with a batch size of 32. A training, validation, and testing split of 80%-10%-10% was used.

2.5 Model Implementation

All discussed networks were implemented using Python 3.7 with the PyTorch library. All training was done on a personal computer with an AMD 3700X CPU and an NVIDIA RTX 2070 GPU.

2.6 Model Evaluation

CNN models were evaluated using accuracy, precision, recall, and F1 score, with F1 score being the primary indicator for classification performance. The metrics are detailed in equations (5) - (8), where TP denotes true positives, TN denotes true negatives, FP denotes false positives, and FN denotes false negatives.

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

$$PRE = \frac{TP}{TP + FP} \tag{6}$$

$$REC = \frac{TP}{TP + FN} \tag{7}$$

$$F1 = 2 \times \frac{PRE \times REC}{PRE + REC} \tag{8}$$

3 Results and Discussion

3.1 Comparison of Preprocessing Methods

Two methods of preprocessing were tested, skull stripping and TorchIO transforms. Figure 11 displays the resulting images after applying preprocessing.

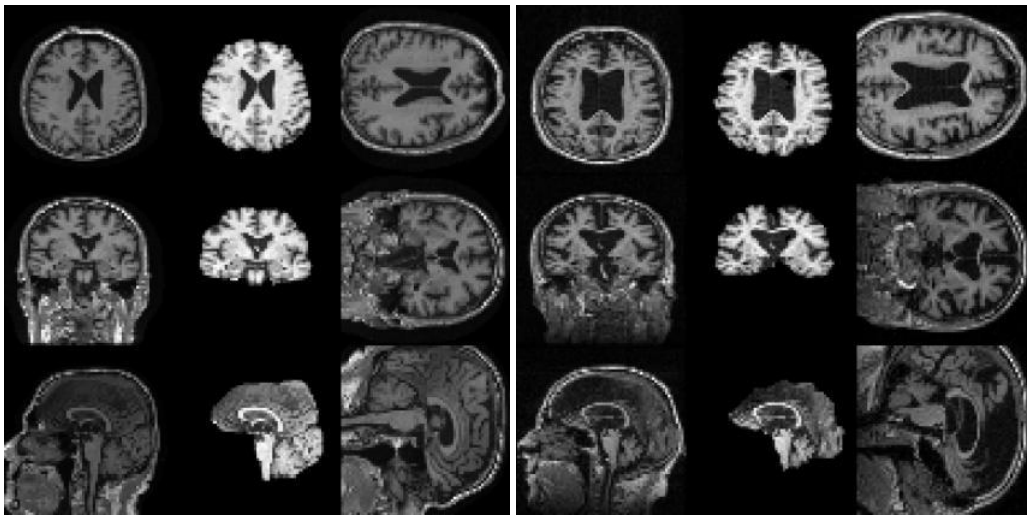


Fig. 11: Normal sample (left) and Alzheimer's sample (right). The original slices are displayed in the first column, the skull stripped samples are displayed in the second, and the TorchIO transformed samples are displayed in the third.

Table 2 compares the results from applying different methods of preprocessing on the three input ResNet50 network as shown in Figure 6. The different transforms were not compatible with each other, so results were obtained separately.

Metric	ResNet50	ResNet50 + Skull Stripping	ResNet50 + TorchIO
Accuracy	0.891	**0.908**	0.907
Precision	**0.932**	0.882	0.897
Recall	0.804	**0.900**	0.854
F1 Score	0.863	**0.891**	0.875

Tab. 2: Comparison of ResNet50 networks with different preprocessing.

The model utilizing TorchIO transforms improved upon the unmodified model, increasing the F1 score from 86.3% to 87.5%. However, the model utilizing skull stripping outperformed both models. This is likely because it improves the consistency among samples in the dataset, which makes it easier for the model to extract important features. The model with skull stripping was kept for the remainder of the study.

3.2 CycleGAN Generation Results

Examples of CycleGAN generated images are shown in Figure 12.

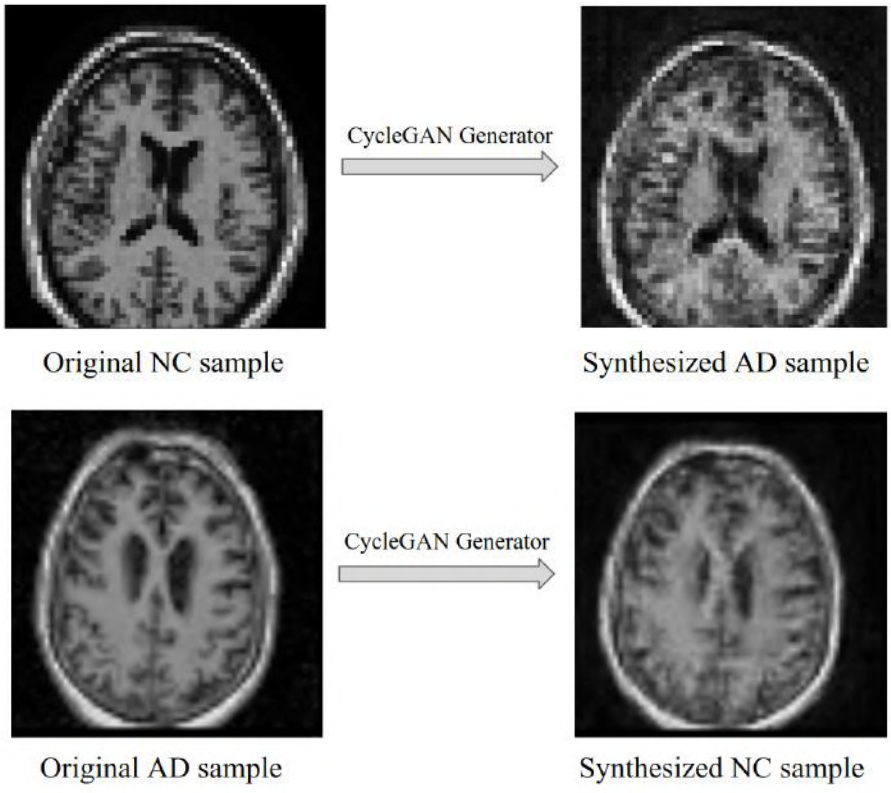


Fig. 12: CycleGAN image synthesis.

By observation, the synthesized Alzheimer's sample displays more dark space throughout the brain when compared to the normal sample that it was transformed from, which is an indication of loss of brain mass and a characteristic of Alzheimer's disease. On the contrary, the synthesized normal sample on the bottom right has much less dark space. While the quality of our synthetic images has not been verified by experts, they exhibit many characteristics of real MRI images.

3.3 Comparison with GAN Augmentation

When using CycleGAN for augmentation, an additional 705 AD samples and 476 NC samples of each orientation were generated, for a total of 1181 images of each class. Table 3 shows that there was a substantial increase in performance to the CNN model when GAN augmentation is applied. The F1 score for the ResNet50 model increased from 0.863 to 0.946, an 9.6% increase. The F1 score for the ResNet50 using skull stripping increased from 0.891 to 0.951, an 6.7% increase.

Metric	ResNet50	ResNet50 + GAN	ResNet50 + SS	ResNet50 + SS + GAN
Accuracy	0.891	**0.954**	0.908	0.949
Precision	0.932	**0.951**	0.882	0.944
Recall	0.804	0.942	0.900	**0.959**
F1 Score	0.863	0.946	0.891	**0.951**

Tab. 3: Comparison of CNN models with GAN augmentation.

These results indicate that the addition of CycleGAN improves CNN classification performance. From this, it is reasonable to infer that the synthesized images had meaningful features that benefited the model.

The increased size and balance among classes in the CycleGAN augmented dataset are also factors that are potentially responsible for the increase in performance. Overall, these results demonstrate the effectiveness of GANs in data augmentation.

4 Conclusions

In this study, we constructed convolutional neural network models utilizing the ResNet50 architecture to diagnose Alzheimer's disease using MRI scans, with variants using one input and three inputs. We also address the problem of size limitations in medical datasets with the use of generative adversarial networks (GANs). Using the ADNI1 dataset, we demonstrated that the addition of GANs can greatly improve deep learning classification accuracy for Alzheimer's disease diagnosis. Specifically, we used CycleGAN to generate images of one class using the other, balancing the dataset and increasing its overall size. Our results show that classification accuracy improved substantially, with F1 scores increasing from 0.863 to 0.946 for the standard model and 0.891 to 0.951 for the model utilizing skull stripping. Due to the lack of large datasets in many medical fields, the results obtained in this study can be generalized to many other fields as well. Overall, with promising results in data augmentation, GANs have potential to significantly improve upon classification tasks across a wide variety of applications.

5 Future Work

5.1 Preprocessing

Upon inspection, the current methods of preprocessing yield some inconsistency between images, particularly in axial scans, due to subjects having slight variation in brain shape. In the future, other methods of preprocessing could be used to improve consistency across each slice taken. Software such as statistical parametric mapping could also be applied to perform preprocessing such as gray matter segmentation and modulation to reduce variation between images.

5.2 Intermediate Stages

The current model serves to classify between MRI scans that either have or don't have Alzheimer's disease. The intermediate stage, mild cognitive impairment (MCI), could also be incorporated into the classifier. Augmented CycleGAN could be implemented to generate images of all three classes.

5.3 MRI and PET Fusion

Diagnosis of Alzheimer's Disease often uses both MRI scans and PET scans. In contrast to MRIs, which provide identification of abnormalities in the brain, PET scans can show areas of low metabolism, allowing for differentiation between Alzheimer's and other types of dementia. Using a fused image with both scans in conjunction would contain more features and will likely improve classification accuracy.

5.4 External features

A full Alzheimer's disease diagnosis requires many elements other than neuroimaging, such as mental status tests and physical exams. Incorporating all of these elements would greatly improve the accuracy of a machine learning system.

References

[1] Thomas G Beach, Sarah E Monsell, Leslie E Phillips, and Walter Kukull. Accuracy of the clinical diagnosis of alzheimer disease at national institute on aging alzheimer disease centers, 2005–2010. *Journal of neuropathology and experimental neurology*, 71(4):266–273, 2012.

[2] Christopher Bowles, Roger Gunn, Alexander Hammers, and Daniel Rueckert. Modelling the progression of alzheimer's disease in mri using generative adversarial networks. In *Medical Imaging 2018: Image Processing*, volume 10574, page 105741K. International Society for Optics and Photonics, 2018.

[3] R Nick Bryan. Machine learning applied to alzheimer disease, 2016.

[4] Antoine Buetti-Dinh, Vanni Galli, Sören Bellenberg, Olga Ilie, Malte Herold, Stephan Christel, Mariia Boretska, Igor V Pivkin, Paul Wilmes, Wolfgang Sand, et al. Deep neural networks outperform human expert's capacity in characterizing bioleaching bacterial biofilm composition. *Biotechnology Reports*, 22:e00321, 2019.

[5] Ammarah Farooq, SyedMuhammad Anwar, Muhammad Awais, and Saad Rehman. A deep cnn based multi-class classification of alzheimer's disease using mri. In *2017 IEEE International Conference on Imaging systems and techniques (IST)*, pages 1–6. IEEE, 2017.

[6] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. *Neurocomputing*, 321:321–331, 2018.

[7] Tanya Glozman and Orly Liba. Hidden cues: Deep learning for alzheimer's disease classification cs331b project final report, 2016.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[9] Ehsan Hosseini-Asl, Robert Keynton, and Ayman El-Baz. Alzheimer's disease diagnostics by adaptation of 3d convolutional network. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 126–130. IEEE, 2016.

[10] Han Woong Kim, Ha Eun Lee, Sangwon Lee, Kyeong Taek Oh, Mijin Yun, and Sun Kook Yoo. Slice-selective learning for alzheimer's disease classification using a generative adversarial network: a feasibility study of external validation. *European Journal of Nuclear Medicine and Molecular Imaging*, pages 1–10, 2020.

[11] Yongsheng Pan, Mingxia Liu, Chunfeng Lian, Tao Zhou, Yong Xia, and Dinggang Shen. Synthesizing missing pet from mri with cycle-consistent generative adversarial networks for alzheimer's disease diagnosis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 455–463. Springer, 2018.

[12] Fernando Pérez-García, Rachel Sparks, and Sebastien Ourselin. Torchio: a python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning. *arXiv preprint arXiv:2003.04696*, 2020.

[13] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[14] Rajat Vikram Singh. Imagenet winning cnn architectures–a review.

[15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

A Supplemental Figures

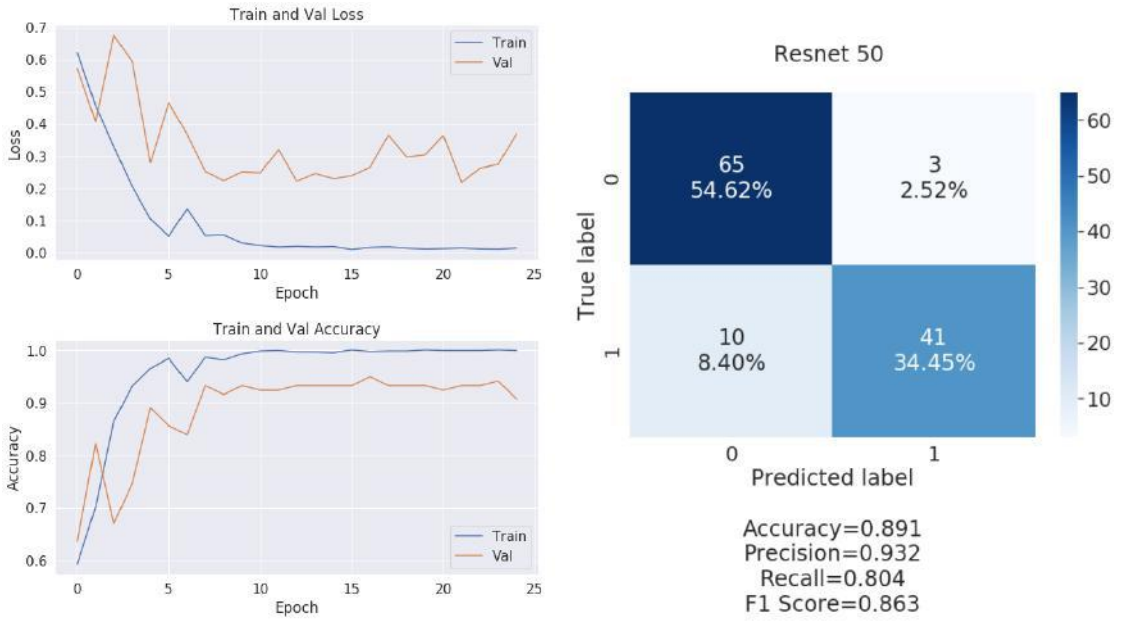


Fig. 13: Results for ResNet50.

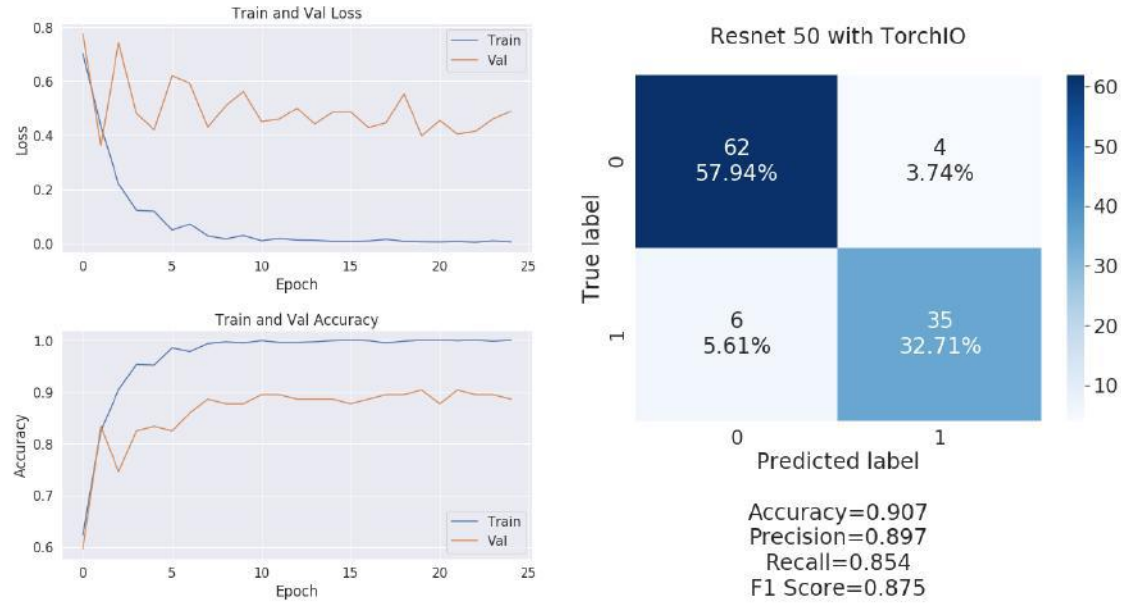


Fig. 14: Results for ResNet50 with TorchIO preprocessing.

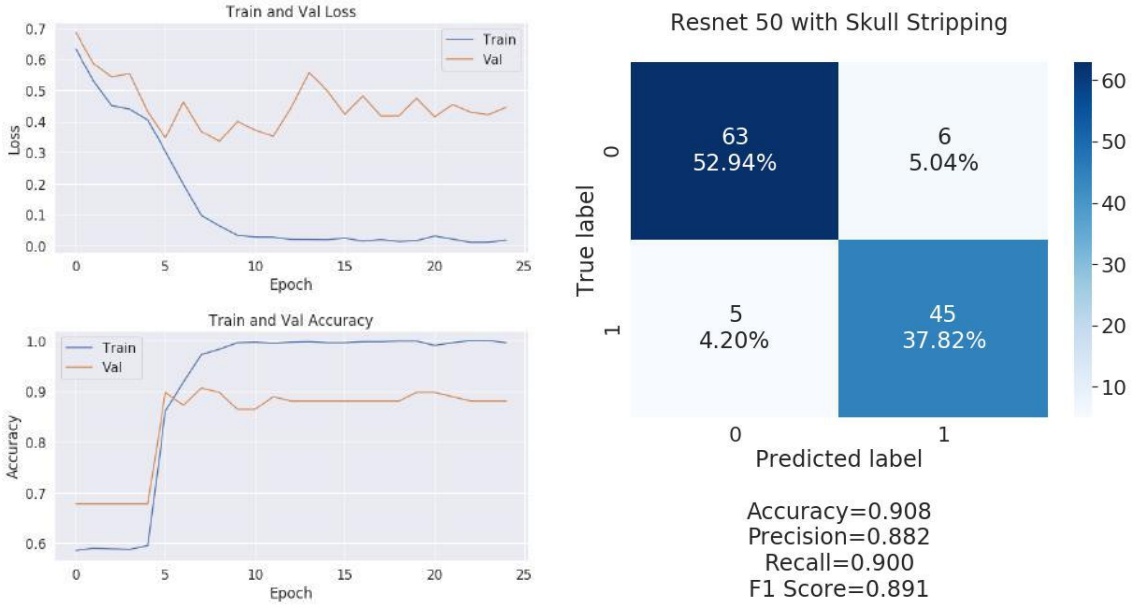


Fig. 15: Results for ResNet50 with skull stripping preprocessing.

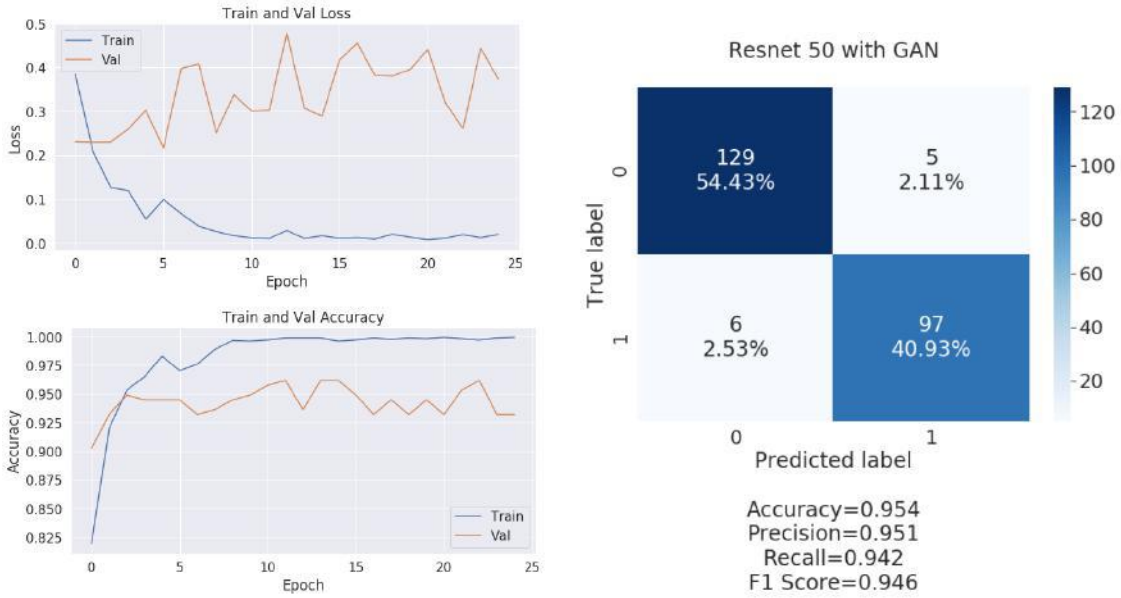


Fig. 16: Results for ResNet50 with GAN data augmentation.

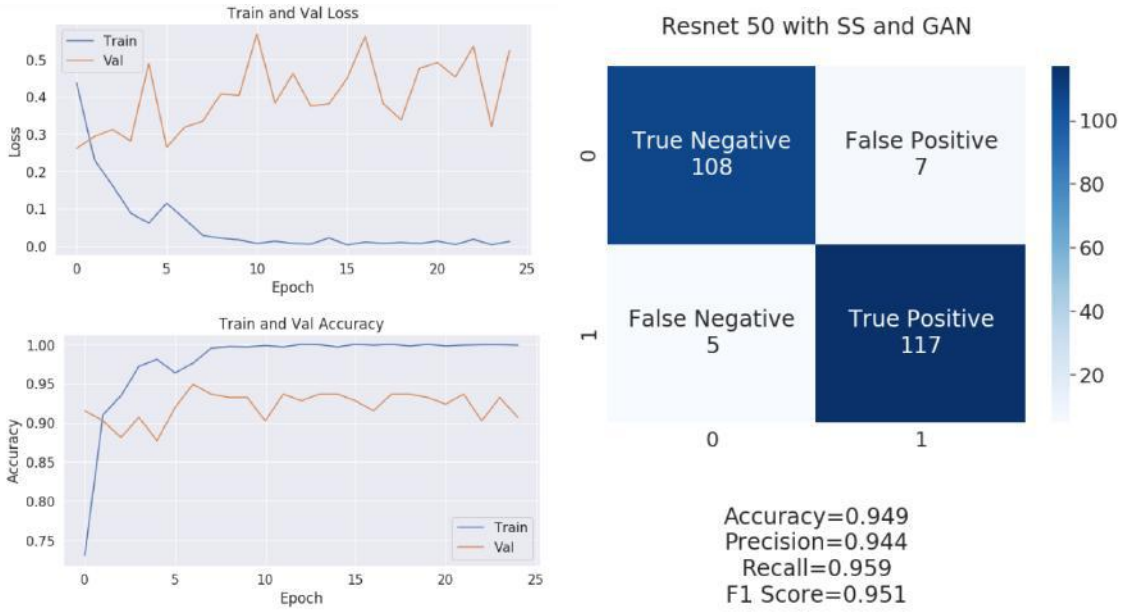


Fig. 17: Results for ResNet50 with skull stripping preprocessing and GAN data augmentation.

Hydra: An Efficient, Provably Secure Website Fingerprinting Defense Based on Traffic Splitting

Alexander Di

Abstract

As Internet privacy becomes an increasing concern in the Digital Age, millions of people have turned to Tor, the world's largest anonymous communication network. However, Tor suffers from website fingerprinting (WF), a type of traffic analysis attack in which an adversary uses a machine learning classifier on a user's web traffic to determine his or her web activity. Although previous studies have proposed various defenses against WF, these defenses either incur high overheads or provide no formal guarantee of privacy. We propose Hydra, a lightweight WF defense that splits the user's traffic among several different Tor nodes. Our results show that Hydra is able to reduce an attacker's accuracy from 92.2% to 7.5% while only incurring 68.2% bandwidth overhead and 59.8% time overhead. In comparison to the existing state-of-the-art defenses, Hydra reduces the attacker's accuracy by over 65% while incurring at least 15% less bandwidth and time overheads.

1 Introduction

In today's Digital Age, the Internet provides users with a platform to freely express their views and disseminate truthful information. However, this is not possible in highly censored countries. Instead, many users ranging from journalists to activists rely on the Tor network, which is the world's largest anonymous communication network with over 2.5 million daily users. [13][12]. The Tor network allows users to evade government censorship and access critical resources unavailable in their country. Tor works by encrypting the web traffic and routing it through various Tor nodes, whereby each Tor node only knows its preceding node and succeeding node, but no other nodes in the circuit, thereby providing privacy for its users.

However, Tor is susceptible to traffic analysis attacks. One such attack that has been extensively studied is website fingerprinting (WF) because it is an easy attack to perform. In a WF attack, an adversary monitors the traffic passing between the victim and the Tor entry node, as shown in Figure 1. The attacker is considered passive in that he does not insert, drop, or delay data packets in the user's traffic; he simply observes.

Although Tor encrypts its data packets and pads each packet to a fixed size of 512 bytes, the adversary can look at the timestamp and direction of the packets to identify patterns in the traffic. During each website visit, the user generates a sequence of data packets with their respective timestamps, which is defined as a *trace*. Typically, the attacker first collects traffic traces of various websites that he seeks to identify. He then extracts features from each of these traffic traces, such as the total number of packets or the average interpacket timing of the trace. Finally, the attacker trains a machine learning classifier on these features and attempts to classify the victim's traffic trace, thereby revealing the website that the victim visited.

Various defenses against WF attacks have been proposed; these defenses seek to modify the traffic trace by delaying certain packets or inserting dummy packets. However, many of these defenses are ineffective as state-of-the-art attacks that employ deep learning have been able to reach over 90% accuracy despite these countermeasures [15]. These attacks leverage the long uninterrupted packet sequences that the defenses fail to break up [10][1]. Other defenses, such as CS-BuFLO and Tamaraw, do significantly decrease attacker accuracy, but at the expense of significant overheads in time and bandwidth [2][3].

In this paper, we propose a highly tunable, provably secure WF defense that is based on traffic splitting and incurs reasonable overheads.

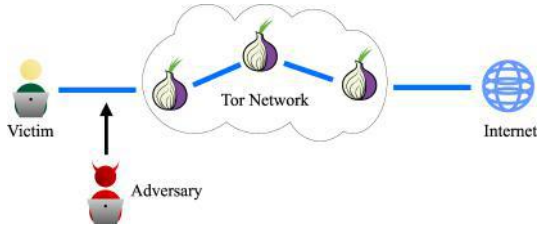


Fig. 1: A local adversary, such as an ISP or a workspace administrator, monitors the traffic between the victim and the Tor entry node. Through the traffic, the attacker can infer the website that the victim is visiting.

2 Literature Review

Various website fingerprinting attacks and defenses have been proposed by researchers over the last few years.

2.1 Existing Attacks

Researchers have proposed various attack models for WF. They evaluate their attacks in two scenarios:
- *Closed world setting*. The victim can only visit a certain amount of web pages. In this setting, the attacker can train on all possible websites that the victim can visit.
- *Open world setting*. The victim can visit any website. The attacker has to also determine whether the victim is visiting a *monitored site*, a site that the attacker trains on, or an *unmonitored site*, a site that the attacker does not train on.

Currently, there are the following major attack models:
• **k-NN** [16]: Wang et al. utilize a k-nearest neighbors classifier to match traffic traces with known websites. k-NN is able to achieve above 80% accuracy on defenses involving traffic morphing.
• **CUMUL** [11]: The attack devised by Panchenko et al. utilizes a support vector machine that takes into account the total number of incoming and outgoing packets, as well as their respective sums. CUMUL is effective in that it does not require training on many instances per website.
• **k-fingerprinting (k-FP)** [6]: Proposed by Hayes et al., this attack uses random forests to classify traffic traces. It is highly effective in an open-world scenario. k-FP allows for classification of other features particular to the relationship between incoming and outgoing packets such as transmission rate per direction [4].
• **Deep Fingerprinting (DF)** [15]: Sirinam et al. proposed an attack that leverages convolutional neural networks. This attack is very effective against defenses such as WTF-PAD (90.7% accuracy) [8] and reaches just below the maximum attacker accuracy of Walkie-Talkie (49.7% accuracy) [17].

2.2 Existing Defenses

Many defenses have been proposed against WF. Typically, defenses are divided into two categories: provably and not provably secure. *Provably secure* defenses have a computable upper bound on the attacker's accuracy. No attacker can achieve a higher accuracy than the upper bound. Such defenses will remain resilient against future attacks.

Defenses also incur overheads, specifically bandwidth and time. *Bandwidth overhead* measures the amount of dummy packets added to a given trace, calculated as the total number of dummy packets divided by the total number of real data packets. *Time overhead* measures the amount of latency or delay in the page load. This is calculated as the extra time taken for the defended page to load divided by the time of an undefended page load.

• **Walkie-Talkie** [17]: This defense pairs sensitive pages with non-sensitive pages so that the morphed traffic trace of both websites are identical. Thus, even if the attacker recognizes the trace, he cannot determine

which website produced the trace. Although WT has a maximum attacker accuracy of 50%, it suffers from top-N prediction since the attacker knows that the trace was produced by one of two websites [7]. In addition, WT requires a database of all the pairings between many different websites.
• **WTF-PAD** [8]: WTF-PAD builds upon Adaptive Padding [14] by sending dummy packets when noticeable gaps appear in the traffic trace. This defense eliminates gaps as a distinguishable feature for an attacker to train on. Although WTF-PAD is a lightweight defense, meaning that it incurs low overheads, it is ineffective against deep learning attacks [15], neither is it provably secure.
• **BuFLO and CS-BuFLO** [5][2]: BuFLO sends packets at a fixed rate such that no distinguishable features are present other than the trace length [5]. If no packets need to be sent, BuFLO sends dummy packets for t seconds either until more data packets enter the queue or until t seconds has passed. Cai et al. built upon BuFLO to create CS-BuFLO, which varies its transmission rate according to the network congestion, thereby reducing the overheads [2].
• **Tamaraw** [3]: Tamaraw, like BuFLO [5], sends packets at a constant rate, hiding the inter-packet timing. However, Tamaraw is able to send outgoing packets at a different rate than incoming packets, further reducing the overheads. In addition, Tamaraw pads each trace to a multiple of a number L to hide the trace length. Tamaraw is especially effective against state-of-the-art attacks [15][6][3]. However, it still suffers from high bandwidth overhead due to the amount of dummy packets injected into the traffic to maintain a constant rate of packet flow, as high as 128% bandwidth overhead.
• **Traffic Splitting** [4]: Instead of padding the traffic, De la Cadena et al. evaluate the several strategies for splitting the traffic among several different Tor entry nodes. They find that using a weighted random splitting strategy greatly reduces the attacker's accuracy (30.27% against DF and 34.09% against k-FP) when traffic is split among three or more different paths.
• **HyWF** [7]: Henri et al. use a different approach by splitting the traffic among two different WiFi networks. This method removes the need to modify Tor nodes but requires the user to have access to two different internet services. HyWF distributes the packets in groups rather than individually and randomly changes the splitting probability, which reduces the attacker's accuracy to around 15.3% and 36.3% against k-NN and k-FP, respectively. Against Deep Fingerprinting, HyWF yields a true positive rate of 48.6%.

3 Purpose

Previous works on WF defenses often incur high overheads or guarantee no upper bound on any attacker's accuracy. The goal of this work is three-fold:

1. Design and implement a provably secure website fingerprinting defense that utilizes traffic splitting (dividing the traffic among multiple Tor entry nodes) while incurring low overheads.
2. Evaluate the effectiveness and efficiency of our defense on existing state-of-the-art attacks.
3. Compare our defense with existing state-of-the-art defenses.

4 Hydra

At a high level, our defense morphs the user's traffic into a constant stream and splits the traffic among several different Tor entry nodes, generating several sub-traces (in this paper, we refer to each individual Tor entry node as a *network* or *path*). In addition, Hydra produces a variety of distinct sub-trace lengths for each website, which allows for a greater chance for the attacker to misclassify traffic traces when he looks for patterns in the trace lengths.

4.1 Hydra Design

Our defense consists of three parts: 1. traffic morphing, 2. traffic splitting, and 3. end padding.

1. *Traffic morphing*. Similar to Tamaraw [3], Hydra utilizes a constant stream of packets. Our defense delays packets and adds dummy packets in order to maintain a pattern of one outgoing packet followed by four incoming packets. The interpacket timing is set to 0.005 seconds.

2. *Traffic splitting*. Hydra utilizes m networks to route the user's traffic. Each network i has its own threshold T_i such that $T_{i-1} < T_i < T_{i+1}$ for $1 \leq i \leq m$, which dictates when to utilize each network. Hydra begins by sending packets through network 1. Since our defense delays packets in order to maintain a strict pattern, packets will accrue in both the server's queue and client's queue. Once the number of packets in either queue reaches T_i, our defense opens up network i and begins sending packets through network i in addition to networks $1, 2, ..., i-1$. Once the number of packets in both queues drops below T_i, network i closes off. After a network is used, it cannot be reopened again. In total, Hydra generates k sub-traces for $1 \leq k \leq m$. For each website visit, the order in which networks are opened is randomized so that the attacker cannot deduce which path he is monitoring.
3. *End Padding*. The traffic morphing part of our defense eliminates distinguishable features such as inter-packet timing and packet ordering. However, the defended trace lengths of a video-streaming service is vastly different than those of a simple search engine. Hydra implements an exponential padding scheme to hide the trace length. Each sub-trace is padded to $\{10\lfloor b \rfloor, 10\lfloor b^2 \rfloor, 10\lfloor b^3 \rfloor, 10\lfloor b^4 \rfloor, ...\}$ packets for $b > 1$. We multiply each power of b by 10 so that our defense, which implements a five-packet pattern, is able to terminate.

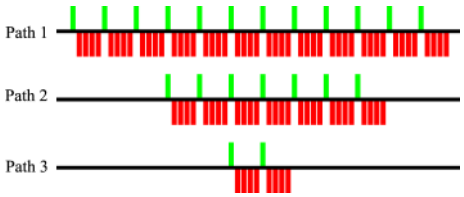


Fig. 2: Hydra opens up new networks as packets accumulate in the queue, which produces sub-traces of varying lengths. In this scenario, an adversary monitoring path 3 does not know whether he is viewing the trace of a small website or that of a large website.

Hydra differs from previous traffic splitting defenses [4][7] in that for every website visit, our defense generates sub-traces of drastically different lengths, further hiding the original trace length (see Figure 2). Thus, when the attacker sees one of the sub-traces, he has no way of determining whether the trace was generated by a large website or a small website.

5 Methods

5.1 Dataset

We used the dataset provided in DynaFlow containing the traffic traces of 100 monitored sites with 100 instances each, as well as 9000 unmonitored sites [9]. However, some traces had unreasonably low packet numbers compared to the total time of the traffic trace. The first quartile of the ratios of packets to total trace time in the original dataset was 140 packets per second. Therefore, we decided to remove all traces that had a packet-time ratio of less than 125 packets per second. The remaining dataset has 91 monitored sites with 50 instances each and 6418 unmonitored sites.

5.2 Implementation

Implementing our defense on the Tor browser would be difficult because it would require modifying the Tor source code. Thus, we wrote Python code to simulate Hydra[1]. With our Python simulation, we produced defended traffic traces from the original traces provided in the dataset. Our Python simulation of Hydra adds delays to certain packets, inserts dummy packets into the traffic, and splits the traffic onto various paths.

Although Hydra creates several different sub-traces from each individual traffic trace, we randomly pick and keep only one of those sub-traces. We make a reasonable assumption that the attacker monitors only one path so that each time the victim visits a website, the attacker will only see one of those sub-traces.

[1] https://github.com/alexdi0421/Hydra

6 Evaluation

We evaluated our defense by running two configurations of Hydra on our modified dataset: (1) For medium security with reasonable overheads, we used three networks and padded to powers of 1.5. (2) For higher security at a greater cost with both overheads and network setup, we used four paths and padded to powers of 2. We then ran k-FP, using the code provided in the paper [6], on the defended dataset generated by Hydra. Finally, we computed an upper bound on any attacker's accuracy.

We did not run DF on Hydra because it uses a large convolutional neural network and was computationally infeasible to run on our machine [15]. Instead, we used k-FP, a state-of-the-art non-deep learning attack [6].

6.1 Closed World Results

We evaluated two configurations of Hydra. The results are summarized in Table 1.

Tab. 1: The accuracy of k-FP on two configurations of Hydra and the original undefended dataset.

Num. Paths	Thresholds	Base	Bandwidth Overhead	Time Overhead	k-FP Accuracy
3	0, 300, 800	1.5	68.2%	59.8%	7.5%
4	0, 200, 400, 800	2	99.7%	43.2%	4.9%
Undefended Traces			N/A	N/A	92.2%

Hydra significantly reduces k-FP's accuracy. In the first configuration with three paths, Hydra is able to significantly reduce the accuracy from 92.2% to 7.5% while only incurring 68.2% bandwidth overhead and 59.8% time overhead. With more paths and more padding, the second configuration further lowers k-FP's accuracy to 4.9% with 99.7% bandwidth overhead and 43.2% time overhead.

6.2 Open World Results

The same configurations of Hydra also significantly decrease the efficacy of k-FP in the open-world (See Figure 3). When no defenses are applied, k-FP easily achieves a 72.6% true positive rate (TPR) with only a 0.8% false positive rate (FPR). When the first configuration of Hydra is applied, k-FP achieves a 3.8% TPR with a 94.8% FPR. As shown in Figure 3, reducing the FPR to 26.4% reduces the TPR to 1.2%. Our second configuration is even stronger. k-FP achieves a 2.5% TPR with a 95.5% FPR, or equivalently, a 0.8% TPR with a 25.9% FPR.

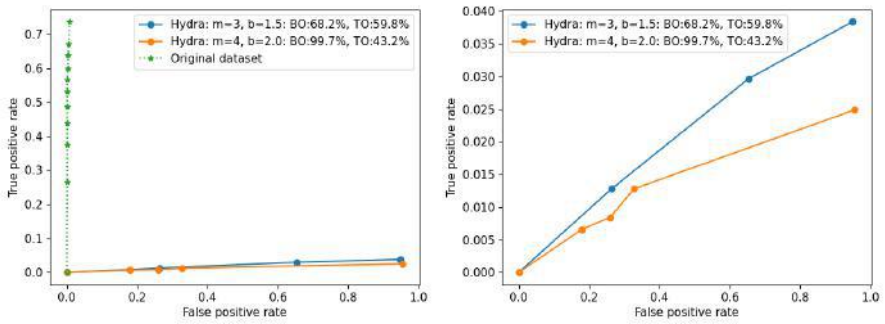


Fig. 3: True positive rate vs. false positive rate of the two configurations of Hydra tested against k-FP. The graph on the left includes the original dataset whereas the graph on the right compares only the two configurations of Hydra. Bandwidth overhead (BO) and time overhead (TO) are labeled in the legend.

6.3 Maximum Attacker Accuracy

Our defense is provably secure because the only distinguishing feature in defended traces is the trace length; other features such as interpacket timing and packet order remain constant for all defended traces. Thus, we can compute an upper bound on any attacker's accuracy. This is the highest accuracy any attacker can achieve assuming that the attacker has perfect knowledge of the nature of the traces generated by each website, i.e., the trace lengths. The MAA is computed as follows:

$$\sum_{i=1} P(l_i) \cdot max\{P(w_1 \mid l_i), P(w_2 \mid l_i), ..., P(w_n \mid l_i)\},$$

where $P(l_i)$ is the probability that a given trace length is l_i and $P(w_k \mid l_i)$ is the probability that website w_k generates a trace with length l_i. $P(w_k \mid l_i)$ is computed as the number of occurrences of l_i in website w_k divided by the total number of occurrences of l_i.

In essence, given a trace of length l_i, the optimal attacker always picks the website that generates the most traces of length l_i. The probabilities of success for each distinct trace length are then summed up to yield an upper bound on any attacker's accuracy (MAA) when Hydra is applied.

6.4 Maximum Attacker Accuracy of Hydra

In the closed world setting, we find that the MAA of Hydra remains below 10% for both configurations (see Table 2). With the same overheads as before, the medium security configuration guarantees a MAA of 8.5% while the higher security configuration promises a 5.6% upper bound on any attacker's accuracy.

Tab. 2: The MAA of both configurations of Hydra.

Num. Paths	Thresholds	Base	Bandwidth Overhead	Time Overhead	MAA
3	0, 300, 800	1.5	68.2%	59.8%	8.5%
4	0, 200, 400, 800	2	99.7%	43.2%	5.6%

In the open world, when Hydra is applied, the optimal attacker still achieves a very low TPR compared to high FPR (see Figure 4). At a 99.5% FPR, the medium configuration of Hydra guarantees a TPR no greater than 3.9%. Similarly, the second configuration of Hydra reduces the maximum TPR to 2.5% with a 99.9% TPR.

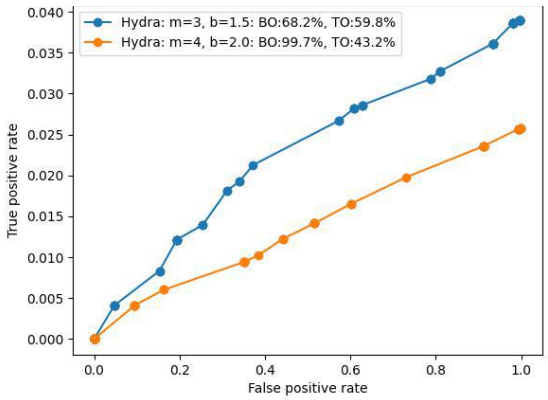


Fig. 4: The attacker's optimal TPRs graphed against their respective FPRs when Hydra is applied.

6.5 Comparison with Other Defenses

We compare our defense with HyWF, the traffic splitting defense proposed by De la Cadena et al. (in this paper, we will refer to this defense as TrafficSplit), as well as Tamaraw, a more heavyweight defense

that is provably secure [3][4][7].

We used our modified version of the dataset provided in DynaFlow [9], which contains 91 monitored sites, each with 50 instances, and 6418 unmonitored sites. Due to compatibility issues, we rewrote the codes for HyWF, TrafficSplit, and Tamaraw [3][4][7]. To run k-FP, we used the original code provided in the paper [6].

6.5.1 Closed World

We use k-FP to evaluate the defenses in addition to the MAA when applicable. The results are summarized in Table 3.

Both TrafficSplit and HyWF do not delay packets, nor do they add dummy packets [4][7]. Thus, both defenses incur no overheads. k-FP is able to achieve a 43.5% TPR against HyWF and 38.6% TPR against TrafficSplit. However, we cannot evaluate an upper bound on the attacker's accuracy because both TrafficSplit and HyWF are not provably secure defenses. Although Hydra incurs moderate overheads, k-FP only achieves a 7.5% and 4.9% accuracy for both the medium security and high security configurations of Hydra, respectively. At the same time, both configurations of Hydra guarantee that any attacker cannot achieve higher than a 8.5% and 5.6% accuracy, respectively.

Tab. 3: The medium and high security configurations of Hydra compared with HyWF, TrafficSplit, and Tamaraw.

Defense	Bandwidth Overhead	Time Overhead	k-FP Accuracy	MAA
Undefended trace	N/A	N/A	92.2%	N/A
HyWF	0%	0%	43.5%	N/A
TrafficSplit	0%	0%	38.6%	N/A
Tamaraw: $L = 500$	52.9%	102.7%	20.1%	21.6%
Tamaraw: $L = 1000$	65.4%	102.7%	13.9%	14.6%
Hydra: $m = 3$, $b = 1.5$	**68.2%**	**59.8%**	**7.5%**	**8.5%**
Hydra: $m = 4$, $b = 2$	**99.7%**	**43.2%**	**4.9%**	**5.6%**

Compared to Tamaraw, our defense yields lower attacker accuracies as well as lower overheads. We adjust Tamaraw's parameters so that it yields low overheads on our dataset. Both configurations of Tamaraw have outgoing and incoming interpacket timings of 0.005 seconds and 0.025 seconds, respectively, and a padding parameter L. The MAA of both configurations of Hydra (8.5% and 5.6%) is lower than that of Tamaraw (21.6% and 14.6%), as well as k-FP's accuracy (7.5% and 4.9% compared to 20.1% and 13.9%). In fact, the aggregate overheads (the sum of bandwidth overhead and time overhead) of both configurations of Hydra, 128.0% and 142.9%, are lower than those of Tamaraw, 155.6% and 168.1%, respectively.

6.5.2 Open World

Both configurations of Hydra outperform TrafficSplit, HyWF, and Tamaraw (see Figure 5). At a 0.0% FPR, TrafficSplit and HyWF only reduce k-FP's TPR to 3.1% and 5.6%, respectively, whereas both configurations of Hydra drop the TPR to 0%.

Hydra outperforms Tamaraw in an open world setting. At a FPR of 98.8%, the higher security configuration of Tamaraw ($L = 1000$) only promises a TPR of 6.1% whereas our medium configuration of Hydra drops the TPR to 2.5% at a FPR of 99.9%. Similarly, at a 38.1% FPR, Tamaraw only guarantees a 3.1% TPR compared to the 1.0% TPR that the higher security configuration of Hydra guarantees. As stated before, the aggregate overheads of Hydra are less than those of Tamaraw.

6.6 Hydra's Parameters

Hydra offers high tunability; it can be configured for a wide range of MAAs, bandwidth overheads, and time overheads. Here, we evaluate the effects of changing Hydra's parameters.

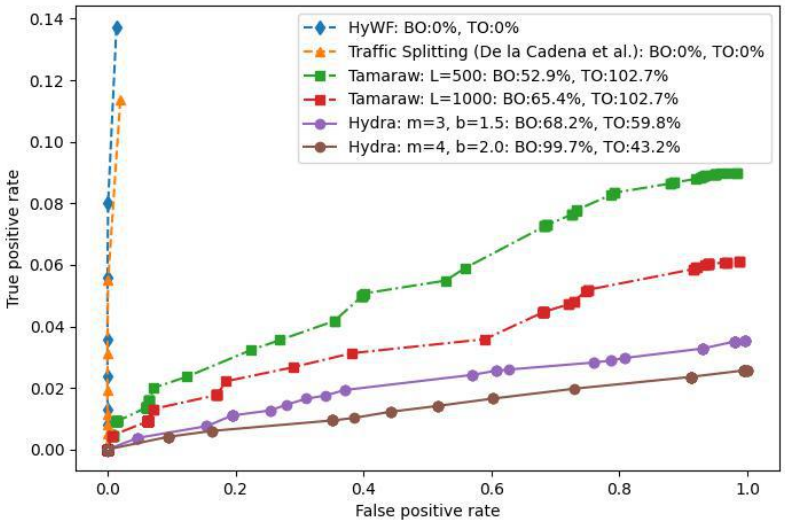


Fig. 5: The TPR graphed against the FPR for Hydra, TrafficSplit, HyWF, and Tamaraw. We used k-FP to evaluate the TPR and FPR of TrafficSplit and HyWF. We graphed the optimal attacker's TPR and FPR when Hydra and Tamaraw were applied.

6.6.1 Number of networks

We run Hydra with a base b of 1.5 on different numbers of networks. As the number of paths increases from two to six, the MAA decreases from 10.1% to 6.9% (see Figure 6). In addition, while the bandwidth overhead remains at 69.2 $\pm$ 1.1%, the time overhead drops from 84.4% with two paths to 37.5% with six paths. However, we noticed that the MAA of six paths is only 0.1% less than that of five paths (6.9% compared to 7.0%). In fact, we suggest that the cost of setting up more than five networks outweighs the guaranteed privacy of five or more networks.

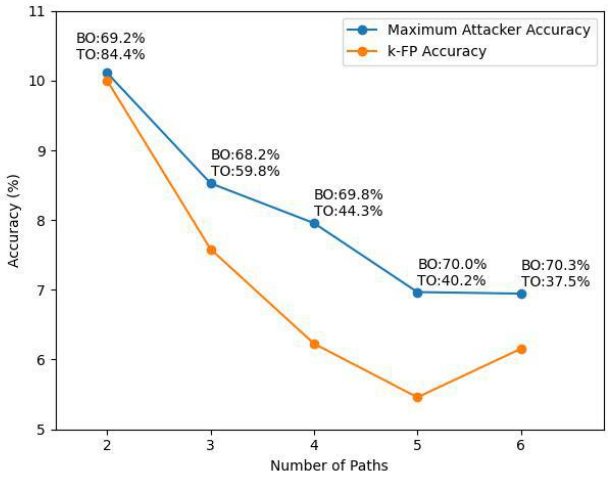


Fig. 6: The change in the maximum attacker accuracy (MAA) and k-FP accuracy (closed world) with the number of paths. Configurations with the same number of paths have the same bandwidth overhead (BO) and time overhead (TO). $b = 1.5$ for each configuration.

6.6.2 Change of Base

Increasing the base b from 1.1 to 1.5 to 2 increases the bandwidth overhead from 44.2±0.5% to 69.0±0.8% to 99.1±0.7%, regardless of the number of paths used (see Table 4). The MAA decreases as well. For instance, when changing the base for four networks, the MAA drops from 12.2% to 8.0% to 5.6% (also see Figure 7).

Tab. 4: The overheads and accuracies of different configurations of Hydra. These results are graphed in Figure 7.

Num. Paths	Thresholds	Base	Bandwidth Overhead	Time Overhead	k-FP Accuracy	MAA
2	0, 400	1.1	43.8%	85.5%	16.2%	18.2%
2	0, 400	1.5	69.2%	84.4%	10.0%	10.1%
2	0, 400	2	99.3%	83.6%	7.1%	7.3%
3	0, 300, 800	1.1	44.0%	61.3%	10.0%	14.1%
3	0, 300, 800	1.5	68.2%	59.8%	7.5%	8.5%
3	0, 300, 800	2	98.4%	58.6%	6.2%	6.5%
4	0, 200, 400, 800	1.1	44.7%	46.2%	8.3%	12.2%
4	0, 200, 400, 800	1.5	69.8%	44.3%	6.2%	8.0%
4	0, 200, 400, 800	2	99.7%	43.2%	4.9%	5.6%

However, for configurations with the same number of paths, the time overhead remains relatively constant, regardless of the base. Configurations with two, three, and four networks incur 84.6±1.0%, 60.0±1.4%, and 44.7±1.5% time overhead, respectively. This is because changing the base does not further delay any packets, but rather changes the amount of padding added to the end of the trace. Thus, the final time overhead remains the same.

Based on these observations, we suggest that if the user wishes for a faster surfing experience, the user should allow more time for Hydra to configure more paths. If the user is concerned with network congestion, the user should use a smaller base.

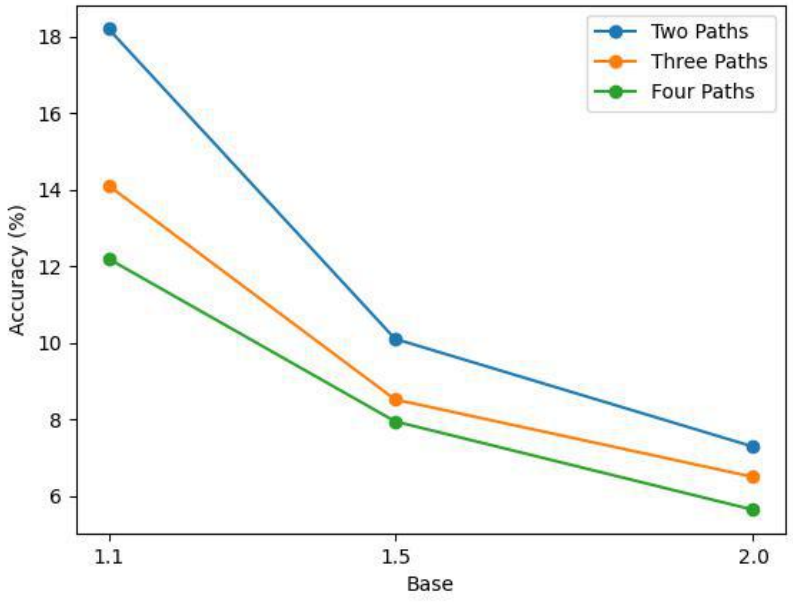


Fig. 7: The effect of changing the base b on the maximum attacker accuracy (MAA) for different numbers of networks.

7 Discussion

Hydra is effective in that it allows the same website to generate traces of different lengths. Since the only distinguishing feature in Hydra's defended traces is the trace length, the attacker is more prone to misclassification due to the variety of trace lengths that a single website can produce.

Additionally, Hydra is a provable defense, which means that our defense guarantees an upper bound on any attacker's accuracy. As future WF attacks become more powerful, Hydra will remain secure due to its provable nature.

In fact, Hydra provides lower overheads than existing state-of-the-art provable defense [3][2]. Even when Hydra produces bandwidth overheads equivalent to those of Tamaraw, not only does Hydra yield lower attacker accuracies, but Hydra also reduces the time overhead by almost 50%. This is in part due to Hydra's ability to split the traffic among several different Tor nodes, thereby speeding up the transmission rate and reducing the latency.

When deployed in the real world, Hydra also offers an additional layer of security. With Hydra's traffic-splitting strategy, some website visits may not trigger the usage of certain networks. This leads some networks to remain closed during the entire website visit. In our evaluation, we calculated the MAA and k-FP's accuracy with the assumption that the attacker would always see a non-empty sub-trace, which is not always the case in the real world. Thus, Hydra provides yet another layer of security in which the attacker may not even be aware that a website visit has occurred.

8 Future Work

In this section, we discuss the possible areas of study for future work.

Other Attacks. In this paper, we did not evaluate the efficacy of DF against Hydra due to the amount of computational power required to run DF. However, we hypothesis that DF will achieve higher accuracies than k-FP. Whereas k-FP focuses mainly on features related to timing [6], DF disregards the timing all together [15]. Thus, we assume that DF may perform better than k-FP since Hydra normalizes interpacket timing. In our future work, we plan to investigate DF's accuracies against Hydra.

Traffic splitting latency. Setting up more networks for Hydra to route the user's traffic may take up more time compared to using one path. At the very least, Hydra must connect to m different Tor entry nodes. Although Hydra shows that increasing the number of paths decreases the time overhead, using too many paths may in fact overload the entry nodes, thereby increasing the time overhead. In our future work, we plan to calculate the cost of setting up networks and bandwidth overload and determine the optimal number of paths to use.

End of page load. Hydra stops sending packets when the total trace length is padded to a power of b and when there are no more packets in the queue. However, if the total trace length reaches a power of b and no packets are in either the client's queue or the server's queue, our defense terminates regardless of whether the website has finished loading or not. This could potentially result in a partial page load. In our evaluation, we kept Hydra running until all packets from the original trace were sent. However, in the real world, Hydra would not know when to stop sending packets. This is also a problem with other constant-stream defenses [5][3]. We propose a solution involving sending a dummy packet at the very end of a page load that signals to Hydra that the page has finished loading.

Parameters of Hydra. Hydra offers high tunability; the user may configure the number of paths to use and the amount of padding to add. However, the relationship between the thresholds and the overheads (as well as the accuracy) is murky. In our evaluation, we tested many different thresholds for different numbers of networks and chose the thresholds that yielded the least overheads, which are listed in Table 5. We plan to determine a formula to optimize the thresholds for any given number of networks in future work.

9 Conclusion

In this paper, we developed Hydra, an efficient, provably secure defense based on traffic splitting that offers high tunability. Hydra splits the user's traffic into various streams and normalizes the traffic trace to

Tab. 5: The thresholds used in our evaluation.

Num. Paths	Thresholds
2	0, 400
3	0, 300, 800
4	0, 200, 400, 800
5	0, 200, 400, 800, 1200
6	0, 200, 600, 400, 800, 1200

remove distinguishable features from the attacker. Each website visit generates several different sub-traces of drastically different lengths, which allows for many sites' trace lengths to collide. Hydra is able to drop any attacker's accuracy from 92.2% to 8.5% while only incurring 68.2% bandwidth overhead and 59.8% time overhead. Thus, Hydra promises both efficiency and efficacy when existing state-of-the-art defenses only guarantee one.

10 Acknowledgments

I would like to thank my mentor David Lu for supporting me and providing valuable resources during this project.

References

[1] Ahmed Abusnaina, Rhongho Jang, Aminollah Khormali, DaeHun Nyang, and David Mohaisen. Dfd: Adversarial learning-based approach to defend against website fingerprinting. 2020.

[2] Xiang Cai, Rishab Nithyanand, and Rob Johnson. Cs-buflo: A congestion sensitive website fingerprinting defense. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, WPES '14, page 121–130, New York, NY, USA, 2014. Association for Computing Machinery.

[3] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A systematic approach to developing and evaluating website fingerprinting defenses. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, page 227–238, New York, NY, USA, 2014. Association for Computing Machinery.

[4] Wladimir De la Cadena, Asya Mitseva, Jan Pennekamp, Jens Hiller, Fabian Lanze, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. Poster: Traffic splitting to counter website fingerprinting. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2533–2535, New York, NY, USA, 2019. Association for Computing Machinery.

[5] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, page 332–346, USA, 2012. IEEE Computer Society.

[6] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 1187–1203, Austin, TX, August 2016. USENIX Association.

[7] Sébastien Henri, Gines Garcia-Aviles, Pablo Serrano, Albert Banchs, and Patrick Thiran. Protecting against website fingerprinting with multihoming. volume 2020, pages 89 – 110, Berlin, 2020. Sciendo.

[8] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. volume 9878, pages 27–46, 09 2016.

[9] David Lu, Sanjit Bhat, Albert Kwon, and Srinivas Devadas. Dynaflow: An efficient website fingerprinting defense based on dynamically-adjusting flows. In *Proceedings of the 2018 Workshop on Privacy*

in the Electronic Society, WPES'18, page 109–113, New York, NY, USA, 2018. Association for Computing Machinery.

[10] N. Mathews, P. Sirinam, and M. Wright. Understanding feature discovery in website fingerprinting attacks. In *2018 IEEE Western New York Image and Signal Processing Workshop (WNYISPW)*, pages 1–5, 2018.

[11] Andriy Panchenko, Fabian Lanze, Jan Pennekamp, Thomas Engel, Andreas Zinnen, Martin Henze, and Klaus Wehrle. Website fingerprinting at internet scale. In *NDSS*, 2016.

[12] The Tor Project. Who uses Tor? 2019. `https://2019.www.torproject.org/about/torusers.html.en`.

[13] The Tor Project. Users - Tor Metrics. 2020. `https://metrics.torproject.org/userstats-relay-country.html`.

[14] Vitaly Shmatikov and Ming-Hsiu Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In *Proceedings of the 11th European Conference on Research in Computer Security*, ESORICS'06, page 18–33, Berlin, Heidelberg, 2006. Springer-Verlag.

[15] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 1928–1943, New York, NY, USA, 2018. Association for Computing Machinery.

[16] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 143–157, San Diego, CA, August 2014. USENIX Association.

[17] Tao Wang and Ian Goldberg. Walkie-talkie: An efficient defense against passive website fingerprinting attacks. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1375–1390, Vancouver, BC, August 2017. USENIX Association.

A Biologically Inspired Search Algorithm for Optimal Network Design

Albert Tam

Abstract

Optimal road network design is a crucial component of the city planning process. The mixed network design problem (MNDP) aims to modify existing links and add new candidate links to optimize network performance, and solutions to it form an important part of road network planning. However, current solutions to the MNDP are computationally intensive and can impede the road planning process. We introduce a novel algorithm, BioNet, to solve the MNDP as well as related road network design problems. Inspired by the slime mold *Physarum*, BioNet makes iterative changes to the links of a network in order to optimize performance. We show that BioNet matches the state-of-the-art on multiple classes of road network design problems, including the MNDP. In particular, its computational time for the MNDP is much lower than that of existing solution methods to the MNDP and delivers a speedup of over 400 times. Due to BioNet's computational performance on the MNDP, it appears to be a promising approach for the rapid development of large network designs. Further directions include considering a wider variety of optimization objectives, such as fault tolerance, and generalizing BioNet to other classes of network design problems outside of road network design.

1 Introduction

A city's transportation system, particularly its road network, is a core component of a city's physical, economic, and social organization and can shape its development. The network design problem (NDP) is concerned with determining the optimal ways to improve an existing road network, given a fixed financial budget. Road networks are represented as graphs with nodes and links. Nodes represent high-density places along the network and are connected by directed links, which are each associated with capacities. Traffic on the network is expressed in terms of link flows, which are numbers associated with each link that represent the amount of traffic on the link. Together with capacities, link flows are a factor in determining the travel time on each link.

There are three main classifications of the NDP: the discrete network design problem (DNDP), the continuous network design problem (CNDP), and the mixed network design problem (MNDP). The DNDP aims to find the optimal set of links to add from a given set of candidate links, and the CNDP aims to find the optimal changes in link capacity to make to existing links. The MNDP mixes the two and considers both the discrete decision variables of the DNDP and the continuous decision variables of the CNDP. Table 1 shows the types of NDPs depending on the types of candidate improvements each problem considers. In general, the MNDP aims to determine the optimal capacity changes and link additions to make, given a network of nodes and links, demand between each origin-destination pair, a set of capacity improvement costs, a set of new candidate links, and budget constraints. Challenges of the problem include its nonlinearity and non-convexity [5]. In this project, we propose BioNet, a novel solution method to the MNDP inspired by the slime mold *Physarum*.

2 Literature Review

The MNDP is typically formulated as a mathematical programming with equilibrium constraints problem and expressed as a mixed, nonlinear bi-level optimization problem [20]. The lower level problem is the traffic assignment or user equilibrium (UE) problem, in which link flows are predicted, and the upper level problem consists of expanding link capacities and adding new links. The lower-level problem

Tab. 1: Types of road network design problems by types of candidate projects considered.

Variation	Capacity expansion	New link addition
Discrete (DNDP)	No	Yes
Continuous (CNDP)	Yes	No
Mixed (MNDP)	Yes	Yes

can be solved using a deterministic user equilibrium model or a stochastic user equilibrium (SUE) model. The SUE model accounts for the fact that perceived travel times may not equal actual travel times [4]. However, calculations using SUE are more difficult, so most studies prefer to use the deterministic user equilibrium model instead [5]. For the overall MNDP, two common formulations appear in the literature: either the objective is a weighted sum of passenger travel times and network construction cost [13] [22], or the objective is the sum of passenger travel times, and construction cost is a constraint [13] [14].

The non-linear and non-convex nature of the problem, which arises from its discrete, multi-objective nature, has prevented the development of an exact solution algorithm that runs in reasonable time. Thus, research has focused almost exclusively on finding suitable heuristic and metaheuristic solutions. Examples of previous heuristics are the link-based mixed-integer linear program (LMILP) method [14]. This method finds linear approximations to the non-linear objective function and solves the simplified linear programming problem using an exact method. Metaheuristics applied to the MNDP include genetic algorithms, simulated annealing, tabu search, and scatter search [2] [9]. Out of these metaheuristics, genetic algorithms and tabu search achieve the highest objective functions on the tested networks.

However, many limitations exist on current solution methods to the MNDP. Finding solutions is still computationally intensive, with the LMILP method taking 35 minutes to run on a medium-sized network [14]. Improving computational speed would expedite the overall road network planning process. Furthermore, the heuristics applied depend heavily on the approximations derived from the mathematical formulation of the MNDP and cannot be readily transferred to related NDPs, such as those for bus and railway network design.

In other fields, biologically inspired algorithms and methods, particularly neural networks and new metaheuristics, have led to significant computational advances. Initially inspired by the brain, neural networks have recently been applied to solve difficult tasks in a wide variety of domains, such as image classification, speech recognition, text generation, and medical diagnosis. Similarly, inspired by the process of evolution, genetic algorithms [10] have been used for a wide variety of optimization problems, including the MNDP itself. Other metaheuristics, such as particle swarm optimization–inspired by the social behavior of swarms [11]–and ant colony optimization–inspired by the behavior of ant colonies–have also been applied to a number of combinatorial optimization problems.

Previously, the slime mold *Physarum polycephalum* has been used to find solutions to the railway NDP for medium-sized networks [17]. In the NDP considered by Tero et al., a new network is built without starting from an existing network, unlike in the MNDP. In these experimental models, food sources were placed at nodes in a dish containing the slime mold. Initially, the organism grew in all directions and branched out to cover the entire dish. Over the next few days, regions of the organism that did not receive nutrients disappeared, until all that remained were tunnels connecting different food sources, whose capacities varied with the amount of food at each tunnel's endpoints. This model of growth, where all links are initialized at first, and their corresponding capacities are changed, has not been applied to the road MNDP, and computational models inspired by *Physarum* could offer a new method.

3 Purpose

1. Implement a solution to the lower-level user equilibrium problem in the form of the Frank-Wolfe algorithm [6].

2. Implement the novel biologically inspired algorithm for the upper-level optimization problem.

3. Evaluate our methods against existing methods by comparing the metrics of including sum of travel times and network building cost.

4 Notation and Problem Formulation

Here, we mathematically formulate the mixed network design problem, the continuous network design problem, and the user equilibrium problem. In order for our method to solve the MNDP and CNDP, it must solve the user equilibrium problem to determine flows on each link.

4.1 Notation

Throughout this paper, we will use the following notation:

A_1	the set of unchanged links
A_2	the set of expanded links
A_3	the set of added candidate links
A	the set of links in the network, which is $A_1 \cup A_2 \cup A_3$
x_a	the traffic flow on link $a \in A$
$\mathbf{x}$	a vector whose elements are x_a
y_a	the incremental capacity on expanded link $a \in A_2$
$\mathbf{y}$	a vector whose elements are y_a
u_a	a 0-1 decision variable which is 1 if link $a \in A_3$ is added and 0 otherwise
$\mathbf{u}$	a vector whose elements are u_a
$g_a(y_a)$	the improvement cost function of link $a \in A_2$
d_a	the construction cost of adding link $a \in A_3$
$t_a(x_a)$	the travel time function of link $a \in A_1$
$t_a(x_a, y_a)$	the travel time function of link $a \in A_2$ with added capacity y_a
$t_a(x_a, u_a)$	the travel time function of link $a \in A_3$
K_{rs}	the set of paths from node r to node s
f_k^{rs}	the flow on path k from node r to node s
$\delta_{a,k}^{rs}$	a variable which equals 1 if link a is on path k from node r to node s and 0 otherwise
$q_r s$	the demand from node r to node s
R	the set of origin nodes, i.e. nodes from which there is demand
S	the set of destination nodes, i.e. nodes to which there is demand
Z	the objective function
θ	the weight of construction cost in Z
I	the construction budget

The travel time function t_a of a link is given by

$$t_a = A_a + B_a \left(\frac{x_a}{c_a} \right)^4,$$

where c_a is the total capacity of the link, and A_a and B_a are constants that vary from link to link.

4.2 Mixed network design problem

We solve two versions of the mixed network design problem. The first, which we call the unconstrained MNDP, aims to minimize an objective function Z, which is a weighted sum of total passenger travel time and network construction cost. It is formulated as

$$\min_{\mathbf{y},\mathbf{u}} Z(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in A_1} x_a t_a(x_a) + \sum_{a\in A_2} x_a t_a(x_a,y_a) + \sum_{a\in A_3} x_a t_a(x_a,u_a) + \theta\left(\sum_{a\in A_2} g_a(y_a) + \sum_{a\in A_3} d_a u_a\right)$$

$$\text{s.t. } y_a \geq 0. \qquad \text{(all capacity changes are nonnegative)}$$

For the other version, which we call the constrained MNDP, the objective function Z is the sum of passenger travel times. The constrained MNDP aims to minimize Z while keeping the network construction cost below a predetermined budget I and is formulated as

$$\min_{\mathbf{y},\mathbf{u}} Z(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in A_1} x_a t_a(x_a) + \sum_{a\in A_2} x_a t_a(x_a,y_a) + \sum_{a\in A_3} x_a t_a(x_a,u_a)$$

$$\text{s.t. } \sum_{a\in A_2} g_a(y_a) + \sum_{a\in A_3} d_a u_a \leq I \qquad \text{(construction cost is under budget)}$$

$$y_a \geq 0 \qquad \text{(all capacity changes are nonnegative)}$$

4.3 Continuous network design problem

In the CNDP, there are no new candidate links to be added, and the only possible changes are capacity expansions of existing links. Thus, it can be thought of as a special case of the MNDP, so our method also solves the CNDP. The CNDP is usually stated in its unconstrained form, where the objective function Z is a weighted sum of total travel time and network construction cost. It is formulated as

$$\min_{\mathbf{y},\mathbf{u}} Z(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in A_1} x_a t_a(x_a) + \sum_{a\in A_2} x_a t_a(x_a,y_a) + \theta \sum_{a\in A_2} g_a(y_a)$$

$$\text{s.t. } y_a \geq 0. \qquad \text{(all capacity changes are nonnegative)}$$

4.4 User equilibrium problem

In the unconstrained MNDP, constrained MNDP, and CNDP, the link flows $\mathbf{x}$ are determined by solving the lower-level user equilibrium problem. We assume that route choice behavior follows Wardrop's user equilibrium model [19], in which travelers choose routes to minimize their individual travel times. The problem of determining the link flows given the network structure is formulated as

$$\min_{\mathbf{x}} \sum_{a\in A_1}\int_0^{x_a} t_a(w)\,dw + \sum_{a\in A_2}\int_0^{x_a} t_a(w,y_a)\,dw + \sum_{a\in A_3}\int_0^{x_a} t_a(w,u_a)\,dw$$

$$\text{s.t.} \sum_{k\in K_{rs}} f_k^{rs} = q_{rs}\,\forall r\in R, s\in S$$

$$f_k^{rs} \geq 0\,\forall r\in R, s\in S, k\in K_{rs}$$

$$x_a = \sum_{r\in R}\sum_{s\in S}\sum_{k\in K_{rs}} f_k^{rs}\delta_{a,k}^{rs}\,\forall a\in A.$$

This problem is solved using the Frank-Wolfe algorithm [6]. In practice, although Frank-Wolfe may yield exact solutions, it takes too long to do so. As a result, the algorithm is usually stopped before an exact solution is reached.

5 BioNet, a Biologically Inspired Solution Method to the MNDP

We propose BioNet, a solution method to the MNDP inspired by the slime mold *Physarum*. Figure 1 shows *Physarum*'s growth over time. Initially, *Physarum polycephalum* grows evenly outward to fill its entire environment [18]. Pathways within the network that do not receive nutrients degenerate, and certain pathways that do transport nutrients are selectively strengthened and widened. Eventually, after enough time, the *Physarum* creates a stable network that only contains pathways connecting food sources. Since a pathway's thickness depends on the amount of nutrients it transports, the *Physarum* essentially attempts to optimize the network for the objective of effective nutrient transportation.

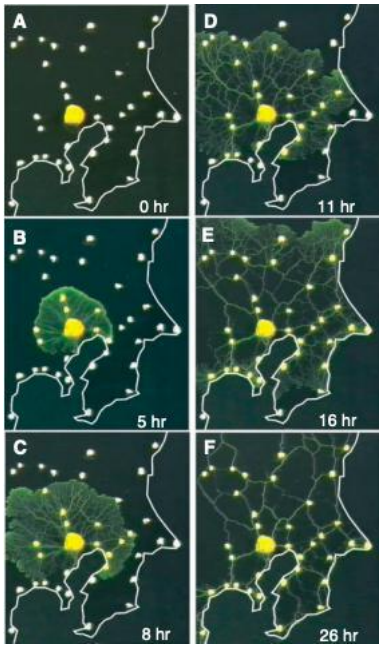


Fig. 1: Growth and network formation of *Physarum polycephalum*, reprinted from Tero et al., 2010. White dots represent food sources, and the organism grows outward from the center of the figure.

To simulate the growth of *Physarum*, which evenly explores its whole environment, the network is initialized with all links set to an arbitrarily high capacity. In practice, we set this arbitrarily high capacity

to be 3 times the capacity of the link with the greatest capacity in the network. In each iteration, the method changes the capacities of each link that is targeted for improvement. These capacity changes are based off of the estimated benefit of improving that link, which we measure with an "effectiveness factor" μ_a calculated for each link in A_2 and A_3. The capacities of links with low μ_a values are reduced more than those of links with high μ_a values, and vice versa.

Since links in A_3 are controlled by a discrete decision variable, they can only either be added to the network with a fixed capacity or not added at all. For such a link $a \in A_3$, we keep track of a temporary capacity y_a, which does not necessarily equal the capacity that a would have if it were added. After the algorithm is complete, we add all links $a \in A_3$ such that y_a is above a certain threshold to the network with their respective candidate capacities.

The process can be controlled by four parameters: m, ℓ, c_1, and c_2. The parameters m and ℓ control the size of the capacity changes made by the algorithm, and c_1 and c_2 control the threshold at which the algorithm accepts or rejects adding or improving links in the network. The procedure can be described as follows:

Step 0: *Initialization*. Set the capacity of all links in A_2 to an arbitrarily high capacity. Set $u_a = 1$ for all $a \in A_3$.

Step 1: *Objective checking*. If the objective function is below the target objective, or the construction cost is below budget, depending on the problem formulation, go to step 5.

Step 2: *Link evaluation*. Solve the user equilibrium problem, and update all link flows. For each link $a \in A_1 \cup A_2$, define the effectiveness factor

$$\mu_a \coloneqq \frac{x_a}{c_a},$$

where c_a is the capacity of link a. For a candidate link $a \in A_3$, define its effectiveness factor to be

$$\mu_a \coloneqq \frac{x_a}{c_a d_a}.$$

Step 3: *Update μ_{max}*. Find the maximum value of μ across all links. If this value is greater than μ_{max}, set μ_{max} to this value. If μ_{max} does not exist, set it to the maximum value of μ.

Step 4: *Update link capacities*. For all links $a \in A_2$, update their capacities by the rule

$$y_a \coloneqq y_a \gamma_a,$$

where

$$\gamma_a = \frac{(m-\ell)\mu + \ell\mu_{max}}{\mu_{max}}.$$

Step 5: *Clean links*. For all links $a \in A_2$ such that $y_a < c_1$, set $y_a = 0$. For all links $a \in A_3$ such that $y_a < c_2$, set $u_a = 0$. Go to step 1.

Step 6: *Re-initialize links in A_3*. For all links $a \in A_3$ such that $y_a > 0$, set $u_a = 1$.

Keeping track of a single μ_{max} value throughout the entire procedure prevents large capacity fluctuations as the algorithm proceeds.

6 Evaluations

In this section, we explain the tests that have been carried out on BioNet, the results of which are in the following section. We implemented the algorithm in Python 3.8.3 and used this implementation for all our tests. To compare BioNet to existing methods, we evaluate its performance on a small 16-link test network [7] and the medium-sized Sioux Falls network [12], which are commonly used for measuring NDP solution performance. Since there are significantly more solution methods for the CNDP than for the MNDP, we also compare BioNet's performance on the CNDP to other solution methods for the CNDP. Table 2 contains a list of methods to which we compare BioNet.

Tab. 2: Existing solution methods to the CNDP and MNDP.

Abbreviation	Algorithm	Problems solved	Source
H-J	Hooke-Jeeves	CNDP only	[1]
EDO	Equilibrium decomposed optimization	CNDP only	[16]
SA	Simulated annealing	CNDP only	[8]
SAB	Sensitivity analysis-based	CNDP only	[21]
GP	Gradient projection	CNDP only	[3]
CG	Conjugate gradient projection	CNDP only	[3]
QNEW	Quasi-Newton projection	CNDP only	[3]
PT	PARTAN gradient projection	CNDP only	[3]
LMILP	Link-based mixed-integer linear program	CNDP and MNDP	[14]
DDIA	Dimension-down iterative algorithm	CNDP and MNDP	[13]
-	BioNet	CNDP and MNDP	This paper

6.1 Notes on tests

To evaluate the total travel time of a network, the user equilibrium problem must be solved for that network. Since the Frank-Wolfe algorithm for UE problems takes an extremely long time to converge on a solution, different thresholds are used by different studies when calculating total travel time of a network. Therefore, to reduce any bias that may result from these differences, we calculate the objective function for all solutions using a common threshold of 100 Frank-Wolfe iterations. While more iterations may be desirable in real-world network design, we cap all testing at 100 Frank-Wolfe iterations to standardize comparisons between algorithms and to speed up tests.

When comparing results, we report the number of UE problems solved for our algorithm instead of computational time, since times can vary between different computing platforms. For reference, on a personal computer with a 1.6 GHz Intel Core i5 and 8 GB of RAM, one UE problem on the 16-link network takes around 20 ms to solve, and one UE problem on the Sioux Falls network takes around 375 ms to solve.

To choose values for the parameters m, ℓ, c_1, and c_2, we ran tests by solving the CNDP over a small 5-link network with a large number of combinations of different parameter values. Eventually, we selected a small number of feasible values for each parameter. When evaluating our method over the test networks, we ran tests for all combinations of the following parameter values:

$$\begin{aligned} m &= 0 \\ \ell &\in \{0.8, 1\} \\ c_1 &\in \{0, 0.1, 0.2\} \\ c_2 &\in \{0.05, 0.1\}. \end{aligned}$$

6.2 Continuous case

For the CNDP, we compare BioNet to the existing CNDP solution methods listed in Table 2. For each method, we evaluate the objective function Z as well as the total travel time and construction cost on the network.

The simple 16-link test network, which is shown in Figure 2, consists of 6 nodes. We use the input data given in Suwansirikul et al. (1987), which consists of the starting network, link improvement costs, and an O-D demand matrix.

There are only 2 origin-destination demand pairs, namely $(1, 6)$ and $(6, 1)$. We consider the case where these pairs have demand 5 and 10, respectively. The objective function to be minimized is

$$Z = \sum_{a \in A_1} x_a t_a(x_a) + \sum_{a \in A_2} x_a t_a(x_a, y_a) + \theta \sum_{a \in A_2} g_a(y_a),$$

where $\theta = 1$ and $g_a(y_a) = k_a y_a$ for a given link improvement cost k_a. The cost k_a varies from link to link and is defined for all 16 links in the network. We used parameter values of $m = 0$, $\ell = 0.8$, and $c_1 = 0.5$. Since no new links are being added, we did not need to provide a c_2 value.

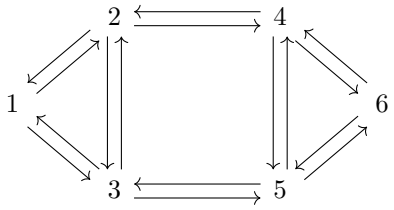


Fig. 2: The 16-link test network. All links are targeted for improvement.

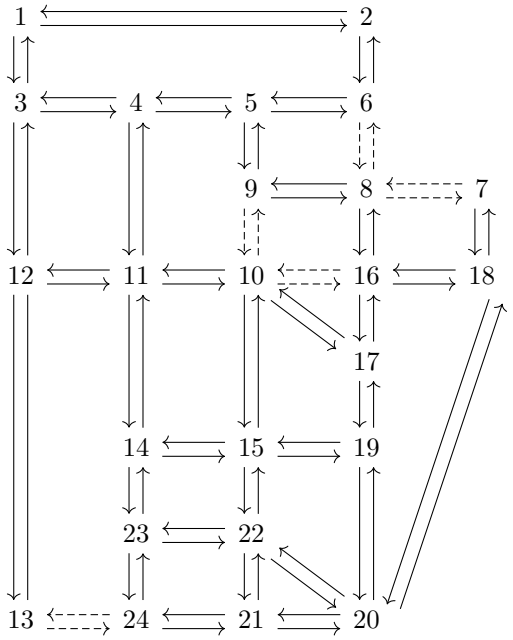


Fig. 3: The Sioux Falls test network. The dashed links are targeted for improvement.

The medium-sized Sioux Falls network, which is shown in Figure 3, consists of 24 nodes and 76 links. Again, we use the input data from Suwansirikul et al. (1987). For the CNDP on this network, the objective function to be minimized is the typical CNDP objective function

$$Z = \sum_{a \in A_1} x_a t_a(x_a) + \sum_{a \in A_2} x_a t_a(x_a, y_a) + \theta \sum_{a \in A_2} g_a(y_a),$$

where $\theta = 0.001$ and $g_a = k_a y_a^2$ for a given link improvement cost k_a. This cost k_a is only defined for the links that are targeted for improvement, which are the dashed links in Figure 3. We used parameter values of $m = 0$, $\ell = 1$, and $c_1 = 0.5$. Again, no new links are being added, so we did not need to provide a c_2 value.

6.3 Mixed case

Figure 4 shows the Sioux Falls network in the MNDP case, with 10 candidate links added. For the MNDP, we compare BioNet to three other solution methods for the MNDP. Due to the relative scarcity of network comparisons, we only evaluate BioNet's performance on the MNDP over the Sioux Falls network. We follow the approach of Luathep et al. (2011) and combine the link improvement costs of Suwansirikul

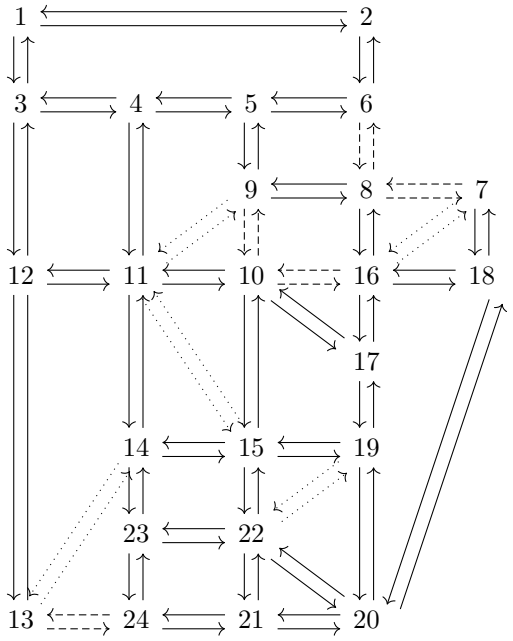


Fig. 4: The Sioux Falls test network for the MNDP case. Dashed links are targeted for capacity expansion, and dotted links are candidates for being added to the network.

et al. (1987) and the new candidate link projects of Poorzahedy and Turnquist (1982). We evaluate BioNet's performance on both the unconstrained and constrained MNDP.

For the unconstrained MNDP, the objective function to be minimized is

$$Z(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in A_1} x_a t_a(x_a) + \sum_{a\in A_2} x_a t_a(x_a,y_a) + \sum_{a\in A_3} x_a t_a(x_a,u_a) + \theta\left(\sum_{a\in A_2} g_a(y_a) + \sum_{a\in A_3} d_a u_a\right),$$

where $\theta = 0.001$ and $g_a(y_a) = k_a y_a^2$. These values are the same as those used for the CNDP over the Sioux Falls network. We used parameter values of $m = 0$, $\ell = 0.8$, $c_1 = 0$, and $c_2 = 0.1$.

We considered the constrained MNDP with a budget of $I = 4000$, or 4 million dollars. The objective function to be minimized is the total travel time on the network, or

$$Z(\mathbf{y},\mathbf{u},\mathbf{x}) = \sum_{a\in A_1} x_a t_a(x_a) + \sum_{a\in A_2} x_a t_a(x_a,y_a) + \sum_{a\in A_3} x_a t_a(x_a,u_a).$$

We used parameter values of $m = 0$, $\ell = 0.8$, $c_1 = 0.2$, and $c_2 = 0.05$.

7 Results

In this section, we apply the proposed solution algorithm to the CNDP and MNDP and compare them to existing methods in the literature. We follow the tests described in the Evaluations section above. Capacity expansion values for each solution method and information on which links were added for each MNDP solution method can be found in the appendix.

7.1 Continuous case

7.1.1 16-link network

The results in Table 3 show that our method finds a solution with a rather low value for Z, outperforming the majority of the methods we compared ours to. Our value for Z is less than 1% greater than the Z value found by LMILP, the best-performing method on this network. The construction cost of our solution is also less than 4% greater than the least expensive method and is much lower than the costs achieved by all other solutions. However, we solve slightly more UE problems than other methods, so our method may run slower than others. However, our method runs in less than a second for the 16-link network, so any such differences would be minimal.

Tab. 3: Results for the CNDP case on the 16-link network.

Metric	H-J	EDO	SA	SAB	GP	CG	QNEW	PT	LMILP	BioNet
Total travel time (10^3 veh)	188.35	187.25	191.45	186.35	186.60	185.93	185.96	**185.88**	186.80	190.47
Construction cost (10^3 \$)	29.80	13.95	**9.89**	17.85	17.16	14.30	14.65	16.54	12.84	10.27
UE problems solved	54	10	18300	**6**	14	7	12	16	-	21
Z	218.15	201.20	201.34	204.21	203.76	200.23	200.62	202.42	**199.63**	200.74

Note: Luathep et al. (2011) did not provide the number of UE problems solved.
Lower travel time, lower construction cost, fewer UE problems solved, and a lower value of Z are better. The best value for each metric is bolded.

7.1.2 Sioux Falls network

Tab. 4: Results for the CNDP case on the Sioux Falls network.

Metric	H-J	EDO	SA	SAB	GP	CG	QNEW	PT	LMILP	DDIA	BioNet
Total travel time (10^3 veh)	76.547	80.255	75.791	**73.564**	78.134	76.226	76.753	77.876	76.246	76.127	77.397
Construction cost (10^3 \$)	5079	**3132**	5487	10796	5973	6626	6451	6296	5027	4910	5216
UE problems solved	58	12	3900	11	10	6	**4**	7	-	-	37
Z	81.626	83.387	81.278	84.360	84.108	82.852	83.204	84.172	81.273	**81.038**	82.612

Note: Luathep et al. (2011) and Liu and Chen (2016) did not provide the number of UE problems solved.
Lower travel time, lower construction cost, fewer UE problems solved, and a lower value of Z are better. The best value for each metric is bolded.

For the CNDP over the Sioux Falls network, Table 4 shows that our method matches performance with existing methods on the continuous problem. Our objective function is closest to that of the CG method. However, in comparison, the construction cost of the network generated by our method is 5.216 million dollars, which is much smaller than the 6.626 million dollar construction cost of the CG method. Although our method solves more equilibrium problems than some other methods and therefore may be slightly slower, our method takes 16 seconds to run on the Sioux Falls network, so differences would also be small.

7.2 Mixed case

For the unconstrained MNDP over the Sioux Falls network, we compare our method to DDIA, another MNDP solution method, and the "solve all CNDPs" method proposed by Liu and Chen (2016). The "solve

Tab. 5: Results for the unconstrained MNDP case on the Sioux Falls network.

Metric	DDIA	Solve all CNDPs	BioNet
Total travel time (10^3 veh)	56.708	56.709	**55.913**
Construction cost (10^3 \$)	**10208**	10249	12067
UE problems solved	-	-	3
Z	**66.916**	66.958	67.980

Note: Liu and Chen (2016) did not provide the number of UE problems solved.
Lower travel time, lower construction cost, fewer UE problems solved, and a lower value of Z are better. The best value for each metric is bolded.

all CNDPs" method enumerates all the possible combinations of adding new links and solves a CNDP for each one.

Table 5 shows the results for the unconstrained MNDP on the Sioux Falls network. Here, our method achieves a similar value on the objective function to DDIA. In addition, our method runs in fewer than 3 seconds. However, computational time is not reported for DDIA or the method of solving all CNDPs, so we cannot compare our time to theirs.

Tab. 6: Results for the constrained MNDP case on the Sioux Falls network.

Metric	LMILP	DDIA	BioNet
Construction cost (10^3 \$)	4000	3970	**3856**
UE problems solved	-	-	9
Z (total travel time, 10^3 veh)	**68.212**	68.651	68.87

Note: Luathep et al. (2011) and Liu and Chen (2016) did not provide the number of UE problems solved.
Lower travel time, lower construction cost, fewer UE problems solved, and a lower value of Z are better. The best value for each metric is bolded.

Table 6 shows the results for the constrained MNDP on the Sioux Falls network with a budget of $I = 4000$, or 4 million dollars. All three methods achieve very similar results. Our total travel time is less than 1% greater than that achieved by LMILP, the best-performing method, and our construction cost is also the lowest out of all the methods. Furthermore, our method achieves a greater than 400 times computational speedup over LMILP, with a runtime of only 5 seconds, compared to 35 minutes for LMILP. [14].

8 Conclusion and Discussion

The mixed network design problem aims to find improvements on an existing road network to optimize an objective function, usually a combination of total travel time and network construction cost. Because of the problem's non-linearity and non-convexity, heuristics and metaheuristics have been developed to solve the problem. However, these methods are computationally intensive and difficult to generalize to different objectives and types of networks. We propose BioNet, a method for solving the MNDP inspired by the slime mold *Physarum*. BioNet finds solutions by initializing all links and selectively keeping those that are effective, which we measure by calculating the ratio of link flow to link capacity. Our method can be applied to different classes of road network design problems, such as the CNDP and DNDP.

To evaluate the effectiveness of our method, we performed tests on a small 16-link network and the medium-sized Sioux Falls network. We compared our method's performance to existing methods on the CNDP, unconstrained MNDP, and constrained MNDP. For all classes of problems, BioNet matches the

performance of current state-of-the-art methods, and its computational time either matches that of existing methods or is lower by far. On the CNDP, our method solves slightly more UE problems than some other methods, but the computational times are still small (< 20 seconds). On the unconstrained and constrained MNDPs, our method also performs similarly to existing methods on the objective function. However, it runs much faster than current solutions on the constrained MNDP.

For both the unconstrained and constrained MNDP, BioNet solves fewer UE problems than for the two CNDP tests. Thus, BioNet performs better with respect to computational time on the MNDP than for the CNDP. We believe that this difference may be due to BioNet's exploratory nature; it naturally considers the effects of adding or removing links, and limiting it to only consider capacity expansions hinders its performance. Furthermore, since BioNet does not solve significantly more problems for the Sioux Falls network than for the small 16-link network, the number of UE problems solved may depend strongly on network size. This may be due to the fact that BioNet changes the entire network at once, so improvements to different parts of the network can be made in parallel in the same number of iterations. As a result, computational time does not appear to vary strongly with network size, so our method may be able to offer fast performance on larger networks.

This paradigm of biologically inspired learning is the first of its kind to be applied to road network design, and it offers a number of benefits over existing methods. Based on the comparisons of computational speed that could be made, our method runs significantly faster than previously published state-of-the-art methods. The biologically inspired framework that our method uses is also more generalizable and can lead to new solution methods to other types of NDPs. Furthermore, our method allows for the consideration of alternative objective functions. While we consider the ratio of flow to capacity as well as improvement cost in μ, other metrics may be incorporated into the calculation of μ to optimize different objectives.

Future directions include considering a wider variety of objectives, evaluating BioNet's performance on larger networks, and incorporating more sophisticated traffic models. In particular, fault tolerance [17], environmental impact, and equity in travel time [5] have been identified as objectives that are of further interest in road network design. As for applying BioNet to larger networks, while we have not evaluated its performance on large networks, BioNet's performance on small and medium-sized test networks in this paper indicate that it may offer a promising approach for large networks. Finally, in order to better model traveler behavior, BioNet may be combined with more realistic models, such as elastic demand, which allows for changes in origin-destination demand over time, and stochastic user equilibrium, which corrects for the possibility that travelers' perceptions of travel time may not reflect actual travel time. Applying these models would make our framework even more applicable to real-life network design.

9 Acknowledgements

I would like to thank my mentor at SSI, Dhaman Kaur, for providing valuable advice on both my project and this paper. I would also like to thank another mentor at SSI, Allison Tam, for her feedback and encouragement on my project and paper. Finally, I would like to thank Dhruvik Parikh, Anne Lee, and Franklyn Wang from SSI for their comments on this paper.

References

[1] Mustafa Abdulaal and Larry J. LeBlanc. Continuous equilibrium network design models. *Transportation Research Part B: Methodological*, 13(1):19–32, March 1979.

[2] G.E. Cantarella, G. Pavone, and A. Vitetta. Heuristics for urban road network design: Lane layout and signal settings. *European Journal of Operational Research*, 175(3):1682–1695, December 2006.

[3] S Chiou. Bilevel programming for the continuous transport network design problem. *Transportation Research Part B: Methodological*, 39(4):361–383, May 2005.

[4] Carlos F. Daganzo and Yosef Sheffi. On Stochastic Models of Traffic Assignment. *Transportation Science*, 11(3):253–274, August 1977.

[5] Reza Zanjirani Farahani, Elnaz Miandoabchi, W.Y. Szeto, and Hannaneh Rashidi. A review of urban transportation network design problems. *European Journal of Operational Research*, 229(2):281–302, September 2013.

[6] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, March 1956.

[7] T. L. Friesz and P. T. Harker. Bounding the Solution of the Continuous Equilibrium Network Design Problem. *Proceedings of the 9th International Symposium on Transportation and Traffic Theory*, 1984.

[8] Terry L. Friesz, Hsun-Jung Cho, Nihal J. Mehta, Roger L. Tobin, and G. Anandalingam. A Simulated Annealing Approach to the Network Design Problem with Variational Inequality Constraints. *Transportation Science*, 26(1):18–26, February 1992.

[9] Mariano Gallo, Luca D'Acierno, and Bruno Montella. A meta-heuristic approach for solving the Urban Network Design Problem. *European Journal of Operational Research*, 201(1):144–157, February 2010.

[10] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[11] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948, Perth, WA, Australia, 1995. IEEE.

[12] Larry J. LeBlanc, Edward K. Morlok, and William P. Pierskalla. An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research*, 9(5):309–318, October 1975.

[13] Xinyu Liu and Qun Chen. An Algorithm for the Mixed Transportation Network Design Problem. *PLOS ONE*, 11(9):e0162618, September 2016.

[14] Paramet Luathep, Agachai Sumalee, William H.K. Lam, Zhi-Chun Li, and Hong K. Lo. Global optimization method for mixed transportation network design problem: A mixed-integer linear programming approach. *Transportation Research Part B: Methodological*, 45(5):808–827, June 2011.

[15] Hossain Poorzahedy and Mark A. Turnquist. Approximate algorithms for the discrete network design problem. *Transportation Research Part B: Methodological*, 16(1):45–55, February 1982.

[16] Chaisak Suwansirikul, Terry L. Friesz, and Roger L. Tobin. Equilibrium Decomposed Optimization: A Heuristic for the Continuous Equilibrium Network Design Problem. *Transportation Science*, 21(4):254–263, November 1987.

[17] A. Tero, S. Takagi, T. Saigusa, K. Ito, D. P. Bebber, M. D. Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki. Rules for Biologically Inspired Adaptive Network Design. *Science*, 327(5964):439–442, January 2010.

[18] Atsushi Tero, Ryo Kobayashi, and Toshiyuki Nakagaki. A mathematical model for adaptive transport network in path finding by true slime mold. *Journal of Theoretical Biology*, 244(4):553–564, February 2007.

[19] J G Wardrop and J I Whitehead. Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers*, 1(5):767–768, October 1952.

[20] Hai Yang and Michael G. H. Bell. Models and algorithms for road network design: a review and some new developments. *Transport Reviews*, 18(3):257–278, July 1998.

[21] Hai Yang and Sam Yagar. Traffic assignment and signal control in saturated road networks. *Transportation Research Part A: Policy and Practice*, 29(2):125–139, March 1995.

[22] Haozhi Zhang and Ziyou Gao. Bilevel programming model and solution method for mixed transportation network design problem. *Journal of Systems Science and Complexity*, 22(3):446–459, June 2009.

A Solution details

Tab. 7: Capacity expansion values for the CNDP case on the 16-link network.

Variable	H-J	EDO	SA	SAB	GP	CG	QNEW	PT	LMILP	BioNet
Links to be expanded										
y_3	1.20	0.13								
y_6	3.00	6.26	3.1639	5.8352	5.8302	6.1989	6.0021	5.9502	5.24	4.531
y_{15}	3.00	0.13		0.9739	0.8700	0.0849	0.1846	0.5798	0.002	
y_{16}	2.80	6.26	6.7240	6.1762	6.1090	7.5888	7.5438	7.1064	7.585	5.743

Note: If no number is given in a cell, the solution method did not choose to expand the corresponding link.

Tab. 8: Capacity expansion values for the CNDP case on the Sioux Falls network.

Variable	H-J	EDO	SA	SAB	GP	CG	QNEW	PT	LMILP	DDIA	BioNet
Links to be expanded											
y_{16}	4.8	4.59	5.38	5.7392	5.4277	4.7691	5.3052	5.0237	5.362	3.9375	6.253
y_{17}	1.2	1.52	2.26	51.7182	5.3235	4.8605	5.0541	5.2158	2.057	1.6875	
y_{19}	4.8	5.45	5.50	4.9591	1.6825	3.0706	2.4415	1.8298	5.486	4.3125	6.254
y_{20}	0.8	2.33	2.01	4.9612	1.6761	2.6836	2.5442	1.5747	1.895	2.9375	
y_{25}	2.0	1.27	2.64	5.5066	2.8361	2.8397	3.9328	2.7947	2.556	2.5000	0.810
y_{26}	2.6	2.33	2.47	5.5199	2.7288	2.9754	4.0927	2.6639	2.618	2.1875	0.907
y_{29}	4.8	0.41	4.54	5.8024	5.7501	5.6823	4.3454	6.1879	3.741	2.9375	4.685
y_{39}	4.4	4.59	4.45	5.5902	4.9992	4.2726	5.2427	4.9624	4.551	4.0000	3.797
y_{48}	4.8	2.71	4.21	5.8439	4.4308	4.4026	4.7686	4.0674	3.741	4.2500	4.831
y_{74}	4.4	2.71	4.67	5.8662	4.3081	5.5183	4.0239	3.9199	4.489	3.875	3.760

Note: If no number is given in a cell, the solution method did not choose to expand the corresponding link.

Tab. 9: Capacity expansion values and added links for the unconstrained MNDP case on the Sioux Falls network.

Origin node	Destination node	DDIA	Solve all CNDPs	BioNet
Links to be expanded				
6	8	3.250	3.500	2.870
7	8	1.250	1.500	2.407
8	6	3.750	4.000	2.953
8	7	1.250	1.750	1.557
9	10	0.750	1.250	1.694
10	9	0.938	0.750	1.625
10	16	4.250	4.000	4.643
13	24	1.000	1.500	1.782
16	10	4.500	4.000	4.348
24	13	1.000	2.000	1.795
New candidate links				
7	16	$-$	$-$	$+$
9	11	$+$	$+$	$+$
11	9	$+$	$+$	$+$
11	15	$+$	$+$	$+$
13	14	$+$	$+$	$+$
14	13	$+$	$+$	$+$
15	11	$+$	$+$	$+$
16	7	$-$	$-$	$+$
19	22	$+$	$+$	$+$
22	19	$+$	$+$	$+$

Note: For candidate links, $+$ indicates that the link was added, and $-$ indicates that the link was not added.

Tab. 10: Capacity expansion values and added links for the constrained MNDP case on the Sioux Falls network.

Origin node	Destination node	LMILP	DDIA	BioNet
Links to be expanded				
6	8	3.173	4.625	2.300
7	8	1.084	1.875	0.338
8	6	2.919	3.000	2.325
8	7	1.078	1.500	0
9	10	1.316	2.250	0.400
10	9	1.564	2.250	0.342
10	16	3.232	3.000	2.054
13	24	2.867	2.000	0.731
16	10	3.232	2.000	2.042
24	13	2.638	2.000	0.722
New candidate links				
7	16	−	−	−
9	11	−	−	−
11	9	−	−	−
11	15	+	+	+
13	14	−	−	−
14	13	−	−	+
15	11	+	+	+
16	7	−	−	−
19	22	−	−	−
22	19	−	−	−

Note: For candidate links, + indicates that the link was added, and − indicates that the link was not added.

Rapid DNA Barcode Demultiplexing

Michelle Lei

Abstract

Coronavirus has emphasized the necessity for inexpensive and rapid testing, which can be done using a handheld, realtime DNA sequencer, the MinION device. DNA barcodes are unique identifiers that can be attached to different people's DNA samples. Through the use of barcodes, multiple people's samples of DNA can be sequenced with the MinION device at one time. Barcode demultiplexing is the process by which the barcodes are sequenced so that the DNA fragments attached can be traced back to the original person. Currently, barcode demultiplexing is very inaccurate when basecalled, due to the noise in the barcodes, or very complex when using deep neural networks. Here, we present a new method by using raw electrical signals to maintain higher classification accuracy while reducing complexity. We simulate barcode reads of two different classes using DeepSimulator, then extract features including the Hjorth parameters and the Hurst exponent that were previously used in EEG data classification. We utilized supervised learning algorithms to classify the reads and compare the statistics and latency to the state-of-the-art Deepbinner. While maintaining the accuracy of Deepbinner, our model has a much lower latency and is less complex. Our model can classify with a 99.20% f1 score, which is comparable to Deepbinner's results. We can achieve this f1 score using around 625 barcodes to train, and the latency using random forest classification is 5.15 seconds. In the future, more than two barcodes can be classified using our model, and real data can be applied instead of simulated data. By reducing the complexity of Deepbinner, using supervised learning instead of deep neural networks, we can increase the efficiency and decrease the cost of demultiplexing to make testing scalable for future pandemics.

1 Introduction

Currently, SARS-CoV-2 is causing an unprecedented global pandemic that requires efficient and cost-effective testing. To test for viruses, the RNA fragments are converted to complementary DNA and are then sequenced. A third-generation sequencing method, the Oxford Nanopore Technology's (ONT) MinION, was popularized in recent years for its higher speed, longer read lengths, and real-time sequencing [6][13]. It is a portable, low cost handheld device, which makes it useful in doctors' offices. The MinION contains an electrically charged flow cell, perforated with nanopores. Adapters ligated to the DNA fragments lead the DNA fragment to one of the 512 channels in the flow cell. When the DNA enters, the nucleotides in each DNA fragment are measured as a sequence of electrical current signals called a "squiggle", which can be converted to a corresponding nucleotide sequence through a process called basecalling. Each squiggle obtained from a DNA fragment is called a read. The MinION has been used to test for Ebola, dengue, influenza, and the Ross River virus [6]. The MinION has the potential for detecting coronavirus in a reasonable time [5]. Current testing was insufficient to meet the demands of the Covid-19.

In order to make testing scalable for future pandemics, a MinION feature, barcoding, has gained some popularity. The LamPORE Assay, an assay designed by ONT to sequence more samples per flow cell, is being developed. It can run between 1 to 1,152 samples on one flow cell and takes about an hour to process the first 1 to 96 samples. The process of sequencing the remaining samples is projected to take around 4 hours. The LamPORE assay uses DNA barcodes to multiplex [3]. Barcodes are unique identifiers; for example, DNA barcodes, which are a part of the DNA of different species, are generally used to identify the species [1]. When multiple people's DNA fragments are mixed together, there is no unique identifier to distinguish the fragments from different individuals. For each individual, a unique barcode can be ligated to that particular individual's DNA fragments so that multiple samples of DNA to be tested for a virus can be inserted into the MinION device at once, or multiplexed. Many different samples can then be tested for a virus, as shown in Figure 1. The DNA sequences can later be matched, or demultiplexed, to the person they belong to through the use of the barcode classification [18]. If a DNA sample tests positive, the DNA barcodes can identify to whom the DNA sample belongs. This would allow for increased and

scalable testing in the future [3]. The barcode, however, becomes very noisy after basecalling. This could be a result of various factors, such as structural similarities in nucleotides, simultaneous influence of multiple nucleotides on the signal, nonuniform speeds of nucleotides through the pore, and the signal not changing within homopolymers [17]. Current solutions to this are very complex or inaccurate, but the problem itself seems simple.

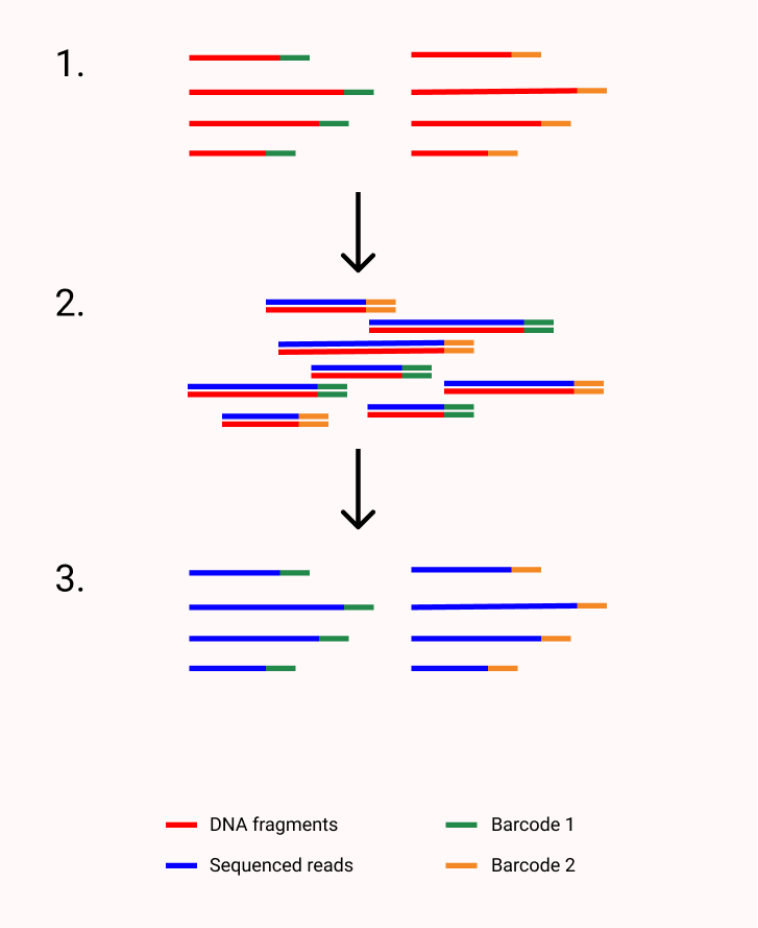


Fig. 1: DNA fragments with barcodes attached can be sequenced. The sequenced reads can be traced back to the original individual based on barcodes.

While very useful for making testing scalable, demultiplexing is a challenge. In the past, basecalling algorithms such as Albacore and Porechop were used to identify barcode reads. However, the basecalling methods are unable to classify many reads and have a 20% error rate [6]. Many reads could not be binned, or assigned to a barcode, because of how noisy the reads are. Instead of basecalling, the state-of-the-art demultiplexer Deepbinner uses raw electrical signals for classifying reads.

Electroencephalography (EEG) signals resemble MinION reads [10]. EEG signals and MinION reads are both nonlinear and nonstationary, and have spontaneous oscillations [16]. This means that from individual to individual, the frequency of the EEG signal differs [14]. Similarly, from read to read, the frequency of the read signal changes. EEG data are classified through feature extraction and supervised learning algorithms. A novel approach to classifying barcode reads would be to utilize these EEG data classification methods. Simple features include the Hjorth parameters and the Hurst exponent [4]. The Hjorth parameters include activity, mobility, and complexity, and they are calculated using the variance of signal waves in a time series [8]. The Hurst exponent measures the long-memory and predictability of a signal in the time domain [16]. A value of 0.5 indicates that the series is random, while a value less than 0.5 indicates the series is anti-persistent, and a value greater than 0.5 indicates the series is persistent [15]. Instead of basecalling

the barcodes, we focus on signal characteristics and utilize the extra information in the raw data, similar to DeepBinner. This extra information that is lost when converted to base calls will help to classify barcodes faster and more efficiently [17]. However, instead of utilizing deep neural networks, we will classify barcodes through extracting these four features used for EEG signal data classification and using supervised learning algorithms, reducing the complexity of DeepBinner while maintaining its accuracy.

Accurately classifying barcodes quickly and simplistically will improve the LamPORE Assay. Multiple DNA samples can be simultaneously sequenced in a single flow cell, increasing the speed and reducing the cost of DNA sequencing for SARS-CoV-2. As future pandemics threaten to infect globally, scalable, fast, and inexpensive testing becomes more important.

The paper will proceed as follows. In Section 2, we review current literature on barcode demultiplexing. In Section 3, we explain the methodology of our research, including its design and implementation. Next, in Section 4, we provide our results and discuss them. Finally, in Section 5, we conclude and describe future directions.

2 Literature Review

Currently, DNA barcodes are being identified using basecalling methods, which are unable to classify many reads and have a 20% error rate [6]. The state of the art barcoding technology right now is Deepbinner, which looks at raw electrical signals to classify reads into the proper bins and operates using deep neural networks. By using signal-space instead of base-space tools, there is greater accuracy because the signals would not first be converted to base calls, where noise could be introduced [18]. Deepbineer has high sensitivity, can bin lower quality reads, and has higher precision. However, training Deepbinner's networks is computationally intensive and they does not generalize well over different flow cells and library preparation kits. The network is composed of groups of convolutional layers, which are the result of randomized trials of various architectures. The "blackbox" nature of Deepbinner makes it difficult to identify the reason behind an error in classification [18]. Because of the complexity, Deepbinner requires supercomputers to operate and a high level of computing for real-time analyses. The latency of Deepbinner is very high. These disadvantages introduce an opportunity to improve upon barcode demultiplexing. While maintaining the high precision and sensitivity of Deepbinner, we look to reduce its complexity.

In 1970, the Hjorth parameters had been discovered in order to classify EEG data. They were able to describe qualities of a time series through the use of variances, which allows them to inherit the additive properties of variance so that the parameter values measured pertain to the average basic elements in the complex curve [8]. In 2000, they were used to distinguish between three different mental states of normal EEG, where the lower dimensions of the feature vectors allowed for simplicity in computation and stability in classification [**?**]. The Hjorth parameters and another EEG feature extraction method Fisher ratio were also used to associate EEG patterns with certain physical tasks [14].

Recently, research was done exploring Hjorth parameters in classifying *E. coli* and bacteriophage DNA. The modulations of the ionic currents at each pore can be graphed as a squiggle when they pass through the Nanopore through Python packages such as Poretools [12]. Feature extraction is simple and undemanding when applying Hjorth parameters, requiring only a linear time complexity. The Hjorth descriptors consist of three features, which are activity, mobility, and complexity [10]. Activity represents the variance of a time function, and it is large if there are many high frequency components to the signal. Mobility is the proportion of the standard deviation of the power spectrum. Complexity indicates how the shape of a signal compares to a pure sine wave, converging to one as the shape of the signal becomes more similar to the sine wave [8]. These values can be calculated by using the variance of the squiggles.

$$Activity = var(y(t))$$

$$Mobility = \sqrt{\frac{var(\frac{dy(t)}{dt})}{var(y(t))}}$$

$$Complexity = \frac{Mobility(\frac{dy(t)}{dt})}{Mobility(y(t))}$$

The three features can be visualized as three-dimensional feature vectors that form clusters, and the DNA can be separated into two different organisms [10]. The transformation is also deterministic, so supervised algorithms can be trained to identify the phage DNA for automatic classification of the reads as the position of the clusters and their centroids will remain constant for a single organism. Possible supervised learning algorithms include Support Vector Machine (SVM), which finds an optimal hydroplane between two sets of datasets [7]; random forest, which assesses various binary trees that split data along nodes to eventually end in the right class [9]; and K-Nearest Number (K-NN), which classifies a data point based on the data points around that point [2]. The success of the Hjorth parameters in identifying phage DNA when mixed with bacteria DNA introduces the possibility of using the parameters for DNA barcode classification.

3 Methods

Instead of basecalling the barcodes into nucleotide sequences, we looked at raw electrical signals over a time-domain and utilized the extra information in them beyond the corresponding nucleotides. This allows for higher sensitivity, which can reduce the number of unclassified reads [19]. To assess classification of barcodes from feature extraction, we simulated a dataset using DeepSimulator, extracted features, and used supervised learning algorithms such as K-NN and SVM.

Simulating a Dataset

We obtained DNA barcodes through DeepSimulator, which generates barcode signals similar to those of real barcodes without sequencing real fragments of DNA. DeepSimulator is the state-of-the-art Nanopore simulator right now, and it is widely known for its accuracy and is commonly used in place of real reads. Two nucleotide barcode sequences, AAGAAAGTTGTCGGTGTCTTTGTG and TCGATTCCGTTTGTAGTCGTCTGT, were inputted into DeepSimulator, with 40,000 reads being generated for each.

DeepSimulator first samples sequences from the genome that correspond to DNA sequences which pass through the Nanopore, with the read length being alterable by the user. The length of DNA barcodes is twenty-four nucleotides, so the read-length set was twenty-four [18]. A context independent pore model, based on the official Nanopore Tech stats of 6-mers, helps generate signals quickly. The signal generator makes generated raw electrical signals in the form of a time series that are very similar to real ones and are of variant qualities [11]. Because barcodes, at the ends of reads, are inherently noisier than regular reads, noise was added to the barcodes acquired from DeepSimulator. Based on average barcode noise levels, the noise on the barcodes was increased two standard deviations. When the signals were read, they appeared as a list of numbers. There were around 240 signals per read. Sometimes, DeepSimulator generated barcode reads of less than 100 signals. When this was the case, the barcode reads were discarded because the short reads could not give adequate information, and the sequencing quality was insufficient.

The barcode signal data stored in Fast5 files were extracted using the ONT Python library. 4,000 signal events were generated through a single second of time. Median/Median Absolute Deviation (MAD) normalization was then applied to the data, following by removing duplicates from the samples. A total of 3410 (1710 each) unique reads were included in further analysis.

Extracting Features

Features are commonly extracted from EEG data through the use of time domain properties. Squiggles from MinION reads are comparable to those from EEG data, being non-linear and non-stationary. Given the physical similarities between the two time series, we extracted common features for barcode classification. However, because MinION reads are only twenty-four nucleotides long and much smaller than EEG data, resulting in around 600 signals per read, choosing a limited number of features to extract should be sufficient. Hjorth parameters have previously been used to classify between bacteria and bacteriophage DNA [10], so we used them to classify the barcodes.

A common tool for evaluating features to classify EEG data is Pyeeg. This Python library consists of several functions and can calculate Hjorth parameters, entropy, and Hurst exponent, amongst other

features. Because our data was standardized using the median and median absolute deviation and the Pyeeg source code operates off of different assumptions, we wrote our own functions similar to that of Pyeeg to find the Hjorth parameters: activity, mobility, and complexity. The Hjorth parameters and the barcode class were added to a dataframe, which was then split into train and test sets, using a random seed of 31.

We classified the reads using SVM and random forest, finding the f1 scores to be around 80%. We added another feature commonly used for EEG data, the Hurst exponent. With the additional feature, the total number of dimensions became four. Since classifying twenty-four nucleotide barcode reads should not be a complex task, we removed one feature. After looking at scatter plots of activity against the other three features, we found that activity did not show a large distinction between the barcode classes. We removed activity and discovered that the f1 score and other assessment statistics increased for SVM and random forest. Activity is the variance of the signal, whereas the other features extracted require more calculation. Therefore, variation between the two barcode classes may have been too similar for activity to have distinguished between them.

Supervised Learning Algorithm

Several supervised learning algorithms were implemented to classify the barcodes, including K-NN, random forest, and SVM. Through plotting various f1 scores from different k values, the k value of 3 resulted in the highest f1 score. Tenfold cross validation was used for random forest. After using GridSearchCV, the RBF kernel with a tenfold cross validation was determined to have the highest f1 score for SVM. We set a constant random seed of 31, and we took statistical metrics after training 20% of the reads and testing on the remaining 80%.

We reserved half of the data for testing and used the other half for training, increasing the training number of reads from 50 to 1700 incrementally by 50 to find the minimum training dataset number for the f1 score to remain using convergence plots. The convergence plots are seen in Figure 1.

The state-of-the-art is Deepbinner, which has a high accuracy, but also a large classification latency. We ran Deepbinner on a Linux machine to assess its latency and accuracy on the same barcodes used for feature extraction and classification. Because we ran the feature extraction classification without a Linux machine, we ran Deepbinner on a single thread so as to measure the prediction latency of the model, not the aptitude of the machine. The barcode sequences used in the current study belong to the sequences in the native barcoding kit, for which a pre-trained model has been built for the Deepbinner software [19]. Therefore, we used Deepbinner to perform classification on the two barcodes with the native model. Had the nucleotide sequences used in the study not been included in the pretrained models, a lot of data and heavy computation would have been needed to train Deepbinner [19].

Four Barcodes

We evaluated the performance of our model when using four barcodes. Using DeepSimulator, we generated two additional twenty-four nucleotide signal sequences. The nucleotide sequences inputted were GAGTCTTGTGTCCCAGTTACCAGG and TTCGGATTCTATCGTGTTTCCCTA. Two standard deviations of noise were added to these sequences as well. After removing duplicates and reads that were less than 100 nucleotides, we standardized the signals. We then extracted mobility, complexity, and the hurst exponent from these reads. Using these features, we tried different supervised learning algorithms, including K-NN, random forest, and SVM. Through plotting the macro f1 score across different k values, we found that a k value of 3 performed the best. We found through GridSearchCV that the polynomial kernel, with degree 3, performed the best. We compared tenfold polynomial SVM, random forest, and 3-NN.

4 Results and Discussion

To evaluate Deepbinner and our model, we looked at statistical metrics, including the f1 score, sensitivity, specificity, and precision. We also looked at the latency of each classification method. The latency of classification is measured as the delay between when the user inputs the command to classify and when the

program produces an output. An attribute of an efficient classification method is a short latency. Sensitivity is the proportion of positive reads binned correctly. Specificity is the proportion of negative reads binned correctly. Precision is the proportion of reads binned positive that are true positives. In all of these metrics, a true positive can be defined as Barcode 1 being classified as Barcode 1 and a true negative can be defined as Barcode 2 being classified as Barcode 2. The f1 score can be defined as

$$f1\ score = \frac{2 \times precision \times sensitivity}{precision + sensitivity}$$

Algorithm	**Sensitivity** (%)	**Specificity** (%)	**Precision** (%)	**F1 Score** (%)	**Latency** (s)
3-NN	99.54	97.15	97.30	98.41	0.29
RBF SVM	99.71	94.77	95.09	99.31	0.03
Random Forest	99.27	99.12	99.13	99.20	5.15
Deepbinner	100	100	100	100	107.56

Tab. 1: The sensitivity, specifity, precision, f1 score, and latency values across different classification algorithms and Deepbinner.

The latency for classification of random forest is 5.15 seconds while the latency of Deepbinner, which is 107.56 seconds. The sensitivity, precision, specificity, and f1 scores of random forest were all above 99%, and Deepbinner's were 100%. In addition, 24.77% of Deepbinner's reads were not binned, which were not considered in the table.

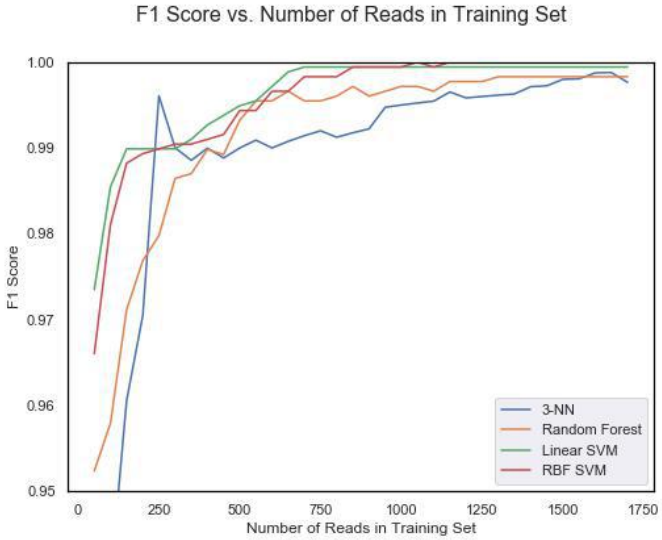


Fig. 2: A convergence plot showing f1 scores of RBF SVM, Linear SVM, Random Forest, and 3-NN across different training set sizes.

After evaluating RBF SVM, 3-NN, and random forest, we found that out of all the supervised learning algorithms used, random forest was shown to have the highest statistical metrics except for sensitivity and f1 score, with values of above 99% for every metric. It achieved a 99.20% f1 score while only training 625 out of all 3420 reads, which is around 19% of the total reads, as seen in Figure 2.

We attempted to classify two more barcodes along with the other two as an extension, for a total of four barcodes. For four barcode classification, we used sensitivity, precision, and macro f1 score, which is the average of the f1 scores for each barcode class, as statistical metrics. We found that random forest performed the best, with sensitivity, precision, and macro f1 score all above 90% as seen in Table 2.

Algorithm	**Sensitivity** (%)	**Precision** (%)	**Macro F1 Score** (%)	**Latency** (s)
3-NN	86.94	86.74	88.93	0.66
Polynomial SVM	88.27	88.16	88.16	0.13
Random Forest	90.23	90.24	90.22	12.96

Tab. 2: The sensitivity, precision, macro f1 score, and latency values across different classification algorithms for four barcode classification.

Using random forest and tenfold cross validation, the DNA barcodes could be classified with comparable accuracy, sensitivity, and precision to DeepBinner, as seen in Table 1. The latency of the supervised learning algorithm classification was much shorter than DeepBinner. DeepBinner must classify using its neural network hidden layers in a very time expensive process, a process which is much simpler when training and classifying using a supervised learning algorithm. The 100% statistical metrics of Deepbinner were the result of Deepbinner being pretrained with a package of reads. If a read is too short, there is not enough information for Deepbinner to bin the read, and these reads that were not binned were not considered for the table.

Our data came from a simulator rather than actual MinION reads because of the lack of accessibility in obtaining real reads. While our model outperformed Deepbinner in latency and maintained similar sensitivity, precision, and f1 accuracy, we did not have real reads to test the two models against, though DeepSimulator is well-known for its accuracy in simulating reads. In our research, we did not perform barcode extraction as would be necessary in real reads; instead, we simulated only barcodes. This is a limitation in the performance of our model, as we do not know how this might affect the latency or the statistical metrics.

Both DeepBinner and demultiplexing using feature extraction utilize raw electrical signals as opposed to basecalling. These raw electrical signals carry extra information in them besides the nucleotides they correspond to that allow them to be binned more often and with higher accuracy. Deep convolutional neural networks, which are used for DeepBinner, are extremely accurate, but they are very complex. They have many hidden layers that require a lot of time and data to train. Using feature extraction of two Hjorth parameters, mobility and complexity, and the Hurst exponent, the data could be plotted with three dimensional feature vectors in Figure 3. They formed evident clusters which could be separated using deterministic algorithms. For similar accuracy to DeepBinner, 3-NN only needed to train 625 reads. Given the simplicity of classifying 24 nucleotide reads, using deep neural networks appears unnecessary. Supervised learning algorithms work just as well with less training, less time, and lower costs.

DeepBinner is computationally intensive and does not generalize well across different flow cells. It is also intensive to install. DeepBinner has many layers to train, so it takes a long time to train and classify barcodes. Right now, there are only two pre-trained models included in Deepbinner software. To train barcodes other than the ones in the models requires a lot of data and a fast computer, preferably with TensorFlow on a big GPU [19]. Its "black box" nature makes the classification process difficult to understand or resolve if there is an error. When using DNA barcodes to multiplex DNA samples for efficient testing, it is important to demultiplex as quickly as possible, with as much accuracy as possible, and with as little training as possible. Instead of making the problem more complex, as is the case with going from basecalling to deep neural networks, the problem should be made simpler by using supervised learning algorithms and extracting features, including the Hjorth parameters and the Hurst exponent. This novel approach to classifying DNA barcodes will allow these barcodes to then be demultiplexed rapidly and with small amounts of training data while resulting in high macro f1 scores, precision, and sensitivity. Barcode demultiplex-

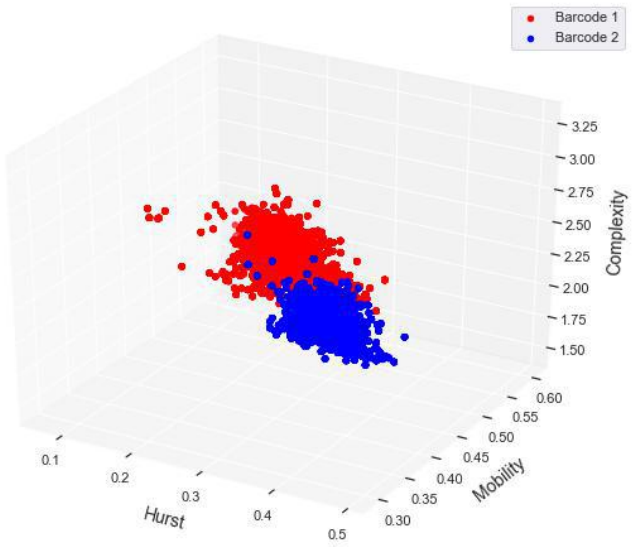


Fig. 3: A convergence plot showing f1 scores of RBF SVM, Linear SVM, Random Forest, and 3-NN across different training set sizes.

ing using feature extraction will contribute to the efficiency and cost-effectiveness of the LamPORE Assay, providing faster and less expensive testing for diseases.

While classifying between two different classes of barcodes works well with feature extraction, the LamPORE Assay has the ability to multiplex up to 1,152 samp in a single flow cell. Currently, Deepbinner is capable of demultiplexing twelve barcodes, which is the number of distinct barcodes in barcode kits, with high accuracy [18]. In the future, more barcodes could potentially be demultiplexed using supervised learning algorithms. We have begun to classify four barcodes, with macro f1 scores from random forest being 90.22%, shown in Table 2. We notice that very few barcodes are misclassified to be barcode 1, while barcodes 3 and 4 are frequently classified as the other. This could be a result of similar nucleotide sequences throughout the barcodes. While multiple barcode demultiplexing can be done, with this model, the barcodes should ideally be very different from each other. The next step is to utilize real data to classify reads, and to explore other features to better classify multiple barcode classes.

5 Conclusion

DNA barcodes allow for fast and inexpensive testing since the price of a single flow cell can be held constant and multiple DNA fragments can be sequenced simultaneously. To demultiplex the noisy DNA barcodes, we use raw electrical signals, for the raw electrical signals contain more information than basecalled sequences. We are not interested in the exact sequence of the barcodes; rather, we want to classify them, to identify to which person's DNA they are attached. Because of this, basecalling becomes unimportant.

Using data generated by DeepSimulator, we were able to reconstruct squiggles from a list of signals. By taking the variances of these signals, we then extracted Hjorth parameters and the Hurst exponent from the reads. We were able to obtain a 99.20% f1 score, a 99.13% precision, a 99.12% specificity, and a 99.27% sensitivity from using random forest, in Table 1. These values were comparable to the performance of the state-of-the-art, Deepbinner, which utilizes deep neural networks. Deepbinner is computationally

intensive, so its classification latency was high. When compared with our model, which had a latency of 5.15 seconds, Deepbinner had a higher latency of 107.56 seconds. Deepbinner also requires much more training than our model, which only required 19% of the data for training to achieve a 99.20% f1 score.

Our model outperformed Deepbinner in latency, being a much simpler solution to barcode demultiplexing. At the same time, our model maintained the statistical performance of Deepbinner. In future research, multiple barcodes can be classified using feature extraction instead of just the two in this research.

The LamPORE Assay has provided an opportunity for multiplexing barcodes through the MinION device to sequence DNA samples faster and for less cost. This will make testing scalable for future pandemics. In the past, because DNA barcodes are noisy, demultiplexing has either been inaccurate or very complex. Our model is able to maintain the state-of-the-art's accuracy while reducing complexity when classifying between two barcode classes.

References

[1] *DNA barcoding - International Barcode of Life*.

[2] *K Nearest Neighbor Algorithm In Python — by Cory Maklin — Towards Data Science*.

[3] *LamPORE Assay*.

[4] F. S. Bao, X. Liu, and C. Zhang. Pyeeg: An open source python module for eeg/meg feature extraction. *Computational Intelligence and Neuroscience*, 2011:1–7, 2020.

[5] B. Chen, E.-K. Tian, B. He, L. Tian, R. Han, S. Wang, Q. Xiang, S. Zhang, T. El Arnaout, and W. Cheng. Overview of lethal human coronaviruses. *Signal Transduction and Targeted Therapy*, 5(1):1–16, 2020.

[6] D. Filloux, E. Fernandez, E. Loire, L. Claude, S. Galzi, T. Candresse, S. Winter, M. L. Jeeva, T. Makeshkumar, D. P. Martin, and P. Roumagnac. Nanopore-based detection and characterization of yam viruses. *Scientific Reports*, 8(1):17879, 2018.

[7] R. Ghandi. *Support Vector Machine – Introduction to Machine Learning Algorithms*.

[8] B. Hjorth. Eeg analysis based on time domain properties. *Electroencephalography and Clinical Neurophysiology*, 29(3):306–310, 1970.

[9] Will Koehrsen. An implementation and explanation of the random forest in python.

[10] K. Kupkova, K. Sedlar, and I. Provaznik. Reference-free identification of phage dna using signal processing on nanopore data. In *2017 IEEE 17th International Conference on Bioinformatics and Bioengineering (BIBE)*, pages 101–105, Washington, DC, 10 2017. IEEE.

[11] Y. Li, S. Wang, C. Bi, Z. Qiu, M. Li, and X. Gao. Deepsimulator1.5: a more powerful, quicker and lighter simulator for nanopore sequencing. *Bioinformatics*, 36(8):2578–2580, 2020.

[12] N. Loman and A. Quinlan. Poretools: a toolkit for analyzing nanopore sequence data. *Bioinformatics*, 30(23):3399–3401, 2014.

[13] Hengyun Lu, Francesca Giordano, and Zemin Ning. Oxford nanopore minion sequencing and genome assembly. *Genomics, Proteomics & Bioinformatics*, 14(5):265–279, 2016.

[14] S.-H. Oh, Y.-R. Lee, and H.-N Kim. A novel eeg feature extraction method using hjorth parameter. *IJEEE*, pages 106–110, 2014.

[15] Bo Qian and Khaled Rasheed. Hurst exponent and financial market predictability. page 7.

[16] B. Rahmani, C. K. Wong, P. Norouzzadeh, J. Bodurka, and B. McKinney. Dynamical hurst analysis identifies eeg channel differences between ptsd and healthy controls. *PLOS ONE*, 13(7):e0199144, 2018.

[17] F. J. Rang, W. P. Kloosterman, and J. de Ridder. From squiggle to basepair: computational approaches for improving nanopore sequencing read accuracy. *Genome Biology*, 19(1):90, 2018.

[18] R. R. Wick, L. M. Judd, and K. E. Holt. Demultiplexing barcoded oxford nanopore reads with deep convolutional neural networks. Technical report, Bioinformatics, 2018.

[19] Ryan Wick. rrwick/deepbinner.

Analyzing the Relation Between Government Anti-Contagion Policy Severity and United States COVID-19 Epidemiological Data

Jenny Wei

Abstract

Since the first emergence of the SARS-CoV-2 virus in Wuhan, China in December of 2019, the virus has spread to 216 countries and territories, infecting over 16.3 million people globally. To combat the COVID-19 pandemic, governments are implementing unprecedented anti-contagion policies with little known effects. Measuring the impact of anti-contagion policies would help governments make informed decisions and design effective policies for stopping the spread of COVID-19. In this work, we introduce an approach to assess United States policy severity by analyzing U.S. state government Twitter tweets using a random forest ensemble model. While previous studies have measured country-level effects of anti-contagion policies, we measure the relation between policy severity on U.S. state-level COVID-19 epidemiological data. We show that our method is able to quantitatively evaluate state government tweet severity and analyze both tweet severity and COVID-19 prevalence, hospitalization rate, and death rate for U.S. states. Future directions involve predicting COVID-19 trends from policy severity to quantify the effects of anti-contagion policies of various strengths.

1 Introduction

Since the first emergence of the SARS-CoV-2 virus in Wuhan, China in December of 2019, the virus has spread to 216 countries and territories around the world, infecting over 16.3 million people [1]. In the United States, the first case of COVID-19 was reported in the state of Washington in January of 2020, and according to the Centers for Disease Control and Prevention, the United States now accounts for over 25% of global COVID-19 cases: over 4.2 million people have been diagnosed with COVID-19 in all 50 states [2]. In the wake of this pandemic, governments are implementing policies designed to limit the spread of the coronavirus. These policies aim to restrict person-to-person contact and consist of, but are not limited to, banning group gatherings, restricting travel, closing schools and businesses, and encouraging social distancing and mask wearing. However, the impacts that these policies have on reducing the spread of COVID-19 are often unknown due to the unprecedented nature of the COVID-19 pandemic [3].

A major challenge to implementing policies lies in the economic costs that come with stay-at-home orders: reduced business activity leads to job losses, drops in economic activity, and increased government debt levels [4]. Many states in the U.S. are hesitant to put in place anti-contagion policies that have the potential to be detrimental to the economy, but do not have clear effects on reducing COVID-19 spread, all the while COVID-19 infection rates steadily rise across the U.S. Analysis of the effects that these anti-contagion policies have on COVID-19 infection rates could provide state governments with knowledge on what policies to implement in order to limit the spread of the coronavirus [5].

Several works address the impact of anti-contagion policies in multiple countries using open-source epidemiological data and policy data [3, 5, 6, 7]. These studies use epidemiological compartmental models (e.g. SIR model) to predict how various categories of anti-contagion policies affect the number of cases in specific locations. However, these works are conducted only on a country-wide scope and do not specifically address the varying severities of policies. Futhermore, these studies are impractical to conduct on a U.S. state-wide scale due to the difficulty of cataloging constantly changing policies that differ between many states.

In this study, we analyze the impact of specific policy severity on COVID-19 cases in the United States. In order to study this, we obtain Twitter tweets regarding anti-contagion guidelines and policies from U.S.

state government Twitter handles. We develop an approach to analyzing guideline severity and the impact of guideline severity on COVID-19 epidemiological data using natural language processing techniques and statistical analysis tools. We show that our method is able to effectively classify tweet severity and analyze the relation between tweet severity and COVID-19 prevalence, hospitalization rate, and death rate for individual U.S. states.

Finally, because our method is able to analyze anti-contagion policy severity, as well as the correlation between policy strength and COVID-19 data, it enables future work to be done on predicting how different levels of policy severity affect various measures of the COVID-19 pandemic, including, but not limited to, case count, hospitalization rate, and death rate. Our hope is that by modeling the relation between anti-contagion policy strength and COVID-19 epidemiological data, governments will be able to make informed decisions on policy implementation to better limit the spread of the COVID-19 disease.

2 Literature Review

2.1 Text Feature Selection Via Topic Modeling

In recent years, topic modeling has emerged as a standard tool for text feature selection, the selection of relevant features for document analysis [8]. A common form of topic modeling is Latent Dirichlet Allocation, an unsupervised probabilistic topic model for analyzing corpora, first proposed by Blei et al [9]. Latent Dirichlet Allocation assumes that for a given corpus containing M documents, the documents represent a distribution of T topics, and each topic is composed of a word distribution θ. For each document in the corpus, a topic z is sampled from the word distribution θ of the document, and a word w in θ that is associated with z is also sampled. This process is repeated N times, N being the number of words in the document, for each document in M. Every document is therefore represented as a probability distribution of T topics, with the distributions for each document varying depending on the word distribution θ of the document [10]. Latent Dirichlet Allocation models have consistent performance on short texts, including tweets, and by representing tweets as a distribution of topics, tweets can be aggregated into groups based on their dominant topic. This is particularly useful for gathering documents from Twitter: during the process of tweet mining, many irrelevant tweets are introduced into the dataset, and Latent Dirichlet Allocation can be used to prune irrelevant tweets, thereby extracting only the relevant features of a text corpora [11].

2.2 Supervised Approaches to Text Classification

Many approaches to text classification exist, the most common algorithms being decision trees, Naive-Bayes, K-nearest neighbors, logistic regression, and support vector machines. Naive-Bayes, K-nearest neighbors, and logistic regression are often used due to their simplicity, but their performance is suboptimal when analyzing text corpora containing larger amounts of documents [12]. Support vector machines are the most popular method for text classification due to their high precision, but tend to have poor recall scores. Decision trees are often combined together into random decision forests, an ensemble classification approach. Random forests are able to achieve high accuracy for text classification, but face the risk of overfitting [13].

2.3 Feature Extraction

A fundamental component of text classification lies in feature extraction, the conversion of unstructured documents to structured feature spaces. In order for a text corpora to be processed by text classification algorithms, the documents must be converted into n-dimensional data [14]. One method of feature extraction is Word2vec, a group of language models proposed by Mikolov et al. that create word embeddings [15]. Unlike rules-based embedding methods such as TF-IDF, which creates a matrix representation of a document by counting the term frequency of individual words, Word2vec models implement a state of the art two layer neural network approach to generate word embeddings, vectors that capture the semantic meaning of a document. One architecture of Word2vec is the Continuous Bag-of-Words model (Fig. 1). For each word in a document, the model one-hot encodes the future and history words surrounding the target word and inputs the encodings into a neural network, with the goal of correctly predicting the target

word. The vector embedding of the target word is then generated by taking the dot product of the neural network's weight and the one-hot encoding of the target word [16].

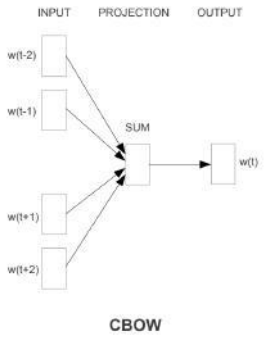


Fig. 1: Word2vec Continuous Bag-of-Words architecture. The CBOW model uses the surrounding words to predict the target word [16].

2.4 COVID-19 Tracking Via Twitter Data

Several studies have been published regarding the analysis of COVID-19 related Twitter data. Most notably, Singh et al. collected geotagged tweets containing phrases related to COVID-19 and evaluated the relation between tweet volume, tweet themes, and common phrases used in tweets with COVID-19 case counts in several countries. Furthermore, the study investigated the quality of information and level of misinformation in tweets, drawing the conclusion that Twitter conversations are a viable source for predicting the spread and outbreak of COVID-19.

Current studies involving the analysis of Twitter data exhibit the applicability of using Twitter data to model and predict the COVID-19 pandemic. However, these studies only compile general user tweets related to COVID-19, and to our knowledge, no publications regarding government COVID-19 tweet analysis have been released.

2.5 Anti-Contagion Policy Effect Analysis

Previously, studies have been conducted to investigate the impact of government interventions on the COVID-19 pandemic. Hsiang et al. [3] evaluated the effects of large-scale anti-contagion policies on the COVID-19 epidemiological data in China, France, Iran, Italy, South Korea, and the United States. A total of 1,717 policies were collected from January to April of 2020, and COVID-19 data regarding cumulative cases, recoveries, and deaths was acquired from each of the six countries. Compartmental SIR models, a type of epidemiological model which divides a population into three compartments (susceptible, infectious, and recovered) and applies differentiation to model the spread of a disease, were used to predict the effect of policy implementation on COVID-19 cumulative case numbers. The study found that implementation of combined policies lead to statistically significant reductions on the number of COVID-19 cases and COVID-19 infection rate (Fig. 2).

Another study by Dergiades et al. [7] measured the effect of government policy strength on COVID-19 mortality rates. Dergiades et al. compiled policy stringency indices for 32 countries from the University of Oxford's COVID-19 Government Response Tracker (OxCGRT) [17]. Policy stringency is a composite measure of anti-contagion policy strength calculated from nine indicators, including, but not limited to, the stringency of travel bans, school closures, and business closures, with each indicator ranging from 0 (least stringent) to 100 (most stringent). Dergiades et al. tested for a break in the slope of the time-series mortality rate data for each country, as changes in slope affect the method in which the time-series data is modeled and estimated. Autoregressive models, types of time-series forecasting models, were then applied to predict the trend of mortality rates in the countries studied. Probit regression, a model for dichotomous

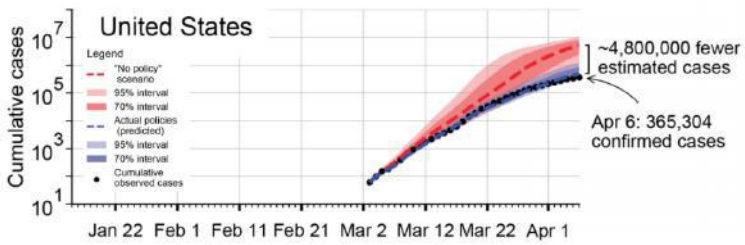


Fig. 2: Estimated cumulative confirmed COVID-19 infections with and without anti-contagion policies in the United States. A no policy scenario in the U.S. is predicted to lead to a 4.8 million increase in COVID-19 cases [3].

variables, was utilized to estimate the probability of attaining an insignificant trend slope for death rate, given the policy stringency index of a country. The study's results indicate that countries with higher stringency indices have a higher probability of attaining a trend slope of 0 (i.e. insignificant increase in death rate), and that increases in policy strength lead to a decrease in the average growth rate of deaths (Fig. 3).

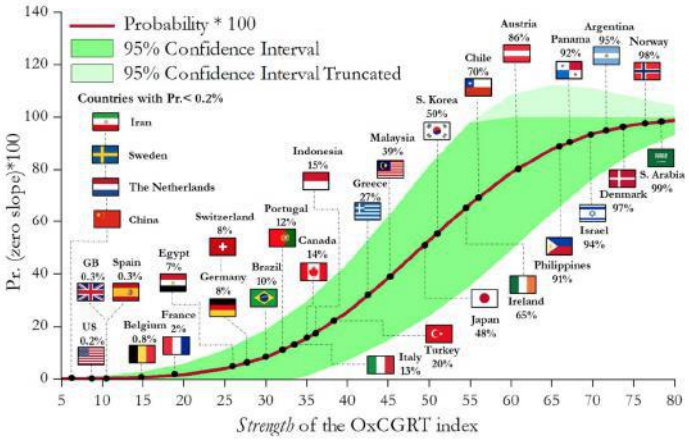


Fig. 3: Predicted probability of attaining an insignificant death rate from a country's OxCGRT policy stringency index [7].

Current studies published on anti-contagion policy display the efficacy of country-level COVID-19 policies, which are gathered from publicly released COVID-19 anti-contagion policy data. However, to our knowledge, no studies have been published on state-level anti-contagion policies in the United States, potentially due to a lack of compiled data on state-level policy analysis.

3 Purpose

1. We aim to analyze the severity of U.S. state anti-contagion policies by analyzing Twitter tweets from state government Twitter accounts. Due to the accessibility of Twitter, many state governments use Twitter to broadcast COVID-19 guidelines. Severity is defined as the stringency of anti-contagion policy, reflected by the language used in government tweets.
2. We aim to evaluate the relation between policy severity and COVID-19 prevalence, hospitalization rate, and death rate by performing regression tests.

4 Methods

4.1 Data Collection

To create a corpus relevant to individual U.S. state anti-contagion policies, we gather past Twitter tweets from U.S. state government accounts using Tweepy, a Python package for Twitter data mining. We acquire 149,633 tweets posted between August 7th, 2018 and July 18th, 2020 from 50 U.S. state government accounts that are labeled with the account handle, the U.S. state in which the account handle corresponds to, and the tweet date. The accounts are primarily individual state health department Twitter handles, but also include governor accounts for states that lack an active state health department Twitter account. The list of Twitter handles is included in Appendix A.

4.2 Data Preprocessing

We preprocess the tweets in five steps: 1) filter out tweets posted before March 1st, 2020, 2) standardize common COVID-19 phrases, 3) normalize the corpus, 4) perform lemmatization, and 5) remove stopwords. Tweets posted before March 1st, 2020 do not pertain to COVID-19 and are removed accordingly. Common COVID-19 related words are standardized into a single form (e.g. replacing $COVID$-19, $coronavirus$, $corona\,virus$, etc. with $covid$; replacing 6 $feet$, 6 ft, $six\,ft$, etc. with $six\,feet$) to reduce unnecessary variation. Normalization consists of removing duplicate tweets, converting the corpus to lowercase, removing URLs and user tags, and removing special characters and numbers. Lemmatization, the process of reducing an inflected form of a word to its base form (e.g. $infecting$ to $infect$), is performed using the spaCy library's $en\_core\_web\_sm$ model. Finally, we remove stopwords using a stopword list we define, due to pre-defined stopword lists removing words significant to tweet severity (e.g. $do\,not$). The stopword list is included in Appendix A.

4.3 Topic Labeling of Tweets

After data preprocessing, 40,327 tweets remain in the corpus. To eliminate remaining tweets unrelated to anti-contagion policy, we implement a two-phase topic modeling feature selection approach. Two topic models are used to filter out all irrelevant tweets, since tweets unrelated to anti-contagion policy remain after implementing only a single LDA model. We utilize Latent Dirichlet Allocation via the Gensim LDA Mallet package to cluster tweets into their primary topics. Both topic models are optimized by fine-tuning T, the number of topics, to maximize the coherence score of the model. For the first model, we find that a T of 24 topics yields the highest coherence score (Fig. 4). Based on the dominant keywords generated for each topic, we filter out topics that do not pertain to anti-contagion policies, reducing the corpus size to 15,170 tweets. For the second model, we find that a T of 15 topics yields the highest coherence score (Fig. 5). After filtering out topics which do not pertain to anti-contagion policies, we reduce the corpus size to 6,410 tweets. The list of topics for both models can be found in Appendix B.

4.4 Document Training Set Generation

To create a training set of documents, we define three classes: $severe$, $moderate$, and $not\,severe$. We randomly select a subset of 200 tweets, and we hand label each tweet with a class. From this labeled subset of tweets, we manually find keywords that are indicative of a tweet's severity level to create a list of keywords for each severity class. We use WordNet, a lexical database which organizes nouns, verbs, and adjectives into sets of synonyms, to obtain synonyms for the initial list of keywords in order to expand our keyword list. The list of keywords is found in Appendix B. The spaCy PhraseMatcher algorithm is used to create the training set of documents, which is defined as tweets containing three or more keywords from one of the three defined classes. In total, the training corpus consists of 194 tweets labeled $moderate$, 439 tweets labeled $not\,severe$, and 236 tweets labeled $severe$.

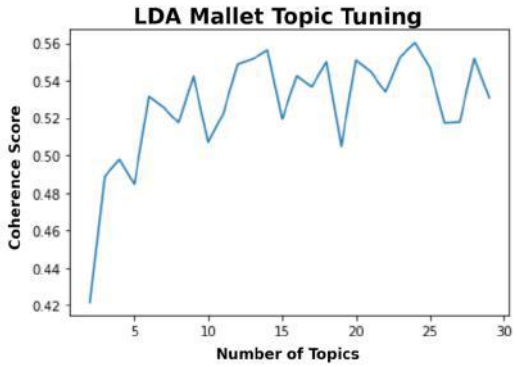


Fig. 4: Phase 1 T tuning of LDA model. 30 models are created with T values ranging from 1 to 30, and a coherence score is calculated for each model. We find that a T value of 24 yields the highest coherence score.

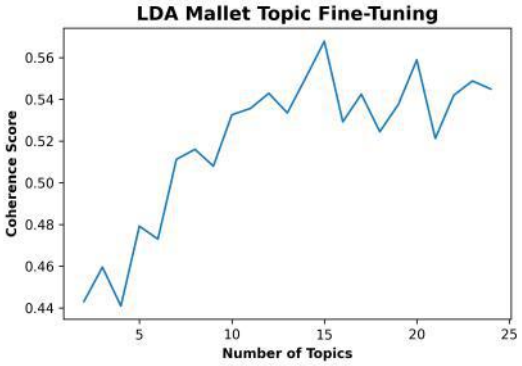


Fig. 5: Phase 2 T tuning of LDA model. 25 models are created with T values ranging from 1 to 25, and a coherence score is calculated for each model. We find that a T value of 15 yields the highest coherence score.

4.5 Tweet Classification

We first utilize the Gensim library to train a Word2vec model on the tweet corpus, generating word embeddings for each tweet. As per convention, a dimension size of 100 is chosen for the word embeddings [15]. To determine the optimal classification model, we train five models on the training corpus using the Scikit-Learn library's: 1) random forest ensemble model, 2) Naive-Bayes model, 3) K-nearest neighbors model, 4) logistic regression model, and 5) support vector machine. The random forest model is optimized by fine-tuning $n\text{–}estimators$, the number of trees in the random forest ensemble, with an $n\text{–}estimator$ value of 7 being chosen (Fig. 6). The K-nearest neighbors model is optimized by fine-tuning k, the number of nearest neighbors, with a k value of 4 being chosen (Fig. 7). The Naive-Bayes model, logistic regression model, and support vector machine model do not require hyperparameter tuning.

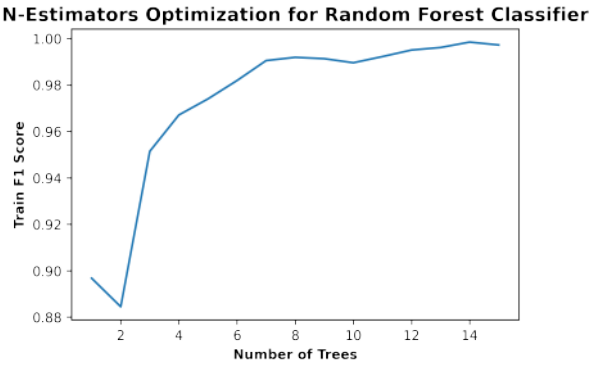


Fig. 6: $N - estimators$ tuning for random forest classifier. A macro-averaged F_1 score (calculated on the training corpus) is obtained for each $n - estimator$. We find an $n - estimator$ value of 7 to be optimal, with values greater than 7 having minimal effects on the F1 score, and thereby potentially over-fitting on the training corpus.

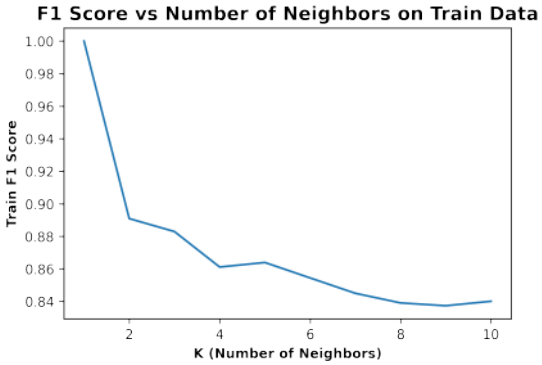


Fig. 7: K hyperparameter tuning for K-Nearest Neighbors classifier. A macro-averaged F_1 score (calculated on the training corpus) is obtained for each k value. Using the elbow methodology [18], we find a k value of 4 to be optimal.

We determine the optimal classification model by calculating a macro-averaged F_1 score from the predicted labels each model generates on the training corpus and the true labels of the training corpus. Confusion matrices for each classification model are found in Appendix B. We find that the random forest model is the optimal model, generating the highest F_1 score of 0.990 (Table 1).

Model	**F_1**
K-Nearest Neighbors	0.869
Logistic Regression	0.827
Naive-Bayes	0.701
Random Forest	0.990
Support Vector Machine	0.827

Tab. 1: Macro-averaged F_1 scores for each classification model, calculated from the predicted tweet labels of the training corpus and the true labels of the training corpus.

4.6 Regression Analysis

We gather epidemiological data from March 1, 2020 to July 18, 2020, from the COVID Tracking Project, an API containing information on COVID-19 statistics from each U.S. state [19]. We use each

state's cumulative case count, cumulative hospitalization count, and cumulative death count, as of July 18, 2020, to calculate the prevalence, hospitalization rate, and mortality rate, respectively. Per the Centers for Disease Control and Prevention's definitions of epidemiology measures, prevalence is defined as $\frac{\text{Cumulative Case Count}}{\text{Population Size}} \times 100$. Hospitalization rate is defined as $\frac{\text{Cumulative Hospitalization Count}}{\text{Population Size}} \times 1000$. Mortality rate is defined as $\frac{\text{Cumulative Death Count}}{\text{Population Size}} \times 1000$ [20]. To analyze a state's policy severity, we define a numerical scale for tweet severity: $severe$ tweets are assigned a value of 2, $moderate$ tweets are assigned a value of 1, and $not\ severe$ tweets are assigned a value of 0.

We complete a regression analysis for four sets of the data: 1) cumulative policy severity and COVID-19 measures for all 50 states, 2) cumulative policy severity and COVID-19 measures for the 20 U.S. states with the largest population density, 3) cumulative policy severity and COVID-19 measures for states with at least 100 anti-contagion policy related tweets (totaling to 18 states), and 4) monthly policy severity and COVID-19 measures for states with at least 100 anti-contagion policy related tweets.

4.6.1 Cumulative Policy Severity and COVID-19 Measures For 50 States

We first analyze cumulative data from all 50 states to investigate the overall relation of policy severity assign every state a mean policy severity score by averaging the severity values of each state's tweets. We then calculate the strength of a correlation between a state's policy severity and prevalence, hospitalization rate, and mortality rate by fitting a least squares regression line on the data, then calculating the coefficient of determination and Pearson correlation coefficient of the regression. A regression with hospitalization rates with rates of zero removed is also performed since some rates of zero are possibly due to the state having less COVID-19 testing kits; the COVID Tracking Project defines hospitalization counts as the number of patients who have tested positive for COVID-19, and it's possible that certain states have reported lower hospitalization counts due to a lower availability or delay of testing results [19].

4.6.2 Cumulative Policy Severity and COVID-19 Measures For 20 States With Highest Densities

To eliminate the potential effect of state density on policy severity, thereby affecting the correlation between policy severity and COVID-19 measures, we complete linear regression analyses on the 20 U.S. states with the highest population densities. We calculate the strength of a correlation between policy severity and prevalence, hospitalization rate, and mortality rate.

4.6.3 Cumulative Policy Severity and COVID-19 Measures For States With Highest Tweet Volume

To eliminate the potential inaccuracy of a state's policy severity due to low tweet volumes, we perform regression on states with a tweet volume of 100 or more anti-contagion policy tweets, accounting for 18 total states (see Appendix C for individual state tweet counts). We calculate the strength of a correlation between policy severity and prevalence, hospitalization rate, and mortality rate.

4.6.4 Monthly Policy Severity and COVID-19 Measures For States With Highest Tweet Volume

We examine the relation between changes in policy severity and COVID-19 measures by analyzing tweet severity on a monthly scale. To reduce potential inaccuracies due to low tweet volumes, we analyze only states with a tweet volume of 100 or more tweets. We divide the tweets (spanning from March 1, 2020 to July 18, 2020) into five groups of 28 days each: 1) March 1 to March 28, 2) March 29 to April 25, 3) April 26 to May 23, 4) May 24 to June 20, and 5) June 21 to July 28. We calculate the average policy severity score, prevalence, hospitalization rate, and death rate for each state during each time frame. The data for the 5 time frames is then combined for regression to be performed, and the coefficient of determination and Pearson correlation coefficient is calculated.

5 Results

5.1 U.S. State Policy Severity

Figure 8 shows the mean anti-contagion policy severity score for U.S. states, rounded to four decimal places. Of all 50 states, Michigan has the highest policy severity score of 0.7037, while Kentucky has the lowest policy severity score of 0.1053. Table 2 shows the total number of tweets belonging to each severity class.

Severity Class	Number of Tweets
Not Severe	4,213
Moderate	1,244
Severe	953

Tab. 2: Total number of tweets in each severity class.

Of the 6,410 tweets in the tweet corpus, 66% of the tweets are classified as not severe, 19% of the tweets are classified as moderate, and 15% of the tweets are classified as severe. Specific counts of tweets by individual states are found in Appendix C.

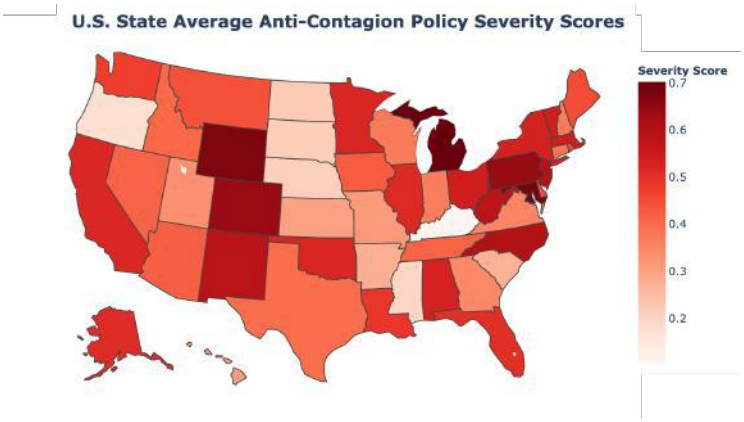


Fig. 8: Mean anti-contagion policy severity score from March 1, 2020 to July 18, 2020, calculated for U.S. states.

5.2 Policy Severity and COVID-19 Epidemiological Data Correlation

5.2.1 Cumulative Policy Severity and COVID-19 Measures For 50 States

Linear regression is performed on the average policy severity of each U.S. state versus the prevalence, hospitalization rate, and mortality rate (Fig. 9, 10, 11, 12). For hospitalization rate, linear regression is first performed for all 50 states, then performed on states with hospitalization rates greater than zero. Table 3 shows the Pearson's correlation coefficient for each linear regression.

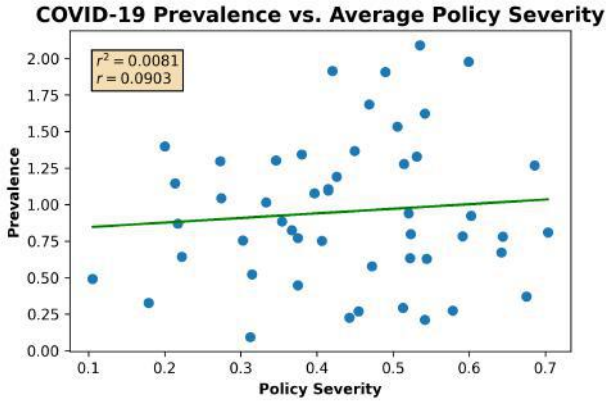


Fig. 9: COVID-19 prevalence vs. average policy severity for all U.S. states. Prevalence is calculated from the cumulative number of cases as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

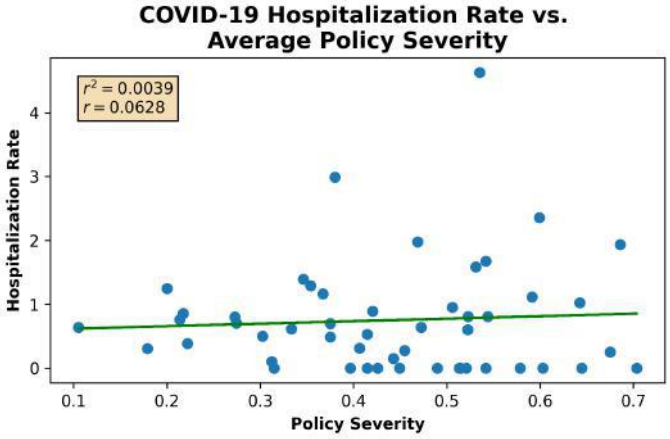


Fig. 10: COVID-19 hospitalization rate vs. average policy severity for all U.S. states. Hospitalization rate is calculated from the cumulative number of hospitalizations as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

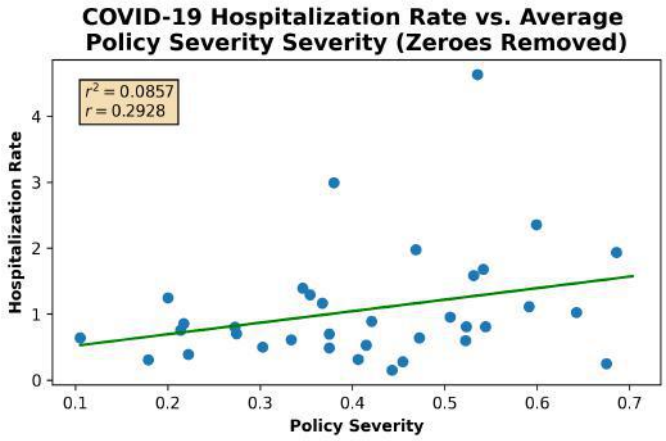


Fig. 11: COVID-19 hospitalization rate vs. average policy severity for all U.S. states, with hospitalization rates of zero removed. Hospitalization rate is calculated from the cumulative number of hospitalizations as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

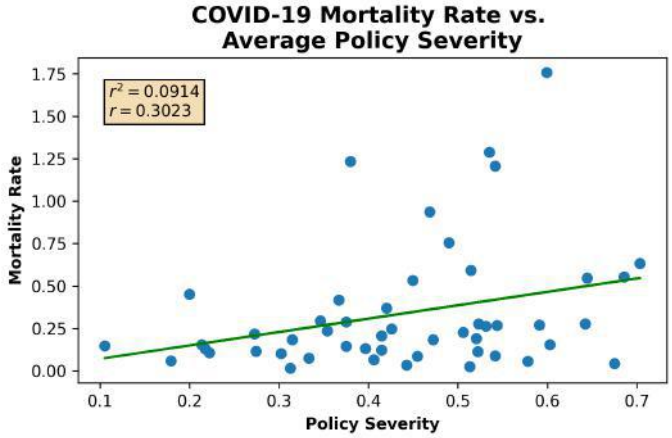


Fig. 12: COVID-19 mortality rate vs. average policy severity for all U.S. states. Mortality rate is calculated from the cumulative number of deaths as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

Linear Regression Model	**Pearson's r**
Prevalence vs. Severity	0.0903
Hospitalization Rate vs. Severity	0.0628
Hospitalization Rate vs. Severity (Zeroes Removed)	0.2928
Mortality Rate vs. Severity	0.3023

Tab. 3: Pearson correlation coefficients (r) for each linear regression model.

All four regressions display positive correlation coefficients. Prevalence, as well as hospitalization rate, have correlation coefficients around 0.08. Hospitalization rate with rates of zero removed, as well as mortality rate, have correlation coefficients around 0.3.

5.2.2 Cumulative Policy Severity and COVID-19 Measures For 20 States With Highest Densities

Linear regression is performed on the average policy severity of the U.S. states with the 20 highest population densities, versus the prevalence, hospitalization rate, and mortality rate (Fig. 13, 14, 15). Table 4 shows the Pearson's correlation coefficient for each linear regression.

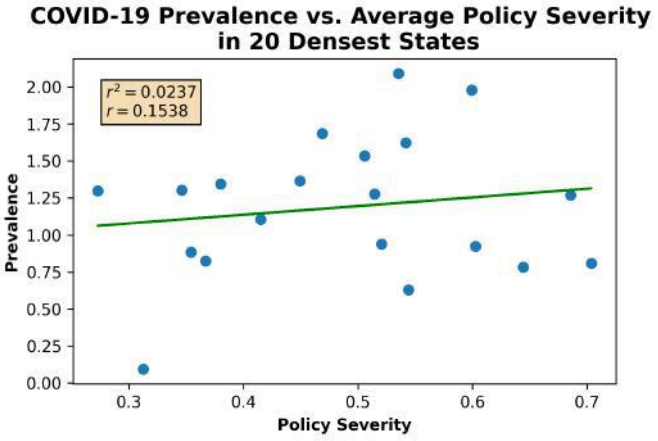


Fig. 13: COVID-19 prevalence vs. average policy severity for U.S. states with 20 highest population densities. Prevalence is calculated from the cumulative number of cases as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1st, 2020 to July 18, 2020.

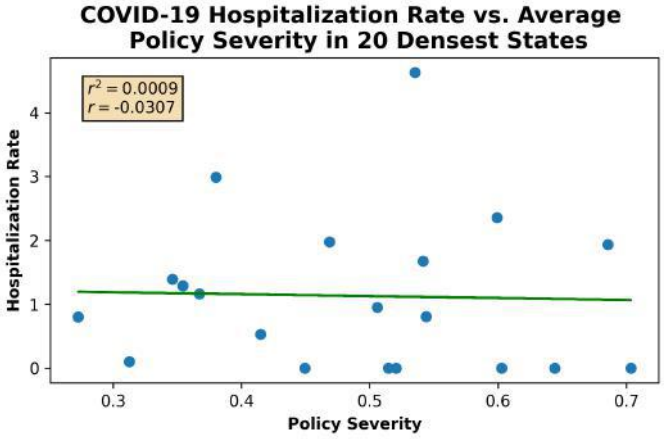


Fig. 14: COVID-19 hospitalization rate vs. average policy severity for U.S. states with 20 highest population densities. Hospitalization rate is calculated from the cumulative number of hospitalizations as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

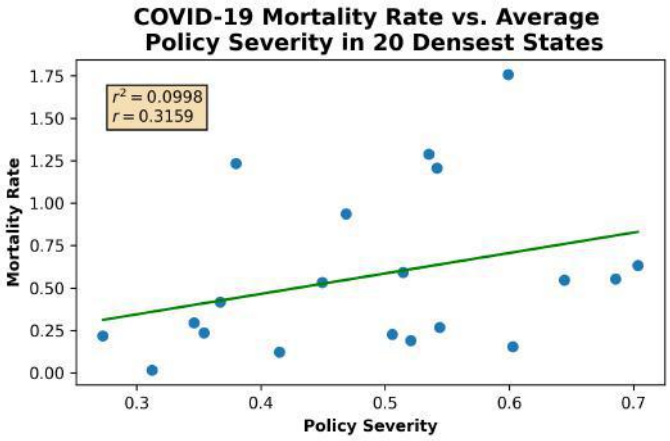


Fig. 15: COVID-19 mortality rate vs. average policy severity for U.S. states with 20 highest population densities. Mortality rate is calculated from the cumulative number of deaths as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

Linear Regression Model	**Pearson's *r***
Prevalence vs. Severity	0.1538
Hospitalization Rate vs. Severity	-0.0307
Mortality Rate vs. Severity	0.3159

Tab. 4: Pearson correlation coefficients (r) for each linear regression model.

The regressions for prevalence and mortality display positive correlation coefficients, while the regression for hospitalization rate displays a negative correlation coefficient. Prevalence and mortality rate exhibit weak correlations with policy severity, and hospitalization rate exhibits no correlation with policy severity.

5.2.3 Cumulative Policy Severity and COVID-19 Measures For States With Highest Tweet Volume

Linear regression is performed on the average policy severity of the U.S. states with tweet volumes of 100 or more tweets, versus the prevalence, hospitalization rate, and mortality rate (Fig. 16, 17, 18). Table 5 shows the Pearson's correlation coefficient for each linear regression.

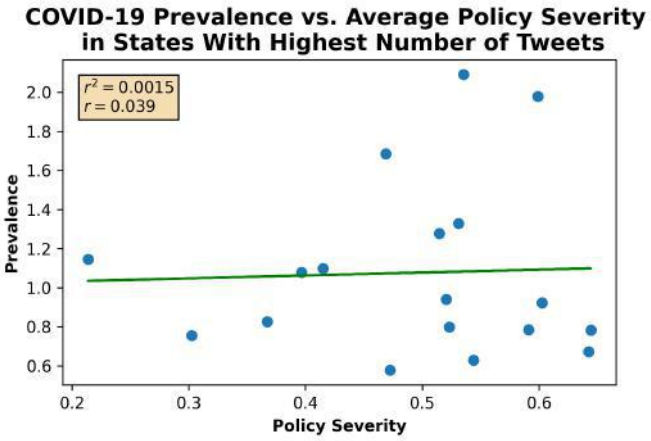


Fig. 16: COVID-19 prevalence vs. average policy severity for U.S. states with tweet volumes of 100 or more tweets. Prevalence is calculated from the cumulative number of cases as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

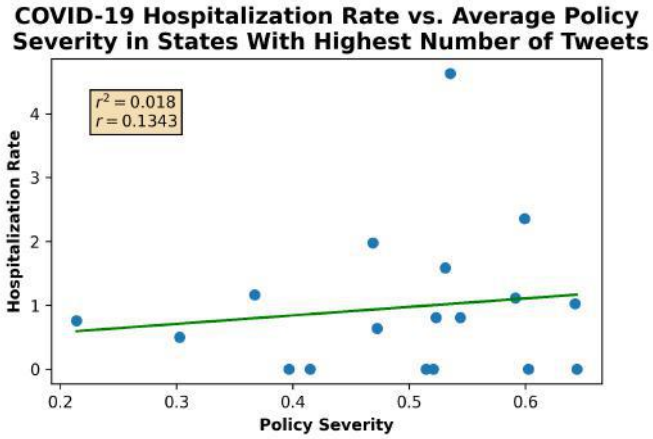


Fig. 17: COVID-19 hospitalization rate vs. average policy severity for U.S. states with tweet volumes of 100 or more tweets. Hospitalization rate is calculated from the cumulative number of hospitalizations as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

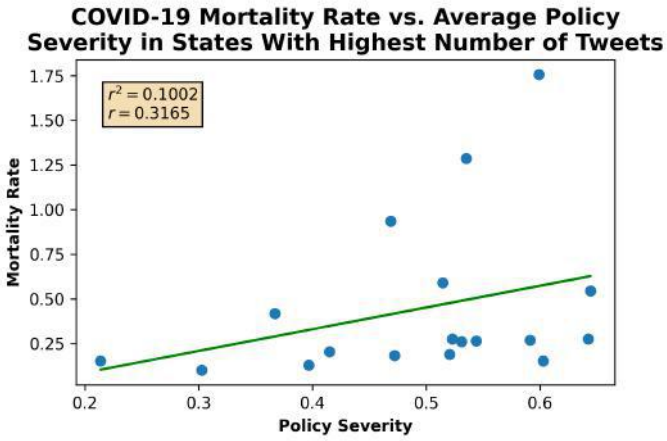


Fig. 18: COVID-19 mortality rate vs. average policy severity for U.S. states with tweet volumes of 100 or more tweets. Mortality rate is calculated from the cumulative number of deaths as of July 18, 2020. Severity is calculated as the mean tweet severity of anti-contagion policy tweets from March 1, 2020 to July 18, 2020.

Linear Regression Model	Pearson's r
Prevalence vs. Severity	0.0390
Hospitalization Rate vs. Severity	0.1343
Mortality Rate vs. Severity	0.3165

Tab. 5: Pearson correlation coefficients (r) for each linear regression model.

All three regressions display positive correlation coefficients. Hospitalization rate and mortality rate exhibit weak correlations with policy severity, and prevalence exhibits no correlation with policy severity.

5.2.4 Monthly Policy Severity and COVID-19 Measures For States With Highest Tweet Volume

Linear regression is performed on the monthly average policy severity of the U.S. states with tweet volumes of 100 or more tweets, versus the prevalence, hospitalization rate, and mortality rate (Fig. 19, 20, 21). Table 6 shows the Pearson's correlation coefficient for each linear regression.

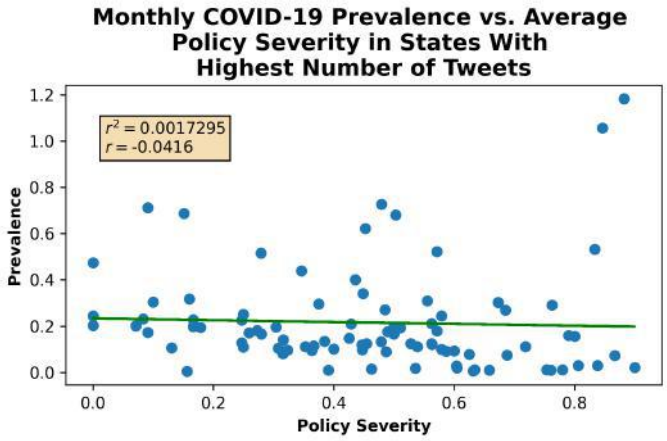


Fig. 19: COVID-19 prevalence vs. monthly average policy severity for U.S. states with tweet volumes of 100 or more tweets. Prevalence is calculated as the cumulative number of cases per 28 days. Severity is calculated as the mean tweet severity of anti-contagion policy tweets per 28 days.

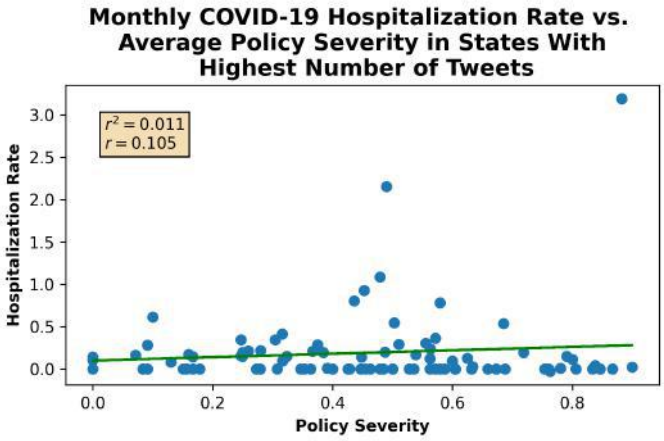


Fig. 20: COVID-19 hospitalization rate vs. monthly average policy severity for U.S. states with tweet volumes of 100 or more tweets. Hospitalization rate is calculated from the cumulative number of hospitalizations per 28 days. Severity is calculated as the mean tweet severity of anti-contagion policy tweets per 28 days.

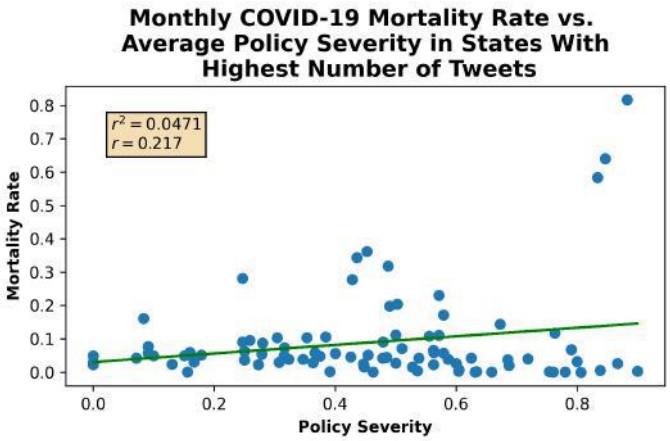


Fig. 21: COVID-19 mortality rate vs. monthly average policy severity for U.S. states with tweet volumes of 100 or more tweets. Mortality rate is calculated from the cumulative number of deaths per 28 days. Severity is calculated as the mean tweet severity of anti-contagion policy tweets per 28 days.

Linear Regression Model	**Pearson's r**
Prevalence vs. Severity	-0.0416
Hospitalization Rate vs. Severity	0.1050
Mortality Rate vs. Severity	0.2170

Tab. 6: Pearson correlation coefficients (r) for each linear regression model.

The regressions for hospitalization rate and mortality rate display positive correlation coefficients, while the regression for prevalence displays a negative correlation coefficient. Hospitalization rate and mortality rate exhibit weak correlations with policy severity, and prevalence exhibits no correlation with policy severity.

6 Discussion

In our first set of regressions, we analyze the average policy severity and COVID-19 measures of all 50 U.S. states. For these analyses, both the regression of prevalence against severity, as well as the regression of hospitalization rate against severity, display correlation coefficients less than 0.10, indicating that there is no linear relation between these two variables and severity. This is perhaps because while some smaller, sparser states with a low amount of cases tend to implement less severe policies due to their lack of need for stricter policies, larger states that also have less severe policies might have much higher case counts and hospitalization rates. Additionally, the lack of correlation between the hospitalization rates and severity for all 50 states can possibly be attributed to variations in the quality of healthcare systems and COVID-19 testing availability for each state. Several states with varying levels of policy severity all exhibit hospitalization rates close to zero, which may be due to these states having less hospital availability or having less COVID-19 testing kits; the COVID Tracking Project defines hospitalization counts as the number of patients who have tested positive for COVID-19, so it's possible that certain states have reported lower hospitalization counts due to a lower availability of testing results [19].

Once hospitalization rates of zero are removed, the regression between the remaining hospitalization rates and policy severity has a correlation coefficient of 0.30, indicating a moderately weak, positive correlation. Since the severity score used in this regression is taken as an average of all severity scores over a 4 month time frame, and the hospitalization rate is taken over the cumulative hospitalizations, it is possible that states that started with less severe policies experienced many hospitalizations in the first months of the pandemic, thereby increasing their hospitalization rate, and as a result, implemented stricter anti-contagion policies, raising their severity score. The regression between mortality rate and policy severity has a correlation coefficient of 0.29, indicating a moderately weak, positive correlation. Surprisingly, the positive correlation between mortality rate and policy severity is opposite the findings of Dergiades et al., who found that increases in policy severity lead to a decrease in mortality rates [7]. One potential explanation is that while Dergiades et al. conducted time-series analyses of mortality rates and policy severity, this regression did not account for changes in policy severity and mortality rate over time. Similar to the regression of hospitalization rates (with zeros removed), a possibility is that states which originally had less severe policies experienced high death rates in the first months of the pandemic, and increased their anti-contagion policy severity in response, raising their average severity score.

In our second set of regressions, we analyze the average policy severity and COVID-19 measures of the 20 U.S. states with the highest population densities. By analyzing only states with the highest population densities, we minimize inconsistencies in tweet severity caused by variations in population density. For these analyses, the regression of hospitalization rate against severity displays a negative correlation coefficient of -0.0307 with an absolute value less than 0.01, indicating that there appears to be no linear relation between hospitalization rate and severity for these states. As mentioned previously, the lack of correlation is potentially due to variations in testing availability and inconsistencies among states in defining COVID-19 hospitalization counts.

The regressions for prevalence and mortality rate exhibit correlation coefficients of 0.15 and 0.32, respectively, indicating there exists a moderately weak, positive relationship between prevalence and mortality rate with policy severity. The positive correlations are at odds with the findings of Dergiades et al. and Hsiang et al. [7, 3], and as mentioned previously, a potential explanation is that these regressions did not account for changes in policy severity over time, causing the trends in the data to differ from the studies of Dergiades et al. and Hsiang et al.

In the third set of regressions, we analyze the average policy severity and COVID-19 measures of U.S. states with tweet volumes of at least 100 tweets in order to reduce inaccurate policy severity values caused by low tweet volumes. In these analyses, the regression of prevalence against severity displays a correlation coefficient less than 0.0390, indicating no linear relationship between tweet severity and prevalence. The regressions of hospitalization rate and mortality rate against policy severity display positive, moderately weak correlation coefficients of 0.13 and 0.32, respectively. Again, these trends contrast with the findings of previous studies, and this may be attributed to the fact that this set of regressions does not account for the effects that variations in population density may have on policy severity.

In the last set of regressions, we analyze the monthly policy severity of the U.S. states with tweet volumes of 100 or more tweets. By performing regression on the monthly average policy severity rather

than the overall mean policy severity, we can evaluate the impacts that changes in policy may have on COVID-19 epidemiological data. We find that the regression of prevalence against severity displays a negative correlation coefficient of -0.04, signifying that there is no linear relationship between prevalence and severity. Similar to the previous regressions, this is perhaps due to the effect that state density may have on prevalence: states with lower densities experience low person-to-person to contact and therefore low disease transmission, and these states may implement less stringent anti-contagion policies. At the same time, some larger states with higher prevalence possibly do not implement stringent policies either, and this causes the regression to appear as if there is no correlation between prevalence and policy severity.

The regressions of hospitalization rate and mortality rate exhibit correlation coefficients of 0.11 and 0.22, indicating that these two variables have positive, weak correlations with policy severity. Again, these results contrast from the studies of Dergiades et al. and Hsiang et al., who found that higher policy stringencies have a negative correlation with mortality and hospitalization rate. One probable explanation for these results is that although the data used in this set of regressions was calculated over monthly intervals, Dergiades et al. and Hsiang et al.'s studies analyzed COVID-19 data on a daily scale, allowing the studies to have a higher degree of specificity and a more accurate reflection of COVID-19 trends. Furthermore, since our regressions were performed on monthly data, it is possible that states which experienced high hospitalization rates or mortality rates in a month implemented more stringent COVID-19 policies, and once these rates decreased, the states lowered the stringency of the anti-contagion policies, decreasing the monthly policy severity score and thereby resulting in a positively correlated trend.

Finally, it is probable that each calculated severity score is not reflective of the state's policy severity, leading to unrepresentative regression results. This can be attributed to several reasons, including that low tweet volumes were obtained for many states, and government handles could potentially be inclined to use more positive language when communicating through Twitter, causing the tweet severity score to inaccurately represent the state's stringency of anti-contagion policies.

7 Conclusion

In this work, we present a method to analyze U.S. anti-contagion policy severity from U.S. state government Twitter tweets based on a random forest classification approach. We show that our method can be used to classify a state's guideline severity and quantify the relation of guideline severity with COVID-19 epidemiological data on a state-level scope. We measure the correlation strength between tweet severity and COVID-19 prevalence, hospitalization rate, and death rate for U.S. states, showing that policy severity and COVID-19 epidemiology data have moderately weak, positive correlations. These are preliminary results, and more analysis on patterns between policy severity and epidemiological data is necessary to predict the effect of policy severity on COVID-19 measures. Through additional time-series modeling of policy severity and COVID-19 data, as well as other influential factors such as testing availability, infection and mortality measures can be predicted from various anti-contagion policies. If the efficacy of U.S. anti-contagion policies can be modeled, governments on local, state, and even national levels can use such findings to implement strategically designed approaches to combating the COVID-19 pandemic.

References

[1] World Health Organization. Novel Coronavirus (2019-nCoV): situation report, 19. Technical documents, 2020.

[2] Centers for Disease Control and Prevention, 2020. Coronavirus Disease 2019 (COVID-19).

[3] Solomon Hsiang, Daniel Allen, Sébastien Annan-Phan, Kendon Bell, Ian Bolliger, Trinetta Chong, Hannah Druckenmiller, Luna Yue Huang, Andrew Hultgren, Emma Krasovich, Peiley Lau, Jaecheol Lee, Esther Rolf, Jeanette Tseng, and Tiffany Wu. The effect of large-scale anti-contagion policies on the COVID-19 pandemic. *Nature*, Jun 2020.

[4] Catrin Sohrabi, Zaid Alsafi, Niamh O'Neill, Mehdi Khan, Ahmed Kerwan, Ahmed Al-Jabir, Christos Iosifidis, and Riaz Agha. World health organization declares global emergency: A review of the 2019 novel coronavirus (covid-19). *International Journal of Surgery*, 76:71 – 76, 2020.

[5] Priyanka and Vicky Verma. Study of lockdown/testing mitigation strategies on stochastic SIR model and its comparison with South Korea, Germany and New York data. *arXiv e-prints*, page arXiv:2006.14373, June 2020.

[6] Giulia Giordano, Franco Blanchini, Raffaele Bruno, Patrizio Colaneri, Alessandro Di Filippo, Angela Di Matteo, and Marta Colaneri. Modelling the COVID-19 epidemic and implementation of population-wide interventions in Italy. *Nature Medicine*, 26, Jun 2020.

[7] Theologos Dergiades, Costas Milas, Elias Mossialos, and Theodore Panagiotidis. Effectiveness of Government Policies in Response to the COVID-19 Outbreak, 2020. Available at SSRN: http://dx.doi.org/10.2139/ssrn.3602004.

[8] Catherine Ordun, Sanjay Purushotham, and Edward Raff. Exploratory Analysis of Covid-19 Tweets using Topic Modeling, UMAP, and DiGraphs. *arXiv e-prints*, page arXiv:2005.03082, May 2020.

[9] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3(null):993–1022, March 2003.

[10] Liangjie Hong and Brian D. Davison. Empirical Study of Topic Modeling in Twitter. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, page 80–88, New York, NY, USA, 2010. Association for Computing Machinery.

[11] Malek Hajjem and Chiraz Latiri. Combining IR and LDA Topic Modeling for Filtering Microblogs. *Procedia Computer Science*, 112:761 – 770, 2017. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France.

[12] Emmanouil Ikonomakis, Sotiris Kotsiantis, and V. Tampakas. Text Classification Using Machine Learning Techniques. *WSEAS transactions on computers*, 4:966–974, 08 2005.

[13] Fabrizio Sebastiani. Machine Learning in Automated Text Categorization. *ACM Comput. Surv.*, 34(1):1–47, March 2002.

[14] Zitao Liu, Wenchao Yu, Wei Chen, Shuran Wang, and Fengyi Wu. Short Text Feature Selection for Micro-Blog Mining. pages 1 – 4, 01 2011.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

[16] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, 2013.

[17] Thomas Hale, Noam Angrist, Beatriz Kira, Anna Petherick, Toby Phillips, and Samuel Webster. Variation in Government Responses to COVID-19 Version 6.0. *Blavatnik School of Government Working Paper*. Available: www.bsg.ox.ac.uk/covidtracker.

[18] Purnima Bholowalia and Arvind Kumar. Article: EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN. *International Journal of Computer Applications*, 105(9):17–24, November 2014. Full text available.

[19] The COVID Tracking Project.

[20] Principles of Epidemiology, May 2012.

Appendix A

State	Twitter Account
Alabama	ALPublicHealth
Alaska	Alaska_DHSS
Arizona	AZDHS
Arkansas	ADHPIO
California	CAPublicHealth
Colorado	CDPHE
Connecticut	CTDPH
Delaware	Delaware_DHSS
Florida	HealthyFla
Georgia	GaDPH
Hawaii	HIgov_Health
Idaho	IDHW
Illinois	IDPH
Indiana	StateHealthIN
Iowa	IAPublicHealth
Kansas	KDHE
Kentucky	KYHealthAlerts
Louisiana	LADeptHealth
Maine	MEPublicHealth
Maryland	MDHealthDept
Massachusetts	MassDPH
Michigan	MichiganHHS
Minnesota	mnhealth
Mississippi	msdh
Missouri	HealthyLivingMo
Montana	GovernorBullock
Nebraska	NEDHHS
Nevada	NVHealthRespon1
New Hampshire	NHPubHealth
New Jersey	NJDeptofHealth
New Mexico	NMDOH
New York	HealthNYGov
North Carolina	ncdhhs
North Dakota	NDDOH
Ohio	OHdeptofhealth
Oklahoma	HealthyOklahoma
Oregon	OHAOregon
Pennsylvania	PAHealthDept
Rhode Island	RIHEALTH
South Carolina	scdhec
South Dakota	SDDOH
Tennessee	TNDeptofHealth
Texas	TexasDSHS
Utah	UtahDepOfHealth
Vermont	healthvermont
Virginia	VDHgov
Washington	WADeptHealth
West Virginia	WV_DHHR
Wisconsin	DHSWI
Wyoming	GovernorGordon

(a) **U.S. State Twitter Handles**

Stopwords	
a	on
about	once
above	only
after	or
again	other
against	our
am	ours
an	ourselves
and	out
any	over
are	own
as	pron
at	same
be	she
because	she's
been	so
before	some
being	such
below	than
between	that
both	the
but	their
by	theirs
can	them
down	themselves
during	then
each	there
few	these
for	they
from	this
further	those
he	through
her	to
here	too
hers	under
herself	until
him	up
himself	very
his	was
how	we
i	were
if	what
in	when
into	where
is	which
it	while
it's	who
its	whom
itself	why
just	will
me	with
my	you
myself	your
now	yours
of	yourself
off	yourselves

(b) **Stopword List**

Appendix B

Topic Number	Keywords	Number of Tweets	Included
0	test, site, covid, free, find	1667	Y
1	county, release, covid, phase, test	608	Y
2	hand, covid, wash, clean, spread	1083	Y
3	covid, symptom, test, people, home	1734	Y
4	food, supply, covid, donate, state	1510	Y
5	office, apply, online, eek, state	419	N
6	family, day, stayhomefornevada, time, lose	1860	N
7	covid, information, contact, question, health	2685	N
8	covid, live, update, watch, today	2607	N
9	test, report, covid, total, case	1276	N
10	covid, health, mental, support, child	2066	N
11	case, covid, datum, update, report	2320	N
12	school, wic, food, learn, student	976	N
13	wear, mask, face, covid, stay	3179	Y
14	risk, learn, condition, high, covid	1253	N
15	order, business, guidance, open, today	2483	Y
16	health, covid, work, public, worker	2079	N
17	stay, safe, tip, time, learn	1569	N
18	care, health, facility, child, covid	1417	N
19	case, covid, death, total, report	2326	N
20	covid, fund, census, business, federal	1183	N
21	covid, visit, patient, positive, information	420	N
22	covid, spread, state, continue, work	2955	Y
23	hospital, patient, covid, care, ventilator	652	N

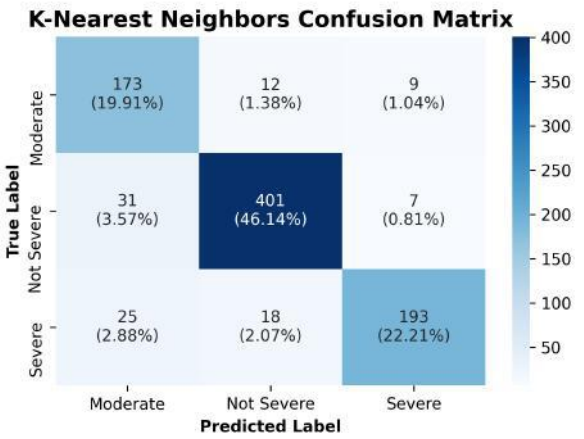
(c) **24 Topic Latent Dirichlet Allocation: List of Topics**

Topic Number	Keywords	Number of Tweets	Included
0	hand, wash, covid, clean, spread	1013	N
1	supply, ppe, mask, state, care	805	N
2	covid, case, spread, people, test	1146	N
3	business, essential, store, employee, service	890	Y
4	phase, reopen, plan, county, today	696	N
5	guidance, close, gathering, event, school	819	Y
6	covid, work, state, health, time	1578	N
7	stay, home, covid, spread, live	1157	Y
8	food, blood, covid, donate, bank	645	N
9	test, county, covid, free, health	699	N
10	test, site, covid, find, location	1246	N
11	wear, face, mask, cover, cloth	1455	Y
12	order, health, covid, state, travel	1013	Y
13	distance, social, stay, practice, covid	1076	Y
14	covid, symptom, provider, care, test	981	N

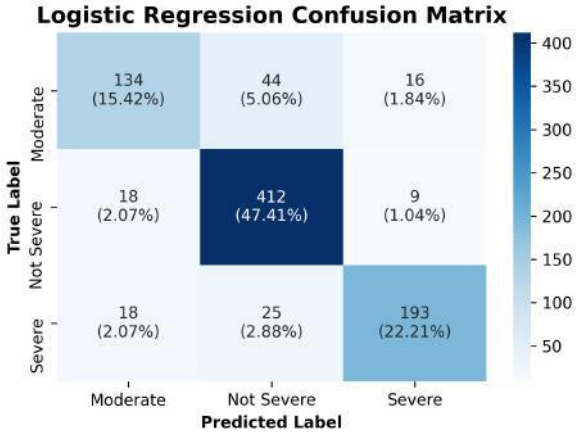
(d) **15 Topic Latent Dirichlet Allocation: List of Topics**

Severe	Moderate	Not Severe
critical	avoid	good idea
immediately	stay home	urge
immediate	do not	can take
right away	restrict	can help
a soon a possible	serious	recommend
must	seriously	encourage
require	minimize	discuss
need	vital	low risk
crucial	necessary	risk be low
terrible	essential	low risk
always	must	no risk
mandatory	close	will evaluate
order	always	evaluate
enforce	close down	fully prepare
prohibit	requisite	reconsider
stay at home order	prevent	limit travel
shut down	not optional	consider
shutdown		keep in mind
executive order		suggest
mandate		when possible
impose		if possible
dire		try to
ordain		permit
ban		volunteer
requirement		not require
		not enforce
		promote
		advise
		advocate
		discourage
		not close
		remain open
		optional

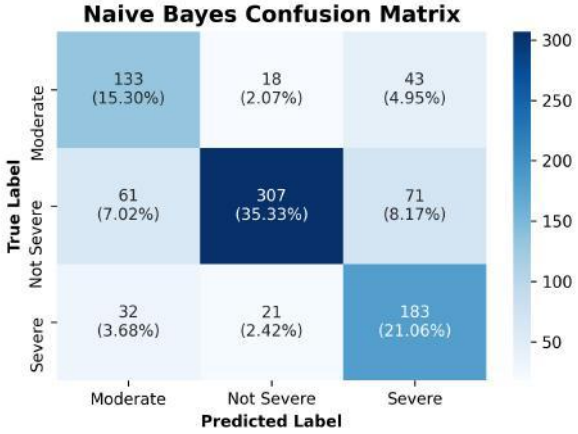
(e) **Severity Keywords**



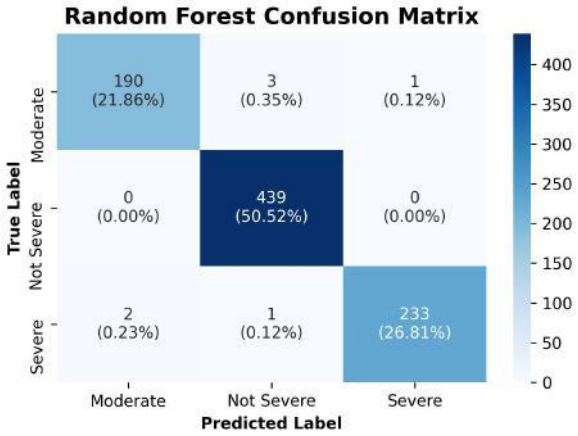
(f) **K-Nearest Neighbors Confusion Matrix**



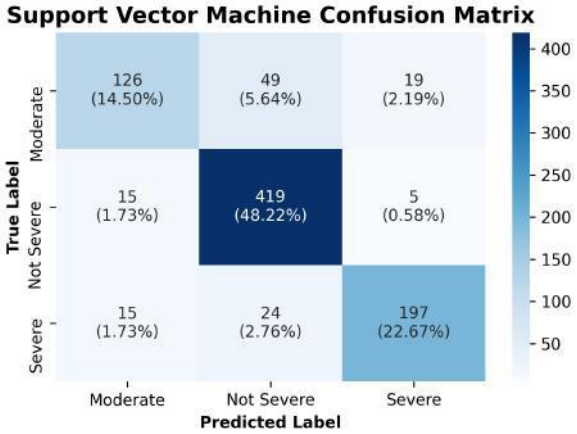
(g) **Logistic Regression Confusion Matrix**



(h) **Naive-Bayes Confusion Matrix**



(i) **Random Forest Confusion Matrix**



(j) **Support Vector Machine Confusion Matrix**

Appendix C

State	Severe Tweets	Moderate Tweets	Not Severe Tweets	Tweet Count
Alabama	53	22	166	241
Alaska	16	7	53	76
Arizona	7	23	58	88
Arkansas	6	2	43	51
California	26	49	119	194
Colorado	50	24	119	193
Connecticut	7	5	38	50
Delaware	13	14	62	89
Florida	16	12	59	87
Georgia	4	10	38	52
Hawaii	3	4	25	32
Idaho	9	8	47	64
Illinois	17	54	100	171
Indiana	22	14	122	158
Iowa	5	13	36	54
Kansas	14	8	97	119
Kentucky	0	2	17	19
Louisiana	14	20	64	98
Maine	2	1	8	11
Maryland	20	8	42	70
Massachusetts	2	9	13	24
Michigan	17	23	41	81
Minnesota	24	43	107	174
Mississippi	1	1	13	15
Missouri	7	3	44	54
Montana	3	21	37	61
Nebraska	10	8	113	131
Nevada	22	34	132	188
New Hampshire	10	4	50	64
New Jersey	48	43	141	232
New Mexico	21	78	104	203
New York	53	84	218	355
North Carolina	73	124	251	448
North Dakota	3	6	45	54
Ohio	70	64	241	375
Oklahoma	15	17	58	90
Oregon	3	6	58	67
Pennsylvania	52	146	190	388
Rhode Island	88	71	368	527
South Carolina	7	10	71	88
South Dakota	2	11	56	69
Tennessee	6	27	61	94
Texas	13	22	86	121
Utah	0	4	8	12
Vermont	11	17	44	72
Virginia	5	7	36	48
Washington	43	42	186	271
West Virginia	22	4	57	83
Wisconsin	7	10	47	64
Wyoming	11	5	24	40

(k) **State Tweet Counts By Severity Category**

Interpretability in Deep Learning Models Used to Classify Building Damage in Satellite Imagery

Thomas Chen

Abstract

Natural disasters ravage the world's cities, valleys, and shores on a monthly basis. Having precise and efficient mechanisms for assessing infrastructure damage is essential to channel resources and minimize the loss of life. Using a dataset that includes labeled pre- and post- disaster satellite imagery, we train multiple convolutional neural networks to assess building damage on a per-building basis. In order to investigate how to best classify building damage, we present a highly interpretable deep-learning methodology that seeks to explicitly convey the most useful information required to train an accurate classification model. We also delve into which loss functions best optimize these models. Our findings include that ordinal-cross entropy loss is the most optimal loss function to use, as well as that including the type of disaster that caused the damage in combination with a pre- and post-disaster image best predicts the level of damage caused. Future work can build on our approach of interpretability by studying more input types. Our research seeks to computationally contribute to aiding in this ongoing and growing humanitarian crisis, heightened by climate change.

1 Introduction

Natural disasters devastate countless vulnerable communities and countries every decade. They are responsible for the deaths of 60,000 people a year worldwide, on average [24]. The timely allocation of resources in the event of these tragedies are crucial to saving lives. Additionally, natural disasters cause varying levels of damage to buildings. The havoc wreaked by them causes widespread infrastructure damage, in some cases leading to a "cascade effect" [17]. The resulting economic impact is colossal. For example, since 1980, the United States has sustained 273 weather and climate disasters that have caused damages exceeding 1 billion US dollars (USD), totalling 1.79 trillion USD [7]. Unfortunately, the frequency and severity of these disasters will only continue to increase, exacerbated by climate change [30]. The catastrophic impact of natural disasters and their increasing prevalence motivates the problem addressed in this work.

In order to prepare for and recover from these terrible but inevitable events, robust emergency response plans must be in place. This is especially needed in a world where weather and climate disasters are increasing in occurrence and brutality. This requires quickly and accurately analyzed data from the disaster site. Because it is almost always slow and hard (and sometimes impossible) to obtain damage assessment and other details from on the ground in a timely manner, satellite imagery has gained popularity in being used to analyze these types of situations. Deep neural networks (DNNs), which are machine learning algorithms that have multiple layers, have been used to locate and classify building damage within satellite imagery [13, 12, 36, 10]. However, the current literature (see Literature Review) is limited in the interpretability of what exactly these neural networks are learning and what is most useful in assessing building damage.

To better address this problem, we present a novel analysis of the most important information that a deep learning model needs to assess building damage. We use a convolutional neural network (CNN) architecture called a residual neural network (ResNet), pre-trained on Imagenet data [15]. In our approach, we train multiple CNNs on xBD satelite imagery data [11], with different modalities of input, as well as using different loss functions, and compare accuracy on the validation set. We aim to explicitly provide insight into the most effective ways to train models to classify levels of building damage, maximizing the efficiency of the emergency response after a natural disaster, which has the potential to save lives and reduce economic strain.

2 Literature Review

2.1 Image Classification

Image classification is a well-studied area with work ranging from classical techniques [14, 19, 33, 21] to deep learning-based techniques [20, 35, 38]. Such deep-learning techniques have been enabled by ImageNet data [6]. Classification enables many opportunities in applications, such as object classification [28], scene classification [2], and action recognition [32]. In this work, we adapt the common image classification problem to damage assessment of buildings, using labeled satellite imagery.

2.2 Computer Vision for Satellite Imagery

Satellite imagery is useful in a plethora of areas, including for assessment of marine ecology [29], weather forecasting [5], and even in studying and predicting the spread of infectious disease [25]. Computer vision is the study of how computers gain high-level understanding from digital images and video. Used in tandem with satellite imagery, there are many possibilities for applications. Lately, there has been increased interest in using satellite imagery for humanitarian purposes such as responding to natural disasters [22]. Satellite imagery also provides insights for agriculture [37] and urban road damage [18]. More generally, change detection, which is the process of identifying differences in the state of an object by observing it at different times, can be used with satellite imagery in a variety of contexts. There are a few primary categories in which change detection approaches fall: algebra-based, transform-based, and classification-based [1]. Change detection has been employed on satellite imagery to study deforestation [31], urban growth [16], and more. In our case, the change between the pre-disaster image and post-disaster image can help in assessing and classifying building damage.

2.3 Building Damage Assessment

One specific area that has garnered significant attention in computer vision and satellite imagery is building damage assessment. Recent works have studied semantic building segmentation [13, 12] and cross-region transfer learning to avoid overfitting [36]. Furthermore, Gueguen and Hamid present a semi-supervised approach [10]. xView2 recently introduces a dataset, xBD, discussed more in detail later in this work [11]. Many teams [34] competed in the xView2 data competition and improved on the baseline model provided [11]. In our work, we focus on building damage assessment via image classification and change detection. We specifically hone in on what information is most useful in accurate classifications of building damage and analyze which loss functions are most fit for training our models and yield the most precise results. Our primary contribution is to improve upon the interpretability of machine learning models of prior works and existing literature in this area by explicitly examining per-building classification prediction accuracy with different combinations of inputted information and loss functions.

3 Purpose

1. Our first goal is to grasp a solid understanding of the dataset, xBD, that we are working with. This is necessary and important because the remainder of the project, which includes training models, hinges on our insight into various aspects of the dataset and its constituent parts. This goal is achieved through data preprocessing. Moreover, we weed out noisy and/or useless data points.

2. As previously mentioned, current literature on building damage classification with satellite imagery lacks interpretability in what exactly deep neural networks are learning and what kinds of information help them train for the highest accuracy. Our first step in tackling this problem is setting up and training a baseline model, which will take in only post-disaster images as input. The recorded accuracy of the validation set on this model will serve as a baseline for comparisons with other models.

3. We train other models, with different types of input. This will allow us to find which types of information are most useful for the network. We also experiment with various loss functions. We compare the results with those of the baseline model.

4 Methods

4.1 Dataset Details

For this project, we utilize the xBD dataset [11], which consists of 22,068 high-resolution, annotated 1024 by 1024 satellite images for the purpose of building damage detection. This dataset covers a wide range of disasters in fifteen countries around the world (Fig. 2), from Guatemala to Portugal to Indonesia (over 850,736 building polygons totalling an area of 45,361 square kilometers). One of xBD's main purposes is to demonstrate changes between pre-disaster and post-disaster satellite imagery to aid in detecting the damage caused. Therefore, each post-disaster building is labeled as one of the following: "unclassified," "no damage," "minor damage," "major damage," or "destroyed." (We later discard the "unclassified" buildings, as described later in this section). The classification technique utilized is called the Joint Damage Scale (JDS), as presented and described in detail in Figure 1. We use the xBD dataset because it incorporates a variety of disaster types, building types, and geographical locations. This allows for diversity in training the model. For example, the wide variety of geographical locations is important for cross-region generalization. Additionally, the high resolution imagery allows for detailed change detection between pre-disaster and post-disaster images. These factors currently make xBD the leading dataset for building damage detection using labeled satellite imagery [11]. Previous satellite imagery datasets were not as comprehensive, and, for example, had only covered singular disaster types or did not have uniform building damage assessment criteria like the JDS used in xBD [9, 3, 8].

Score	Label	Visual Description of the Structure
0	No damage	Undisturbed. No sign of water, structural damage, shingle damage, or burn marks.
1	Minor damage	Building partially burnt, water surrounding the structure, volcanic flow nearby, roof elements missing, or visible cracks.
2	Major damage	Partial wall or roof collapse, encroaching volcanic flow, or the structure is surrounded by water or mud.
3	Destroyed	Structure is scorched, completely collapsed, partially or completely covered with water or mud, or no longer present.

Fig. 1: Joint Damage Scale descriptions, which are used in the labeling of the xBD dataset. Table taken from xBD dataset release paper [11].

4.2 Data Preprocessing

We use the Python programming language [26] to preprocess the data and train CNNs. The dataset consists of 1024 pixels by 1024 pixels satellite images. In order to zero in on the changes we begin by collecting bounding boxes of the buildings in each image. As aforementioned, the dataset provides segmentation ground truth masks with classification labels for each corresponding building polygon, which is a collection of Cartesian coordinates. Thus, we take the maximum and minimum x coordinates of each polygon, $x_{\max}$ and $x_{\min}$, respectively, and the maximum and minimum y coordinates, $y_{\max}$ and $y_{\min}$, respectively, and construct a bounding box with a lower left corner at coordinates $(x_{\min}, y_{\min})$, width $x_{\max} - x_{\min}$, and height $y_{\max} - y_{\min}$. A visual representation of bounding boxes is displayed in Figure 3.

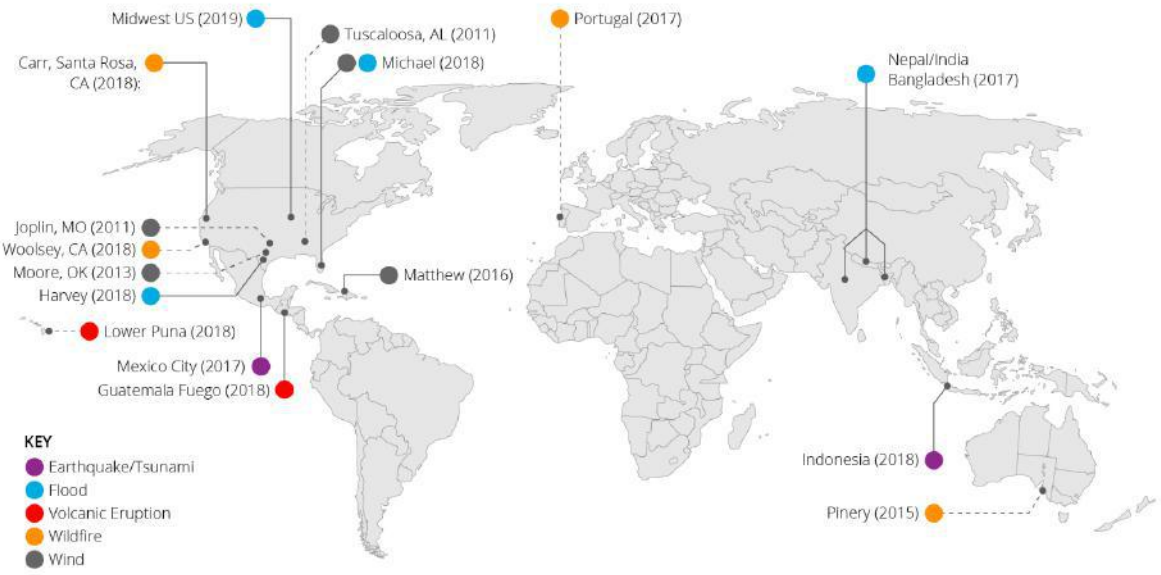
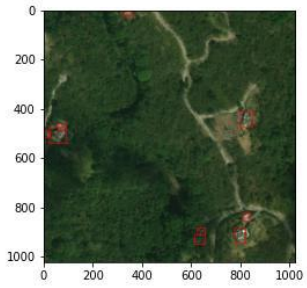
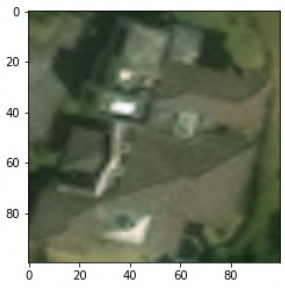


Fig. 2: Map of locations worldwide from which satellite imagery data was taken. Figure taken from xBD dataset release paper [11].



(a) Bounding Boxes on an Image



(b) One such bounding box.

Fig. 3: Left: Nine bounding boxes drawn in red on an arbitrary image from the dataset, each containing a building. Right: A building crop (the interior of one of these bounding boxes), resized to 100 pixels by 100 pixels.

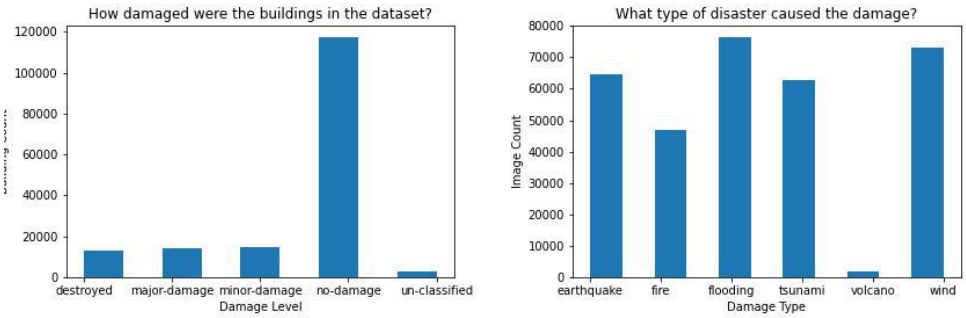


Fig. 4: Left: Distribution of buildings by damage level. Right: Distribution of buildings by damage type.

We create graphical representations of the data, plotting the number of buildings per disaster type and per damage classification (Fig. 4). Furthermore, we analyze the distribution of building bounding box sizes (Fig. 5), noting that buildings that are too small and blurred will not be valuable training data and may hinder the model from achieving accurate results (Fig. 6). We discard buildings that have a bounding box size of less than 2,000 pixels. We also discard any buildings with the classification ground truth label of "unclassified," because this information is not useful for our purposes. We could then randomly shuffle the dataset and split it into training and validation sets according the ratio 0.8:0.2. However, in order to maintain an equal distribution over JDS classification (damage level) in our training and validation sets so that we can properly assess model accuracy, we provide for an equal number of buildings of the categories "destroyed," "major damage," "minor damage," and "no damage" in each set, while still maintaining a 0.8:0.2 ratio between train and validation. The xBD dataset is deliberately created with a disproportionately large volume of buildings with no damage (see Figure 4) [11], but training on such a lopsided data distribution would yield artificially high accuracy numbers and not give valuable results.

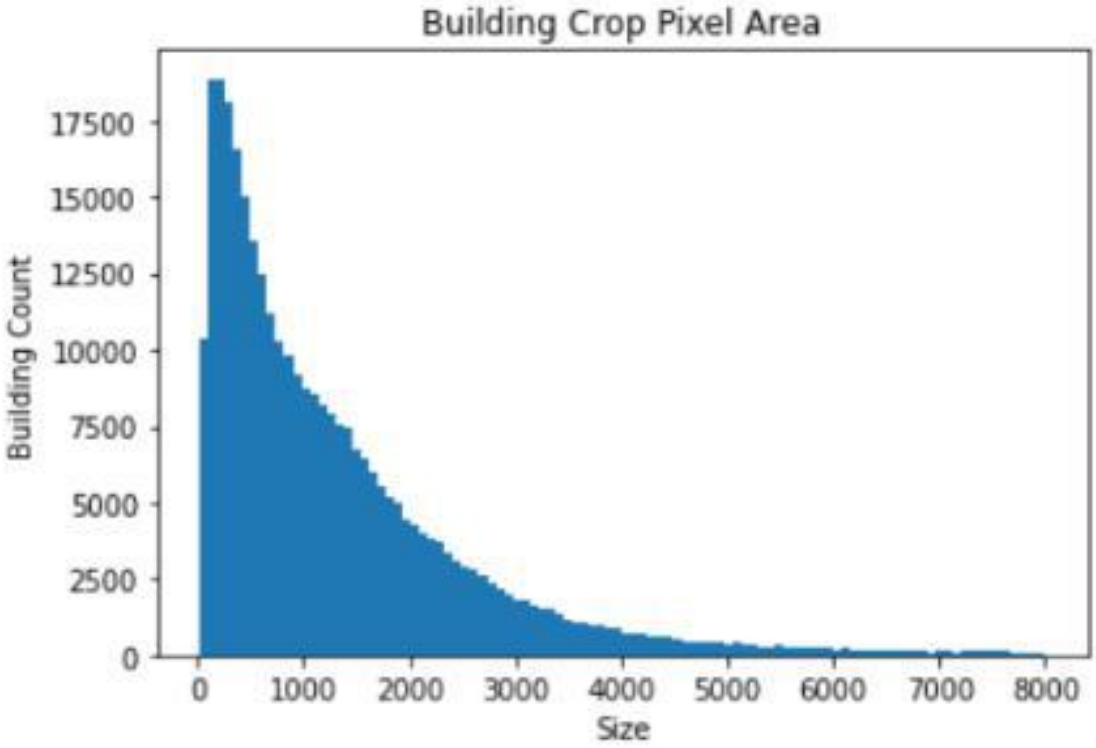
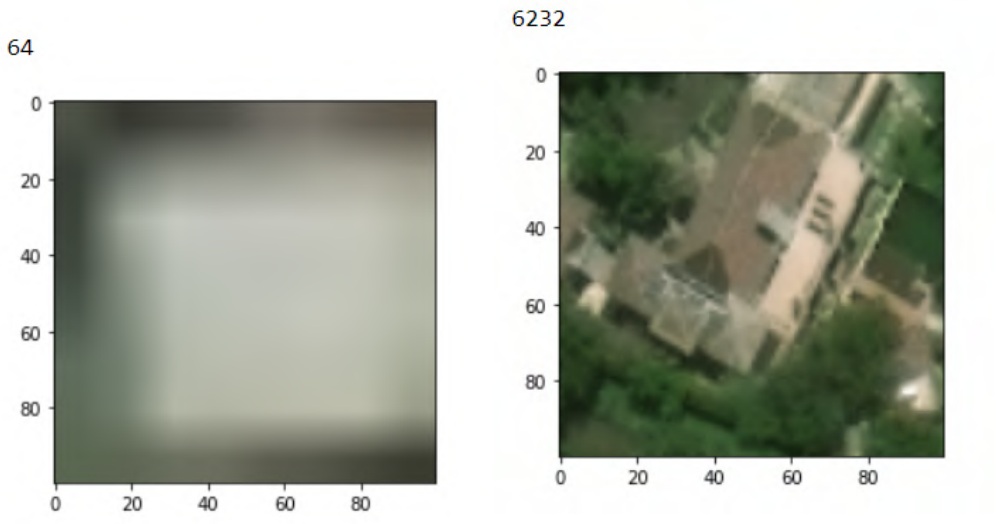


Fig. 5: A histogram representing the distribution of building bounding box areas in the xBD dataset. Outliers (bounding boxes with a pixel area greater than 8000) have been removed from the graph.

4.3 Baseline Model

We train a baseline classification model to classify buildings by damage level, as defined by the Joint Damage Scale (Fig. 1). The model input is only the post-disaster image. Notably, our baseline model does not use change detection. Because the data is labeled, this is a supervised approach. The model architecture is ResNet18, an 18 layer CNN, and was pre-trained on ImageNet data [6]. This baseline model uses the cross-entropy loss function, which is defined as

$$-\sum_{c=1}^{4} y_{o,c} \log(p_{o,c}),$$



(a) A building with a bounding box of 64 pixels. Resized to 100 by 100 pixels for better viewing.

(b) A building with a bounding box of 6232 pixels. Resized to 100 by 100 pixels for better viewing.

Fig. 6: A comparison of two vastly differently sized buildings in the dataset. Clearly, the blurred building of size 64 pixels would not be a useful data point for any deep learning model to learn from and yield accurate results. Instead, buildings like this are noisy and are discarded in our method.

where $y_{o,c}$ is a binary indicator (either 0 or 1) of whether c, as a label, correctly classifies observation o, and $p_{o,c}$ is the predicted probability that observation o is of the class c. Cross-entropy loss is defined, in other words, as the negative sum of the expression $y_{o,c} \log(p_{o,c})$ across all 4 possible classes c: no damage, minor damage, major damage, and destroyed. The network is trained on 12,800 buildings crops with a batch size of 32. The Adam optimizer with a learning rate of 0.001 is used. The model trained for 100 epochs on NVIDIA Tesla K80 GPUs. We use PyTorch [23] to train this model, as well as any subsequent models.

4.4 Improvements

We train other models that improve upon the performance of the baseline model. To do this, we introduce other model inputs, namely the pre-disaster image (in combination with the post-disaster image) and the type of disaster (e.g. volcano, wind, etc.) that caused the building damage. To train a model that takes in both pre-disaster images and their corresponding post-disaster images, we concatenate the RGB channels of the two and use that as input. To train a model that takes in the pre-disaster image, post-disaster image, and disaster type, we do the same, but also concatenate a one-hot encoded representation of the disaster type in one of the later layers of the CNN.

Furthermore, we experiment with other loss functions, namely mean squared error loss and ordinal cross-entropy loss to train these models. We define mean squared error as

$$\frac{1}{b}\sum_{i=1}^{b}(y-\hat{y})^2,$$

where b is the batch size, y is the ground truth (a class from 0 to 3 representing each damage level), and $\hat{y}$ is the prediction. Ordinal cross-entropy loss differs from cross-entropy loss in the sense that it takes into account the distance between the ground truth and the predicted class (hence "ordinal"). Since the building damage classification problem involves different and increasing levels of damage from no damage to destruction, this function is useful to distinguish between different categories. To implement ordinal cross-entropy loss as the loss function, we treat it as generic multi-class classification and encode the classes no damage, minor damage, major damage, and destroyed as [0, 0, 0], [1, 0, 0], [1, 1, 0], and [1, 1, 1],

Tab. 1: Comparison of the Validation Accuracy on 9 Different Models

Model Accuracy on Validation Set with Chosen Loss (100 epochs)			
Model Input	**Loss Function**		
	Mean Squared Error	**Cross-Entropy Loss**	**Ordinal Cross-Entropy Loss**
Post-Disaster Image Only	45.3%	59.5%	64.2%
Pre-Disaster, Post-Disaster Images	50.2%	68.3%	71.2%
Pre-Disaster, Post-Disaster Images, Disaster Type	49.7%	72.7%	74.6%

respectively. [4]. The other aspects of the training process (optimizer, learning rate, number of epochs, etc.) remain the same. These improved models contribute to our understanding of what information leads to the most accurate prediction results for building damage assessment.

4.5 Qualitative Interpretability

In addition to the quantitative approaches achieved above, we also make progress in the study of model interpretability in a qualitative manner. One important question that one might ponder in the sphere of examining how, explicitly, models assess and classify building damage, is where the model is looking to make its decision. To address this question in a qualitative way, we create gradient class activation maps [27], or "heat maps," to represent which parts of an image (building crop) are most integral for the model's prediction. We do this after already training a model and saving its weights. For our purposes, we only perform this task on the baseline model, which takes in just the post-disaster image as input.

5 Results

5.1 Quantitative Results

In Table 1, we present model accuracy on the validation set across nine different models, which are differentiated by three different input combinations and three different loss functions. The baseline model, which is trained with post-disaster data only and the cross-entropy loss function, has an accuracy of 59.5%, as shown. It is important to note that all models were trained and validated on data that is evenly split between building crops of each class (no damage, minor damage, major damage, and destroyed), so a purely blind guessing model would achieve approximately 25% accuracy.

When the model is trained and validated on both pre-disaster and post-disaster building imagery as opposed to solely the post-disaster data, we see an 8.8% increase in accuracy on the validation set in comparison to the baseline model, while keeping the loss function constant. Adding the disaster type as a third type of input subsequently increases accuracy by another 4.4%.

Reverting back to the baseline model, changing the loss function utilized to ordinal cross-entropy loss instead of simply cross-entropy loss, we see a 4.7% accuracy jump on the validation set. Sticking with ordinal cross-entropy loss, adding the pre-disaster image as a mode of input increases accuracy by another 7.0%, while adding the disaster type as yet another mode of input increases accuracy by an additional 3.4%.

5.2 Qualitative Results

In Figure 7, we present some qualitative results that we generated in the form of gradient class activation maps [27], as discussed earlier. Semantically, we can see that portions of the buildings that showed evidence of damage, such as cracks and debris, are depicted to have been the most useful to the model for classification.

6 Discussion

Much of our results conform to our hypotheses. Firstly, accuracy on the validation set improves when more modes of useful information are inputted into the model (accuracy generally increases moving down the rows of Table 1). This is reasonable because the more information that the model has to work with,

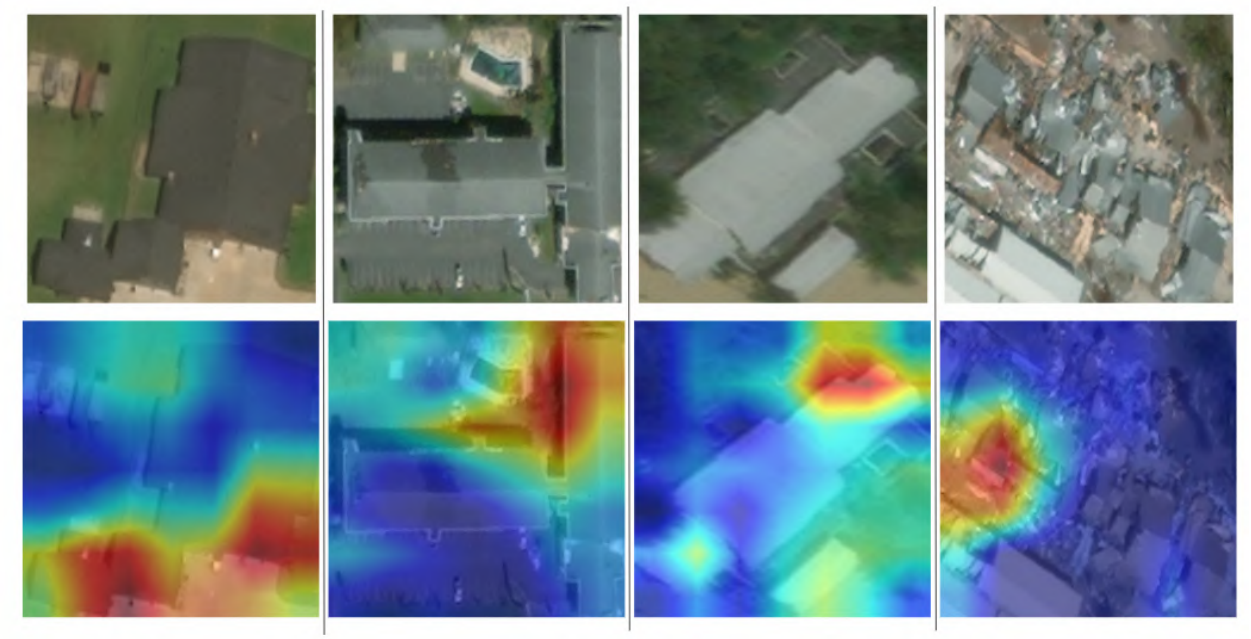


Fig. 7: Gradient class activation maps depict which parts of the building crop lead the baseline model to predict a certain classification. On the top are the original images (crops) and on the bottom are the corresponding gradient class activation maps. The images included are only post-disaster images. From left to right: (1) A building with label "no damage," after flooding in the Midwestern United States, (2) A building with label "minor damage," after Hurricane Michael, (3) A building with label "major damage," after Hurricane Harvey, and (4) A building with label "destroyed," after Hurricane Michael.

the more accurate predictions it should make. A large part of our research was addressing which types of input aid the convolutional neural networks in making accurate predictions. From the results generated, it seems that having the aspect of change detection (when the pre-disaster image is concatenated with the post-disaster image and inputted) is useful, along with the type of disaster.

We also note that models using ordinal cross-entropy loss as their criterion for optimization perform the most accurately. This is also reasonable because, as previously mentioned, ordinal cross-entropy loss is most specifically applicable for a classification problem that involves an ordinal scale (in this case, the JDS), as opposed to categories with no intrinsic ordering. Mean squared error (MSE), not surprisingly, showed itself to be the least effective loss function to use for training. This is a justifiable because MSE is primarily used in regression problems, not classification problems. We find that cross-entropy loss models fall somewhere in between.

In summation, Table 1 shows that we were able to improve upon the baseline model by adding two more modes of input, as well as switching the loss function used to ordinal-cross entropy loss.

However, overall, we note that none of the accuracy numbers are necessarily optimal. This can be explained by the fact that the differences between categories, particularly between minor-damage and major-damage, are in many cases difficult to discern, for both humans and computers. This is certainly a challenge that comes with non-binary classification tasks with building damage, and it has been acknowledged by many, including Gupta et. al [11]. In addition, there is some noisy data in the dataset and cleaning it more thoroughly would most likely yield marginally more accurate model predictions.

Furthermore, the qualitative results generated in the form of gradient class activation maps display a more visual interpretation of what our baseline model is "seeing." These visuals can help in the study of other types of useful model inputs in the future.

These results, taken together, contribute to the research area of building damage detection by addressing the limited interpretability of current literature in regards to what types of information are most useful to building damage classification models as well as what loss functions are the best criterion.

7 Conclusion

The main insights that can be drawn from our work include using individualized building crops instead of semantic segmentation to train models, performing experiments with various combinations of model inputs and loss functions to explicitly examine their differences, and using overlaid heat maps to qualitatively analyze which aspects of a building are useful to derive predictions. Our work's main contribution to the field is presenting a novel, more interpretable and step-by-step, analysis of how to classify building damage most accurately and effectively in the event of a natural disaster. Practically, our work and others in the field advance methods for more robust emergency responses and more efficient allocation of resources, which saves lives and property. This research is especially important now, when climate change is ramping up the frequency and intensity of these devastating events.

Building on our work with improving interpretability, in the future, one could investigate the prediction performance of deep learning models with other types of input added, such as neighboring building damage levels. Experiments with other loss functions, if any, are also an aspect of future work in this area. Additionally, other ways to combine information such as pre-disaster and post-disaster images (instead of a simple concatenation like we did here) should yield interesting results. Moreover, it should be noted that the qualitative results we present are only from the baseline model. In the future, one could generate these types of gradient class activation maps for the models that take in both the pre-disaster and post-disaster images, or other models.

8 Acknowledgements

I would like to thank Ethan Weber for his outstanding mentorship and everyone involved in the Summer STEM Institute for the excellent research and bootcamp programs. This project would not have been possible without their support.

References

[1] Anju Asokan and J Anitha. Change detection techniques for remote sensing applications: a survey. *Earth Science Informatics*, 12(2):143–160, 2019.

[2] Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771, 2004.

[3] Sean Andrew Chen, Andrew Escay, Christopher Haberland, Tessa Schneider, Valentina Staneva, and Youngjun Choe. Benchmark dataset for automatic damaged building detection from post-hurricane remotely sensed imagery. *arXiv preprint arXiv:1812.05581*, 2018.

[4] Jianlin Cheng, Zheng Wang, and Gianluca Pollastri. A neural network approach to ordinal regression. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1279–1284. IEEE, 2008.

[5] BJ Conway and Keith Anthony Browning. Weather forecasting by interactive analysis of radar and satellite imagery. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 324(1579):299–315, 1988.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] National Centers for Environmental Information. Billion-dollar weather and climate disasters: Overview.

[8] R Foulser-Piggott, R Spence, K Saito, DM Brown, and R Eguchi. The use of remote sensing for post-earthquake damage assessment: lessons from recent events, and future prospects. In *Proceedings of the Fifthteenth World Conference on Earthquake Engineering*, page 10, 2012.

[9] Aito Fujita, Ken Sakurada, Tomoyuki Imaizumi, Riho Ito, Shuhei Hikosaka, and Ryosuke Nakamura. Damage detection from aerial images via convolutional neural networks. In *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, pages 5–8. IEEE, 2017.

[10] Lionel Gueguen and Raffay Hamid. Large-scale damage detection using satellite imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1321–1328, 2015.

[11] Ritwik Gupta, Richard Hosfelt, Sandra Sajeev, Nirav Patel, Bryce Goodman, Jigar Doshi, Eric Heim, Howie Choset, and Matthew Gaston. xbd: A dataset for assessing building damage from satellite imagery. *arXiv preprint arXiv:1911.09296*, 2019.

[12] Rohit Gupta and Mubarak Shah. Rescuenet: Joint building segmentation and damage assessment from satellite imagery. *arXiv preprint arXiv:2004.07312*, 2020.

[13] Hanxiang Hao, Sriram Baireddy, Emily R Bartusiak, Latisha Konz, Kevin LaTourette, Michael Gribbons, Moses Chan, Mary L Comer, and Edward J Delp. An attention-based system for damage assessment using satellite imagery. *arXiv preprint arXiv:2004.06643*, 2020.

[14] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Ibrahim Rizk Hegazy and Mosbeh Rashed Kaloop. Monitoring urban growth and land use change detection with gis and remote sensing techniques in daqahlia governorate egypt. *International Journal of Sustainable Built Environment*, 4(1):117–124, 2015.

[17] Farid Kadri, Babiga Birregah, and Eric Châtelet. The impact of natural disasters on critical infrastructures: A domino effect-based study. *Journal of Homeland Security and Emergency Management*, 11(2):217–241, 2014.

[18] Peijun Li, Haiqing Xu, and Benqin Song. A novel method for urban road damage detection using very high resolution satellite imagery and road map. *Photogrammetric Engineering & Remote Sensing*, 77(10):1057–1066, 2011.

[19] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[20] Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 international interdisciplinary PhD workshop (IIPhDW)*, pages 117–122. IEEE, 2018.

[21] Siddhartha Sankar Nath, Girish Mishra, Jajnyaseni Kar, Sayan Chakraborty, and Nilanjan Dey. A survey of image classification methods and techniques. In *2014 International conference on control, instrumentation, communication and computational technologies (ICCICCT)*, pages 554–557. IEEE, 2014.

[22] German Novikov, Alexey Trekin, Georgy Potapov, Vladimir Ignatiev, and Evgeny Burnaev. Satellite imagery analysis for operational damage assessment in emergency situations. In *International Conference on Business Information Systems*, pages 347–358. Springer, 2018.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[24] Hannah Ritchie and Max Roser. Natural disasters. *Our World in Data*, 2014.

[25] David J Rogers, Sarah E Randolph, Robert W Snow, and Simon I Hay. Satellite imagery in the study and forecast of malaria. *Nature*, 415(6872):710–715, 2002.

[26] Michel F Sanner et al. Python: a programming language for software integration and development. *J Mol Graph Model*, 17(1):57–61, 1999.

[27] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[28] Zheng Song, Qiang Chen, Zhongyang Huang, Yang Hua, and Shuicheng Yan. Contextualizing object detection and classification. In *CVPR 2011*, pages 1585–1592. IEEE, 2011.

[29] Richard W Stoffle, David B Halmo, Thomas W Wagner, and Joseph J Luczkovich. Reefs from space: satellite imagery, marine ecology, and ethnography in the dominican republic. *Human Ecology*, 22(3):355–378, 1994.

[30] Maarten K Van Aalst. The impacts of climate change on the risk of natural disasters. *Disasters*, 30(1):5–18, 2006.

[31] Andrés Viña, Fernando R Echavarria, and Donald C Rundquist. Satellite change detection analysis of deforestation rates and patterns along the colombia–ecuador border. *AMBIO: A Journal of the Human Environment*, 33(3):118–125, 2004.

[32] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *CVPR 2011*, pages 3169–3176. IEEE, 2011.

[33] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3360–3367. IEEE, 2010.

[34] Ethan Weber and Hassan Kané. Building disaster damage assessment in satellite imagery with multi-temporal fusion. *arXiv preprint arXiv:2004.05525*, 2020.

[35] Jiajun Wu, Yinan Yu, Chang Huang, and Kai Yu. Deep multiple instance learning for image classification and auto-annotation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3460–3469, 2015.

[36] Joseph Z Xu, Wenhan Lu, Zebo Li, Pranav Khaitan, and Valeriya Zaytseva. Building damage detection in satellite imagery using convolutional neural networks. *arXiv preprint arXiv:1910.06444*, 2019.

[37] Chenghai Yang, James H Everitt, Qian Du, Bin Luo, and Jocelyn Chanussot. Using high-resolution airborne and satellite imagery to assess crop growth and yield variability for precision agriculture. *Proceedings of the IEEE*, 101(3):582–592, 2012.

[38] Xiaofei Yang, Yunming Ye, Xutao Li, Raymond YK Lau, Xiaofeng Zhang, and Xiaohui Huang. Hyperspectral image classification with deep learning models. *IEEE Transactions on Geoscience and Remote Sensing*, 56(9):5408–5423, 2018.

Prostate Lesion Detection and Salient Feature Assessment Using Zone-Based Classifiers

Haoli Yin

Abstract

Multi-parametric magnetic resonance imaging (mpMRI) has a growing role in detecting prostate cancer lesions. Thus, it's pertinent that medical professionals who interpret these scans reduce the risk of human error by using computer-aided detection systems. The variety of algorithms used in system implementation, however, has yielded mixed results. Here we investigate the best machine learning classifier for each prostate zone. We also discover salient features to clarify the models' classification rationale. Of the data provided, we gathered and augmented T2 weighted images and apparent diffusion coefficient map images to extract first through third order statistical features as input to machine learning classifiers (Logistic Regression, SVM, Random Forest, and XGBoost). For our deep learning classifier, we used a convolutional neural net (CNN) architecture for automatic feature extraction and classification. The CNN results' interpretability was improved by saliency mapping to uncov er the black box and understand the classification mechanisms within. Ultimately, we concluded that effective detection of peripheral and anterior fibromuscular stroma (AS) lesions depended more on statistical distribution features, whereas those in the transition zone (TZ) depended more on textural features. Ensemble algorithms worked best for PZ and TZ zones, while CNNs were best in the AS zone. These classifiers can be used to validate a radiologist's predictions and reduce inter-reader variability in patients suspected to have prostate cancer. The salient features reported in this study can also be investigated further to better understand hidden features and biomarkers of prostate lesions to look for in prostate cancer detection with mpMRIs.

1 Introduction

Prostate Cancer is the most common malignancy and the second leading cause of cancer for men in the United States [31]. Thus, it is vital that early and accurate diagnosis of prostate cancer is achieved for appropriate treatment and improved prognosis. Diagnostic tests such as the prostate specific antigen (PSA) serum test and the digital rectal exam (DRE) have been used in the past but have proven to deliver unreliable results. PSA can deliver false positive results since it is also secreted by normal, hyperplastic and malignant tissue. In addition, DRE is an unnecessary invasive procedure that can precipitate pain among other side effects and does not significantly reduce mortality. This ineffectiveness stems from the fact that with DRE, only peripheral zone tumors can be identified in asymptomatic men and sometimes low-grade tumors can even go unnoticed [25]. Ultimately, both tests may lead to a higher chance of false positives and negatives than if another diagnostic test were performed. Such results would negatively affect risk stratification and clinical cancer staging, which are essential to providing the appropriate treatment level.

To better assess the presence of prostate cancer through noninvasive procedures, medical imaging techniques such as multiparametric Magnetic Resonance Imaging (mpMRI) have shown promising results in diagnosis, localization, risk stratification, and staging of clinically significant local prostate cancer [14]. mpMRIs consist of different scanning methods that develop into four main sets of images: T2-weighted images (T2WI), diffusion-weighted imaging (DWI), dynamic contrast-enhancement (DCE), and MR spectroscopy [33]. However, even though mpMRI has provided a much improved rationale for diagnosis, the analysis and interpretation of these scans through the standardized Prostate Imaging Reporting And Data System (PI-RADS) by radiologists are largely subjective and inconsistent due to varied experience and training in the field of prostate cancer diagnostics. This may result in the increased possibility of false positives/negatives. These inconsistencies may be addressed with computer aided detection (CAD) algorithms to provide a more standardized, objective diagnosis that minimizes the risk of human error [35].

In the last few years, several machine learning algorithms and pipelines have been proposed for this exact purpose [21, 10] and have achieved results that outperform those of trained radiologists. However,

the design of such algorithms for feature extraction in mpMRIs require a certain level of domain expertise in order to increase the accuracy of the classification algorithms. With this range of experience in prostate cancer diagnostics, different researchers have reported various algorithmic implementations. Even more recently, the advent of deep convolutional neural network (CNN) techniques has led to great success in natural image processing [23, 20, 32]. For medical imaging, however, the effectiveness of deep learning in a clinical setting is hampered by three major challenges: the heterogeneous raw data, the relatively small sample size classification [11], and the "black box" of CNNs, which is due to their multilayer nonlinear structure that is non-transparent and uninterpretable to humans [16]. In this paper, we address these challenges by incorporating heterogeneous data with proper preprocessing to detect lesions in three zones of the prostate: the peripheral zone (PZ), the transition zone (TZ), and the anterior fibromuscular stroma (AS). Detection pipelines are established for each zone independently as there has been proven to be significant differences in features and signs seen in each zone [29]. From these features respective to each zone, non-deep learning and deep learning algorithms are compared regarding their performance in accurate lesion detection. Furthermore, important (salient) features in lesion determination from each non-deep learning classifier are determined to better understand the rational for classification. As a supplementary objective, we also hope to better unveil the black box of CNNs by improving the interpretability of results through saliency mapping to highlight the salient features of an image that the CNNs uses to justify its output in lesion detection.

2 Literature Review

In the last decade, there have been significant developments in machine learning applications in the medical field, especially in cancer diagnostics. Chan et al. [6] were the first to implement a multi-parametric CAD system for the diagnosis of prostate cancer. In their approach, they used line-scan diffusion, T2, and T2-weighted images in combination with an SVM classifier, to identify predefined areas of the peripheral zone of the prostate for the presence of prostate cancer. This study set the foundation for future algorithmic developments in prostate cancer detection and classification using today's traditional machine learning classifiers.

Building on this, Litgens et al. [21] made the first prostate MRI CAD system that was evaluated on a per-patient basis and was compared with the prospective performance of radiologists. This fully automated CAD system included a novel combination of segmentation, voxel classification, candidate extraction, and candidate classification for a more accurate diagnostic outcome compared with the ground truth of a biopsy. Even though performance evaluation showed that it outperformed the state-of-the-art at the time, this comparison had its limitations due to different evaluation data sets involved. To address this issue in our paper, we used the same mpMRI dataset for both non-deep learning classifiers and deep learning classifiers so that inputs were standardized and allowed for valid comparison between classifiers.

Focusing more on deep learning developments, in a 2019 study, collaborators developed a deep-learning system (DLS) for the prostate cancer grading scale (Gleason scoring) based on whole tissue images [1]. This DLS achieved a diagnostic accuracy of 0.7 (scale of 0.5 (random) to 1). This was an impressive result when compared with 29 expert pathologists' diagnostic accuracy of only 0.61. Furthermore, follow-up data confirmed a better patient risk stratification by the DLS. Overall, this study demonstrates how machine learning can improve well established standards such as the Gleason scoring by eliminating subjective evaluation by the human eye, thus yielding to more precise prognostication. While this study used whole tissue samples as an input, we used mpMRIs instead of a biopsy for noninvasive diagnosis to increase patient comfort and to hopefully yield similar results. Finally, Liu et al. [23] constructed a accurate CNN model, Xmasnet, to detect prostate lesions using an image of the entire prostate. To instead determine the performance of CNNs in each zone, we created our own CNN architecture (Fig. 1) inspired by Xmasnet to fit our smaller input images.

3 Purpose

1. Extract statistical and textural features from mpMRIs in separate zones of the prostate.

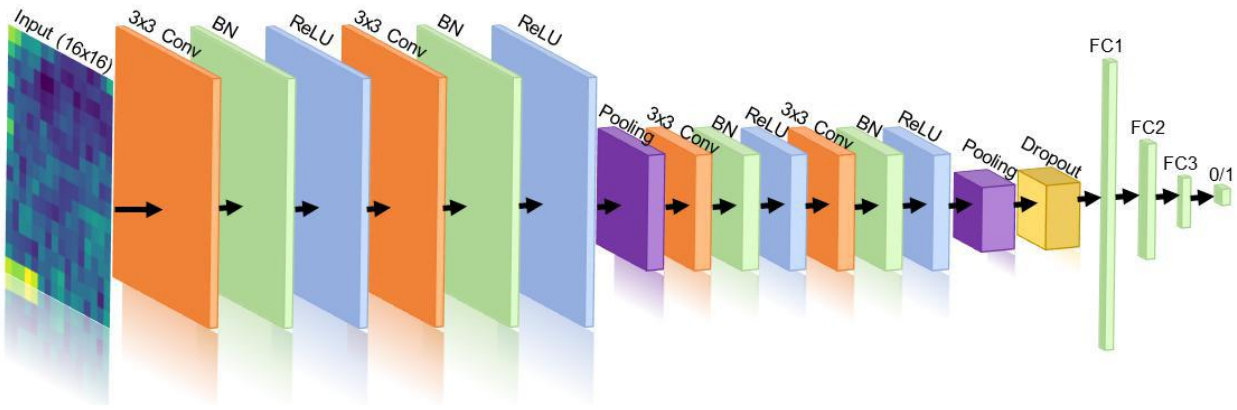


Fig. 1: Custom CNN Architecture. Conv: convolutional layer; BN: batch normalization layer; ReLU: rectified linear unit; Pooling: max pooling layer; FC: fully connected layer.

2. Implementation and analysis of machine learning classifiers to determine the best classifier in each zone.
3. Increase interpretability of CNN results through saliency mapping.

4 Methods

4.1 Data Set

All data used for this paper was provided by the 2017 PROSTATEx Challenge held in conjunction with the 2017 SPIE Medical Imaging Symposium. This collection is a retrospective set of prostate MR studies. Studies include T2-weighted (T2W), diffusion-weighted (DW) imaging, and dynamic contrast-enhanced (DCE).

T2-weighted imaging Given a proton from a hydrogen atom, whenever the magnetic dipole moment is placed into an external magnetic field, it will feel a torque that will tend to align it with the external magnetic field. The magnetic dipole moments will try its best to align with the magnetic field lines, but it will not line up exactly. Instead, it will precess around on an axis, which can be either spin-up or spin-down (with spin-down having more energy). In this case, the external magnetic field is the electromagnetic (EM) pulse given by the MRI machine to give spin-up protons enough energy to become spin-down protons. Once the EM pulse is turned off, the excited proton returns to equilibrium and gives off photons which the MRI machine can measure. While the amount and locations of photons are measured in T1 imaging, T2-weighted images are created based on T2 relaxation times, which is the time taken for spinning protons to lose phase coherence among other proton-containing nuclei. Since T2-weighted imaging assigns pixel intensities based on the density of hydrogen atoms in an area, it can map relative water content within the body to create recognizable structures and tissues. Within the prostate, the peripheral zone has the highest water content and is thus the brightest part of the image.

Diffusion-weighted imaging This is a form of MR imaging based upon measuring the random Brownian motion of water molecules within a voxel of tissue. In general, highly cellular tissues or those with cellular swelling exhibit lower diffusion coefficients, which are represented as lower intensity values in an apparent diffusion coefficient map. This type of imaging is particularly useful for determining tumor presence as more tissue results in lower diffusion of water due to the restriction of space, leading to lower signal intensities.

dynamic contrast-enhanced imaging This sometimes also referred to as permeability MRI and is one of the main MRI perfusion techniques which calculates perfusion parameters by evaluating T1 shortening induced by a gadolinium-based contrast bolus passing through tissue. The most commonly calculated parameter is k-trans and is the one given in this dataset. Ktrans is measure of capillary permeability and has higher values (corresponding with higher pixel intensities) for areas with more blood flow. This is a significant tool when it comes to detecting tumors, especially those that have induced angiogenesis to get more blood flow to the tumor.

The images were acquired on two different types of Siemens 3T MR scanners, the MAGNETOM Trio and Skyra. T2-weighted images were acquired using a turbo spin echo sequence and had a resolution of around 0.5 mm in-plane and a slice thickness of 3.6 mm. The DCE time series were acquired using a 3-D turbo flash gradient echo sequence with a resolution of around 1.5 mm in-plane, a slice thickness of 4 mm and a temporal resolution of 3.5 s. Finally, the DWI series were acquired with a single-shot echo planar imaging sequence with a resolution of 2 mm in-plane and 3.6 mm slice thickness and with diffusion-encoding gradients in three directions. Three b-values were acquired (50, 400, and 800), and subsequently, the apparent diffusion coefficient (ADC) map was calculated by the scanner software. All images were acquired without an endorectal coil [21, 22, 9]. For our paper, we mainly focused on using T2-weighted images and ADC maps for feature extraction, feature extraction, and the resulting binary classification for lesion determination.

4.2 Data Storage

The raw data provided included the acquired MR images encoded in DICOM format while Ktrans images were provided in MHD format. Ktrans is a key pharmacokinetic parameter computed from the available Dynamic contrast-enhanced T1-weighted series. Each patient had one corresponding Ktrans image but can have up to three lesions, so detection was evaluated on a per-lesion basis. The annotations for the mpMRIs were provided in csv files that contained information such as the de-identified patient number (compliant with HIPAA rules), a finding number corresponding to each lesion found in a single patient, a centroid location marked by a radiologist (biopsied for the ground truth), and a DICOM description to match the annotation to the image data. The overall data was already separated into train and test sets by the image repository with a total of 327 lesions in the train set and 206 in the test set. All data was sorted through and compiled automatically through scripts written in-house. The lesion pixel arrays and corresponding annotations were stored in an hierarchical data format version 5 (HDF5) for efficient storage and querying of data. This storage method allowed for almost instantaneous speeds of data extraction from the raw data and significantly reduced runtime when querying the data [12].

4.3 Preprocessing

We recognized that images which contained similar objects or scenes often had different intensity ranges that made it difficult to compare them manually, especially in T2WIs. Since the focus of this study was to primarily use T2-weighted images (T2WI) and ADC map images, we decided to apply an intensity range standardization algorithm provided by Medpy.filter to transform T2WI image intensity ranges to a common standard intensity space without any loss of information. This was done with a multi-segment linear transformation model. Thus, this method allowed for valid comparison of prostates without signal intensity bias created from different scanning conditions. This algorithm works by defining a standard intensity space through an intensity value range. During the training phase with the T2WIs in the training set, the intensity values at certain cut-off percentiles of each image were computed and a single-segment linear mapping from them to the standard intensity space range limits were created. Then the images' intensity values at several landmark percentiles were extracted and passed to the linear mapping to be transferred roughly to the standard intensity space. The mean of all these mapped landmark intensities formed the model learned. When the model was applied on a new test image, the image's intensity values were extracted at the cut-off percentile as well as at the landmark percentile positions. This resulted in several segments. Using these segments, the corresponding standard intensity space range values, and the learned mean landmark values, a multi-segment linear transformation model was created for the image.

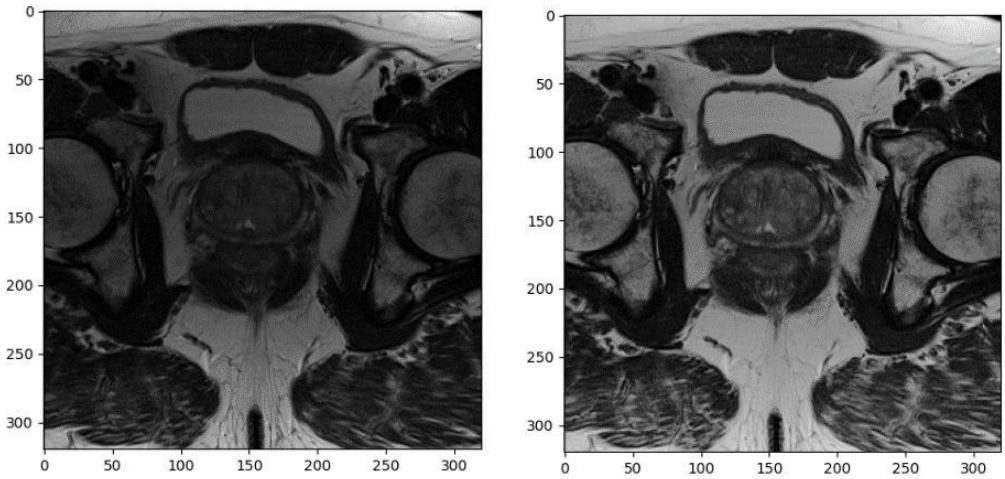


Fig. 2: Before and after images of Intensity Range Standardization.

This was then applied to the image's intensity values to map them to the standard intensity space, creating the new, intensity-standardized image (Fig. 2), ready for feature extraction [24]. This signal intensity range standardization algorithm was not applied to ADC map images as ADC is a quantitative measurement [5].

4.4 Data Augmentation

To split the data, the number of lesions from each zone were counted. In the training set, there was found to be 188 peripheral zone (PZ) lesions, 82 transition zone (TZ) lesions, 55 anterior fibromuscular stroma (AS) lesions, and 2 seminal vesicle (SV) lesions. In the test set, there were 113 PZ lesions, 59 TZ lesions, 34 AS lesions, and 2 SV lesions. Since there were no malignant SV lesions for our model to learn from, we excluded SV lesions from our study, similar to the procedure in Chen et al. [7]. This was a part of our study that we hoped other studies could address given a more inclusive data set.

The main limitation for the provided data set was that the test set's ground truth was not given (still used in an ongoing challenge). The original purpose of that challenge was to classify lesions to an appropriate clinical significance score, in which clinical significance was defined as having a Gleason score of greater than 7. Knowing that all lesions contained some lesion grade, all images under a zone label were denoted to have a lesion (truth label of '1'), denoting that they were lesions. Negative labels (truth label of '0') for lesions were manually extracted from patient prostates with no noted lesions noted for a zone, meaning that there was no lesion there, thus preventing data leakage. Within both training and test sets, the T2WI and ADC image sets were split into 3 zones: PZ, TZ, and AS. To generate the samples needed for feature extraction, a 16x16 pixel area for T2WI and 6x6 pixel area for ADC images were cut out from the centroid of the positive lesion marked by the radiologist. To generate the same size areas from negative MRI samples, random sampling was done at the manually bounded area until the positive and negative ratio was 1 to 3 (25% positive, 75% negative) for each zone. In addition, during the sampling process, T2WI and ADC images were paired per location so that the features extracted would be relevant for a single location towards binary classification of the possible lesion in the experimental stage. A summary for sample quantities is recorded in table 1.

4.5 Feature Extraction

For each zone and MRI type combination, 13 features were extracted, totalling 26 features for each sample. First order statistics based on the pixel signal intensities were extracted such as the 10th percentile and average intensity [27]. Second order texture features were extracted based on Tamura and Haralick features. These are considered second order with respect to how they consider the relationship between groups of two pixels in an image.

	Zone Type		
Data Set	PZ	TZ	AS
Train	188/748	82/322	55/215
Test	113/453	59/239	34/132

Tab. 1: Data augmented so that there were 25% positive, 75% negative samples in each zone. Number of lesions were the same in T2WI & ADC images.

The most common texture analysis for image classification is based on the gray-level co-occurrence matrix (GLCM) by Haralick et al. [15]. The GLCM is a matrix that is defined over an image to be the distribution of co-occurring pixel values (gray level intensity) at a given offset. This matrix is then used to make various texture measure calculations. In our case, we automatically calculated the GLCM matrix for each lesion MRI sample with an offset of one pixel and zero degrees as the direction for the neighbor pixel. We originally experimented with multiple degrees and then summing the output values together, but the classification results did not show significant improvement from only calculating zero degrees.

There are three main categories of Haralick features totaling to 13 features, of which we used six. In the first category, the contrast group uses weights related to the distance from the GLCM diagonal (weights on the diagonal show no contrast) to calculate a weighted average value as an output. The contrast group includes the contrast value (sum of squares variance) where weights increase exponentially, dissimilarity where weights increase linearly, and homogeneity (inverse difference moment) where weights decrease exponentially from the diagonal. In the second category, the orderliness group features define how regular pixel values differences are within a preset frame. The weights to calculate the weighted average value are based on how many times a given pair occurs. The orderliness group includes the angular second moment (ASM) and energy which have high values when the frame contents are very orderly. In the final category, descriptive statistics of the GLCM matrix are synthesized such as correlation, which measures the linear dependency of gray levels on those of neighboring pixels. In summary, we used GLCM texture features (formulas in table 2) of contrast, dissimilarity, homogeneity, ASM, energy, and correlation to be incorporated into each sample's feature set.

Tamura et al. [34] presented that six textural characteristics had high correlation to human visual perception, which were coarseness, contrast, directionality, line-likeness, regularity, and roughness. Of these six, we will use coarseness, contrast, and roughness as features (formulas in table 2). Coarseness in this case refers to the distances of notable spatial variations of grey levels which implicity describes the size of primitive elements (texels) forming a texture. Contrast measures how much gray levels vary in an image and to what extent their distribution is biased towards black or white. The final roughness feature is given by summing coarseness and contrast values.

Finally, we extracted third order statistical features of skewness and kurtosis. Skewness measures the lack of symmetry, which is medically significant as a heavy bias toward lower values is associated with the presence of a lesion, especially in ADC maps. Kurtosis measures whether the data is heavy-tailed or light-tailed compared to a normal distribution. This can help us determine the impact of outlier signal intensities within a sample. Once a feature was calculated for each lesion, the feature vector was appended as another column to an existing feature matrix and used as input to classifiers.

4.6 Non-Deep Learning Classifiers

For our method, each zone had an independent classification pipeline, meaning that each classifier did not incorporate any data between different zones. For this section, we evaluated the performance of

Texture Feature	Formula
Haralick Angular Second Moment	$\sum_i \sum_j p(i,j)^2$
Haralick Contrast	$\sum_i \sum_j (i-j)^2 p_d(i,j)$
Haralick Correlation	$\frac{\sum_i \sum_j (ij)p(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y}$
Haralick Dissimilarity	$\sum_{i,j=0} P_{i,j} \lvert i-j \rvert$
Haralick Energy	$\sqrt{\text{ASM}}$
Haralick Homogeneity	$\sum_i \sum_j \frac{1}{1+(i-\mu)^2} p(i,j)$
Tamura Coarseness	$\sum_{i=x-2^{k-1}}^{x+2^{k-1}-1} \sum_{j=y-2^{k-1}}^{y+2^{k-1}-1} f(i,j)/2^{2k}$
Tamura Contrast	$\sigma/(\alpha_4)^n$ where $\alpha_4 = \mu_4/\sigma^4$
Tamura Roughness	Coarseness + Contrast

Tab. 2: Formulas to calculate Tamura and Haralick features from a Gray-Level Co-occurrence Matrix (GLCM).

non-deep learning classifiers such as Logistic Regression, Support Vector Machine (SVM), Random Forest, and XGBoost. The results of these classifiers were compared with each other and with the deep learning classifier. Except for XGBoost [8], all non-deep learning classifier algorithms were provided by the python library Scikit-learn [26].

Logistic Regression A logistic regression model is formulated mathematically by relating the probability of some event conditional on the vector of explanatory variables to attain a binary output at a given threshold value. Some of the variables may be useful for the classification, but when combined with other variables, they may be ineffective if they are highly correlated with other variables. Variables that contribute redundant information can be omitted so that only those variables with the best diagnostic accuracy are retained for the model. For our purposes, a L1/Lasso regularization method was first applied as a feature selection technique to select for significant features with non-zero coefficients to decrease noise and to be used in classification. Then, a 10-fold cross validation was performed on training zone data for hyperparameter tuning. In the end, the logistic regression algorithm used was set to have a L1 penalty and liblinear solver. This setting was used for all zones as a baseline classifier.

Support Vector Machine An SVM generates maximal margin hyperplanes which separates the data into different groups. If the data is not linearly separable, then a kernel function is used to move the data into a higher dimension so that separation may be easier in that space. Since the number of observations were greater than the number of features for our data, a nonlinear Gaussian Radial Basis Function (RBF) kernel was used with a C value of 0.05. This setting was used for all zones.

Random Forest The Random Forest algorithm consists of many individual decision trees that operate as an ensemble. Each individual tree in the random forest produces a class prediction and the class with the most votes becomes our model's prediction. The main advantage with using random forests is that many relatively uncorrelated models (trees) operating as a committee outperforms any of the individual constituent models. While each deep tree in the random forest tends to overfit (leading to low bias and high variance), averaging relatively uncorrelated high variance models via random forest produces a much

	Hyperparameters					
Zone	Colsample_bytree	Gamma	Eta	Max depth	N estimators	Subsamples
PZ	0.73	0.009	0.058	4	122	0.63
TZ	0.70	0.255	0.155	2	132	0.65
AS	0.71	0.013	0.143	2	117	0.99

Tab. 3: XGBoost hyperparameters were tuned using three-fold cross validation in each zone: peripheral zone (PZ), transition zone (TZ), and anterior fibromuscular stroma (AS).

lower variance ensemble to optimize the bias-variance tradeoff. The best hyperparameters were determined manually so that the function criterion to measure the quality of a split was 'entropy' and the number of trees in the forest were 1000. Entropy here refers to criterion for calculating information gain, which decision trees in the random forest use to split a node and determine the structure of the tree in a way that reduces entropy. This setting was used for all zones.

XGBoost XGBoost is an implementation of a gradient boosted tree algorithm. Gradient boosting refers to the sequence of classification models that are learned from the data in which each classifier gives higher weight to incorrectly classified instances. This weight allows the next classifier in the sequence to more likely sample those instances with higher weights. In contrast to random forest, sequential methods such as XGBoost are used to train an ensemble of high bias models (shallow trees in this case) to increase the variance and decrease the bias. Due to the nature of this ensemble, this technique is highly prone to overfitting, but the learning rate (eta), gamma, and subsampling parameters can be adjusted to control overfitting. During hyperparameter tuning, A few other parameters were also adjusted. The Colsample_bytree parameter is the subsample ratio of columns when constructing each tree. Max depth is the maximum depth of the tree with increased depth leading to increased complexity and chance of overfitting. Subsample is the subsample ratio for the training instances used, in which lower values signifies that XGBoost would randomly sample less of the training data prior to growing trees, preventing overfitting. Finally, n_estimators is the number of trees used. To automatically tune hyperparameters on the training set, three-fold cross validation was performed and Scikit-learn's RandomizedSearchCV was used to return the best hyperparameters for each independent zone. Refer to table 3 for the hyperparameter values set for each zone.

4.7 Deep Learning Classifier

A deep learning convolutional neural net (CNN) was also established for each zone. The CNN assigns importance (learnable weights and biases) to various image features in the image to be able to detect the presence of a lesion. In this paper, our architecture (Fig. 1) was composed of various layers with an input of 16x16 T2WI MRIs as ADC images were too small to be used as input. Convolutional layers were the core part of this model as they allowed the the neural net to essentially extract features from an input image through many filters. Then, the output of the convolutional layer was processed by a batch normalization (BN) layer. This was important as it ensured that each layer saw the same distribution of inputs from the previous layer at each iteration of training, thus learning happened at faster rate as the model did not also have to deal with distributional (covariate) shifts between iterations. After this layer, a nonlinear Rectified Linear Unit (ReLU) activation function was used to process the basic outputs of the neuron and allowed for complex mappings of outputs to the next layer of the architecture. The purpose of specifically using ReLU was to provide both a nonlinearity in addition to a constant gradient (unlike a tanh activation function). After this, a max pooling layer was used to reduce the number of extracted features and to avoid

overfitting. In addition to BN layers, dropout layers also prevented overfitting and improved training time, albeit through a different mechanism. Thus, we decided to experiment with them and concluded with slightly improved results in doing so [13]. Finally, we included an Adam optimizer and a cross entropy loss function as defined by Xmasnet [23].

5 Results

The performance of each classifier in its respective zone was evaluated through a Receiver Operating Charactistic (ROC) curve, Precision Recall Curve (PRC), and F1 score. The ROC curve is defined as a plot of true-positive ratio (TPR) against false-positive ratio (FPR) when the threshold c moves on a real number line. The area under the ROC curve (AUC) was taken as a measure of separability for binary lesion classification. However, since the data set was imbalanced with 25% positives and 75% negatives, this was accounted for by using PRCs and F1 scores, which do not introduce true negative results in their calculation. This allowed for the accuracy of measuring true positives to be better interpreted compared with the AUC of a ROC curve. The F1 score is determined by calculating the harmonic mean between precision (positive predictive value) and recall (sensitivity) to combine these two statistics into one value. The PRC plots the precision and recall at different threshold values into a graphical form so that the AUC can be taken. ROC and PRC curves with their corresponding AUC scores and F1 scores can be found in appendices A, B, and C.

5.1 Salient Features for Non-Deep Learning Classifiers

After the best classification result was achieved on the test set of a zone, the coefficient vector for each classifier (except for SVM) was used to generate a bar graph and salient features used for classification were reported. A coefficient vector could not be taken from SVM because a nonlinear kernel was used, and the mapping function could not be analytically determined from an infinite-dimensional transformed space.

Logistic regression The logistic regression algorithm mainly used second and third order statistical features. Specifically, there was a high preference for texture features such as Haralick correlation and Tamura coarseness and third order statistics such as skewness and kurtosis. There was not a preference for either T2WI or ADC maps as both were highly prioritized for their respective features.

Random forest The random forest algorithm performed the best using first and second order statistical features. Specifically, there was a high preference for 10% and average intensity values and second order texture features such as Haralick dissimilarity and contrast. There was not a preference for either MRI type as both were highly prioritized for the features mentioned.

XGBoost The XGBoost algorithm utilized all orders of features to derive the best performing model. From ADC images, 10% intensity values were found to salient as were Haralick energy and ASM. From T2WIs, Haralick correlation and Tamura roughness were found to be salient. From both MRI types, features included average signal intensity and skewness.

5.2 Salient Features for each Zone

There were various trends noted for which salient features corresponded to each zone to attain an accurate classification.

Peripheral zone Features that were salient for PZ lesions included first order statistical features such as 10% and average intensity as well as second order texture features such as Tamura courseness and Haralick energy and dissimilarity. There was not a significant preference for either MRI type, except for Haralick features where ADC maps were more significant.

Transition zone Features that were salient for TZ lesions included ADC 10% and second order texture features such as Tamura roughness and Haralick, correlation, contrast, and dissimilarity. There was no preference for either MRI type for texture features.

Anterior fibromuscular stroma zone Features that were salient for AS lesions included first order features such as 10% and average intensity and third order features such as skewness and kurtosis. There was no preference for either MRI for salient features.

Throughout all the features, 10% and average intensity as well as skewness contributed the most information for classification, regardless of MRI type.

6 Discussion

We will discuss the experimental results from comparing non-deep learning and deep learning classifiers for prostate lesion detection with respect to the region of the prostate that samples were extracted from.

It has been reported that a radiologist following the Prostate Imaging-Reporting and Data System (PI-RADS) to detect lesions from mpMRIs can achieve an AUC of 0.81 to 0.84 [19, 18, 28]. It seems that a few base classifiers without any hyperparameter tuning can attain a similar perfomance. While the radiologist-like performance is already satisfactory, we believe that there is more room to improve. After extensive hyperparameter tuning on training sets, the models from cross validation transferred over relatively well to the test set and achieved AUC values that were significantly better than the baseline performance. It seems that some non-deep learning classifiers such as logistic regression, random forest, and XGBoost were able to achieve a higher accuracy through hyperparameter tuning. Overall, Xgboost performed similar if not better compared with Random Forest. Logistic Regression with L1 regularization was slightly worse, but still performed better than a well-trained radiologist would in lesion detection. Finally, SVM with a radial basis function (RBF) kernel performed with metrics close to random chance. The poor perfomance of SVM can be attributed to having many heterogenous and independent features as input. The RBF kernel gives equal weighting to noisy and informative features, thus decreasing the performance of the model. However, SVM has proven to achieve great results in prostate lesion detection [3, 36]. Hopefully, future studies may augment the data used in this study in such a way to decrease noise and increase the performance of SVM. Even with optimal tuning, these classifiers achieved various results in different zones, which can be explained by the increased medical significance of some features over others.

Peripheral zone For lesions present in the peripheral zone (PZ), first and second order features have the most medical significance for lesion determination as round or ill-defined low-signal intensity masses are an indication of a lesion in both ADC and T2WI MRIs [17]. Healthy peripheral zone tissue gives off the highest signal intensity out of all regions owing to its high-water content. Thus, it is relatively easier to diagnose lesions in this zone. Generally, there is also a higher rate of interobserver agreement for PZ lesions as reported by Schoots et al. [30], which is reflected by the highest performance measurements of the area under ROC and PRC for classifiers in this zone. In regards to the best classifiers of this zone, ensemble algorithms such as random forest and xgboost performed significantly better than others as it combines multiple models together and delivers superior prediction power. Surprisingly, F1 scores were significantly lower than area under PRC throughout all the zones, especially in PZ. We discovered that models tend to classify more lesions as positives as evidenced by the high recall and low precision for positive results. While misclassification is present, from a medical standpoint, it is much more beneficial to detect a possible lesion as a false positive result than to completely miss the lesion as a false negative.

Transition zone For lesions present in the transition zone (TZ), first order statistics and second order texture features influenced classification the most. The pixel intensities found in this zone have a more heterogenous distribution for healthy tissue, emphasizing the importance of texture analysis. Lesions in TZ appear often as a homogenous mass possessing ill-defined edges with lenticular or "water-drop" shapes on a T2WI [2, 4]. However, not all lesions are obvious in a heterogenous background, making the classification

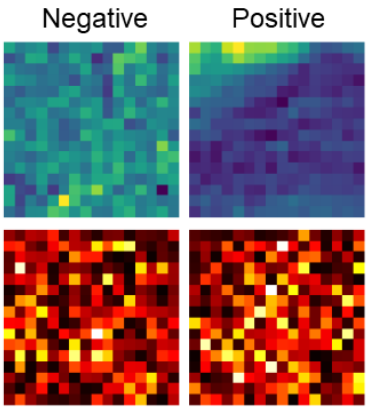


Fig. 3: Saliency Map from AS zone to interpret the high performance of the deep learning model. Bottom images are the saliency maps derived from the top T2WI images, in which orange to white colors indicate high gradients while darker colors indicate low gradients. Based on the gradients, we can determine which pixels contributed most to the CNN's decision of classification.

problem more difficult than in PZ. The classifiers' performances reflect this issue, having the lowest classifier performance metrics of all zones. Nonetheless, XGBoost performed significantly better than random forest in this zone compared with other zones. This is likely due to the correlated texture features having a beneficial, correctional effect during the iterative training process in XGBoost. For TZ lesion detection, we believe that more texture features could be incorporated in future studies to give more information to a model to train on. However, proper regularization techniques would still need to be utilized to reduce noise.

Anterior fibromuscular stroma zone For lesions present in the anterior fibromuscular stroma zone (AS), first and third order statistical features influenced classification the most. Lesions in this area present with a significant hypointensity, lower than that typically expected for other zones [37]. Thus, it is understandable that texture features are not as salient and features that directly relate to signal intensity are much more important in classification. Most importantly, the deep learning CNN performed the best in AS and overall as it was not limited by the pre-extracted features provided to all non-deep learning models. It was able to learn its own feature set and classify based off of that. To interpret the salient features the CNN used in classification, a saliency map was generated to explain the high performing classifier (Fig. 3). From this image, we can understand that the model clearly recognizes the shape and hypointensity of lesions found in the AS zone. Other than deep learning CNNs, performance in this zone is mid-level with metric values mostly falling between those of other zones. Another surprising finding is that random forest performed better than XGBoost with respect to resulting metrics. This is likely due to salient feature independence that benefits the independence of decision trees within the random forest in contrast to what was seen in TZ. Another important result to note is that the area under the ROC curve for logistic regression is significantly worse than that of other zones, demonstrating the lowered ability to differentiate between positives and negatives. However, it also has a significantly higher area under the PRC than other zones, indicating the increased accuracy of predicting true positives. This means that if the logistic regression classifier labels a lesion as positive, it has a higher chance of being a true positive compared to the same classifier found in other zones.

7 Conclusion and Future Work

In this study, we uncovered the best type of classifier and salient features for each zone of the prostate. For the PZ and TZ zones, ensemble algorithms performed the best while the deep learning CNN soared in performance in the AS zone. We also discovered that the PZ and AS zones were more dependent on statistical features while the TZ zone was more dependent on textural features. The development of CAD

systems for prostate lesion detection holds to be the most promising noninvasive imaging diagnostic rationale which leads to high accuracy of detection and minimizes human error. This would allow for medical professionals to promptly begin treatment after lesion detection and grading to deliver the appropriate level of treatment for a patient to end in a favorable prognosis. While the results were encouraging, we believe that classification models used in this study can be improved with more diverse samples from other data sets to not only validate the results of this study but also improve the performance of these models with more rigorous training. Another limitation that prevented better CNN classification was that this model was not pre-trained on existing images, but rather an original creation inspired by other models and trained on the augmented images within this dataset. If more datasets can be acquired and augmented in a similar way to provide more samples for the CNN to train on, then the performance of the CNN will be improved for more reliable and accurate prostate lesion detection as a CAD system.

Since we determined salient features on a per zone basis, future studies may utilize these indicators to build reliable and accurate supervised machine learning classifiers for whole prostate lesion detection to avoid the blackbox of CNNs altogether. We hope that our work will provide a more standardized basis regarding which algorithms work best for each zone of the prostate and provide future direction in investigation of improving the types of machine learning classifiers highlighted for implementation of CAD systems.

8 Acknowledgements

I would like to offer my special thanks to the SSI staff for providing this amazing opportunity to learn advanced concepts and apply them in my first time in research. Their willingness to sacrifice their summer to organize this science institute is very much appreciated. I am particularly grateful for the assistance given by my mentor, Nithin Buduma, for helping me brainstorm ideas, overcome any technical issues I had with my dataset or code, and explain high level concepts for my understanding and implementation in this paper. Finally, I would like to thank my lab group for supporting my ideas, delivering helpful feedback on my presentations, and asking insightful questions.

References

[1] Machine learning in cancer diagnostics. *EBioMedicine*, 45:1–2, jul 2019.

[2] Oguz Akin, Evis Sala, Chaya S. Moskowitz, Kentaro Kuroiwa, Nicole M. Ishill, Darko Pucar, Peter T. Scardino, and Hedvig Hricak. Transition zone prostate cancers: Features, detection, localization, and staging at endorectal MR imaging. *Radiology*, 239(3):784–792, jun 2006.

[3] Yusuf Artan, Masoom A Haider, Deanna L Langer, Theodorus H van der Kwast, Andrew J Evans, Yongyi Yang, Miles N Wernick, John Trachtenberg, and Imam Samil Yetik. Prostate cancer localization with multispectral MRI using cost-sensitive support vector machines and conditional random fields. *IEEE Transactions on Image Processing*, 19(9):2444–2455, sep 2010.

[4] Jelle O. Barentsz, Jonathan Richenberg, Richard Clements, Peter Choyke, Sadhna Verma, Geert Villeirs, Olivier Rouviere, Vibeke Logager, and Jurgen J. Fütterer. ESUR prostate MR guidelines 2012. *European Radiology*, 22(4):746–757, feb 2012.

[5] David Bonekamp, Simon Kohl, Manuel Wiesenfarth, Patrick Schelb, Jan Philipp Radtke, Michael Götz, Philipp Kickingereder, Kaneschka Yaqubi, Bertram Hitthaler, Nils Gählert, Tristan Anselm Kuder, Fenja Deister, Martin Freitag, Markus Hohenfellner, Boris A. Hadaschik, Heinz-Peter Schlemmer, and Klaus H. Maier-Hein. Radiomic machine learning for characterization of prostate lesions with MRI: Comparison to ADC values. *Radiology*, 289(1):128–137, oct 2018.

[6] Ian Chan, William Wells, Robert V. Mulkern, Steven Haker, Jianqing Zhang, Kelly H. Zou, Stephan E. Maier, and Clare M. C. Tempany. Detection of prostate cancer by integration of line-scan diffusion, t2-mapping and t2-weighted magnetic resonance imaging a multichannel statistical classifier. *Medical Physics*, 30(9):2390–2398, aug 2003.

[7] Quan Chen, Shiliang Hu, Peiran Long, Fang Lu, Yujie Shi, and Yunpeng Li. A transfer learning approach for malignant prostate lesion detection on multiparametric MRI. *Technology in Cancer Research & Treatment*, 18:153303381985836, jan 2019.

[8] Tianqi Chen and Carlos Guestrin. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, aug 2016.

[9] Kenneth Clark, Bruce Vendt, Kirk Smith, John Freymann, Justin Kirby, Paul Koppel, Stephen Moore, Stanley Phillips, David Maffitt, Michael Pringle, Lawrence Tarbox, and Fred Prior. The cancer imaging archive (TCIA): Maintaining and operating a public information repository. *Journal of Digital Imaging*, 26(6):1045–1057, jul 2013.

[10] Henry R. Ehrenberg, Daniel Cornfeld, Cayce B. Nawaf, Preston C. Sprenkle, and James S. Duncan. Decision forests for learning prostate cancer probability maps from multiparametric MRI. In Georgia D. Tourassi and Samuel G. Armato, editors, *Medical Imaging 2016: Computer-Aided Diagnosis*. SPIE, mar 2016.

[11] Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118, jan 2017.

[12] Mike Folk, Gerd Heber, Quincey Koziol, Elena Pourmal, and Dana Robinson. An overview of the HDF5 technology suite and its applications. In *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*. ACM Press, 2011.

[13] Christian Garbin, Xingquan Zhu, and Oge Marques. Dropout vs. batch normalization: an empirical study of their impact to deep learning. *Multimedia Tools and Applications*, 79(19-20):12777–12815, jan 2020.

[14] Sangeet Ghai and Masoom A Haider. Multiparametric-mri in diagnosis of prostate cancer. *Indian Journal of Urology*, 31(3):194–201, 2015.

[15] Robert M. Haralick, K. Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, nov 1973.

[16] Yoichi Hayashi. New unified insights on deep learning in radiological and pathological images: Beyond quantitative performances to qualitative interpretation. *Informatics in Medicine Unlocked*, 19:100329, 2020.

[17] H Hricak, RD Williams, DB Spring, KL Moon, MW Hedgcock, RA Watson, and LE Crooks. Anatomy and pathology of the male pelvis by magnetic resonance imaging. *American Journal of Roentgenology*, 141(6):1101–1110, dec 1983.

[18] Daniel Junker, Georg Schäfer, Michael Edlinger, Christian Kremser, Jasmin Bektic, Wolfgang Horninger, Werner Jaschke, and Friedrich Aigner. Evaluation of the PI RADS scoring system for classifying mpMRI findings in men with suspicion of prostate cancer. *BioMed Research International*, 2013:1–9, 2013.

[19] Moritz Kasel-Seibert, Thomas Lehmann, René Aschenbach, Felix V. Guettler, Mohamed Abubrig, Marc-Oliver Grimm, Ulf Teichgraeber, and Tobias Franiel. Assessment of PI-RADS v2 for the detection of prostate cancer. *European Journal of Radiology*, 85(4):726–731, apr 2016.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, may 2017.

[21] Geert Litjens, Oscar Debats, Jelle Barentsz, Nico Karssemeijer, and Henkjan Huisman. Computer-aided detection of prostate cancer in MRI. *IEEE Transactions on Medical Imaging*, 33(5):1083–1092, may 2014.

[22] Geert Litjens, Oscar Debats, Jelle Barentsz, Nico Karssemeijer, and Henkjan Huisman. Spie-aapm prostatex challenge data, 2017.

[23] Saifeng Liu, Huaixiu Zheng, Yesu Feng, and Wei Li. Prostate cancer diagnosis using deep learning with 3d multiparametric MRI. In Samuel G. Armato and Nicholas A. Petrick, editors, *Medical Imaging 2017: Computer-Aided Diagnosis*. SPIE, mar 2017.

[24] L.G. Nyul, J.K. Udupa, and Xuan Zhang. New variants of a method of MRI scale standardization. *IEEE Transactions on Medical Imaging*, 19(2):143–150, 2000.

[25] Rufus W Ojewola, Emmanuel A Jeje, Kehinde H Tijani, Moses A Ogunjimi, and Charles C Anunobi. Clinico-pathological correlation of digital rectal examination findings amongst nigerian men with prostatic diseases: A prospective study of 236 cases. *Nigerian Journal of Surgery*, 19(1):26–31, 2013.

[26] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python.

[27] Yahui Peng, Yulei Jiang, Cheng Yang, Jeremy Bancroft Brown, Tatjana Antic, Ila Sethi, Christine Schmid-Tannwald, Maryellen L. Giger, Scott E. Eggener, and Aytekin Oto. Quantitative analysis of multiparametric prostate MR images: Differentiation between prostate cancer and normal tissue and correlation with gleason score—a computer-aided diagnosis development study. *Radiology*, 267(3):787–796, jun 2013.

[28] Stephan Polanec, Thomas H. Helbich, Hubert Bickel, Katja Pinker-Domenig, Dietmar Georg, Shahrokh F. Shariat, Wolfgang Aulitzky, Martin Susani, and Pascal A. Baltzer. Head-to-head comparison of PI-RADS v2 and PI-RADS v1. *European Journal of Radiology*, 85(6):1125–1131, jun 2016.

[29] P. Puech, A. Sufana Iancu, B. Renard, A. Villers, and L. Lemaitre. Detecting prostate cancer with MRI — why and how. *Diagnostic and Interventional Imaging*, 93(4):268–278, apr 2012.

[30] Ivo G. Schoots. MRI in early prostate cancer detection: how to manage indeterminate or equivocal PI-RADS 3 lesions? *Translational Andrology and Urology*, 7(1):70–82, feb 2018.

[31] Jonathan L. Silberstein, Sumanta Kumar Pal, Brian Lewis, and Oliver Sartor. Current clinical challenges in prostate cancer. *Translational Andrology and Urology*, 2(3), 2013.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition.

[33] Samuel K. Stephenson, Edward K. Chang, and Leonard S. Marks. Screening and detection advances in magnetic resonance image guided prostate biopsy. *Urologic Clinics of North America*, 41(2):315–326, may 2014.

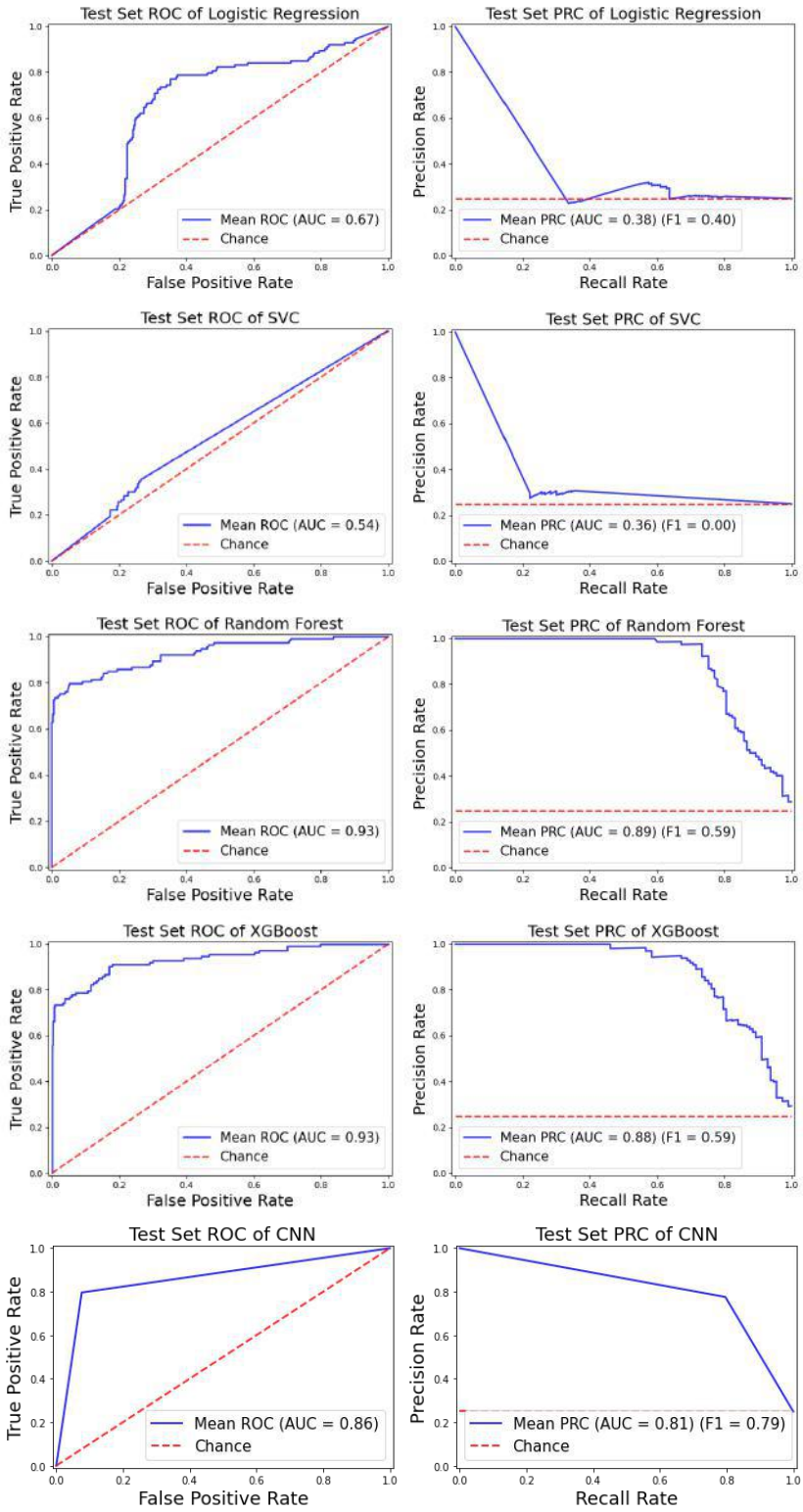
[34] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(6):460–473, 1978.

[35] Baris Turkbey and Peter L. Choyke. Future perspectives and challenges of prostate MR imaging. *Radiologic Clinics of North America*, 56(2):327–337, mar 2018.

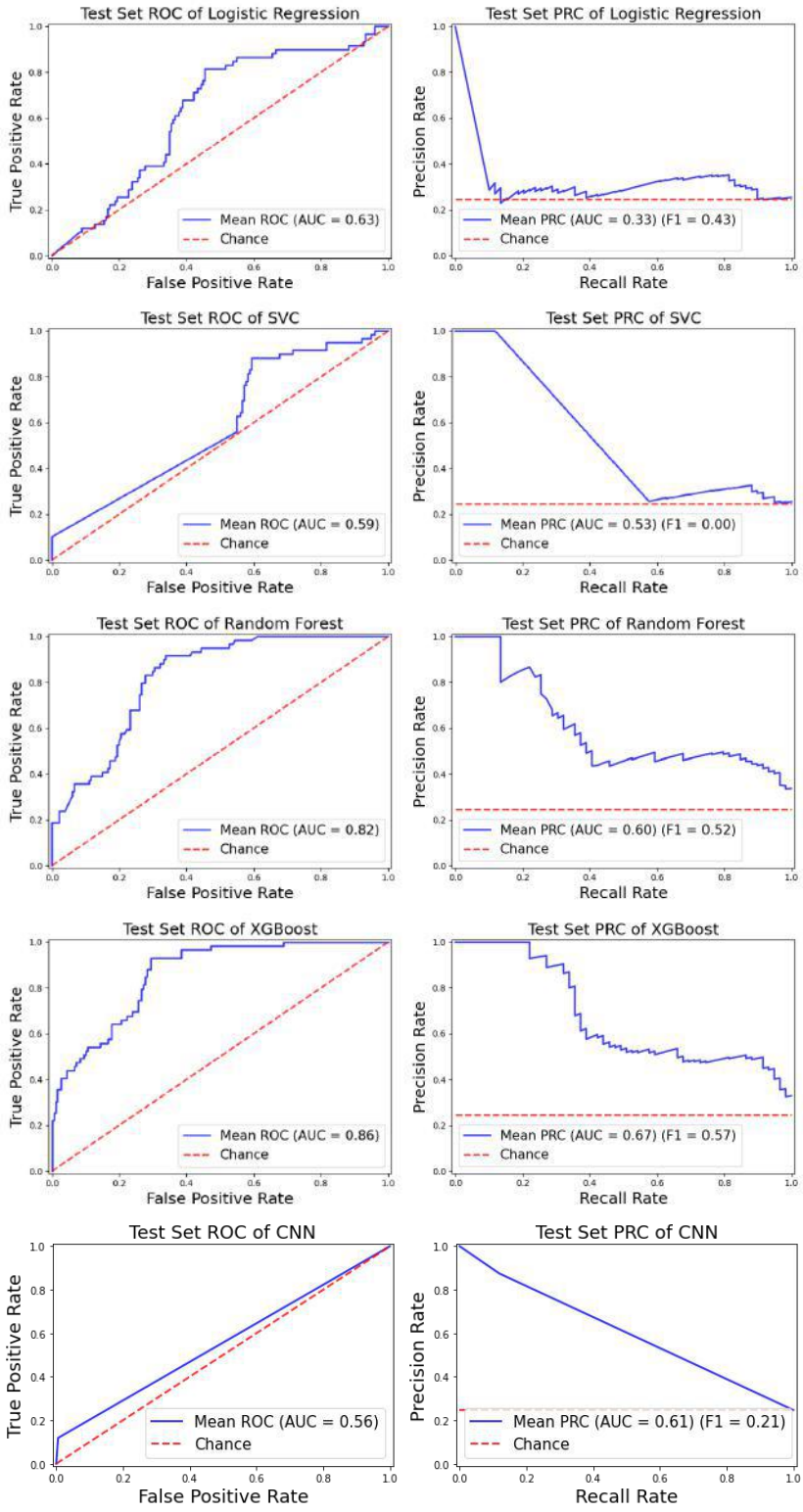
[36] Jing Wang, Chen-Jiang Wu, Mei-Ling Bao, Jing Zhang, Xiao-Ning Wang, and Yu-Dong Zhang. Machine learning-based analysis of MR radiomics can help to improve the diagnostic performance of PI-RADS v2 in clinically relevant prostate cancer. *European Radiology*, 27(10):4082–4090, apr 2017.

[37] Jinxing Yu, Ann S Fulcher, Sarah G Winks, Mary A Turner, Ryan D Clayton, Michael Brooks, and Sean Li. Diagnosis of typical and atypical transition zone prostate cancer and its mimics at multiparametric prostate MRI. *The British Journal of Radiology*, 90(1073):20160693, may 2017.

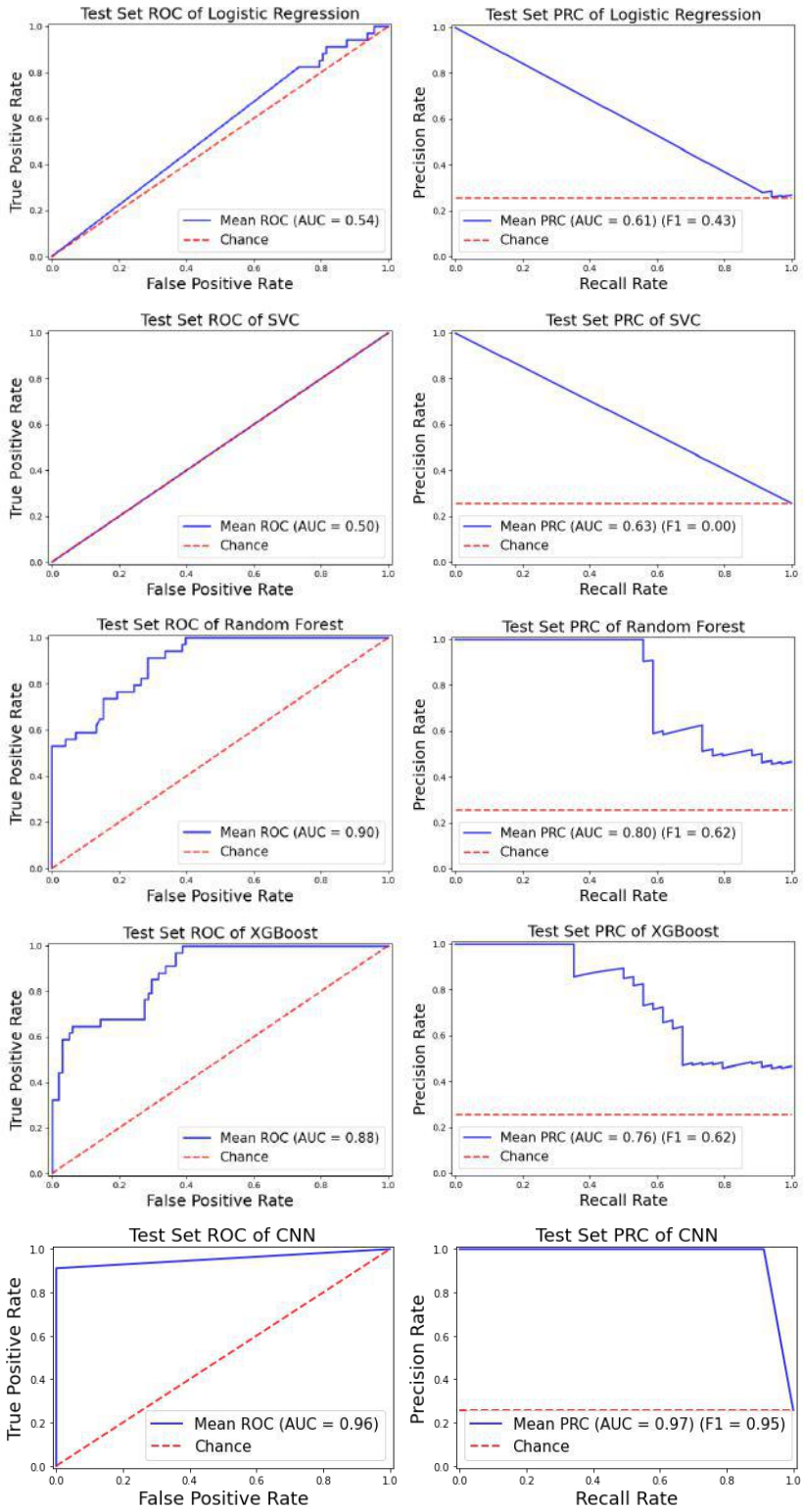
A Peripheral Zone Graphs



B Transition Zone Graphs



C Anterior Fibromuscular Stroma Zone Graphs



Implementing Quantum Error Correcting Codes on the IBM Melbourne Quantum Computer

Ali Hindy

Abstract

We study the performance of different circuits implementing the Shor code circuit, a quantum error correcting code that can correct arbitrary single qubit error. Assuming unbiased noise, we implement and compare the performance of two different circuits implementing the Shor Code , one with an intermediate syndrome measurement and one without. We show that the Shor code without intermediate syndrome measurements can provide sufficient protection against arbitrary single qubit errors. Implementing the Shor Code using IBM Qiskit, we calculated the efficacy of both circuits. We also derived an upper bound for the probability of error occurring across the circuit and optimized the circuit in order to reduce this upper bound.

1 Introduction

Quantum Error Correcting Codes is a sub-field of quantum computing generally related to correcting the error that real quantum computers make when executing computations. Quantum computers have the potential to analyze data much faster than classical computers, and to discover new medicines, predict stocks, improve cybersecurity, and simulate complicated chemical equations [8]. However, quantum computers must run under very low temperatures. If the temperature rises, the computer may make a small error, which can eventually snowball into massive errors that completely ruin results [1].

A quantum circuit is similar to a classical circuit in the sense that quantum bits (qubits) pass through gates (matrix operations) analogous to the classical gates NOT, AND, OR. One such circuit is a quantum error correcting code called the Shor Code, which encodes a single qubit state into a nine qubit state [9].

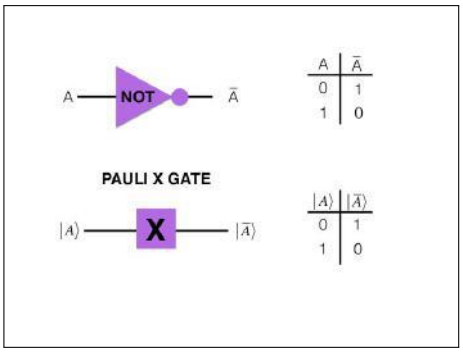


Fig. 1: A Pauli X gate, which is analogous to the classical NOT gate

Instead of 0s and 1s (classical bits), quantum computers operate off of $|0\rangle$s and $|1\rangle$s (qubits) and generally rely on quantum entanglement in order to operate. $|0\rangle$ and $|1\rangle$ (Dirac notation) are the basis vectors for two-dimensional complex space. Quantum entanglement is the property of two or more qubits being highly correlated, meaning they are in a quantum state that generally can not be written as a product state of independent single qubits. An example of an entangled state is the 2 qubit $|\Phi^+\rangle$ Bell State.

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

In order to construct this Bell State, a quantum gate called a CNOT gate, can be applied to two qubits. Quantum gates are matrices that can be applied to one or more qubits. With multiple quantum gates, a quantum circuit can be formed. Quantum entanglement allows multiple qubits in an entangled state to be acted upon simultaneously, which is not possible in classical computing as classical bits can only have one state at a time. A physical qubit is a qubit on hardware while a logical qubit is an abstract (not necessarily hardware or software) qubit. One such application of quantum entanglement is with the Shor Code, which encodes one logical qubit onto nine physical qubits so that if an error on a single qubit occurs, we can figure out what the nine qubit state looked like.

Quantum error correcting codes work by using ancilla (helper) qubits in order to add redundancy to the original data so that if some data gets corrupted, the original data can still be recovered. Our implementation of Quantum Error Correction works in three steps: a single qubit is encoded onto multiple other qubits, then random error is simulated, and finally the error is simultaneously corrected and decoded to recover the original state. In the case of the Shor Code, $|0\rangle$ and $|1\rangle$ are encoded as follows:

$$|0_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)$$

$$|1_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle)$$

In both of these equations, a single logical qubit is encoded onto nine physical qubits, which is the first step of quantum error correction. The states are encoded using tensor ($\otimes$) products. Each of the $(|000\rangle + |111\rangle)$ terms refers to the groups of qubits (1,2,3), (4,5,6), and (7,8,9), which is used to correct X errors. With these nine qubits, if one mistake occurs, we can correct it using the decoding circuit. The next step of quantum error correction is to simulate error by doing some arbitrary random operation. The Shor Code can correct any arbitrary single qubit error, which can be composed by an X, bit flip error, and a Z, phase flip error. An X error turns a $|0\rangle$ into a $|1\rangle$ and a $|1\rangle$ into a $|0\rangle$, while a Z error turns a $|0\rangle$ into a $|0\rangle$ and a $|1\rangle$ into a $-|1\rangle$ These two types of error correspond to two Pauli matrices:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Every error matrix can be written as a linear combination of I,X,Z,and XZ, and, the Shor Code can hence correct every type of arbitrary single qubit error.

1.1 The Project

The goal of this project is to implement the Shor Code on a real quantum computer, the IBM Melbourne, using Python packages such as IBM Qiskit in order to measure if the theoretical efficacy is the same as the practical efficacy. We plan on contributing to IBM Ignis, an open source Python framework for quantum computing that does not currently include the Shor Code. There also exist in the literature other quantum error correcting codes including the Repetition Code and the Bacon-Shor Code, but the only one in the IBM Ignis documentation is the Repetition Code.

Figure 2 depicts the Shor Code circuit, which includes the encoding, random error (E), and decoding. This circuit does not include intermediate measurements, and only measures the first qubit at the end of all the operations. As seen in Figure 2, a quantum circuit is comprised of matrix operations such as the CNOT gate and the Hadamard gate.

The IBM Melbourne is a fifteen qubit quantum computer in Melbourne, Australia. The quantum computer was made publicly available only recently, which has led to a spike in the number of research papers that practically implement quantum algorithms. The IBM Melbourne has the greatest number of qubits that IBM publicly offers. Novel quantum technology called Noisy Intermediate State Quantum (NISQ) (50-100 qubit devices) is not publicly available; however, it has the potential to completely change the field of quantum computing.[7] .[4] IBM computers like the IBM Rochester have powerful capabilities for the future of quantum computing. Furthermore, quantum error correcting codes like the Shor Code

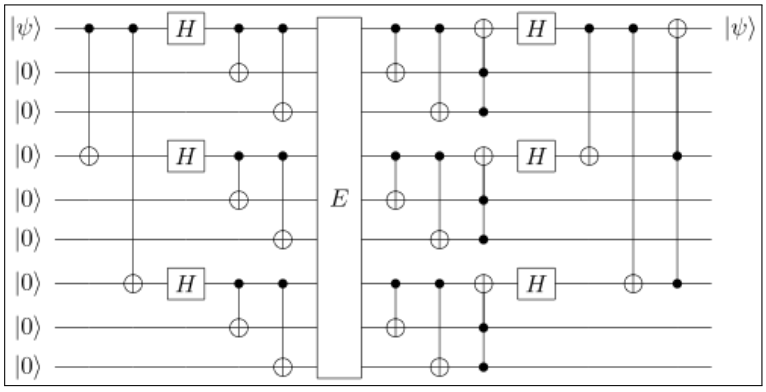


Fig. 2: The Shor Code circuit

will become significantly more important in reducing noise on a quantum circuit. NISQ technology will facilitate novel cybersecurity methods, the discovery of new medicines, and much more. By implementing the Shor Code on a fifteen qubit device, progress will be made towards making quantum computations more precise.

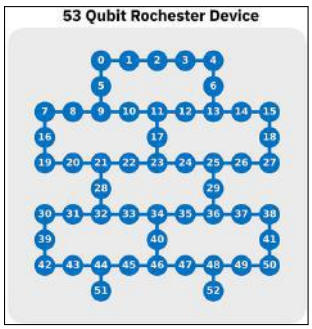


Fig. 3: The IBM Rochester, IBM's largest quantum computer (not publicly available)

Noise models are a useful tool for simulating the error on real quantum computers. We consider noise models such that qubits in the code block are subject to both bit flip (X) errors and phase (Z) errors, where the bit flips occur with probability p_X and phase errors occur with probability p_Z. We assume that the noise acts independently on each qubit,and that the X and Z errors are uncorrelated. We also assume that each error happens with equal probability, which is known as unbiased noise. However, on the real IBM Melbourne, there are other types of error such as crosstalk error that cannot be accurately accounted for in a simulator.

2 Literature Review

Quantum error correction is a relatively new field of science (from the past three decades) and new papers are habitually produced that make progress in the field. However, there are some limitations in the field, which include lacking practical results.[5] Most papers only theorize the quantum error correcting codes instead of actually implementing them, since quantum computers were not widely available until recently. The advent of IBM Qiskit has allowed for an exponential increase in the number of research papers related to practical implementation of previously theoretical ideas.

The IBM Melbourne's architecture provides for connectivity-3 operations, which is ideal for the Shor Code. A connectivity-3 operation is an operation involving connections with 3 qubits, which the Melbourne provides in its base architecture.[4] The IBM Melbourne also comes with a quantum transpiler, which compiles and transforms the circuit in order to minimize the number of gates before it runs on the Melbourne.[3]

For example, the Shor Code has a connectivity-4 qubit on qubit 0, which is not possible on the Melbourne given the architectural constraints. However, the transpiler avoids this issue by adding multiple gates and using extra gates in order to avoid this issue.

We read papers that ran other quantum error correcting codes such as the Bacon-Shor Code on a simulator.[**?**] Such papers numerically calculate the performance of the codes and at best use a simulator. However, in order to test the real performance of quantum error correcting codes, one must run code on a real quantum computer, as opposed to a simulator or theorizing the accuracy.

3 Purpose

1. Learn more about quantum computing / encode a real circuit
2. Contribute to the open source IBM Qiskit Project
3. Successfully encode, run a random noise generator, and decode on the IBM Melbourne

4 Methods

For programming, we used Python coupled with scientific computing packages like Numpy, as well as plotting packages like Matplotlib. Additionally, we used IBM Qiskit, a python library that allows interfacing with real quantum computers, in order to code a quantum circuit. In terms of studying the theory of quantum computing, we read the Nielsen-Chuang textbook, lecture notes from graduate classes at Harvard and the University of Waterloo, as well as more recent papers in quantum computing [2]. We first implemented the Shor Code on the IBM Melbourne with an encoding circuit, error generator, and decoding circuit all in that order. With an input of 0, we measured the probability of each bit occurring. If the input is a 0 and the output is a 1, then we know that the circuit failed to correct the error. Since the IBM Melbourne was down for maintenance for 2 weeks, we used a noise model and simulator for the Melbourne and compared the results.

4.1 Connectivity

Although the Melbourne is a powerful quantum computer, it is limited in the number of connections that certain qubits have with each other. There is a limited number of connections for 2 qubit operations, with some qubits like qubit 7 only being able to execute a 2 qubit operation with qubit 8.

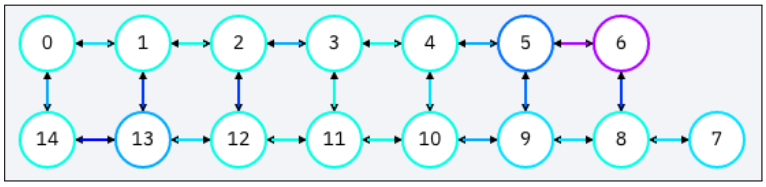


Fig. 4: The Architecture of the IBM Melbourne, with each circle representing a qubit and each line representing a connection for a possible 2-qubit operation

As seen in Figure 7, 3 qubit operations are not possible on the IBM Melbourne. When the transpiler receives a 3 qubit operation such as the Toffoli gate, it must decompose the gate into a series of 2 qubit operations. The Toffoli gate, as given by

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

can be deconstructed into 6 CNOT gates and a combination of H, T, and $T^\dagger$ gates [1]. However, as previously mentioned, the CNOT gate has a high probability of error and as such makes the circuit less efficient in correcting error.

4.2 Circuits

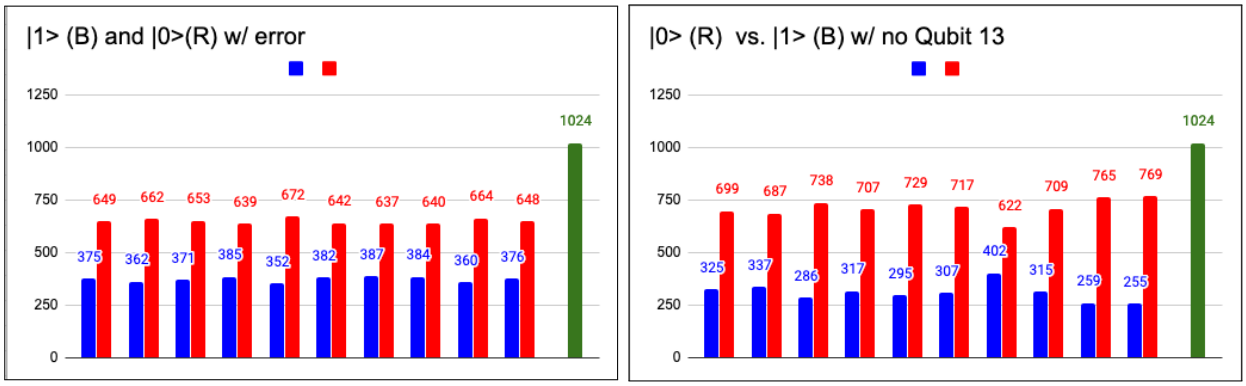
Since the current architectures that IBM provides do not allow for intermediate measurements, we applied the Principle of Deferred Measurement, which states that intermediate measurements can always be moved to the end of the circuit. [6] However, on the simulator, intermediate measurements can be made with the same noise model as the Melbourne. 7 Different experiments were executed:

1. Shor code without added error (no intermediate measurement) on a simulator
2. Shor Code with added error (no intermediate measurement) on a simulator
3. Optimized Shor Code circuit with the added error on a simulator
4. Shor Code with an intermediate measurement without added error
5. Shor Code with an intermediate measurement with added error on a simulator
6. Shor Code without added error on the IBM Melbourne
7. Shor Code with added error on the IBM Melbourne

5 Results and Discussion

5.1 Results

For each circuit, we conducted multiple trials, with each trial consisting of 1024 iterations of the circuit, and graphed the results. Additionally, the upper bound for the probability of error (p_e) was manually calculated.

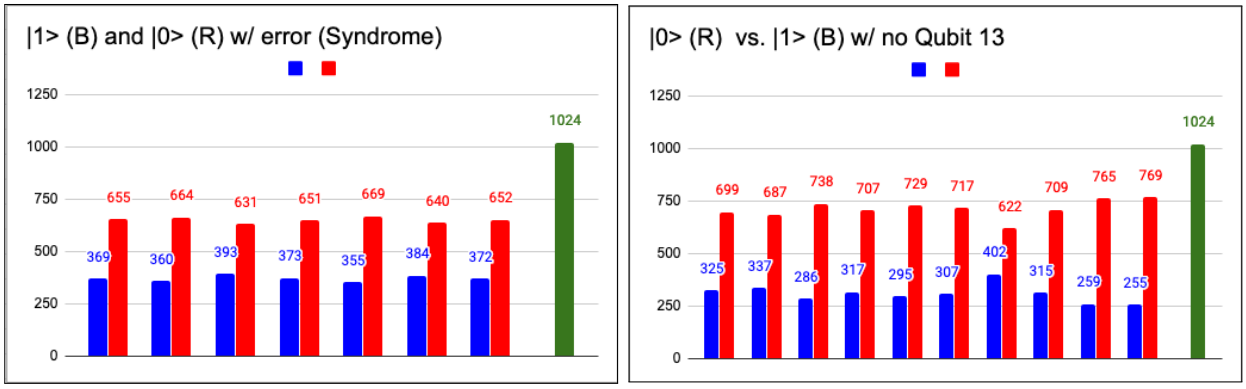


(a) Results with an Error Generator

(b) Results with an Optimized Circuit

Fig. 5: Comparison of Non-Optimized vs. Optimized Circuit Performance

As seen in Figure 4, the results with the optimized circuit on the simulator are better than without the optimized circuit. After 1024 iterations of each circuit, the number of $|0\rangle$s and $|1\rangle$s were measured, with the red bar showing the number of $|0\rangle$s and the blue bar showing the number of $|1\rangle$s. Ideally, the output of the circuuit should all be $|0\rangle$ since the Shor Code is equivalent to the identity matrix for Qubit 0. Each pair of bars represents one experiment, where the circuit was ran 1024 times.In order to optimize the circuit, certain qubits with a high CNOT error rate, such as Qubit 13, were replaced with qubits with a lower CNOT error rate, such as Qubit 8. The probability of error for the non-optimized circuit was 36.45 percent while the probability of error for the optimized circuit was 30.25 percent. The green bar shows the ideal performance, where all error is corrected. However, the Shor Code only corrects for single qubit errors, and other errors such as crosstalk add noise that the Shor Code can not correct.



(a) Results with an Optimized Syndrome Circuit

(b) Results with an Optimized Deferred Circuit

Fig. 6: Comparison of Syndrome vs. Deferred Circuit Performance

As seen in Figure 5, the results with the optimized circuit without an intermediate measurement surpass the results with the optimized circuit with an intermediate measurement. The intermediate syndrome measurement occurs directly after the error occurs, while the deferred measurement happens at the end, which looks like Figure 2. This higher accuracy is most likely due to the higher error rate that measurement operations have on the Melbourne. Additionally, the syndrome measurement circuit has 85 CNOT gates, while the non-syndrome measurement circuit has only 78. CNOT gates have a high probability of error, so higher amounts of CNOT gates decreases the performance of the circuit. The probability of error for the syndrome measurement circuit was 43.8 percent while the probability of error for the optimized non-syndrome circuit was 30.25 percent.

5.2 Dedicated Use / Maintenance

One setback I faced was the dedicated use of the Melbourne for other researchers. More prominent researchers at IBM have first priority to the Melbourne, and during Week 4 they queued hundreds of circuits to the Melbourne, essentially blocking my access to the computer. In order to overcome this issue, I experimented with other circuits such as the 5-qubit stabilizer code and used the noise model and simulator. The noise model captured the error parameters of the IBM Melbourne and applied that to each qubit in the circuit. Additionally, the Melbourne was down for 2 weeks during my research, and as a result I had to turn to the simulator in order to continue getting results. No other publicly avaliable quantum computer has more than 5-qubit capability, meaning that my two options were to use a simulator or try a 5-qubit stabilizer code. However, the 5-qubit stabilizer code requires 4 ancilla qubits, so I could not use any other publicly avaliable IBM architecture.

5.3 Transpiler

On IBM Qiskit, every circuit that runs through a real quantum computer is sent through the transpiler, which changes the circuit in order to match the qubits of the IBM Melbourne. The transpiler is

not optimal for running the circuit, as it merely tries to match the circuit to the qubits of the Melbourne without taking into consideration the number of CNOT gates. Since CNOT gates have a high probability of error, the upper bound for the probability of a non-optimized circuit is 92.48 percent.

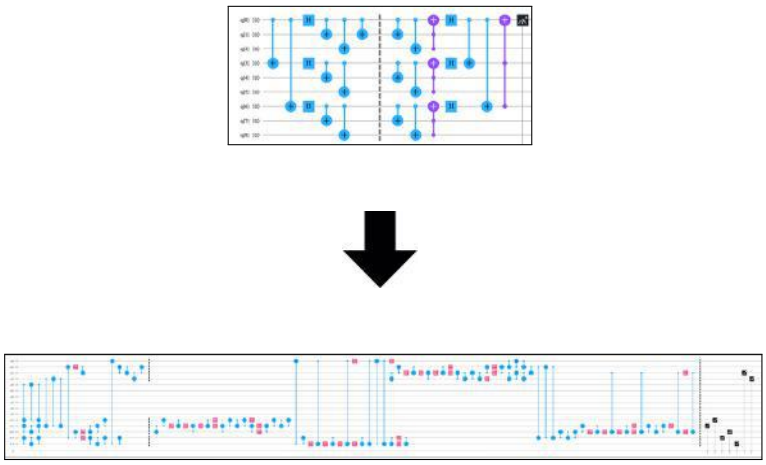


Fig. 7: Initial Circuit (top) vs Transpiled Circuit (bottom)

As seen in Figure 4, the initial circuit has 18 CNOT operations and 4 Toffoli operations, whereas the transpiled circuit has 78 CNOT operations, which drastically increases the probability that error occurs. Additionally, the transpiler uses some qubits (for example, Qubit 13) that have a high CNOT error percentage relative to other qubits. As a consequence, we had to optimize the circuit by trying variations of the transpiled circuit as well as using different qubits with a lower CNOT error rate.

5.4 Architecture Limitations

Although the IBM Melbourne is a state of the art quantum computer, it does not support intermediate syndrome measurements. As such, we could not run the intermediate measurement circuit on the IBM Melbourne and had to use the simulator.

Tab. 1: Error Parameters for the IBM Melbourne

Qubit	Measurement Error	Single Qubit Error	CNOT Error
Q0	2.75E-02	9.59E-04	$\text{cx}0_1 : 2.433e-2, cx01_4 : 4.758e-2$
Q1	5.15E-02	1.10E-03	$\text{cx}1_0 : 2.433e-2, cx1_2 : 1.088e-2, cx1_13 : 6.045e-2$
Q2	2.05E-02	8.02E-04	$\text{cx}2_1 : 1.088e-2, cx2_3 : 2.008e-2, cx2_12 : 3.508e-2$
Q3	6.60E-02	4.49E-04	$\text{cx}3_2 : 2.008e-2, cx3_4 : 1.775e-2, cx3_11 : 3.060e-2$
Q4	3.40E-02	1.70E-03	$\text{cx}4_3 : 1.775e-2, cx4_5 : 2.513e-2, cx4_10 : 2.430e-2$
Q5	6.40E-02	2.54E-03	$\text{cx}5_4 : 2.513e-2, cx5_6 : 6.955e-2, cx5_9 : 5.338e-2$
Q6	2.55E-02	4.18E-03	$\text{cx}6_5 : 6.955e-2, cx6_8 : 1.662e-1$
Q7	5.10E-02	2.85E-03	$\text{cx}7_8 : 3.683e-2$
Q8	2.70E-01	7.11E-04	$\text{cx}8_6 : 1.662e-1, cx8_7 : 3.683e-2, cx8_9 : 3.717e-2$
Q9	5.15E-02	3.18E-03	$\text{cx}9_5 : 5.338e-2, cx9_8 : 3.717e-2, cx9_10 : 5.028e-2$
Q10	2.30E-02	1.15E-03	$\text{cx}10_4 : 2.430e-2, cx10_9 : 5.028e-2, cx10_11 : 2.536e-2$
Q11	4.25E-02	6.11E-04	$\text{cx}11_3 : 3.060e-2, cx11_10 : 2.536e-2, cx11_12 : 1.826e-2$
Q12	2.90E-02	5.15E-04	$\text{cx}12_2 : 3.508e-2, cx12_11 : 1.826e-2, cx12_13 : 2.821e-2$
Q13	1.09E-01	1.74E-03	$\text{cx}13_1 : 6.045e-2, cx13_12 : 2.821e-2, cx13_14 : 4.870e-2$
Q14	3.45E-02	6.98E-04	$\text{cx}14_0 : 4.758e-2, cx14_13 : 4.870e-2$

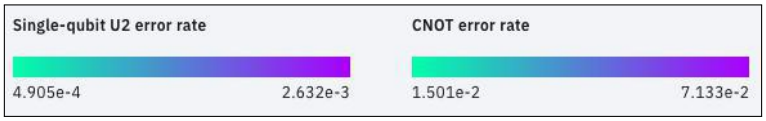


Table 1 shows the Measurement Error, Single Qubit Error, and CNOT Error Calibrations for the IBM Melbourne. The probability of a Single Qubit Error is much lower than both the Measurement Error and CNOT Error, and as a result, CNOTs and measurement operations are the most likely to cause an error. As such, the goal of optimizing the circuits was to minimize the number of CNOT operations. Since there was only 1 final measurement on Qubit 0 in all of the circuits that were passed through the Melbourne, minimizing the number of CNOTs was the priority. While the transpiler drastically increased the number of CNOT operations, applying CNOT operations on certain qubits (such as between qubit 1 and 0) will yield less error than between qubits 8 and 6, which has a 0.1662 percent chance of error. However, some qubits have more possible connections than others, which yields the issue of connectivity.

6 Conclusion

We have studied the performance of multiple versions of Shor Code on the IBM Melbourne and on the simulator against unbiased noise for single qubit errors, concluding that it is possible to optimize the circuit for the given quantum computer. Additionally, using the principle of deferred measurement to minimize the number of measurements also decreases the probability of error occuring.

Further study is required to test the performance of other quantum error correcting codes. Using real hardware instead of a simulator raises a variety of problems, including connectivity issues and other types of error such as crosstalk. Optimizing circuits for different computers requires knowledge of the computer's architecture, and further study is required to learn whether there is a general solution for minimizing the number of gates and hence the probability of error. We expect, however, that this analysis of the performance of the Shor Code will help guidance towards fault tolerant architectures.

7 Acknowledgements

I would like to thank my mentor, Abhijit Mudigonda, for his constant support of my project. Without him, I could never have done this project. Only with his clear explanations of concepts that made no sense to me and his motivation for me to continue my research was I able to complete my project!

References

[1] Panos Aliferis and Andrew W. Cross. Subsystem fault tolerance with the bacon-shor code. *Physical Review Letters*, 98, 05 2007.

[2] Daniel Chiu, Franklyn Wang, and Scott Duke Kominers. Generalization by recognizing confusion. *arXiv:2006.07737 [cs, stat]*, 06 2020.

[3] Cramer J. Repeated quantum error correction on a continuously encoded qubit by real-time feedback, 2016.

[4] N. David Mermin. Shor's 9-qbit error-correcting code. *Quantum Computer Science*, pages 207–209.

[5] Mikio Nakahara and Tetsuo Ohmi. *Quantum computing : from linear algebra to physical realizations*. Crc Press, 2008.

[6] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge Cambridge University Press, 2019.

[7] R. Padma Priya and A. Baradeswaran. An efficient simulation of quantum error correction codes. *Alexandria Engineering Journal*, 57:2167–2175, 09 2018.

[8] S. A. Selesnick. Foundation for quantum computing ii. *International Journal of Theoretical Physics*, 46:984–1002, 12 2006.

[9] Sixia Yu, Qing Chen, C. H. Lai, and C. H. Oh. Nonadditive quantum error-correcting code. *Physical Review Letters*, 101, 08 2008.

The Correlation of Gene Expression in Breast Cancer Patients to Treatment and Survival

Gabriella Troy

Abstract

Breast cancer affects a large portion of the human population. Current methods of treatment have varying levels of success, however, which can in part be explained by genetic mutations and levels of gene expression. In this study, we aim to identify gene expression that correlates with survival and treatments breast cancer patients receive. We use patient samples in the METABRIC dataset from cBioPortal; these patients received different combinations of chemotherapy, hormone therapy, radiotherapy, and breast surgery, and are classified as living, died of disease, or died of other causes. We first perform dimensionality reduction using principal component analysis (PCA) and singular value decomposition (SVD). Next, to determine whether gene expression can be used to predict treatments and survival, we use logistic regression (LR) and support vector machines (SVM), and evaluate these models with accuracy, area under the curve, sensitivity, and specificity scores. Finally, we interpret these models by using weights from LR and SVM to scale PCA and SVD data matrices before transforming the matrices back to their original space. This enables us to compare gene expression in the original and scaled data matrices. Our results show that gene expression alone cannot be used to predict survival, and that nonlinear relationships best represent genetic data. We also identify genes such as WDR48, PLD6, ART3, and SEPN1 that highly positively correlate with all four types of treatment. Studying these genes could illuminate a new direction for the development of effective breast cancer treatment.

1 Introduction

Breast cancer is the most common cancer found among women, and the second leading cause of cancer death in women after lung cancer: it is found occasionally in men, but 12% of women develop the cancer within their lifetimes, and 30% of them die from it [9]. With such a high rate of death, it is imperative that effective treatments are developed and widely distributed. As with many cancers, however, breast cancer proves difficult to cure: its ability to metastasize and affect multiple parts of the body increases the number of tumors treatments must target, and its emergence as a result of various genetic mutations make its course difficult to predict. Treatment is further complicated by the possibility of patient resistance or a resurgence of cancer after it seems to have been eradicated.

Although there are numerous options available for treatment–surgery, radiation, chemotherapy, hormone therapy, immunotherapy, and targeted therapy–there are many cases in which the treatment does not work or has detrimental side effects and another method must be applied. New methods in which deleterious genes and proteins are targeted or inhibited through therapy continue to emerge as the human genome and its association with cancer is studied. Despite the genetic mutations and roles of proteins in signalling pathways that have already been identified, however, there is still much to be discovered in breast cancer genetics that could explain why the cancer metastasizes, exhibits resistance to treatment, and continues to kill a large portion of patients. It is in this genomic direction that a cure for cancer may one day arise.

By studying the METABRIC clinical and genetic data of a group of 1,881 breast cancer patients, we aim to identify differences in gene expression. Because this group of patients has received various combinations of surgery, radiation, chemotherapy, and hormone therapy, we also aim to determine whether gene expression can be used to predict the treatments each patient receives and their ultimate survival. We do this through singular value decomposition (SVD), principal component analysis (PCA), logistic regression (LR), and support vector machines (SVM); using different combinations of PCA, SVD, LR, and SVM allows us to maximize the validity of our models, as, to our knowledge, it has not yet been determined which methods are truly optimal. Any correlation between gene expression, treatment, and survival among these breast cancer patients could illuminate drivers of therapeutic resistance or simply signify gene expression

most likely to result in death. Genetic differences we identify have the possibility of broadening the understanding of genetics behind breast cancer and informing the future development of treatment that could increase the chance of patient survival.

2 Literature Review

2.1 Genetics

Many genes and proteins have already been identified as playing a major role in breast cancer proliferation and resistance to treatment. The most commonly mutated genes are TP53 and PIK3CA [21]. Mutation driver genes can also be linked to different subgroups of breast cancer, such as estrogen receptor positive (ER+), which accounts for 70% of patients: SMAD4 and ERBB2 are common, in addition to KRAS, ARID1A, CDKN2A, PBRM1, KDM6A, MEN1, FOXP1, USP9X, BAP1, which are known to be mutation drivers of other cancers [28]. Although breast cancer can be categorized into groups such as ER+, however, patient survival and response to therapy is widely varied within these groups. Resistance to treatment may arise from a patient's unique combination of genetic mutations or cellular interactions, but in cases such as endocrine, or hormone therapy, resistance may be acquired [11].

A common inhibitor in endocrine therapy is Tamoxifen, which targets the ability of estrogen to bind to estrogen receptors and activate signalling pathways that end in gene transcription and cancer cell growth. Tamoxifen is normally successful in reducing recurrence and mortality in ER+ breast cancer, but resistance proves to be a challenge in prolonged adjuvant tamoxifen therapy, as tamoxifen begins to stimulate cell growth. This may in part be due to tamoxifen's classification as a selective estrogen receptor modulator (SERM), meaning it blocks estrogen receptors on breast cancer cells, but acts like estrogen in other tissues [37]. Although tamoxifen is still an effective antiestrogen that inhibits classical ER receptor-target genes, it increases the non-genomic activity of ER receptors through membrane associated molecules as estrogen does in acquired endocrine resistant patients (see Figure 1). In this case, signalling pathways with several other tyrosine kinase and growth factor receptors are involved. Tyrosine kinase receptors, which are most commonly inhibited to overcome endocrine-resistant breast cancer, activate downstream pathways such as MAPK and PI3K and cross talk with estrogen receptors to evade anti-hormone therapy [4]. When MAPK and Akt are activated, estrogen receptors are phosphorylated, enabling ligand-independent transcription and the cellular activity endocrine therapy aims to inhibit. Growth factors similarly increase these pathways by redistributing ER receptors from the nucleus to extra nuclear areas, where more estrogen is able to bind to the receptors. Although many therapeutic drugs such as lapatinib, a dual inhibitor of the tyrosine kinases epidermal growth factor receptor (EGFR) and human epidermal growth factor receptor 2 (HER2) [3], aim to target components of these signalling pathways, resistance to endocrine therapy continues. This suggests that more genes and proteins may be involved in resistance, which this study aims to determine through the identification of genes whose expression correlates with hormone therapy.

Chemotherapeutic drugs, when successful, indirectly induce apoptosis, a natural cellular function that is efficient at killing cancerous cells and is triggered by various signalling pathways. Chemotherapeutic drugs are used to reduce the size of a tumor before it must be removed through surgery, to kill cancerous cells left behind after surgery and undetected by imaging tests, or to target metastasized cancer that cannot easily be combated with surgery. Resistance to chemotherapy is so often found, however, because chemotherapeutic drugs require apoptotic pathways to be intact, yet initial tumorigenesis depends upon the inhibition of these pathways [17].

Several components of apoptotic pathways could be targeted to reduce resistance to chemotherapy (see Figure 2). The expression of Bcl-2, for example, which regulates the permeability of the mitochondria, from which cytochrome-c must be released to activate the caspase cascade and ultimately apoptosis, could be targeted [29]. Inhibiting anti-apoptotic PI3K with LY294002 could also induce apoptosis and increase sensitivity to tamoxifen, even if Akt, a downstream target of PI3K, has been activated [8]. Because many components, such as PI3K, are involved in several signalling pathways, we may find additional genes that impact apoptosis or the success of chemotherapy.

Like chemotherapy, radiotherapy can induce apoptosis by damaging cell DNA, yet it fails 7% of the time in decreasing the likelihood of cancer recurrence after surgery. Resistance to radiotherapy can be linked to a low expression of the 26S proteasome [10]. Proteasomes are responsible for regulating the

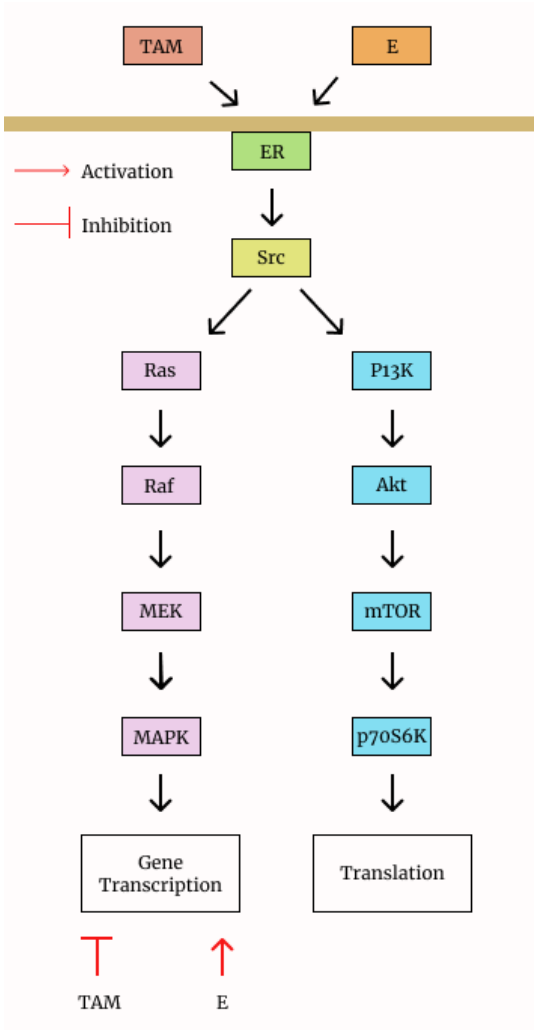


Fig. 1: A model of signal transduction pathways regulated by estrogen (E) and tamoxifen (TAM) in the case of tamoxifen-resistance. Tamoxifen differs from estrogen in that it blocks the gene activation of classical ER-target genes, but both tamoxifen and estrogen increase non-genomic activity through molecules such as Src, whose signalling cascades lead to acquired endocrine resistance.

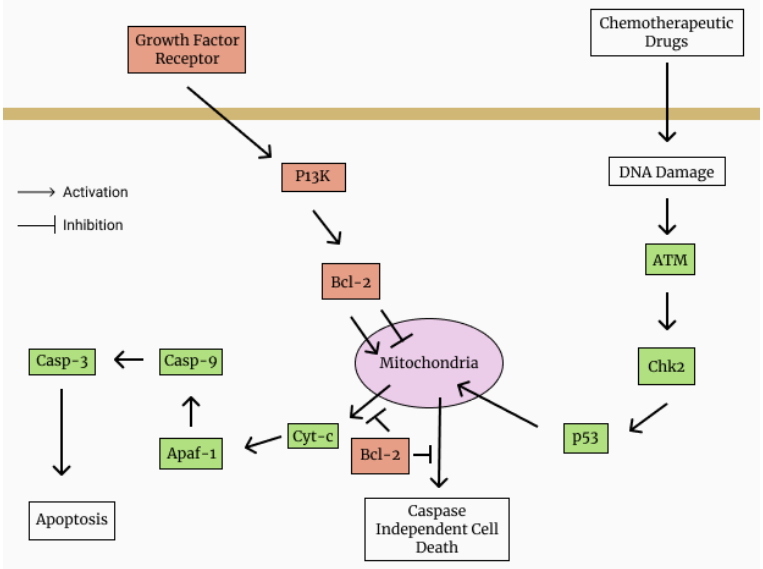


Fig. 2: A model of apoptotic pathways and proteins that may regulate tumor development and response to therapy. Red components inhibit apoptosis, while green components promote it.

level of proteins in cellular processes such as apoptosis. They do so by breaking down proteins that are unneeded or damaged and have been tagged with ubiquitin. 26S, specifically, can break down proteins such as Bcl-2 and p53, a master regulator that senses cellular stressors such as oncoproteins, direct DNA damage, hypoxia, and survival factor deprivation that initiate an apoptotic pathway.

Radiotherapy resistance could be combated by inhibiting HER2, a member of the human epidermal growth factor receptor family. HER2 is overexpressed in 25-30% of breast cancer patients and is associated with reduced survival rates; it plays a key role in the growth and metastasis of breast cancer, and despite a low rate in cancer recurrence today, it is still the breast cancer subtype most associated with recurrence [25]. In the presence of radiation, HER2 activates STAT3, a transcription factor that transduces oncogenic signals from cytokines and growth factors (see Figure 3). Inhibition of HER2 and STAT3 by chemical inhibitors such as lapatinib and S3I-201, and siRNA inhibition of survivin, a downstream target of STAT3, decreases both radiation-induced STAT3 phosphorylation and survivin expression, and ultimately increases radiation-induced cell death [20].

It is critical that steps in inhibition are taken before breast cancer has the chance to metastasize; cancer becomes increasingly difficult to treat effectively once it has spread to other parts of the body. A possible preventative measure is to increase the expression of CCN6, an extracellular matrix protein that acts as a tumor suppressor by preventing the BMP4 protein from inducing the signalling pathway with p38 [26]. When phosphorylated, p38 promotes invasiveness. On the contrary, however, BMP4 can sensitize ER+ breast cancer cells to anti-estrogen, which is significant for therapeutic development not only because 70% of women with breast cancer are ER+, but because a third of them and nearly all of those with metastatic disease develop anti-estrogen-resistant disease [34]. With all the nuanced ways in which proteins and genes such as CCN6 and BMP4 interact, studying each and the effect of their inhibition is necessary to determine the most effective ways of treating breast cancer. Additionally, there exist many genes whose role in breast cancer and therapeutic resistance is yet to be discovered.

2.2 Machine Learning

Many researchers have studied the genetics behind cancer using algorithms that can be used to classify cancer, select feature genes from DNA microarrays, and predict patient survival. The first challenge researchers face is to reduce the extensive dimensions of genetic data in order to extract relevant features and visualize their findings. Principal Component Analysis (PCA) has been the most common method for

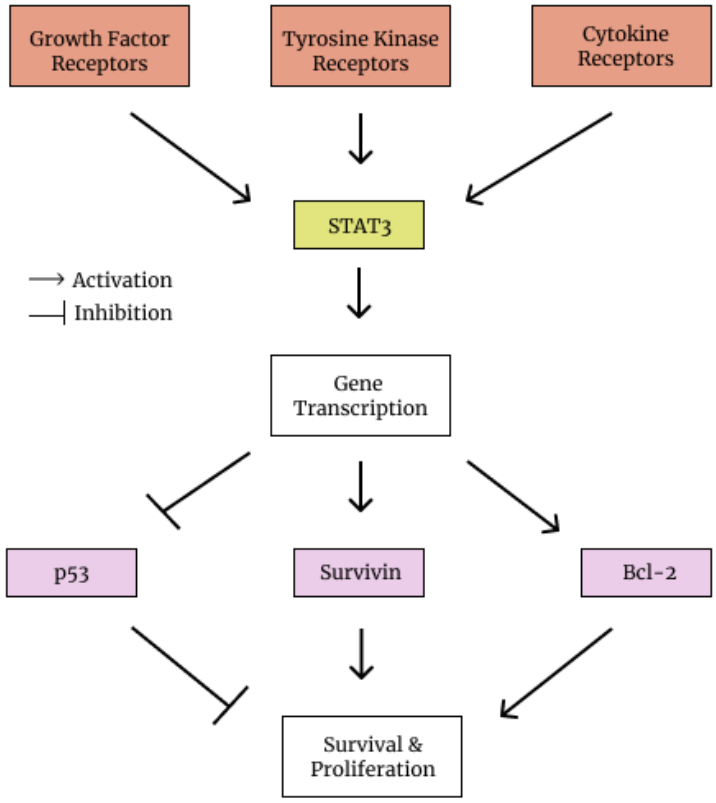


Fig. 3: A model of the role of STAT3 in gene transcription and its targets that promote cancer cell survival and proliferation. Receptors such as growth factor receptors, the family in which HER2 belongs, activates STAT3. STAT3 then proceeds to activate targets that will inhibit apoptosis or increase cancer cell growth and proliferation, resulting in resistance to treatment.

dimensionality reduction, and has borne many success in cancer research alone [16]. PCA does have the problem, however, that all variables must be included in the principal components, regardless of whether it is a noise and thus has no impact on the model. This has partially been solved through the evolution of sparse PCA to reduce the number of nonzero coefficients; nonzero coefficients reflect a poor estimation, as it is expected for only a small portion of genes to be expressed in a given tissue or process [24]. Sparse PCA has not been satisfactorily successful, however. Singular Value Decomposition (SVD) is an equivalent to PCA that has increasingly been used in genetic analysis, but no benefit of one method over the other has arised [27]. Chen et al use SVD to extract gene features of ovarian cancer patients that are linked to chemotherapy responses [6]. Horn et al use truncated SVD to compress the dimensions of a gene sample matrix before applying a novel clustering method [15]. Truncated SVD, a further minimization of dimensions after SVD, is often done arbitrarily and used in inversion [41].

Other methods for dimensionality reduction not explored in this study have previously been used to analyze cancer data. Shen et al find that performing Partial Least Squares (PLS) in addition to SVD results in better and faster results in regression models used to classify cancer [35]. Another method, Multifactor Dimensionality Reduction (MDR), is a nonparametric and model-free approach that has been used to identify and characterize interactions between estrogen-metabolism genes in breast cancer [30]. Although Linear Discriminant Analysis (LDA) cannot be used for feature extraction in cancer classification because of a small sample size compared to a large feature space, Gradient LDA achieves a lower misclassification error than LDA when using gene expression datasets [33].

Once dimensionality reduction has been completed, logistic regression (LR) is a powerful, yet simple tool that can be used for classification; LR is less practical for dealing with high-dimensional data [30].

LR is especially valuable because it can be extended to multiple classes, as is often present in genetic data [36]. LR has formed the basis of algorithms used to study cancer genetics, like the LRpath [32] and an extension of the Gauss-Seidel method [36]. LASSO regularization in the LR model has been a key step in high-dimensional cancer classification for both gene coefficient estimation and gene selection. LASSO has, however, been criticized for being biased in gene selection [1].

Support Vector machines (SVMs) have also been useful for the classification of data with large dimensions; they have had greater accuracy than LR in establishing a classification of breast cancer survival patterns [5]. SVM has proven the ability to accelerate this classification process and to increase generalization performance. It has also been beneficial for selecting feature genes with a greater classification performance, and for differentiating between metastatic and non-metastatic breast cancer samples [39].

Like dimensionality reduction, there are other methods and models to examine cancer datasets. In contrast to LASSO regularization, extreme gradient tree boosting approach (XGBoost) can integrate nonlinear interactions into prediction models in a non-additive form and has been used to identify single nucleotide polymorphisms (SNPs) that contribute most to breast cancer risk [2]. Additional models that have been widely applied include Artificial Neural Networks (ANNs), Bayesian Networks (BNs), and Decision Trees (DTs) [22]. Chou et al suggest that ANN models have a greater predictive power than DTs when analysing gene expression in breast cancer from microarray datasets [7]. Using ANNs, they were able to identify the 21 genes most associated with cancer recurrence within 5 years, and further analyzed these genes with Cox regression. ANNs are also successful in profiling gene expression in neuroblastoma tumors to predict survival [40]. BNs can uncover interactions between genes based on various expression measurements, and are especially useful for learning from noisy observations [13]. Using the emasurement of phosphorylated proteins and phospholipids, BNs can also order the connections between components of signalling pathways and predict novel interpathway causalities, which is applicable to understanding signalling in cancer cells and drug actions [31]. DTs are mainly used to classify a disease and predict both its recurrence and the survival of a patient, which assists physicians in managing breast cancer patients [18]. Fuzzy DTs in particular seem to be more accurate and balanced [19]. In this study, we use LR and SVM to detect patterns in gene expression data and subsequently identify target genes that have the potential to revolutionize the world of breast cancer treatment.

3 Purpose

1. To determine what combination of principal component analysis, singular value decomposition, logistic regression, and support vector machines is most effective for analyzing genetic data
2. To apply a novel inverse-transformation of reduced matrices method to interpret our models and determine the correlation between gene expression in breast cancer patients, the treatments they receive, and their ultimate survival
3. To identify largely unexplored genes that could be targeted in the future development of treatments for breast cancer

4 Methods

4.1 METABRIC Breast Cancer Data from cBioPortal

The 2012 METABRIC clinical and genetic data of 2,509 breast cancer patients with primary tumors is a freely accessible dataset available for download at cBioPortal. Clinical data includes treatments received, type of breast cancer and its subgroups (ER and HER2), age at diagnosis, survival, size of tumor, nottingham prognostic index, and more. Genetic data includes gene expression of 24,368 genes, which is commonly measured by quantifying levels of gene product such as proteins, and specifics on genetic mutations: SNP, location on chromosomes, type of mutation, etc.

4.2 Preprocessing

In this study, we only use gene expression data and treatment and vital status columns from the clinical data (see 1). To efficiently study these columns in relation to the gene expression data, we merge gene expression, treatments (chemotherapy, hormone therapy, radiotherapy, and breast surgery), and vital status (living, died of disease, or died of other causes) into one dataframe based on patient IDs. In this dataframe, each row is a patient, and each column is a different gene, treatment, or vital status. The chemotherapy, hormone therapy, and radiotherapy columns denote "yes" or "no" whether a patient has received that type of treatment, while the breast surgery column denotes "mastectomy" or "breast conserving," which implies that every patient had to undergo surgery at some point during their treatment. Patients undergo a breast conserving surgery if a tumor is small enough to only remove the part of the breast with cancerous cells, or a mastectomy to remove the entire breast. After removing rows without values, the dataframe is reduced to 1,881 patient samples.

Patient ID	RERE	RNF165	Chemo	Horm	Radio	Surg	Vital
MB-0362	8.677	6.075	YES	YES	YES	MASTECTOMY	Died of Disease
MB-0503	9.274	5.909	NO	YES	YES	BREAST CONSERVING	Living
MB-0597	8.758	6.681	YES	YES	YES	MASTECTOMY	Died of Other Causes

Tab. 1: A sample of the dataframe we use to conduct our study. The dataframe includes expression of genes (RERE, RNF165), reception of chemotherapy (Chemo), hormone therapy (Horm), radiotherapy (Radio), and breast surgery (Surg), and vital status (Vital). Each component is linked to a patient ID, which denotes which patient sample the information corresponds to.

4.2.1 Standardization

We standardize our dataframe such that columns have mean 0 and variance 1; after calculating the mean and standard deviation of each variable, the mean is subtracted from the variable, which is then divided by the variable's standard deviation. The resulting standardized values represent the number of standard deviations above or below the mean an observation falls. We do this to ensure that variables measured at a different scale do not disproportionately bias a model. Although it may cut down on precision marginally, the Python library scikit-learn can quickly execute standardization through StandardScaler.

4.3 Dimensionality Reduction: PCA and SVD

We perform dimensionality reduction on the 24,368 features of our dataframe in order to reduce computation time. We do this with both principal component analysis (PCA) and singular value decomposition (SVD).

4.3.1 Principal Component Analysis (PCA)

Our first step in performing Principal Component Analysis (PCA) is to compute a covariance matrix. This covariance matrix contains variances–how much a single variable varies–on the diagonal, and covariances–how much two variables vary together–in all other entries. Given a data matrix X of size n×p, where n is the number of patient samples and p is the number of variables, a square covariance matrix of size p×p covariance can be calculated using the following equation:

$$\sigma_{jk} = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{xj})(x_{ik} - \bar{x_k})$$

where σ is a single entry in the covariance matrix, j and k are columns in the feature vector x_i from matrix X, and $\bar{x}$ is the mean of the denoted columns.

Because we have already standardized the data, X is centered with mean 0, and the equation can be simplified to:

$$C = \frac{XX^T}{n-1}$$

The covariance matrix is symmetric, so we can diagonalize it to:

$$VLV^T$$

where V is a matrix of eigenvectors and L is a diagonal matrix with eigenvalues in decreasing order. Once we project data onto the eigenvectors, also known as principal axes or principal directions, we have new, transformed variables–the principal components. We linearly combine these principal components into a matrix of size n$\times$k, XV, where k is the number of components we have chosen and include the components with the greatest eigenvalues. The j-th principal component is the j-th column of XV, and the i-th data point in the principal component space is the i-th row of XV.

We can perform all these steps with scikit-learn's PCA function. We use 500 components to achieve an explainced variance ratio of approximately 0.95.

4.3.2 Singular Value Decomposition (SVD)

A singular value decomposition (SVD) of X is functionally equivalent to PCA, but is given by the equation:

$$X = USV^T$$

where U is a unitary matrix such that $UUT = I$ and S is a diagonal matrix of singular values–the square root of the eigenvalues. The equivalence of PCA and SVD can be seen in the computation of the covariance matrix:

$$C = \frac{VSU^TUSV^T}{n-1} = V\frac{S^2}{n-1}V^T$$

where V is still the principal directions and the singular values are related to the eigenvalues of the covariance matrix, as $L = \frac{S2}{n-1}$. Principal components can thus be given the same way in XV. Truncated SVD simply takes SVD a step further by computing the top k eigenvectors and eigenvalues of X and keeping only the first k columns of XV.

We execute truncated SVD with the scikit-learn TruncatedSVD function. We use 500 components to achieve an explainced variance ratio of approximately 0.95.

4.4 Label Encoding

We convert categorical values from the treatment and vital status columns in our dataframe to numerical values. We use label encoding to assign numbers to each categorical value: 0, 1, or 2. In the treatment columns, 0 denotes *no* or *breast conserving* and 1 denotes *yes* or *mastectomy*. In the vital status column, 0 dentoes *living*, 1 denotes *died of disease*, and 2 denotes *died of other causes*. A common concern with label encoding, which is often solved with one-hot encoding, is that these numerical values give classes rank within the dataframe. Because the models we implement are regression and classification models, this is not a concern. Furthermore, having three classes in the vital status column means a one-vs-all approach must be used, and we do this manually. We use scikit-learn's LabelEncoder for label encoding.

4.5 Models: Logistic Regression (LR) and SVM

After splitting the data into train and test sets, we use both logistic regression (LR) and support vector machines (SVM) to predict whether a patient has received some combination of the four treatments and what their vital status. Each treatment prediction is run as its own separate binary classification task, while vital status prediction is run as a multi-class classification problem. Because these models are supervised machine learning models, we pass gene expression data from our dataframe through these models with labels from the treatment and vital status columns.

4.5.1 Logistic Regression (LR)

Logistic regression (LR) is a supervised machine learning model that learns linear relationships from input data in order to classify data points by assigning a probability for each class. These classes, which LR separates with a linear decision boundary, are yes/no for chemotherapy, hormone therapy, and radiotherapy, breast conserving/mastectomy for breast surgery, and three different one-vs-all classes for vital status such as living/not living. LR classifies data points by taking a linear combination of the feature vector from the dataframe along with the bias, b:

$$a = b + w_1x_1 + w_2x_2 + ... + w_px_p$$

where the weight w_i corresponds to the feature i that has a value x_i. We then pass this score (a) through a sigmoid function $f(x) = \frac{1}{1+e^{-a}}$ so that it is narrowed down between 0 and 1–the range in which probabilities always lie.

We use the LogisticRegrerssion function of scikit-learn to implement the LR model. We set the penalty parameter to "l1" for LASSO regularization, which adds the absolute value of weights as a penalty term to the loss function and generally performs better when there is a small number of significant parameters and the others are close to zero.

4.5.2 Support Vector Machines (SVM)

While LR focuses on maximizing the probability of the data, support vector machines (SVM) try to find the hyperplane that maximizes the margin between the hyperplane and the points closest to it. This hyperplane divides data points into two classes, and the closest points are support vectors. In a standard binary classification setting, the margin divides a negative (-1) and positive (+1) class. The typical hypothesis function used for SVM is:

$$h(x_i) = \begin{cases} 1, & wx + b \geq 0 \\ -1, & wx + b < 0 \end{cases}$$

We maximize the margin by minimizing the cost function:

$$\frac{1}{n}\sum_{i=1}^{n} max(0, 1 - y_i(wx_i - b)) + \lambda||w||^2$$

whose cost is 0 when the predicted value and actual value have the same sign. We calculate a loss value and take a partial derivative of the cost function in order to update the gradient. The gradient defines the direction of the steepest descent in which the gradient descent optimization algorithm must iteratively move towards in order to minimize a function.

If data is nonlinear, we are still able to use SVM to create a linear decision boundary by using a kernel to raise the data to higher dimensional space and compute a dot product of every vector in the dataset while in that space before coming back down. The default of scikit-learn's svm.SVC function is linear, but we use the radial basis function (rbf) kernel to account for the possibility of the gene expression data not being linear.

4.6 Evaluation Metrics

We find accuracy and area under the curve (AUC) to determine how well our models predict whether a patient has received a certain type of treatment or survived. AUC, which represents the area under the receiver operating characteristic curve (ROC)–a graph showing the performance of a classification model based on true and false positive rates–is often a better measure than accuracy because it cannot be skewed by unbalanced data. We obtain these scores from scikit-learn via accuracy_score and roc_auc_score. To understand the AUC scores, we also find the sensitivity and specificity of the train and test sets, which measure the prediction of true positives and true negatives respectively. We do this with scikit-learn's confusion_matrix.

4.7 Interpretability

Now that we have used and evaluated the logistic regression and support vector machine models, we must understand what they have learned about gene expression in relation to treatments and vital status. We do this by making a comparison between information from the models and the original gene expression matrix.

4.7.1 Scaled Inverse Transform of PCA and SVD Matrices

We must transform the PCA and SVD data matrices back to their original space before determining whether a gene in breast cancer patients has any significance. We can easily do this with the attribute inverse_transform in scikit-learn's PCA and TruncatedSVD functions. If $XV = N$, where N is the new matrix transformed into PCA space, then X can be derived from multiplying each side by V^T, as V is a unitary matrix and will be canceled out when multiplied by its transpose (see 4). Before completing the inverse transform, however, we multiply the SVD and PCA matrices by a diagonal matrix of the weights, or coefficients, from LR and SVM corresponding to each component so that the matrices, when transformed back into their original space, are scaled by these weights. Scaling the matrices allows us to make a comparison with the original gene expression data.

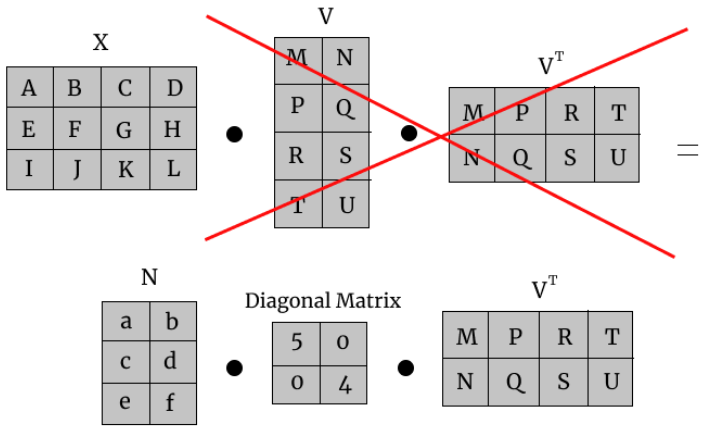


Fig. 4: Example of matrix operations performed for the scaled inverse transform of principal component analysis (PCA) and singular value decomposition (SVD) matrices. X is the original matrix, V is a matrix of eigenvectors, N is the new PCA or SVD matrix, and the diagonal matrix contains the logistic regression or support vector machines weights on the diagonal.

After scaling and transforming the PCA and SVD data matrices back to their original space, we sum over each column. We also sum over the columns of the original gene expression matrix so that we can divide the scaled sums by the original sums. This division enables us to identify genes with the greatest value, whose expression should be highly positively correlated with breast cancer treatments.

5 Results

We perform singular value decomposition (SVD), principal component analysis (PCA), logistic regression (LR), and support vector machines (SVM) on the gene expression of 1,881 breast cancer patients. For these supervised machine learning algorithms, we obtain labels from the treatment and vital status columns of the clinical data.

5.1 Accuracy

We calculate accuracy to determine how well the logistic regression (LR) and support vector machines (SVM) models performed in predicting the treatments and vital status of a breast cancer patient. As shown in Tables 2 and 3, accuracy scores of the train sets are high. The lowest training accuracy is 0.811 for predicting the reception of radiotherapy using a LR model after principal component analysis (PCA), and 0.845 for predicting whether a patient has died of other causes using a LR model after PCA.

Accuracy scores of the test sets, however, are less optimal. In Table 2, we obtain a testing accuracy of 0.841 when predicting chemotherapy using a linear SVM model after singular value decomposition (SVD), but we obtain a testing accuracy of 0.533 when predicting radiotherapy using a linear SVM model after SVD. In Table 3, we obtain a testing accuracy of 0.751 when predicting whether a patient has died from cancer using SVM with a radial basis function (rbf) kernel after PCA, but we obtain a testing accuracy of 0.570 when predicting whether a patient is living using linear SVM after PCA.

Pred	DR	Model	Tr ACC	Te ACC	Tr AUC	Te AUC
Chemo	SVD	LR	1.000	0.828	1.000	0.822
		SVM rbf	0.961	0.806	1.000	0.865
		SVM lin	1.000	0.841	1.000	0.850
	PCA	LR	1.000	0.780	1.000	0.724
		SVM rbf	0.941	0.822	0.989	0.821
		SVM lin	1.000	0.769	1.000	0.711
Horm	SVD	LR	1.000	0.629	1.000	0.658
		SVM rbf	0.947	0.714	0.996	0.752
		SVM lin	1.000	0.631	1.000	0.646
	PCA	LR	0.861	0.631	0.939	0.653
		SVM rbf	0.880	0.735	0.970	0.772
		SVM lin	0.911	0.626	0.927	0.625
Radio	SVD	LR	1.000	0.541	1.000	0.544
		SVM rbf	0.963	0.610	0.998	0.614
		SVM lin	1.000	0.533	1.000	0.554
	PCA	LR	0.811	0.592	0.883	0.608
		SVM rbf	0.884	0.660	0.971	0.638
		SVM lin	0.845	0.586	0.873	0.592
Surg	SVD	LR	1.000	0.560	1.000	0.554
		SVM rbf	0.961	0.618	1.000	0.642
		SVM lin	1.000	0.589	1.000	0.591
	PCA	LR	0.833	0.578	0.911	0.608
		SVM rbf	0.886	0.623	0.986	0.669
		SVM lin	0.871	0.597	0.899	0.608

Tab. 2: Accuracy (ACC) and area under the curve (AUC) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) chemotherapy (Chemo), hormone therapy (Horm), radiotherapy (Radio), and surgery (Surg) using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction (DR). For example, predicting whether a breast cancer patient in the train set received hormone therapy using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using area under the curve can be found in the row Horm, PCA, LR and in the column Tr AUC. Green elements represent the highest value of a column, while red elements represent the lowest value of a column.

Vital Metric	SVD LR	SVD SVM rbf	SVD SVM lin	PCA LR	PCA SVM rbf	PCA SVM lin
Tr0 ACC	1.000	1.000	0.969	0.866	0.904	0.979
Tr1 ACC	1.000	0.959	1.000	0.899	0.908	1.000
Tr2 ACC	1.000	0.952	1.000	0.845	0.881	0.979
Te0 ACC	0.613	0.663	0.605	0.586	0.647	0.570
Te1 ACC	0.692	0.729	0.716	0.660	0.751	0.645
Te2 ACC	0.602	0.594	0.581	0.597	0.594	0.576
Tr0 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Tr1 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Tr2 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Te0 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Te1 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Te2 AUC	0.500	0.500	0.500	0.500	0.500	0.500

Tab. 3: Accuracy (ACC) and area under the curve (AUC) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) vital status using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction. There are three scores for each evaluation metric, found using a one-vs-all approach, for each train and test set to accommodate the three classes in vital status. 0 represents living, 1 represents died of disease, and 2 represents died of other causes. For example, predicting whether a breast cancer patient in the train set is living using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using area under the curve can be found in the row Tr0 AUC and the column PCA LR. The green element represents the highest testing accuracy, while the red elements represent the lowest accuracy and AUC scores.

5.2 Area Under the Curve (AUC)

We also calculate area under the curve (AUC) to evaluate logistic regression (LR) and support vector machines (SVM) models. As shown in Table 2, training AUC scores are often 1.000. The lowest training AUC is 0.873, which we obtain when predicting radiotherapy with a linear SVM model after principal component analysis (PCA). Not all the testing AUC scores are high, but many are high enough to increase our chance of identifying target genes later in our study.

Based on the AUC scores from the test sets, our LR and SVM models can most accurately predict whether a patient has received a certain type of treatment in the following order: chemotherapy, hormone therapy, breast surgery, and radiotherapy. The lowest testing AUC in our predictions of treatments, as shown in Table 2, is 0.544 for predicting radiotherapy using LR after singular value decomposition (SVD). The greatest AUC we obtain is 0.865 for predicting chemotherapy using SVM with a radial basis function (rbf) kernel after SVD. We visualize this optimal AUC in Figure 5. We can also discern from Table 2 that we obtain the majority of the greatest AUC scores from the test sets predicting the other types of treatment using SVM rbf models after SVD. In terms of linear models after SVD, LR performs slightly better than SVM linear.

Our models are not optimal for predicting the vital status of a breast cancer patient. As evident in Table 3, these models are only able to guess; we return 0.500 for every model using both the train and test sets. AUC scores of 0.5 are the worst possible values that we could return.

5.3 Sensitivity and Specificity

We calculate sensitivity and specificity to understand why we obtain low area under the curve (AUC) scores. As shown in Tables 4 and 5, the sensitivity and specificity scores for predicting the reception of a

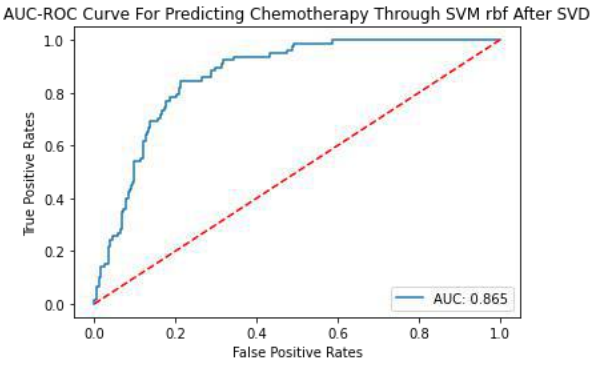


Fig. 5: Area under the receiver operating characteristic curve (AUC-ROC) for predicting a breast cancer patient receiving hormone therapy through a logistic regression (LR) model after support vector machine (SVD) dimensionality reduction.

treatment or vital status in the train sets are 1.000's for the most part. The lowest sensitivity scores are 0.723 sensitivity for predicting chemotherapy using a support vector machines model with a radial basis function (rbf) kernel after principal component analysis (PCA) and 0.734 for predicting whether a patient has died from cancer using a SVM rbf model after PCA.

In the test sets, however, there are only a few cases in which both the sensitivity and specificity scores are similar and around 0.6; otherwise most of our predictions of treatment and vital status return a high value for either sensitivity or specificity, and a low value for the other. The lower value is most often the test specificity. The best combination of values we find is in predicting the reception of hormone therapy using logistic regression (LR) after singular value decomposition (SVD), where the sensitivity and specificity scores are 0.684 and 0.538 respectively. We find a close second in predicting whether a patient has died of other causes using SVM rbf after SVD, where the sensitivity is 0.698 and the specificity is 0.518.

5.4 KFold Cross Validation

We perform KFold cross validation with 5 splits instead of the 2 using scikit-learn's simple train_test_split. Our aim in doing so is to increase the values we return from the test sets, to narrow the gap between train and test set values, and to decrease the difference between the sensitivity and specificity values from the test sets. Cross validation has little effect on the values we return, however. Tables 6, 7, 8, and 9 reveal mixed results in increasing values and decreasing gaps between either train and test sets or sensitivity and specificity scores.

Training accuracy scores remain high; many are 1.000. In Table 6 we observe that the lowest training accuracy in our predictions of treatments is slightly higher than the lowest accuracy of 0.811 we obtain without cross validation; the training accuracy of predicting radiotherapy using a logistic regression (LR) model after singular value decomposition (SVD) is 0.823. The lowest training accuracy in our predictions of vital status after cross validation is also slightly higher; we find that the lowest training accuracy in Table 8 is 0.862 instead of 0.845 for our prediction of whether a patient has died of other causes using a LR model after principal component analysis (PCA).

The testing accuracy scores after cross validation are generally lower. Although the lowest testing accuracy in our predictions of treatments is slightly higher than the original 0.533–in Table 6 we obtain a testing accuracy score of 0.538 when predicting radiotherapy using LR after SVD–the highest testing accuracy is reduced from 0.841 to 0.818 when predicting chemotherapy using either linear support vector machine (SVM) after SVD or SVM with a radial basis function (rbf) kernel after PCA. In our predictions of vital status after cross validation, Table 8 shows that the lowest testing accuracy score changes from 0.570 to 0.564 when predicting whether a patient is living using linear SVM after PCA. Instead of 0.751, the highest testing accuracy is 0.743 when predicting whether a patient has died from cancer using SVM rbf after SVD.

Training area under the curve (AUC) scores also remain high after cross validation in our predictions of treatments; many are 1.000. Table 6 shows that the lowest training AUC is higher than the original 0.811:

Pred	DR	Model	Tr Sens	Te Sens	Tr Spec	Te Spec
Chemo	SVD	LR	1.000	0.500	1.000	0.913
		SVM rbf	0.813	0.192	0.999	0.967
		SVM lin	1.000	0.462	1.000	0.940
	PCA	LR	1.000	0.420	1.000	0.873
		SVM rbf	0.723	0.244	0.997	0.973
		SVM lin	1.000	0.385	1.000	0.870
Horm	SVD	LR	1.000	0.684	1.000	0.538
		SVM rbf	0.990	0.855	0.876	0.483
		SVM lin	1.000	0.722	1.000	0.483
	PCA	LR	0.901	0.739	0.795	0.455
		SVM rbf	0.970	0.902	0.734	0.462
		SVM lin	0.940	0.714	0.864	0.483
Radio	SVD	LR	1.000	0.611	1.000	0.437
		SVM rbf	0.997	0.907	0.914	0.166
		SVM lin	1.000	0.597	1.000	0.437
	PCA	LR	0.752	0.850	0.664	0.483
		SVM rbf	0.997	0.885	0.745	0.325
		SVM lin	0.884	0.633	0.786	0.517
Surg	SVD	LR	1.000	0.584	1.000	0.523
		SVM rbf	1.000	0.929	0.904	0.152
		SVM lin	1.000	0.646	1.000	0.503
	PCA	LR	0.877	0.606	0.768	0.536
		SVM rbf	0.988	0.894	0.735	0.219
		SVM lin	0.907	0.628	0.818	0.550

Tab. 4: Sensitivity (Sens) and specificity (Spec) of train (Tr) and test (Te) sets. Each set predicts (Pred) chemotherapy (Chemo), hormone therapy (Horm), radiotherapy (Radio), and surgery (Surg) using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction (DR). For example, predicting whether a breast cancer patient in the train set received hormone therapy using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using sensitivity can be found in the row Horm, PCA, LR and in the column Tr Sens. The red element represents the lowest value from the training sensitivity and specificity scores, while the green elements represent the best combination of testing sensitivity and specificity scores.

Vital Metric	SVD LR	SVD SVM rbf	SVD SVM lin	PCA LR	PCA SVM rbf	PCA SVM lin
Tr0 Sens	1.000	0.933	1.000	0.778	0.825	0.967
Tr1 Sens	1.000	0.860	1.000	0.789	0.734	1.000
Tr2 Sens	1.000	0.994	1.000	0.836	0.931	0.975
Te0 Sens	0.423	0.390	0.496	0.350	0.407	0.431
Te1 Sens	0.316	0.284	0.347	0.400	0.358	0.316
Te2 Sens	0.560	0.698	0.478	0.491	0.648	0.415
Tr0 Spec	1.000	0.986	1.000	0.909	0.942	0.984
Tr1 Spec	1.000	0.993	1.000	0.936	0.967	1.000
Tr2 Spec	1.000	0.922	1.000	0.852	0.845	0.982
Te0 Spec	0.705	0.795	0.657	0.701	0.764	0.638
Te1 Spec	0.819	0.879	0.840	0.748	0.883	0.755
Te2 Spec	0.633	0.518	0.656	0.674	0.555	0.693

Tab. 5: Sensitivity (Sens), and specificity (Spec) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) vital status using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction. There are three scores for each evaluation metric, found using a one-vs-all approach, for each train and test set to accommodate the three classes in vital status. 0 represents living, 1 represents died of disease, and 2 represents died of other causes. For example, predicting whether a breast cancer patient in the train set is living using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using sensitivity can be found in the row Tr0 Sens and the column PCA LR. The red element represents the lowest value from the training sensitivity and specificity scores, while the green elements represent the best combination of testing sensitivity and specificity scores.

we obtain 0.887 when predicting radiotherapy using linear SVM after PCA. Our lowest testing AUC is almost the same as the original 0.544. The lowest testing AUC after cross validation is 0.546 when predicting radiotherapy using LR after SVD. We still obtain the highest testing AUC in our prediction of chemotherapy using support vector machines (SVM) with a radial basis function (rbf) kernel after singular value decomposition (SVD), but this value is reduced from 0.865 to 0.847.

We find the most concerning result in Table 8, where AUC scores of predicting vital status are still poor and remain 0.5. This is the case for both train and test sets.

The sensitivity and specificity scores from train sets are generally higher; in our predictions of both treatments and vital status, many values remain 1.000. The lowest training sensitivity or specificity scores of both treatment and vital status predictions are slightly higher. In Table 7, the lowest training sensitivity is 0.733 when predicting chemotherapy using SVM rbf after PCA instead of 0.723. In Table 9, the lowest training sensitivity increases from 0.734 to 0.752 when predicting whether a patient has died from cancer using SVM rbf after PCA.

Testing sensitivity and specificity scores are not greatly changed after cross validation. Table 7 shows that the best combination of testing sensitivity and specificity scores after cross validation for our predictions of treatments are 0.647 sensitivity and 0.556 specificity for predicting whether a patient has received hormone therapy using logistic regression (LR) after SVD. The best combination of testing sensitivity and specificity scores after cross validation for our predictions of vital status in Table 9 are 0.682 sensitivity and 0.517 specificity for predicting whether a patient has died of other causes through SVM rbf after SVD.

The small differences we find after performing KFold cross validation with 5 splits are not great or positive enough to merit the expensive run time of the models. We therefore use our results obtained without cross validation for the rest of our study.

Pred	DR	Model	Tr ACC	Te ACC	Tr AUC	Te AUC
Chemo	SVD	LR	1.000	0.787	1.000	0.806
		SVM rbf	0.968	0.815	1.000	0.847
		SVM lin	1.000	0.818	1.000	0.830
	PCA	LR	1.000	0.748	1.000	0.741
		SVM rbf	0.942	0.818	0.922	0.844
		SVM lin	1.000	0.751	1.000	0.741
Horm	SVD	LR	1.000	0.617	1.000	0.657
		SVM rbf	0.942	0.710	0.995	0.732
		SVM lin	1.000	0.638	1.000	0.655
	PCA	LR	0.878	0.640	0.946	0.646
		SVM rbf	0.888	0.715	0.970	0.733
		SVM lin	0.921	0.619	0.934	0.624
Radio	SVD	LR	1.000	0.538	1.000	0.546
		SVM rbf	0.965	0.600	0.999	0.597
		SVM lin	1.000	0.545	1.000	0.560
	PCA	LR	0.823	0.559	0.901	0.562
		SVM rbf	0.891	0.606	0.975	0.597
		SVM lin	0.863	0.571	0.887	0.568
Surg	SVD	LR	1.000	0.556	1.000	0.573
		SVM rbf	0.963	0.606	1.000	0.638
		SVM lin	1.000	0.577	1.000	0.600
	PCA	LR	0.842	0.595	0.917	0.601
		SVM rbf	0.898	0.625	0.988	0.644
		SVM lin	0.886	0.595	0.905	0.606

Tab. 6: Accuracy (ACC) and area under the curve (AUC) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) chemotherapy (Chemo), hormone therapy (Horm), radiotherapy (Radio), and surgery (Surg) using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction (DR). For example, predicting whether a breast cancer patient in the train set received hormone therapy using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using area under the curve can be found in the row Horm, PCA, LR and in the column Tr AUC. Values represent the average of values returned after 5 splits in KFold cross validation. Green elements represent the highest value of a column, while red elements represent the lowest value of a column.

Pred	DR	Model	Tr Sens	Te Sens	Tr Spec	Te Spec
Chemo	SVD	LR	1.000	0.545	1.000	0.855
		SVM rbf	0.844	0.277	1.000	0.960
		SVM lin	1.000	0.461	1.000	0.916
	PCA	LR	1.000	0.503	1.000	0.817
		SVM rbf	0.733	0.289	0.996	0.960
		SVM lin	1.000	0.493	1.000	0.823
Horm	SVD	LR	1.000	0.647	1.000	0.556
		SVM rbf	0.990	0.884	0.864	0.426
		SVM lin	1.000	0.730	1.000	0.484
	PCA	LR	0.917	0.718	0.816	0.506
		SVM rbf	0.970	0.872	0.753	0.459
		SVM lin	0.951	0.707	0.872	0.469
Radio	SVD	LR	1.000	0.548	1.000	0.530
		SVM rbf	0.997	0.872	0.915	0.217
		SVM lin	1.000	0.623	1.000	0.434
	PCA	LR	0.875	0.611	0.746	0.478
		SVM rbf	0.969	0.827	0.772	0.292
		SVM lin	0.909	0.638	0.793	0.472
Surg	SVD	LR	1.000	0.561	1.000	0.548
		SVM rbf	1.000	0.951	0.907	0.095
		SVM lin	1.000	0.650	1.000	0.467
	PCA	LR	0.880	0.652	0.786	0.500
		SVM rbf	0.995	0.902	0.753	0.212
		SVM lin	0.913	0.647	0.846	0.505

Tab. 7: Sensitivity (Sens) and specificity (Spec) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) chemotherapy (Chemo), hormone therapy (Horm), radiotherapy (Radio), and surgery (Surg) using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction (DR). For example, predicting whether a breast cancer patient in the train set received hormone therapy using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using sensitivity can be found in the row Horm, PCA, LR and in the column Tr Sens. Values represent the average of values returned after 5 splits in KFold cross validation. The red element represents the lowest value from the training sensitivity and specificity scores, while the green elements represent the best combination of testing sensitivity and specificity scores.

Vital Metric	SVD LR	SVD SVM rbf	SVD SVM lin	PCA LR	PCA SVM rbf	PCA SVM lin
Tr0 ACC	1.000	0.966	1.000	0.877	0.911	0.991
Tr1 ACC	1.000	0.963	1.000	0.919	0.913	1.000
Tr2 ACC	1.000	0.954	1.000	0.862	0.888	0.991
Te0 ACC	0.573	0.655	0.603	0.612	0.652	0.564
Te1 ACC	0.698	0.743	0.713	0.690	0.738	0.686
Te2 ACC	0.581	0.596	0.608	0.586	0.601	0.574
Tr0 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Tr1 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Tr2 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Te0 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Te1 AUC	0.500	0.500	0.500	0.500	0.500	0.500
Te2 AUC	0.500	0.500	0.500	0.500	0.500	0.500

Tab. 8: Accuracy (ACC) and area under the curve (AUC) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) vital status using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction. There are three scores for each evaluation metric, found using a one-vs-all approach, for each train and test set to accommodate the three classes in vital status. 0 represents living, 1 represents died of disease, and 2 represents died of other causes. For example, predicting whether a breast cancer patient in the train set is living using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using area under the curve can be found in the row Tr0 AUC and the column PCA LR. These train and test sets were obtained through KFold cross validation with 5 splits; each number represents an average of returned values from each split. The green element represents the highest testing accuracy, while the red elements represent the lowest accuracy and AUC scores.

Vital Metric	SVD LR	SVD SVM rbf	SVD SVM lin	PCA LR	PCA SVM rbf	PCA SVM lin
Tr0 Sens	1.000	0.925	0.800	0.833	0.825	0.985
Tr1 Sens	1.000	0.877	1.000	0.831	0.752	1.000
Tr2 Sens	1.000	0.992	1.000	0.850	0.934	0.990
Te0 Sens	0.423	0.424	0.486	0.408	0.454	0.453
Te1 Sens	0.274	0.265	0.317	0.390	0.316	0.284
Te2 Sens	0.504	0.682	0.514	0.495	0.636	0.427
Tr0 Spec	1.000	0.986	1.000	0.914	0.949	0.994
Tr1 Spec	1.000	0.992	1.000	0.949	0.966	1.000
Tr2 Spec	1.000	0.926	1.000	0.870	0.854	0.992
Te0 Spec	0.644	0.766	0.661	0.712	0.746	0.618
Te1 Spec	0.837	0.905	0.842	0.790	0.890	0.809
Te2 Spec	0.626	0.517	0.664	0.641	0.556	0.671

Tab. 9: Sensitivity (Sens), and specificity (Spec) scores of train (Tr) and test (Te) sets. Each set predicts (Pred) vital status using logistic regression (LR), support vector machine (SVM) with radial basis function (rbf) kernel, and SVM with linear (lin) kernel models after either principal component analysis (PCA) or singular value decomposition (SVD) dimensionality reduction. There are three scores for each evaluation metric, found using a one-vs-all approach, for each train and test set to accommodate the three classes in vital status. 0 represents living, 1 represents died of disease, and 2 represents died of other causes. For example, predicting whether a breast cancer patient in the train set is living using a logistic regression model after principal component analysis dimensionality reduction and evaluating this prediction using sensitivity can be found in the row Tr0 Sens and the column PCA LR. These train and test sets were obtained through KFold cross validation with 5 splits; each number represents an average of returned values from each split. The red element represents the lowest value from the training sensitivity and specificity scores, while the green elements represent the best combination of testing sensitivity and specificity scores.

5.5 Genetic Interpretability

Each component obtained from running principal component analysis (PCA) and singular value decomposition (SVD) is assigned a weight from the logistic regression (LR) and support vector machines (SVM) models (see Figure 6). Because the PCA and SVD matrices contain information on the gene expression of a patient, and the models use labels from the treatment columns, the weights represent the reliability of the models predicting whether a patient has received a treatment based on their gene expression; the more closely related the gene expression and treatments are, the greater the reliability of the model and the higher the weight is. These weights thus represent the correlation between gene expression and the type of treatment a patient received. When a diagonal matrix containing these weights follows the PCA or SVD matrix in multiplication, the columns, or groups of gene expression, of the PCA and SVD matrices are scaled by the weights. Depending on how great these weight values are, they signify the importance of groups of genes and their expression in breast cancer.

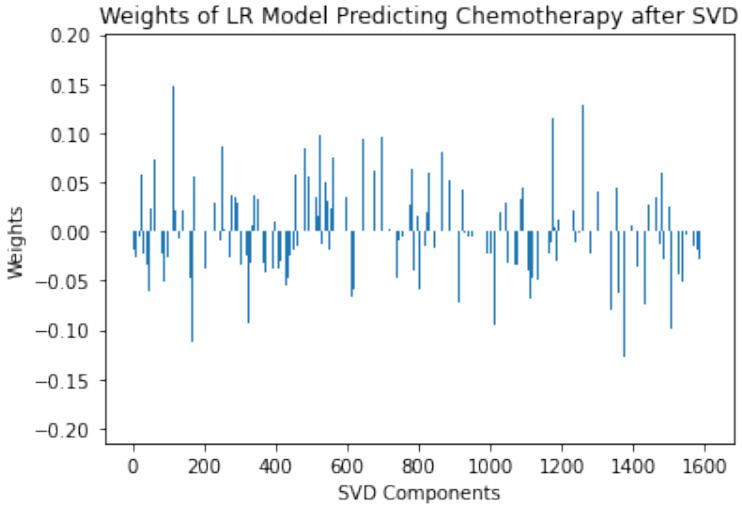


Fig. 6: Weights, or coefficients, from a logistic regression (LR) model. The X axis contains each of the 1600 principal components from the gene expression matrix reduced by singular value decomposition (SVD), and the Y axis contains the corresponding weight of each component. The weights represent the probability that a patient received chemotherapy based on each principal component; negative values indicate a negative relationship between the independent and dependent variables, or that as the gene expression of a principal component increases, the probability of receiving chemotherapy decreases. Conversely, positive weights indicate that an increase in gene expression results in a greater probability of receiving chemotherapy.

5.5.1 Scaled Inverse Transform of PCA and SVD Matrices

We perform an inverse transform of the SVD and PCA matrices after scaling them by the weights of each principal component from LR and SVM (see Figure 6). This results in levels of gene expression from each gene in the dataset scaled by their correlation to breast cancer treatments. We compare these levels to the original gene expression data to identify genes whose expression is significant in terms of treatments and their impact on breast cancer.

The highest level of expression in proportion to the original gene expression, 66.500–a level significantly higher than levels from other treatment models–comes from the gene ANKRD42, which we return in every model predicting hormone therapy (see Figure 7). The gene with the highest expression in predicting chemotherapy (29.605) is FAM125A, although we obtain this gene three times as well in predicting hormone

therapy. We find the genes WDR48, PLD6, ART3, and SEPN1 in predicting each of the four treatments. Although we obtain TMEM30B for all treatments except breast surgery, we find it most often in predicting chemotherapy. We similarly obtain DDAH2 most often in predicting chemotherapy, but we obtain it for all treatments except hormone therapy. We obtain CDC4EP2 three times in predicting radiotherapy, although we also obtain the gene once in predicting chemotherapy and hormone therapy. We likewise obtain SUCLG1 three times in predicting breast surgery, but we find the gene once in predicting radiotherapy.

6 Discussion

Because our models do not all perform optimally or consistently, it is important to note that the importance of genes we identify can only be ascertained through further research and experiments. Accuracy scores are commonly used to evaluate models, but results are often skewed when datasets contain unbalanced data. We thus rely on area under the curve (AUC) from test sets for our main evaluation of each model.

6.1 Area Under the Curve (AUC) Scores

Learning a good model can sometimes be limited by the need for interpretability; the ability to use and understand our results prevails over obtaining the greatest possible results. The results, in this case, are the area under the curve (AUC) scores we obtain from our models. AUC scores of the test sets predicting chemotherapy and hormone therapy, in particular, are high enough to suggest that genes we identify may indeed have an important correlation to treatments and an effect on breast cancer; most of these scores are between 0.7 and 0.9, with the highest being 0.865 for predicting chemotherapy using support vector machines (SVM) with a radial basis function (rbf) kernel after singular value decomposition (SVD). The pattern apparent in most predictions of treatment and vital status is that higher results are returned from SVM rbf models after SVD. This suggests that for this study, SVD is a better tool for dimensionality reduction, and that genetic data may not be represented best with a linear relationship. Linear relationships are most interpretable, however, as weights cannot be obtained from a SVM rbf model. We require these weights for our inverse transform and comparison to the original gene expression data. Results returned from logistic regression (LR) and linear SVM are not remarkably lower than results from SVM rbf, so we proceed to use them in our inverse transform to identify target genes.

AUC scores from the test sets predicting vital status are all 0.500, which indicates that our models are only capable of guessing whether a breast cancer patient is living, has died from cancer, or has died from other causes. This may be explained by other factors in the clinical dataset, such as age or tumor size, which could impact a patient's survival along with gene expression. Because of such poor predictions, we do not attempt to identify target genes based on models predicting vital status.

6.2 Sensitivity and Specificity

Across some models, we observe poor performance in terms of sensitivity and specificity where one score is often much higher than the other. This reveals that even models predicting chemotherapy and hormone therapy, which return optimal area under the curve (AUC) scores, may be guessing; sensitivity and specificity scores are inversely related, as guessing positive the majority of the time results in a high sensitivity score, but a low specificity score. We need both high sensitivity and high specificity scores to confidently rely upon the validity of our models. This is difficult in studies on genetic data, however, because genetic mutations and levels are gene expression vary widely among cancer patients. For this reason, we view any sensitivity and specificity scores that are close to one another and greater than 0.6 more positively than we may have in studying other types of data.

6.3 KFold Cross Validation

Regardless of how high our testing area under the curve (AUC) scores are, the gap between these scores and the AUC scores from test sets suggests that our models overfit. Overfitting poses the problem

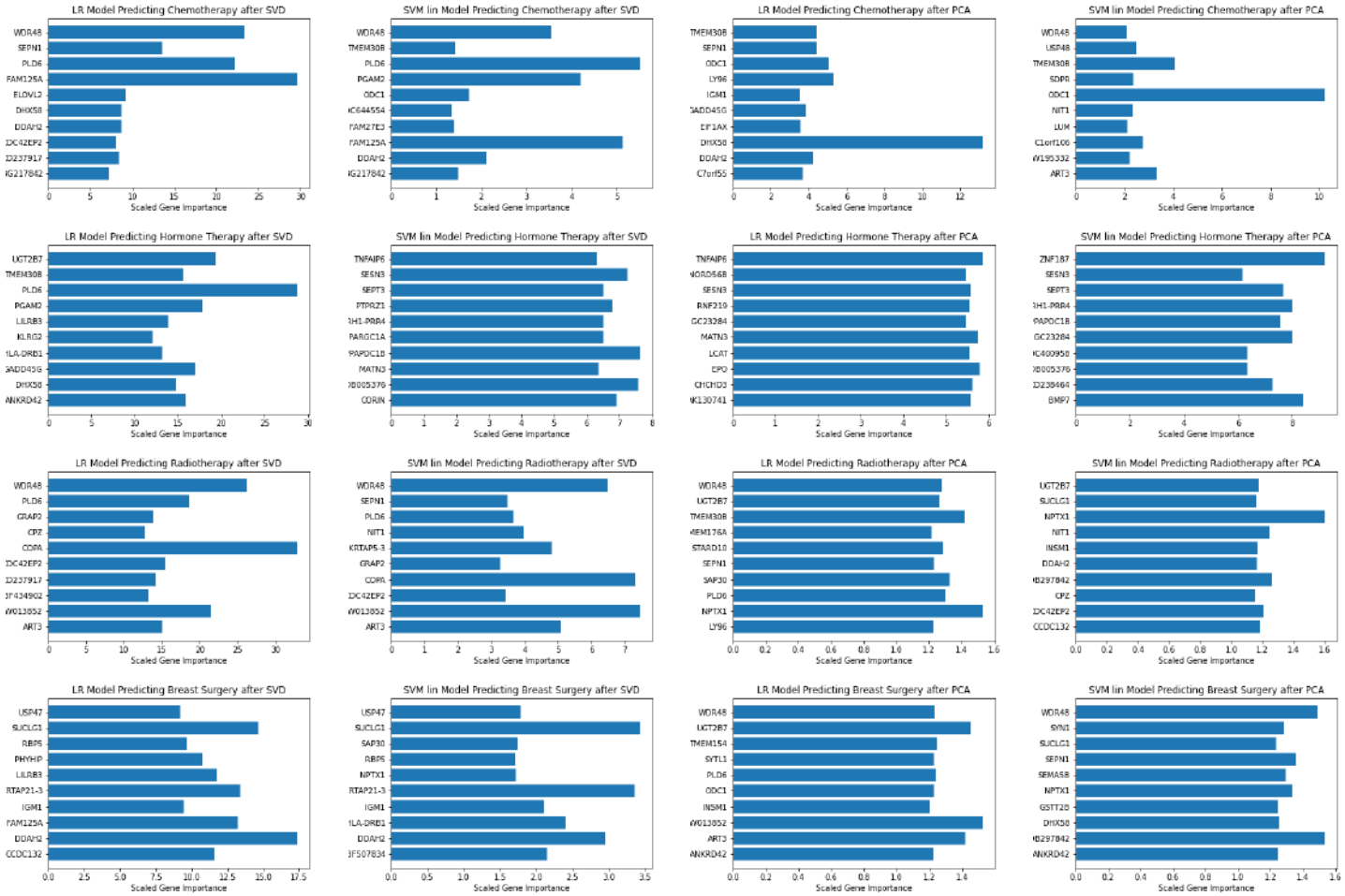


Fig. 7: Ten genes for each model with the highest positive correlation of gene expression to treatments. The scaled gene importance on the X axes represents gene expression levels scaled by their correlation to patients' reception of treatment in proportion to the original level of gene expression. Y axes represent ten genes with the greatest of these proportions. The first row contains models predicting chemotherapy, the second contains models predicting hormone therapy, the third contains models predicting radiotherapy, and the fourth contains models predicting breast surgery. The first column uses logistic regression (LR) models after singular value decomposition (SVD), the second uses support vector machines (SVM) models after SVD, the third uses LR models after principal component analysis (PCA), and the fourth uses SVM models after PCA. In the LR model predicting chemotherapy after SVD, the gene whose expression has the greatest positive correlation to the reception of chemotherapy is FAM125A, although TMEM30B is the gene most commonly found in the models predicting chemotherapy.

of generalizing our study to breast cancer patients as a whole; it is possible that genes we identify and their relation to breast cancer treatments are only applicable to the 1,881 patients in our dataframe. We thus perform KFold cross validation with 5 splits so that our models gain experience with various combinations of data in an attempt to counter overfitting. The lack of better results from cross validation, however, suggests that genes grouped into train and test sets are particularly different. For this reason, we use all gene expression data in our dataframe to identify target genes.

6.4 Scaled Inverse Transform of PCA and SVD Matrices

A common concern of dimensionality reduction is that information is lost from a dataset, but using weights from logistic regression (LR) and linear support vector machine (SVM) to scale our reduced dataset before completing an inverse transform allows us to get all information back while simultaneously retaining information learned from the models. This in turn allows us to view each gene in the original dataset whose expression has been scaled by how important they are in relation to treatments for breast cancer. By dividing the total scaled expression of each gene by the total original expression of each gene, we are left with values that represent exactly how the expression of each gene correlates with whether a patient has received chemotherapy, hormone therapy, radiotherapy, and breast surgery.

This study uses linear models to identify genes that may have a major role in breast cancer. The genes we identify have high positive correlations to the treatments breast cancer patients received, suggesting that genetics has an important role in determining a course of treatment. The expression of the genes WDR48, PLD6, ART3, and SEPN1 have the highest positive correlation to all four treatments: chemotherapy, hormone therapy, radiotherapy, and breast surgery. Individually, the expression of DDAH2 and TMEM30B is highly positively correlated with patients who have received chemotherapy, ANKRD42 ia highly positively correlated with patients who have received hormone therapy, CDC4EP2 is highly positively correlated with patients who have received radiotherapy, and SUCLG1 is highly positively correlated with patients who have received breast surgery.

Research thus far indicates that some of the genes we identified may, in fact, have an effect on breast cancer and the signalling pathways that are involved. WDR48 downregulates Akt activation and consequently promotes apoptosis and suppresses the proiferation of tumor cells [14]. PLD6, which localizes to the mitochondrial outer membrane and inhibits apoptosis, has recently been found to be overexpressed in breast cancer tumors [12]. ART3 is specifically overexpressed in triple negative breast cancer, where it activates Akt and ERK and increases cancer cell proliferation, invasion, and survival [38]. Although it is not related to breast cancer in our knowledge today, SUCLG1 has been identified as a genetic marker for kidney cancer [23]. Researching such genes and understanding their role in breast cancer may illuminate a successful direction in which the development of treatments could advance.

7 Conclusion

We perform singular value decomposition (SVD), principal component analysis (PCA), logistic regression (LR), and support vector machines (SVM) on gene expression data of 1,881 breast cancer patients from the METABRIC Breast Cancer dataset. Although we perform dimensionality reduction using SVD and PCA, we are still able to perform interpretability analysis and retain both original information and information gained through our models. We achieve this by scaling each principal component by weights from LR and linear SVM models before inversely transforming the scaled SVD and PCA matrices back to their original space. We are thus able to identify genes that highly positively correlate to a patient's reception of chemotherapy, hormone therapy, radio therapy, and breast surgery. The genes we identify that have a high positive correlation to all four treatments are WDR48, PLD6, ART3, and SEPN1, while DDAH2 and TMEM30B, ANKRD42, CDC4EP2, and SUCLG1 individually have a greater impact on chemotherapy, hormone therapy, radiotherapy, and breast surgery respectively. Further studies of these genes, perhaps in a lab setting, are necessary to determine how they effect breast cancer treatments and whether they could be linked to treatment resistance or death of patients; our suboptimal area under the curve (AUC) scores suggest that genes we identify may only be applicable to our 1,881 patients.

Better results in studying genetic data may be obtained through the advancement of nonlinear machine learning algorithms. We find the best AUC scores in predictions using SVM with a radial basis func-

tion (rbf) kernel, yet we are unable to use these results because weights for our inverse transform cannot be acquired through this model. Linear models are currently the most interpretable, but much of the data in the real world, such as genetic data, lacks linear relationships.

Difficulty in treating breast cancer, as with many cancers, arises in part from genetic mutations and levels of gene expression. Abnormalities vary greatly across patients, however, making it imperative that a wide range of genes and their effects on both cancer and existing treatments are understood. Patient attributes such as age and tumor size may also shed more light on patient outcomes; our inability to predict patient survival solely on gene expression attests to the complexity of cancer cells. Every discovery made in genetics and the interrelatedness of genes and patient attributes in cancer has the potential to revolutionize breast cancer treatment and increase the survival rate of patients.

8 Acknowledgements

Considering I dove into this project without any prior knowledge of coding, I am indebted to the SSI community for all I learned. First, thank you to Anne Lee for giving me my first opportunity to conduct research and consistently checking in on my progress. Thank you to Alex Tsun and Aleks Jovcic for teaching me the basics of python programming. Thank you to Franklyn Wang for showing me how to read research papers and demonstrating the power of statistics. Thank you to Dhruvik Parikh for teaching me how to write a research paper and for commenting on my writing. Thank you to the students who looked over my presentations and supported me with motivational words. Most of all, thank you to my mentor Andrew Shea for exchanging ideas and answering all the questions I sent his way.

References

[1] Zakariya Yahya Algamal and Muhammad Hisyam Lee. Penalized logistic regression with the adaptive lasso for gene selection in high-dimensional cancer classification. *Expert Systems with Applications*, 42(23):9326–9332, 2015.

[2] Hamid Behravan, Jaana M Hartikainen, Maria Tengström, Katri Pylkäs, Robert Winqvist, Veli-Matti Kosma, and Arto Mannermaa. Machine learning identifies interacting genetic variants contributing to breast cancer risk: A case study in finnish cases and controls. *Scientific reports*, 8(1):1–13, 2018.

[3] Howard A Burris III. Dual kinase inhibition in the treatment of breast cancer: initial experience with the egfr/erbb-2 inhibitor lapatinib. *The oncologist*, 9:10–15, 2004.

[4] Minsun Chang. Tamoxifen resistance in breast cancer. *Biomolecules & therapeutics*, 20(3):256, 2012.

[5] Cheng-Min Chao, Ya-Wen Yu, Bor-Wen Cheng, and Yao-Lung Kuo. Construction the model on the breast cancer survival analysis use support vector machine, logistic regression and decision tree. *Journal of medical systems*, 38(10):106, 2014.

[6] Yan Chen, Bin Han, Lei Zhu, Weiwei Wang, Lihua Li, and Jonathan M Lancaster. A supervised svd approach to ovarian cancer chemotherapy response prediction with across factor normalization. In *2009 3rd International Conference on Bioinformatics and Biomedical Engineering*, pages 1–4. IEEE, 2009.

[7] Hsiu-Ling Chou, Chung-Tay Yao, Sui-Lun Su, Chia-Yi Lee, Kuang-Yu Hu, Harn-Jing Terng, Yun-Wen Shih, Yu-Tien Chang, Yu-Fen Lu, Chi-Wen Chang, et al. Gene expression profiling of breast cancer survivability by pooled cdna microarray analysis using logistic regression, artificial neural networks and decision trees. *BMC bioinformatics*, 14(1):100, 2013.

[8] Amy S Clark, Kip West, Samantha Streicher, and Phillip A Dennis. Constitutive and inducible akt activity promotes resistance to chemotherapy, trastuzumab, or tamoxifen in breast cancer cells. *Molecular cancer therapeutics*, 1(9):707–717, 2002.

[9] Carol DeSantis, Jiemin Ma, Leah Bryan, and Ahmedin Jemal. Breast cancer statistics, 2013. *CA: a cancer journal for clinicians*, 64(1):52–62, 2014.

[10] Dalia ELFadl, Victoria C Hodgkinson, Ervine D Long, Lucy Scaife, Philip J Drew, Michael J Lind, and Lynn Cawkwell. A pilot study to investigate the role of the 26s proteasome in radiotherapy resistance and loco-regional recurrence following breast conserving therapy for early breast cancer. *The Breast*, 20(4):334–337, 2011.

[11] Ping Fan and V Craig Jordan. New insights into acquired endocrine resistance of breast cancer. *Cancer drug resistance (Alhambra, Calif.)*, 2:198, 2019.

[12] Kristen Fite. *Dysregulation of Phospholipase D (PLD) isoforms increases breast cancer cell invasion*. PhD thesis, Wright State University, 2017.

[13] Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using bayesian networks to analyze expression data. *Journal of computational biology*, 7(3-4):601–620, 2000.

[14] Narmadha Reddy Gangula and Subbareddy Maddika. Wd repeat protein wdr48 in complex with deubiquitinase usp12 suppresses akt-dependent cell survival signaling by stabilizing ph domain leucine-rich repeat protein phosphatase 1 (phlpp1). *Journal of Biological Chemistry*, 288(48):34545–34554, 2013.

[15] David Horn and Inon Axel. Novel clustering algorithm for microarray expression data in a truncated svd space. *Bioinformatics*, 19(9):1110–1115, 2003.

[16] Ying-Lin Hsu, Po-Yu Huang, and Dung-Tsa Chen. Sparse principal component analysis in cancer research. *Translational cancer research*, 3(3):182, 2014.

[17] Ricky W Johnstone, Astrid A Ruefli, and Scott W Lowe. Apoptosis: a link between cancer genetics and chemotherapy. *Cell*, 108(2):153–164, 2002.

[18] Muhammad Umer Khan, Jong Pill Choi, Hyunjung Shin, and Minkoo Kim. Predicting breast cancer survivability using fuzzy decision trees for personalized healthcare. In *2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 5148–5151. IEEE, 2008.

[19] Umer Khan, Hyunjung Shin, Jong Pill Choi, and Minkoo Kim. wfdt: weighted fuzzy decision trees for prognosis of breast cancer survivability. In *Proceedings of the 7th Australasian Data Mining Conference-Volume 87*, pages 141–152. Citeseer, 2008.

[20] Jae-Sung Kim, Hyun-Ah Kim, Min-Ki Seong, Hyesil Seol, Jeong Su Oh, Eun-Kyu Kim, Jong Wook Chang, Sang-Gu Hwang, and Woo Chul Noh. Stat3-survivin signaling mediates a poor response to radiotherapy in her2-positive breast cancers. *Oncotarget*, 7(6):7055, 2016.

[21] Ji-Yeon Kim, Eunjin Lee, Kyunghee Park, Woong-Yang Park, Hae Hyun Jung, Jin Seok Ahn, Young-Hyuck Im, and Yeon Hee Park. Clinical implications of genomic profiles in metastatic breast cancer with a focus on tp53 and pik3ca, the most frequently mutated genes. *Oncotarget*, 8(17):27997, 2017.

[22] Konstantina Kourou, Themis P Exarchos, Konstantinos P Exarchos, Michalis V Karamouzis, and Dimitrios I Fotiadis. Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, 13:8–17, 2015.

[23] Rasmus Krempel, Pranav Kulkarni, Annie Yim, Ulrich Lang, Bianca Habermann, and Peter Frommolt. Integrative analysis and machine learning on cancer genomics data using the cancer systems biology database (cancersysdb). *BMC bioinformatics*, 19(1):156, 2018.

[24] Donghwan Lee, Woojoo Lee, Youngjo Lee, and Yudi Pawitan. Super-sparse principal component analyses for high-throughput genomic data. *BMC bioinformatics*, 11(1):296, 2010.

[25] Paul L Nguyen, Alphonse G Taghian, Matthew S Katz, Andrzej Niemierko, Rita F Abi Raad, Whitney L Boon, Jennifer R Bellon, Julia S Wong, Barbara L Smith, and Jay R Harris. Breast cancer subtype approximated by estrogen receptor, progesterone receptor, and her-2 is associated with local and distant recurrence after breast-conserving therapy. *Journal of clinical oncology*, 26(14):2373–2378, 2008.

[26] Anupama Pal, Wei Huang, Xin Li, Kathy A Toy, Zaneta Nikolovska-Coleska, and Celina G Kleer. Ccn6 modulates bmp signaling via the smad-independent tak1/p38 pathway, acting to suppress metastasis of breast cancer. *Cancer research*, 72(18):4818–4828, 2012.

[27] Meisen Pan and Fen Zhang. Research on equivalence of svd and pca in medical image tilt correction. *Journal of Fiber Bioengineering and Informatics*, 8(3):453–460, 2015.

[28] Bernard Pereira, Suet-Feung Chin, Oscar M Rueda, Hans-Kristian Moen Vollan, Elena Provenzano, Helen A Bardwell, Michelle Pugh, Linda Jones, Roslin Russell, Stephen-John Sammut, et al. The somatic mutation profiles of 2,433 breast cancers refine their genomic and transcriptomic landscapes. *Nature communications*, 7(1):1–16, 2016.

[29] John C Reed. Bcl-2 family proteins. *Oncogene*, 17(25):3225–3236, 1998.

[30] Marylyn D Ritchie, Lance W Hahn, Nady Roodi, L Renee Bailey, William D Dupont, Fritz F Parl, and Jason H Moore. Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. *The American Journal of Human Genetics*, 69(1):138–147, 2001.

[31] Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

[32] Maureen A Sartor, George D Leikauf, and Mario Medvedovic. Lrpath: a logistic regression approach for identifying enriched biological groups in gene expression data. *Bioinformatics*, 25(2):211–217, 2009.

[33] Alok Sharma and Kuldip K Paliwal. Cancer classification by gradient lda technique using microarray gene expression data. *Data & Knowledge Engineering*, 66(2):338–347, 2008.

[34] Kevin Shee, Amanda Jiang, Frederick S Varn, Stephanie Liu, Nicole A Traphagen, Philip Owens, Cynthia X Ma, Jeremy Hoog, Chao Cheng, Todd R Golub, et al. Cytokine sensitivity screening highlights bmp4 pathway signaling as a therapeutic opportunity in er+ breast cancer. *The FASEB Journal*, 33(2):1644–1657, 2019.

[35] Li Shen and Eng Chong Tan. Pls and svd based penalized logistic regression for cancer classification using microarray data. In *Proceedings of the 3rd Asia-Pacific Bioinformatics conference*, pages 219–228. World Scientific, 2005.

[36] Shirish Krishnaj Shevade and S Sathiya Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.

[37] Jiang Shou, Suleiman Massarweh, C Kent Osborne, Alan E Wakeling, Simale Ali, Heidi Weiss, and Rachel Schiff. Mechanisms of tamoxifen resistance: increased estrogen receptor-her2/neu cross-talk in er/her2–positive breast cancer. *Journal of the National Cancer Institute*, 96(12):926–935, 2004.

[38] Ling Tan, Xiaodan Song, Xin Sun, Ning Wang, Ying Qu, and Zhijun Sun. Art3 regulates triple-negative breast cancer cell function via activation of akt and erk pathways. *Oncotarget*, 7(29):46589, 2016.

[39] Youlin Tuo, Ning An, and Ming Zhang. Feature genes in metastatic breast cancer identified by metade and svm classifier methods. *Molecular medicine reports*, 17(3):4281–4290, 2018.

[40] Jun S Wei, Braden T Greer, Frank Westermann, Seth M Steinberg, Chang-Gue Son, Qing-Rong Chen, Craig C Whiteford, Sven Bilke, Alexei L Krasnoselsky, Nicola Cenacchi, et al. Prediction of clinical outcome using gene expression profiling and artificial neural networks for patients with neuroblastoma. *Cancer research*, 64(19):6883–6891, 2004.

[41] Peiliang Xu. Truncated svd methods for discrete linear ill-posed problems. *Geophysical Journal International*, 135(2):505–514, 1998.

Impact of Gentrification-Induced Displacement on a National Scale

Benny Sun

Abstract

Gentrification is the controversial process of renovating a poor neighborhood to conform to upper-class tastes, resulting in housing price spikes and expense concerns. While the link between gentrification and displacement is well-known, there is little research over the impact of displacement itself in an economic, political, and social context on a national scale. This paper compares nearby recipient neighborhoods of displaced families to gentrifying neighborhoods in several aspects, including pollution, median life expectancy, test-scores, and incarceration rates. This paper identifies 1148 gentrifying tracts and 1907 recipient tracts in the United States. It concludes that families who become displaced from gentrifying neighborhoods tend to move areas with worse outcomes. For example, recipient neighborhoods contain 31% greater toxicity concentration, an 8-percentile increase in the incarceration rate, a 10-percentile reduction in test scores, and a lower median life expectancy by 1 year. There is still a similar result even after controlling for household income and percent-African American. Overall, this research paper provides one answer to an important question: what is the impact of gentrification-inducement on families? These results add further context to the urgent issue of gentrification in America's cities, demonstrating how necessary steps are required to protect America's poor urban residents from gentrification.

1 Introduction

Neighborhoods are the building blocks of American society. It has an over-arching impact on one's life, including one's friend choices, job status, happiness, and even life expectancy. Unsurprisingly, the ability to reside in a stable neighborhood is a crucial priority for any family living in the United States. The field of Urban Economics focuses on this issue, analyzing trends in housing, education, and local governments in city areas. However, over several decades, many researchers this field have observed an alarming trend: Gentrification. Described as an influx of wealthy residents or investment into an at-risk neighborhood, many have criticized this process for increasing housing costs for poorer residents. As gentrification means added renovations to housing units or extra amenities to conform to upper-class tastes, the living expenses too often become unbearable for many [28]. If rent prices are too high for existing residents, many must choose between forgoing rent, medication, or food just to survive.

Consequently, studies have linked gentrification to significant hikes to housing costs [14]. Other studies show that gentrification can lead to higher levels of displacement [4, 27, 28, 16]. These papers focus on specific geographic areas such as London, Boston, or San Francisco. Because of gentrification, poor households are forced out of their homes (some through evictions and others willingly) to move to cheaper-cost neighborhoods. While some have argued about the expanded job opportunities or municipal resources for existing residents, [13], others have demonstrated the poorer residents cannot live there for long [28].

However, despite this wide literature base, few studies have addressed the impacts of displacement on a national scale. While [17] and [25] quantify how many Americans are displaced each year, there is little research around the concrete effects of displacement in economic, social, and political terms. This could be due to a lack of publicly available data of movers, or the is data limited to a single metropolitan area. Unfortunately, the gap in research is hampering efforts to clarify the gravity of gentrification for urban policymakers. Without a clear story of what happens to displaced households, there is no urgency to act on the issue itself.

To address this gap, this study will compare gentrifying neighborhoods with recipient neighborhoods to evaluate the impact of displacement. Recipient neighborhoods are defined as destination census tracts that displaced families from gentrifying neighborhoods move to for cost reasons. This paper will use the

NBER's Census Tract Distance Data Set to identify nearby census tracts to gentrifying neighborhoods [1]. Candidates for recipient tracts will include cheap neighborhoods with higher population growth rates. After identifying gentrifying and recipient tracts, this study will then draw comparisons between them on environmental pollution, educational quality, crime rate, and life expectancy. If the general quality of life is worse in recipient tracts, then gentrification-induced displacement leads to families moving to worse-off areas. Thus, this study seeks to understand the consequences of displacement on a national scale.

Overall, this research is significant because it reveals the full story of displacement in the United States. By answering a major question of whether Americans are negatively impacted by displacement, it becomes the logical next step from other important studies. Instead of proving the link between gentrification and displacement, it contextualizes why displacement itself is an urgent problem in the field of Urban Economics. This paper is organized as followed. Section 2 discusses the Literature Review. Section 3 shows the research goal. Section 4 describes the methodology. Section 5 includes Results & Discussion. Finally, Section 6 ends with a conclusion.

2 Literature Review

The term gentrification originates from the 1950s and 1960s, describing the influx of 'gentry' into London's lower echelon neighborhoods [28]. Since then, researchers have used gentrification as a term for rent or housing price hikes due to renovations, investment, or migration of wealthy residents. Gentrification has been referred to as a form of "neighborhood change" characterized by rising property values [22]. Often at the street level, there is a visible upgrading of buildings with housing being refurbished and businesses being established. However, these upgrades often come at the cost of accessibility for its residents. [9] found that gentrifying neighborhoods in Philadelphia lost low-cost housing options at a rate five times faster than non-gentrifying neighborhoods. The brunt of these costs is placed predominantly on the shoulders of the disadvantaged.

There are multiple causes of gentrification. [11] finds that environmental quality improvements such as solar panel installments or the upgrading of parks in poorer communities can increase living expenses for residents nearby. Even investments with good-intentions can have unintended consequences for gentrification. Moreover, [20] find that market-rate housing can cause luxury housing development near at-risk neighborhoods. In turn, as home developers continue to renovate to attract wealthy clients, there is a subsequent price hike for neighboring houses. Others are also now analyzing the effect of the public sector on gentrification such as transportation investment [28]. The production of railways or train stations brings economic activity closer together, increasing the demand for housing in these areas as well. Overall, the causes of gentrification is a complex issue that has been researched in depth.

In general, studies in this field have varying definitions of gentrification and even displacement, which often changes the outcome of certain results. For instance, [6] defines gentrification by analyzing changes in rental costs, the creation of new housing stock, and increases in income. Conversely, [26] measures an increase in educated or wealthy residents in a given area. For displacement, some have only analyzed the evictions rate in a gentrifying neighborhood [13] while others focus on exit rates of all poor neighborhoods [4]. This added confusion creates unequal or conflicting results across the academic space. This study will attempt to choose the most logical definitions to have the most accurate results.

Moreover, most studies try to find a link between gentrification and displacement in a specific metropolitan area. [5] finds that half of the studies on gentrification discuss the displacement link, rather than analyzing the consequences of displacement. There has been several conclusions generated around this topic. For instance, [4] utilizes Longitudinal Census Data in London from the 1980s to the 1990s and analyzes changes in household income and population demographics. They find that gentrification leads to greater displacement of vulnerable groups in London. There was a reduction of unskilled workers by 78% and inactive workers by 46%.

[26] also analyzes the link between displacement and gentrification through focusing on exit rates of poor households in the Boston Area, finding contradictory results because of differing definitions. [24] utilizes the 1980s and 1990s housing data from New York City to demonstrate how gentrification leads to displacement for poorer residents. Most of these studies follow a similar pattern of analyzing a single city for gentrification and displacement. Thus, there have been little to none national-level studies on gentrification in this field.

Several studies have indicated that gentrification has little to no impact on displacement [13, 10, 12]. However, others have argued that these studies focus on comparing the mobility rates of all low-income groups. The issue becomes that low-income groups are more likely to move in general, but the movement occurring in gentrifying neighborhoods is specifically because of displacement [22]. Furthermore, when analyzing gentrifying neighborhoods specifically, the amount of low-income households moving into these areas declines significantly, revealing evidence of exclusionary displacement [10]. For this reason, gentrifying neighborhoods have a net-loss of poor households as rent and housing price increases force the disadvantaged out and the wealthy in.

Alarmingly, gentrification has a disproportionate impact on socially vulnerable and racial minorities. [19] finds strong evidence of how displacement occurs in areas with high concentrations of racial minorities such as African-Americans. Thus, many studies have used a reduction of racial minorities as another measure to identify gentrifying tracts. [27] utilizes the New York City Housing and Vacancy Survey data from 2002 to 2008 to measure how vulnerable groups are affected by gentrification. They conclude that renters were nearly twice as likely to become displaced in New York City compared to home-owners, demonstrating the precarious nature of unstable housing prices. Therefore, gentrification has been described as a 'human rights violation' with its origins in racial segregation and historical white flight.

Overall, by using the annual housing survey to track movers and displaced households, [25] found that nearly 500,000 households were forced to move away from 1974 to 1976. Another study analyzes 1 neighborhood in 5 American cities and concludes that 23% of movers in these areas considered their movement as a form of displacement [23]. While these studies are dated, they provide important insight into the prevalence of displacement.

Unfortunately, the methodology for analyzing displacement's impact on poor families is very limited. By employing Census Tract Data from 1990 and 2000, [18] discovers that migrants entering into gentrifying neighborhoods are more likely to be educated and wealthy, while those exiting these neighborhoods are likely poor and uneducated. This means that the demographic composition of gentrifying tracts changes throughout the gentrification process, becoming less diverse economically and racially. However, the study does not track how the movers are affected in this process. Using the LA FANs Survey, a comprehensive data set that tracks residential movement, [21] finds that genuine results of gentrification-induced displacement. Namely, displaced households moved to areas with higher crime rates, greater pollution, and worse education. However, this study is limited to the area of Los Angeles.

The impact of displacement has crucial consequences on one's outcomes. [7] focuses on the impacts of neighborhoods themselves through analyzing de-identified tax records. They conclude that living in a neighborhood below one's income percentile can lower their future incomes by 0.7% for each year. [8] also discovers the neighborhoods are vastly unequal across the United States, where opportunity is significantly lacking in certain areas. Overall, this study aims to contribute to existing research gaps by comparing recipient neighborhoods of displaced households to gentrifying tracts in social, economic, and political terms.

3 Purpose

1. Identify Gentrifying and Recipient Tracts in America using a clear pass-check test.
2. Compare Tracts on Pollution, Educational Quality, Crime, and Life Expectancy to Determine Quality of Life Difference
3. Derive a Conclusion on Whether Displaced Families Move to Worse-Off Areas Using the Data as Evidence

4 Methods

This study aims to compare gentrifying neighborhoods to recipient neighborhoods of displaced households. This paper must first identify both groups by assessing changes at the census tract level (the basic unit of a neighborhood). Therefore, John Logan of Brown University's Longitudinal Tract Data Base is used because it compares the characteristics of census tracts in all 50 states from 2000 and 2013[15]. To analyze

census tracts in urban areas (where gentrification traditionally occurs), the paper only conducts data analysis on census tracts with a CBSA code, which amounts to around 69,000 census tracts. From here, this paper adopts Freeman's methodology of identifying gentrifying tracts (the most cited paper on gentrification[13]). This includes passing several checks.

2000 Census Eligibility	2010 Census Eligibility
Median Home Value below 40th percentile within CBSA	Increase in Median Home Value above 60th percentile within CBSA
Median Household Income below 40th percentile within CBSA	Increase in Median Household Income above 60th percentile within CBSA
Population Greater than 500	Population Greater than 500
Any College-Educated Residents	Increase in College-Educated Residents above 60th percentile within CBSA

Figure 1: Identifying Gentrifying Neighborhoods

Because the process of gentrification is caused by an influx of richer, more educated residents moving into a poorer area, changes in household income, %-college-educated, and home values are analyzed.

While this study is focused on identifying and quantifying the impacts of displacement, there was no publicly available movers data to track displacement. Instead, this study will analyze population changes in nearby census tracts of gentrifying neighborhoods which are called "Recipient Neighborhoods". This study assumes that the nearby-poorer neighborhoods with high population growth rates are the neighborhoods that displaced families move to after living in a gentrifying area. This provides the best alternative data on displacement.

Some other assumptions include the fact that most displaced households will live within the same CBSA code and have the same job. Thus, census tracts within 10 miles of a gentrifying tract are considered as candidates. Here, the NBER's Census Tract Distance Database is utilized to find census tracts within 10 miles of a gentrifying tract. Then, the paper adopts a similar eligibility test for recipient neighborhoods. Generally, poorer neighborhoods were chosen because they are cheaper to live in. The numbers/qualifications of the eligibility test were chosen to provide a comparable amount of recipient census tracts with the gentrifying tracts.

Recipient Census Tract Eligibility in 2013
Population Growth Rate above 60th percentile (from 2000 to 2013)
Median Household Income below 30th percentile
Median Home Value below 30th percentile

Figure 2: Identifying Recipient Neighborhoods

After identifying gentrifying and recipient census tracts, this study will compare each on four different factors: toxicity concentration to measure pollution, incarceration rates to measure crime, 3rd Grade math test-scores to measure education quality and life expectancy. For toxicity concentration, this paper utilizes the Census Bureau's Risk-Environment Screening Index (RSEI) which provides data on toxicity concentration on a census-tract level[2]. For test scores and incarceration rates, this paper uses Dr. Raj Chetty's Opportunity Insights dataset[8]. For life expectancy, this research uses the CDC's Life Expectancy 2010-2015 data set[3].

Comparisons among gentrifying tracts and recipient tracts will include forming a percentile-based number within a CBSA for some variables (incarceration rates and test-scores) while just comparing the national averages for other variables(toxicity concentration and life expectancy). This is because some cities have a different level of expectations for these factors, so the variables had to be adjusted accordingly. The statistical comparisons will occur in four major tests.

1. Comparing the mean of each variable between recipient and gentrifying tracts

2. Conducting a t-test to measure the significance in difference of means
3. Conducting a non-parametric rank sums test to further show the difference in means given that the standard deviation is large
4. Performing linear regression with gentrification (as a categorical variable), household income, and %-African American as independent variables and the four factors as the dependent

This is the equation for the linear regression model:

$$Y_i = b_0 + b_1 X_1 + b_2 X_2 + b_3 X_3 + \varepsilon$$

The first variable is the intercept. The second term contains gentrification as a categorical variable (0 = gentrified and 1 = recipient). The third term is median household income. The fourth term is %-African American. The last term is a normal zero-mean random variable. The model controls for household income and %-African American so this paper can understand more accurately how gentrification impacts the dependent variable. Y_i are the four factors that will be tested individually: pollution, test-scores, incarceration rates, and life expectancy.

If recipient tracts have a generally worse quality of life, it demonstrates the negative effects of displacement and is a strong proof that policymakers and private individuals should account for the effects of gentrification when renovating or investing in a given area.

5 Results and Discussion

5.1 Identifying Gentrified and Recipient Tracts

After cleaning the LTDB, 1148 gentrifying census tracts and 1907 recipient census tracts in the United States were identified. This means that around 10% of poorer census tracts with CBSA codes gentrified from 2000 to 2013. This is comparable to [22] who found 1049 gentrifying census tracts with a similar methodology. Overall, this study builds a stronger base for how gentrification is a pressing issue in the United States. In one instance, a gentrifying tract in Laramie County, Wyoming experienced the average home value increase by 700% in just 10 years. When analyzing the top 10 cities with the largest amount of gentrifying tracts, they were as follows:

Figure 3: Gentrifying Tracts in Top Cities

The top 10 cities in the United States with the most gentrifying tracts accounted for 28% of total gentrified tracts. From Figure 3, New York City has the most amount of gentrifying tracts by far with 77. Afterward, Washington DC had 44, Philadelphia had 39, and Atlanta had 37. Ostensibly, this result demonstrates that large pockets of gentrification occur in the near the oldest and largest cities in the United States. When doing further analysis, gentrification tends to occur in areas near central business districts,

where employers may be seeking more college-educated workers. Wealthier residents are also drawn into these areas because they prefer the bustling urban lifestyle. However, further research should be conducted to analyze how city-characteristics may impact the rate or occurrence of gentrification.

In Figure 4, most of the gentrifying tracts occurred in the states of California, Texas, New York, Florida, and Pennsylvania. Most states with a large amount of gentrifying tracts were near the coast, which is typically where there is a plethora of economic activity and urban development. Unsurprisingly, most rural-majority states such as the Midwest had a small amount of gentrifying tracts, while most gentrifying tracts occurred in populated states with large metropolitan areas.

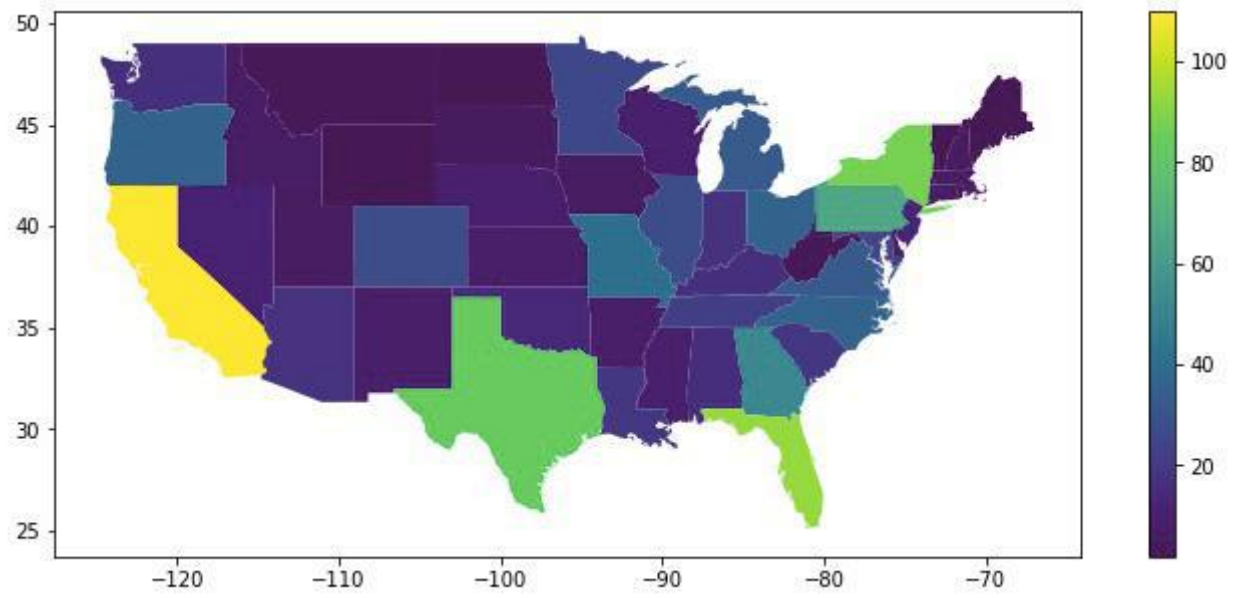


Figure 4: Gentrifying Tracts in America

When looking closer at statistics of figure 4, nearly 37% of all metropolitan areas or 342 CBSA areas had at least one gentrifying tract. However, instead of being spread out, most of the gentrification occurs in concentrated areas. Just 24 CBSA areas that had greater than 10 gentrifying tracts accounted for nearly half of all gentrification in the United States. This means that gentrification is prevalent and focused on certain cities within America, rather than being a universal issue found in all places.

Recipient neighborhoods followed a similar pattern of concentration because the study mainly looked at poorer neighborhoods with higher population growth rates that were nearby to gentrifying neighborhoods. Again, this method is not perfect, and based on predicting where people move, so more accurate depictions of movers data could improve the methodology in the future.

5.2 Comparing Gentrified and Recipient Tracts

Now, this section will compare the 1148 gentrifying tracts with the 1907 recipient tracts of displaced households. This will aid in identifying and quantifying the impact of displacement from a gentrifying neighborhood.

5.2.1 Toxicity Concentration

The toxicity concentration score from the RSI provides comprehensive data on the total amount of toxic pollutants released into the atmosphere and water supply of a given census tract. The sources of these pollutants mostly originate from factories, chemical plants, cars, and water leakage. After comparing the average toxicity concentration score, this important result was achieved in Figure 5:

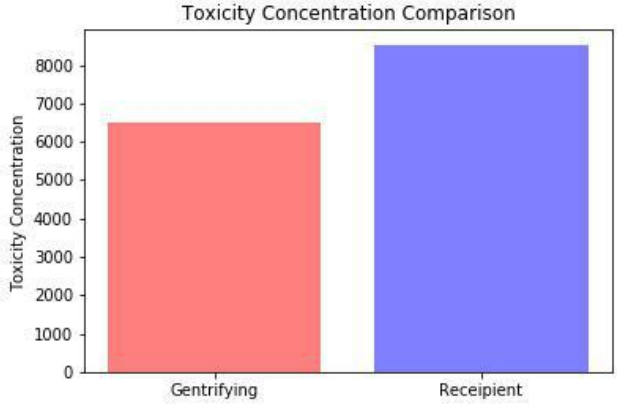


Figure 5: Toxicity Concentration Comparison

On average, recipient neighborhoods have 31% greater toxicity concentration, highlighting how families can be displaced to areas with greater pollution. After conducting a t-test on the difference of means, there is a p-value of 0.0042. Because the data was not normally distributed and the standard deviation was so high, rank sums were also employed with a p-value of 0.0022. Thus, the null hypothesis can be rejected with 95% confidence that both means are the same.

	p-value	**co-efficient**
Gentrification	0.039	0.08112
Household Income	0.023	-0.0441
% African American	0.183	0.2183

Figure 6: Toxicity Linear Regression Model (R-Squared of 0.059)

Because the standard deviation of the toxicity concentration was so high, we had to take the log of the data before normalizing everything else into the z-score. This made the results more accurate in the linear regression. Because co-efficient is positive and the p-value is below the 5% margin, there is strong evidence that gentrified neighborhoods have significantly less water and air pollution, when controlling for household income and %-African American. Overall, this result makes sense because wealthy residents would choose to move into areas with less pollution-producing plants nearby such as factories or chemical plants. Because poorer families have less financial resources and political power, they are unable to escape the pollution prevalent in poorer neighborhoods. Moreover, the renovation process could make neighborhoods cleaner (such as solar panel installments or closing coal plants) at the expense of rising housing expenditures. In turn, gentrified areas have less pollution because there is less highly-polluting activity. On the other hand, other factors such as cleaner cars tend to be more expensive which reduces pollution, and thus less prevalent in poorer neighborhoods. The link between environmental discrimination and displacement has not been discussed before, so this result could shed some important light on gentrification.

5.2.2 Incarceration Rates

To analyze the differing incarceration rates among census tracts, the median percentile within a CBSA was measured for a more fair comparison. This is because some cities are more aggressive with their policing than others.

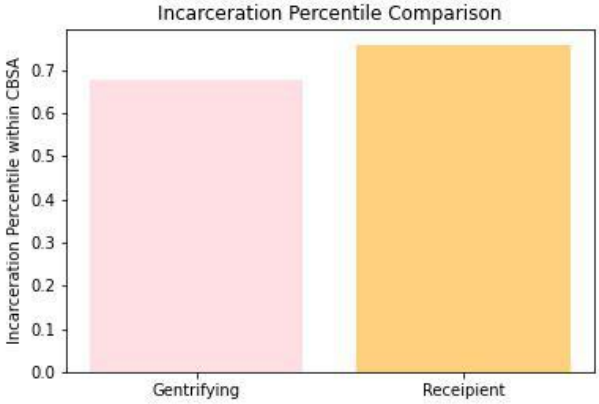


Figure 7: Incarceration Percentile Comparison

From Figure 7, moving to a recipient neighborhood increases the incarceration-rate percentile by 8 points on average. In turn, families must face higher crime rates in their respective areas compared to a gentrifying neighborhood.

After conducting a t-test and non-parametric rank sums, there is further evidence of the clear difference in incarceration rates. The t-test displayed a 1.7×10^{-17} in p-value, while the rank sums showed a 4.1×10^{-12} in p-value. Both values were below the 5% margin.

	p-value	**co-efficient**
Gentrification	0	0.1347
Household Income	0	-0.1632
% African American	0	0.3488

Figure 8: Jail Linear Regression Model (R-Squared of 0.192)

Because the co-efficient of gentrification is positive, recipient neighborhoods face higher incarceration rate-percentiles. Controlling for household income and %-African American, being a recipient community is equivalent in incarceration by 3.68 percentile points.

Typically, an influx of wealthier residents may mean stronger political support for comprehensive policing measures, which could deter crime in the long term. However, for families who are displaced, they do not receive the same benefits and are instead forced into poorer neighborhoods with greater crime rates. Moreover, because recipient neighborhoods lack the same economic or job opportunities that gentrifying neighborhoods have, these areas usually have more crime. While proponents of gentrification argue that the reduction in crime is a positive impact, these results indicate that gentrification could simply be pushing the brunt of the costs onto less-fortunate census tracts. Thus, this result further proves another negative impact of displacement.

5.2.3 Education Quality

Educational quality was based on percentiles within a CBSA of the median 3rd-grade math test score, taken from Dr. Raj Chetty's Opportunity Insights Data Set. Again, education standards vary across state lines, so adjustments had to be made accordingly.

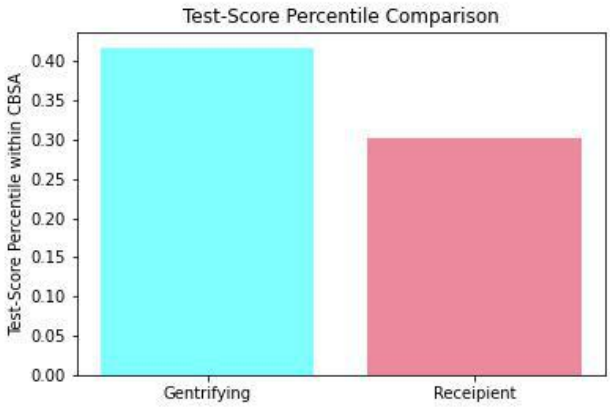


Figure 9: Test-Score Percentile Comparison

From this result, recipient neighborhoods have on average a 33% lower test-score percentile within a CBSA or around 11 percentile points lower. Using t-test and rank sums, this conclusion is statistically sound. For the t-test, the p-value was 2.0×10^{-21} and for rank sums, it was 1.5×10^{-19}. Thus, the null hypothesis can be rejected, which proves that recipient neighborhoods on average have lower test scores.

	p-value	**co-efficient**
Gentrification	0	0.1347
Household Income	0.192	-0.1632
% African American	0	0.3488

Figure 10: Test Linear Regression Model (R-Squared of 0.068)

Surprisingly, household income is not statistically significant. From this, results show that gentrification must play a greater role in determining test-scores. Also, when controlling for household income and % African American in the linear regression, living in a recipient community decreases the average math score by 4.52 percentile points. Overall, when property values go up, so do revenues from property taxes. Therefore, local budgets increase in the long term, allowing municipalities to spend more on school budgets. This could be in the form of hiring more teachers, buying new books, adding more classrooms, or implementing more technology. Conversely, poorer, recipient neighborhoods have a weaker tax base because the median household income is so much lower, so they are unable to afford these amenities. While test scores are not a perfect indicator of educational quality, it does show if the school has the resources to prepare their students on standardized tests in elementary school. Recipient neighborhoods are lacking in this department, demonstrating the need for more equal educational opportunities.

5.2.4 Life Expectancy

From the data analysis, gentrified neighborhoods had an average life expectancy of one more year than recipient neighborhoods (from 77 years to 76 years).

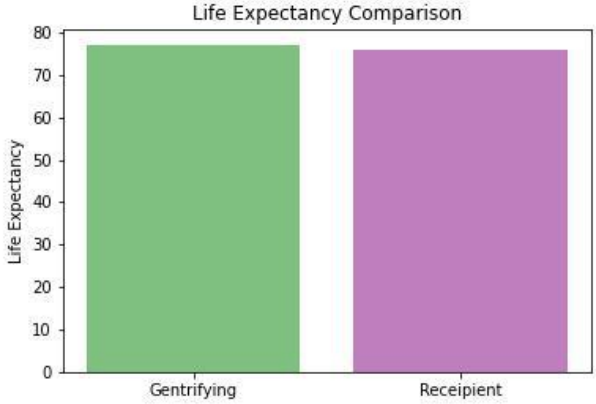


Figure 11: Life Expectancy Comparison

From this result, we can conclude based that the three other factors (environmental pollution, crime rate, and educational systems) have an impact on mortality. Even though one year may seem insignificant, the quality of life may be extremely impacted by the displacement event.

Using both the t-test and rank sums, the null hypothesis is rejected that the life expectancy means of gentrified and recipient neighborhoods were the same. The t-test p-value returned 6.1×10^{-9} while the rank sums p-value returned 7.4×10^{-13}.

	p-value	**co-efficient**
Gentrification	0	0.1390
Household Income	0	0.3989
% African American	0	0.017

Figure 12: Life Linear Regression Model (R-Squared of 0.273)

The linear regression model however had a strange result. It shows gentrification in the positive direction, meaning that recipient neighborhoods have a greater life expectancy even though previous analysis suggests otherwise. This may be because household-income and gentrification are highly-correlated, so household income affected the results of gentrification. Otherwise, this result suggests further more accurate research in the future.

6 Conclusion

This study provides one answer to the question: what are the impacts of gentrification-induced displacement? First, this paper utilized the Freeman method of identifying gentrifying tracts through examining increases in household income, professionals in the population, and median home value. For recipient tracts, this paper used its unique methodology of finding nearby census tracts with lower median home values and higher population growth rates. Initially, 1148 gentrifying tracts and 1907 recipient tracts were identified in the United States. Most of these neighborhoods were concentrated in larger urban areas rather than being spread out.

Afterward, this paper compared gentrifying tracts and recipient tracts on four factors: toxicity concentration, 3rd-Grade Math Test Scores, Incarceration Rates, and Average Life Expectancy. After using multivariate linear regression (controlling for household income and % African American) and t-test and rank sums, this study found that displaced households tend to move to areas with a worse quality of life. For example, recipient neighborhoods contain 31% greater toxicity concentration, an 8-percentile increase in the incarceration rate, a 10-percentile reduction in test scores, and a lower median life expectancy by 1 year. While the life-expectancy linear regression generated strange results, this is most likely because household income and gentrification were highly correlated in this situation.

These results should be alarming for both policymakers and residents when deciding housing policies near at-risk neighborhoods. While prior studies have focused on the link between gentrification and displacement, none have conducted comprehensive research on the concrete consequences of displacement itself. Even when analyzing displacement effects, most are limited to one metropolitan area. Thus, this paper contributes to the growing literature base by quantifying the effects of displacement through analyzing the environmental, social, and economic factors on a national scale. It proves how displacement can create genuine harm for the poorest of Americans. Hopefully, this paper will spark greater research into the aspect of displacement to establish a greater picture of the injustice in gentrification.

Future research should focus on collecting national movers data to further collaborate on the results of this paper. Because this private information is not readily available, many assumptions had to be made in this study, including focusing on nearby poor neighborhoods with high population growth rates. More accurate data could further provide greater insight into the complexities of displacement in gentrifying neighborhoods. Moreover, a more comprehensive approach towards analyzing the quality of life would be better. Instead of just analyzing these four features, other research papers could focus on more elaborate connections with displacement. For example, studies could compare changes in environmental pollution, crime, job opportunity, or educational quality over 13 years to find interesting patterns. Overall, this paper only scratches the surface on the issue of displacement, and more research could further establish how gentrification is a prevailing problem in the United States.

Acknowledgements

I would first like to thank my research mentor, Eric Xia. He was instrumental in helping me find data sets, conduct the right statistical tests, and write a quality research paper. I would also like to thank my weekly deliverable group for listening and providing constructive criticism: Ammar, Ally, Eric F, Harshal, John, Rohan, and Shriya. Finally, I would like to thank my parents for giving the incredible opportunity to follow academic pursuits over the summer.

References

[1] Census tract distance database. National Bureau of Economic Research, 2015.

[2] Risk-screening environmental indicators (rsei) model. Environmental Protection Agency, 2018.

[3] U.s. life expectancy at birth by state and census tract - 2010-2015. Center for Disease Control and Prevention, 2020.

[4] Rowland Atkinson. The hidden costs of gentrification: Displacement in central london. In *Journal of Housing and the Built Environment*, 2000.

[5] Rowland Atkinson. Does gentrification help or harm urban neighborhoods? an assessment of the evidence-base in the context of the new urban agenda. In *ESRC Centre for Neighborhood Research*, 2002.

[6] H. Banzhaf and E. McCormick. Moving beyond cleanup: Identifying the crucibles of environmental gentrification,. 2006.

[7] Raj Chetty and Nathaniel Hendren. The effects of neighborhoods on intergenerational mobility i: Childhood exposure effects. In *Quartely Journal of Economics*, 2017.

[8] Raj Chetty, Nathaniel Hendren, Maggie Jones, and Sonya Porter. The opportunity atlas: Mapping the childhood roots of social mobility. In *National Bureau of Economic Research*, 2018.

[9] Seth Chizeck. Gentrification and changes in the stock of low-cost rental housing in philadelphia, 2000 to 2014. In *Cascade Focus*, 2016.

[10] Lei Ding, Jackelyn Hwang, and Eileen Divringi. Gentrification and residential mobility in philadelphia. In *Regional Science and Urban Ecnonomics*, 2016.

[11] Adam Eckerd. Cleaning up without clearing out? a spatial assessment of environmental gentrification. In *Urban Affairs Review*, 2010.

[12] Ingrid Ellen and Katherine O'Regan. How low income neighborhoods change: Entry, exit, and enhancement. In *Regional Science and Urban Ecnonomics*, 2011.

[13] Lance Freeman and Frank Braconi. Gentrification and displacement new york city in the 1990s. In *Journal of the American Planning Association*, 2004.

[14] Chris Hamnet. Gentrification and the middle-class remaking of inner london, 1961-2001. In *Urban Studies*, 2003.

[15] John Logan, Zengwang Xu, and Brian Stults. Interpolating us decennial census tract data from as early as 1970 to 2010: A longitudinal tract database. In *Professional Geographer*, 2018.

[16] M. Lyons. Gentrification, socioeconomic change, and the geography of displacement. In *Journal of Urban Affairs*, 1996.

[17] P. Marcuse. Abandonment, gentrification and displacement: the linkages in new york city. 1986.

[18] Terra McKinnish, Randall Walsh, and Kirk White. Who gentrifies low-income neighborhoods. In *Journal of Urban Economics*, 2010.

[19] Paul Mohai and Bunyan Bryant. Race, poverty, and the environment. In *HeinOnline*, 1992.

[20] Kathe Newman and Elvin Wyly. The right to stay put, revisited: Gentrification and resistance to displacement in new york city. In *Urban Studies*, 2006.

[21] Ashley Qiang, Christopher Timmins, and Wen Wang. Displacement and the consequences of gentrification. Duke University, 2020.

[22] Jason Richardson, Bruce Mitchell, and Juan Franco. Shifting neighborhoods: displacement in american cities. National Community Reinvestment Coalition, 2020.

[23] M. Schill and R. Nathan. Revitalizing america's cities: Neighbourhood reinvestment and displacement. Albany: State University of New York Press, 1983.

[24] Tom Slater. Missing marcuse: On gentrification and displacement. In *Analysis of Urban Change, Theory, Action*, 2009.

[25] H. Sumka. Neighbourhood revitalization and displacement. a review of the evidence. In *Journal of the American Planning Association*, 1979.

[26] Jacob Vigdor, Douglas Massey, and Alice Rivlin. Does gentrification harm the poor? In *Brookings-Wharton Papers on Urban Affairs*, 2002.

[27] Elvin Wyly, Kathe Newman, and Alex Scafran. Displacing new york. In *Environment and Planning A: Economy and Space*, 2010.

[28] Mariam Zuk, Ariel Bierbaum, and Karen Chapple. Gentrification, displacement, and the role of public investment. In *Journal of Planning Literature*, 2017.