Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

TOWARDS ADVERSARIAL ROBUSTNESS WITH 01 LOSS MODELS, AND NOVEL CONVOLUTIONAL NEURAL NET SYSTEMS FOR ULTRASOUND IMAGES

by
Meiyan Xie

This dissertation investigates adversarial robustness with 01 loss models and a novel convolutional neural net systems for vascular ultrasound images.

In the first part, the dissertation presents stochastic coordinate descent for 01 loss and its sensitivity to adversarial attacks. The study here suggests that 01 loss may be more resilient to adversarial attacks than the hinge loss and further work is required.

In the second part, this dissertation proposes sign activation network with a novel gradient-free stochastic coordinate descent algorithm and its ensembling model. The study here finds that the ensembling model gives a high minimum distortion (as measured by HopSkipJump) compared to full precision, binary, and convolutional neural networks, and explains this phenomenon by measuring the transferability between networks in an ensemble.

In the last part, this dissertation tackles three important segmentation problems for vascular ultrasound images with novel convolutional neural networks. More specifically, these three problems are: (1) vessel segmentation in the internal carotid artery, (2) vessel segmentation in the entire carotid system, and (3) vessel and plaque segmentation in the entire carotid system. The study here represents a first successful step towards the automated segmentation of vessel and plaque in carotid artery ultrasound images and is an important step in creating a system that can independently evaluate carotid ultrasounds.

TOWARDS ADVERSARIAL ROBUSTNESS WITH 01 LOSS MODELS, AND NOVEL CONVOLUTIONAL NEURAL NET SYSTEMS FOR ULTRASOUND IMAGES

by
Meiyan Xie

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

Department of Computer Science

August 2021

APPROVAL PAGE

TOWARDS ADVERSARIAL ROBUSTNESS WITH 01 LOSS MODELS, AND NOVEL CONVOLUTIONAL NEURAL NET SYSTEMS FOR ULTRASOUND IMAGES

Meiyan Xie

Dr. Usman W. Roshan, Dissertation Advisor — Date
Associate Professor of Computer Science, New Jersey Institute of Technology

---

Dr. Zhi Wei, Committee Member — Date
Professor of Computer Science, New Jersey Institute of Technology

---

Dr. Iulian Neamtiu, Committee Member — Date
Associate Professor of Computer Science, New Jersey Institute of Technology

---

Dr. Justin Ady, Committee Member — Date
Assistant Professor of Surgery, Rutgers University, New Brunswick, NJ

Dr. William W. Graves, Committee Member — Date
Associate Professor of Psychology, Rutgers University, Newark, NJ

BIOGRAPHICAL SKETCH

Author: Meiyan Xie

Degree: Doctor of Philosophy

Date: August 2021

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 2021
- Bachelor in Applied Mathematics
  Guangdong University of Technology, Guangdong, China, 2013

Major: Computer Sciences

Presentations and Publications:

M. Xie, Y. Li, Y. Xue, L. Huntress, W. Beckerman, S. A. Rahimi, J. W. Ady, and U. W. Roshan, "Vessel lumen segmentation in carotid artery ultrasounds with the U-Net convolutional neural network," *In the Proceedings of the 3rd Workshop in Artificial Intelligence Techniques for BioMedicine and Healthcare, IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2680-2684, 2020.

M. Xie, Y. Li, Y. Xue, L. Huntress, W. Beckerman, S. A. Rahimi, J. W. Ady, and U. W. Roshan, "Two-stage and dual-decoder convolutional U-Net ensembles for reliable vessel and plaque segmentation in carotid ultrasound images," *In the Proceedings of the 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1376-1381, 2020.

M. Xie, Y. Li, Y. Xue, R. Shafritz, S. A. Rahimi, J. W. Ady, U. W. Roshan, "Vessel lumen segmentation in internal carotid artery ultrasounds with deep convolutional neural networks," *In the Proceedings of the 3rd International Workshop on Deep Learning in Bioinformatics, Biomedicine, and Healthcare Informatics (DLB2H), Workshop at IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2393-2398, 2019.

M. Xie, Y. Xue, U. Roshan, "Stochastic coordinate descent for 0/1 loss and its sensitivity to adversarial attacks," *In the Proceedings of the 18th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 299-304, 2019.

M. Xie, U. Roshan, "Exploring classification, clustering, and its limits in a compressed hidden space of a single layer neural network with random weights," *In the Proceedings of the 15th International Work-Conference on Artificial Neural Networks (IWANN)*, pages 507-516, 2019.

Y. Xue, M. Xie, Z. Yang, U. Roshan, "Defending against black-box adversarial attacks with gradient-free trained sign activation neural networks," (Submitted).

A. Aljouie, Y. Xue, M. Xie, and U. Roshan, "Challenges in predicting glioma survival time in multi-modal deep networks," *In the Proceedings of the 3rd Workshop in Artificial Intelligence Techniques for BioMedicine and Healthcare, IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2558-2562, 2020.

Y. Xue, M. Xie, and U. Roshan, "Towards adversarially robust classification with 01 loss neural networks, " *In the Proceedings of the 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1304-1309, 2020.

Y. Xue, M. Xie, and U. Roshan, "On the transferability of adversarial examples between convex and 01 loss models," *In the Proceedings of the 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1460-1467, 2020.

L. Huntress, M. Xie, S. G. Huang, W. Beckerman, S. A. Rahimi, U. Roshan, J. W. Ady, "An Ensemble-Based Confidence Score to Increase Accuracy of Carotid Ultrasound Vessel Lumen Segmentation by Convolutional Neural Networks," *Journal of Vascular Surgery*, Volume 72, Issue 1, E235-E236, 2020.

Y. Xue, M. Xie, F. Farhat, O. Boukrina, A. M. Barrett, J. R. Binder, U. W. Roshan, and W. W. Graves, "A multi-path decoder network for brain tumor segmentation," *The MICCAI Multimodal Brain Tumor Segmentation Challenge (BraTS)*, 2019.

Y. Xue, M. Xie, F. Farhat, O. Boukrina, A. M. Barrett, J. R. Binder, U. W. Roshan, William W. Graves, "A fully 3D multi-path convolutional neural network with feature fusion and feature weighting for automatic lesion identification in brain MRI images," *extended abstract in ML4H: Machine Learning for Health, Workshop at NeurIPS*, 2019.

To My Beloved Parents (Huiquan Xie and Lingji Cai),
Brother (Wenxuan Xie), Sister (Wenting Xie) and
Yunzhu Li.
To My Dearest Zoe, I love you.

ACKNOWLEDGMENT

First and foremost, I would like to express my sincere gratitude to my dissertation advisor Professor Usman Roshan for his invaluable advice, patience, kindness to my study and research. His guidance and encouragement have helped me all the time in doing research and writing this dissertation. The journey of being his Ph.D. student is a truly precious experience and a lifelong benefit.

I am grateful to Dr. Justin Ady for providing very useful feedback and knowledge on vascular ultrasound, without which this dissertation would not be possible. I also thank Dr. Zhi Wei, Dr. Iulian Neamtiu and Dr. William W. Graves for their insightful comments and feedbacks on my research and serving on my dissertation committee.

In addition, I would like to thank the Computer Science Department for offering me teaching assistant support. The financial support was critical for me to continue my PhD program, and I am very grateful for that. I also appreciate the timely support from faculty and staff in the Department of Computer Science when I had questions and challenges.

I would also like to thank my labmates: Yunzhe Xue, Haodi Jiang, Hao Liu for their support and assistance.

Lastly, I would like to thank my family members: Huiquan Xie, Lingji Cai, Wenxuan Xie and Wenting Xie for their support and understanding throughout my life. In particular, I am extremely grateful to Yunzhu Li, the father of our lovely baby, for his love, support and encouragement.

TABLE OF CONTENTS

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Stochastic Coordinate Descent for 01 Loss

The 01 loss while hard to optimize is least sensitive to outliers compared to its continuous differentiable counterparts, namely hinge and logistic loss. In Chapter 2, we present a stochastic coordinate descent (SCD) heuristic for 01 loss based on the original stochastic gradient descent method [10]. We implement and study our heuristic on real datasets from the UCI machine learning archive and find our method to be comparable to the support vector machine in accuracy and tractable in training time. We conjecture that the 01 loss may be harder to attack in a black box setting due to its non-continuity and infinite solution space. We train the linear classifier in a one-vs-one multi-class strategy on CIFAR10 and STL10 image benchmark datasets. In both cases we find the classifier to have the same accuracy as the linear support vector machine but more resilient to black box attacks. On CIFAR10 the linear support vector machine has 0% on adversarial examples while the 01 loss classifier hovers about 10% while on STL10 the linear support vector machine has 0% accuracy whereas 01 loss is at 10%. Our work here suggest that 01 loss may be more resilient to adversarial attacks than the hinge loss and further work is required.

1.2 Boundary Blackbox Attack

While machine learning models today can achieve high accuracies on classification tasks, they can be deceived by minor imperceptible distortions to the data. These are known as adversarial attacks and can be lethal in the black-box setting which does not require knowledge of the target model type or its parameters. Binary neural networks that have sign activation and are trained with gradient descent have been shown to be harder to attack than conventional sigmoid activation networks but their

improvements are marginal. In Chapter 3, we instead train sign activation networks with a novel gradient-free stochastic coordinate descent algorithm and propose an ensemble of such networks as a defense model. In order to explain our model's robustness we show that an adversary targeting a single network in our ensemble fails to attack (and thus non-transferable to) other networks in the ensemble. Thus, a datapoint requires a large distortion to fool the majority of networks in our ensemble and is likely to be detected in advance. This property of non-transferability arises naturally from the non-convexity of sign activation networks and randomization in our gradient-free training algorithm without any adversarial defense effort.

1.3 Convolutional Neural Networks for Vascular Ultrasound Images

Stroke is the 5th leading cause of death in the United States [45]. Annually, it is responsible for billions of dollars in lost income and health care costs. For this reason, there is significant effort and investment in the prevention of stroke. Ischemic strokes account for 87% of all strokes. Narrowing and deposition of plaque in the carotid arteries due to atherosclerosis is the most common cause of ischemic stroke. Carotid ultrasound is a safe, low-cost procedure that is used as a screening test in patients with risk factors for atherosclerosis [64]. It allows physicians to stratify the stroke risk of a patient and identify those patients that will most benefit from medical therapy or surgical intervention.

During a vascular ultrasound, high-frequency sound waves are transmitted into your body. The sound waves are reflected back to the probe when they encounter the boundaries between different tissues in the body. This information is then utilized to create a 2D image of the vessel and surrounding tissue structures. Physicians utilize ultrasound images of the carotid artery in stroke prevention. During their evaluation, physicians must first identify the vessel in the image. They then identify any atherosclerotic plaque within the wall and lumen of the vessel and finally they evaluate

the physiologic impact of those plaques on the flow of blood within the vessel. This is a time intensive and resource intensive process that requires highly skilled technicians and physicians to perform and interpret the results. As physician workload has increased and healthcare systems investigate ways to streamline processes and cut costs, automating the interpretation of vascular ultrasounds has great potential.

Evaluation of a carotid ultrasound requires segmentation of the vessel wall, lumen, and plaque of the carotid artery.

In Chapter 4, we propose and evaluate single and multi-path convolutional U-neural networks for vessel lumen segmentation in internal carotid artery ultrasound images. With a basic simple convolutional U-Net, we obtained a 10-fold cross-validation accuracy of 95%. We also evaluated a dual-path U-Net where we modified the original image and used it as a synthetic modality but we found no improvement in accuracy. We found that the sample size made a considerable difference and thus expect the accuracy to rise with adding more training samples to the model.

In Chapter 5, we explore a U-Net for the entire carotid artery system that includes internal, external, and common carotid arteries. We also explore images containing bifurcations, longitudinal images, and images with ultrasound shadowing, plaque, and gray shading, all of which make vessel segmentation even harder. With the convolutional U-Net, we obtained a 10-fold cross-validation accuracy of 94.3%. We see that the U-Net correctly segments the lumen even in the presence of significant plaque, calcified wall, and ultrasound shadowing, all of which make it difficult to outline the vessel. We also see that the common carotid artery vessels are easiest to segment with a 96.6% cross-validation accuracy whereas internal and external carotid are harder both with 92.7% and 91.9% cross-validation accuracies, respectively.

In Chapter 6 we consider segmentation of both vessel and plaque. In 10-fold cross-validation, all models attain over 90% accuracy for vessel segmentation. With a basic convolutional U-Net, we obtained an accuracy of 66.8% for plaque segmentation.

With the dual-decoder model, we see an improvement to 68.8% whereas the two-stage model falls behind at 65.1% accuracy. However, if we gave the two-stage model the true correct vessel as input its plaque accuracy rises to 81.7% suggesting that the method has potential and needs more work. We ensemble the U-Net and dual decoder U-Net models to obtain confidence scores for segmentations. By considering high confidence outputs above the 60% and 80% thresholds, the accuracy of our dual decoder U-Net rises to 75.2% and 87.3%, respectively.

CHAPTER 2

STOCHASTIC COORDINATE DESCENT FOR 01 LOSS

2.1 Background and Related Work

The problem of determining the hyperplane with minimum number of misclassifications in a binary classification problem is known to be NP-hard [8]. In mainstream machine learning literature, this is called minimizing the 01 loss [60] as given in Objective 2.1,

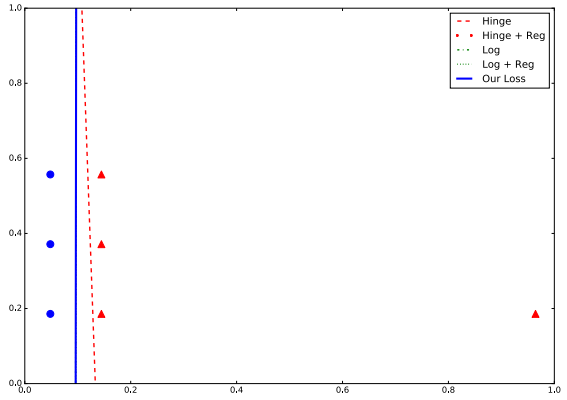
$$\frac{1}{2n} \arg\min_{w,w_0} \sum_i (1 - sign(y_i(w^T x_i + w_0))) \tag{2.1}$$

where $w \in R^d$, $w_0 \in R$ is our hyperplane solution, and $x_i \in R^d, y_i \in \{+1, -1\}. \forall i = 0...n-1$ are our training data. Popular linear classifiers such as the linear support vector machine, perceptron, and logistic regression [4] can be considered as convex approximations to this problem that yield fast gradient descent solutions [7]. However, they are also more sensitive to outliers than the 01 loss [46].

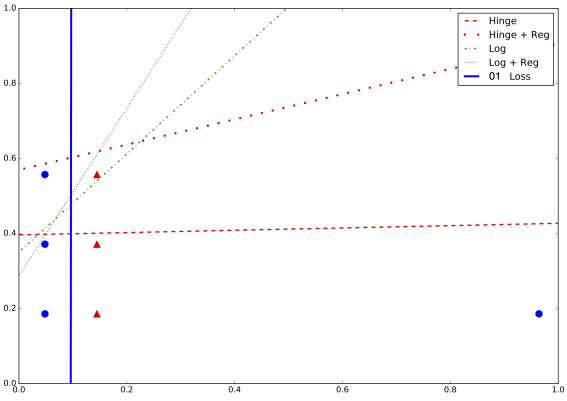
In Figure 2.1, we demonstrate the effect of a single outlier on the hinge, logistic loss, their regularized versions, and 01 loss. In both cases, we intuitively desire a vertical hyperplane that divides (1,1), (1,2), and (1,3) from (3,1), (3,2), and (3,3) since this would likely minimize test error. When the outlier is of the same class as in Figure 2.1(a) all five objectives give similar vertical hyperplanes. The 0/1 loss alone has infinite solutions though and we show a single one here.

When we switch the label of the outlier in Figure 2.1(b), both the hinge and logistic along with their regularized counterparts, give skewed hyperplanes that make several misclassifications on the training data. This is due to the fact that misclassified points increase the hinge and logistic objective (the farther misclassified the point the more the effect); and so in order to lower the objective, the hyperplane is skewed

towards it. The 0/1 loss however is not affected by distances of outliers and still gives a desired hyperplane.



(a)



(b)

Figure 2.1 In (a) we see that an outlier of the same class is not a problem for hinge, logistic, and 0/1 loss objective. However, when we switch its label it affects hinge and logistic considerably while the 0/1 loss decision surface remains the same (although there are an infinite number of solutions we show just one here).

We present here a stochastic coordinate descent (SCD) heuristic for 0/1 loss based on the original stochastic gradient descent method [10]. While the gradient gives the direction of best descent here we perform a heuristic coordinate descent for each stochastic batch. We evaluate our method by comparing it against a cross-validated linear support vector machine (SVM) on real datasets from the UCI machine

learning archive [5]. We find that the cross-validated linear SVM performs slightly better in average and median error but not by a statistically significant margin.

We explore the SCD's sensitivity to a black-box adversarial attack [27, 51], a method that treats the classifier to be attacked as a black box whose model and parameters are unknown. We implement a simple single layer neural network that we use to estimate the black box's gradient and to produce adversarial examples targeting the black box. We perform two separate tests on both SCD 01 loss and the linear SVM on each CIFAR10 and STL10 image object detection benchmarks. On CIFAR10, we find that the linear SVM quickly approaches a 0% accuracy after a few epochs of the black box attack whereas the SCD 01 loss fluctuates and stays above 5%. We see the same on STL10 except there 01 loss stays above 10% accuracy.

While greater exploration is needed, such as a larger neural network as the gradient approximator and experiments on more image benchmarks, we see that the 01 loss has some potential for defending against adversarial attacks. This may be due to its discrete search space and (infinite) non-unique solutions.

2.2 Methods

2.2.1 Coordinate descent

We describe in Algorithm 1 our local search based on coordinate descent. In brief, we start with a random w, make changes to it one coordinate at a time, determine the optimal w_0 for each setting of w, and stop when we reach a local minimum. There are several aspects of our local search worth discussing here.

First, we cycle the coordinates randomly. Since we modify only a single coordinate of w at a time, we can update the projection $w^T x_i$ for all $i = 0..n-1$ in $O(n)$ time — this update is required to determine the optimal w_0 and the objective value. We perform at most ten modifications to a given coordinate (as given by the loop '**for** $j = 1$ to 10 **do**') before considering the next one. This gives all coordinates

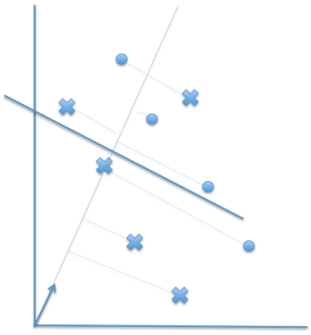
a fair chance before we reach a local minimum. In the same loop, we also update the objective if a better one is found and exit if modifying the coordinate does not improve the objective further. An alternative is to update the objective only after cycling through all the coordinates. However, we find our approach yields a faster search than the alternative while giving similar objective values.

Another aspect of our search is the determination of the optimal w_0. For each setting of w_i (the i^{th} coordinate of w) we determine the optimal value of w_0 by considering all $O(n)$ settings of w_0 between sorted successive projected points $w^T x_k$ and $w^T x_{k+1}$ Since we modify w locally the new projection is similar to the previous (sorted) one and hence insertion sort (that we use for sorting the projection) takes much less than the worst case $O(n^2)$ time.

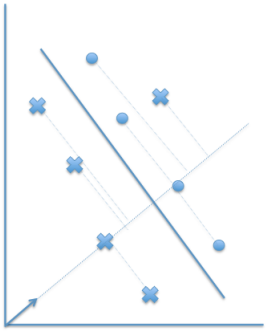
For an initial w it takes $O(n)$ to determine the optimal w_0. After that, as we change w the new w_0 is less likely to be much different than the previous one. And so we don't need to consider all $O(n)$ points again to determine the optimal w_0. Instead, if the initial w_0 was found right after the projected point i then we only consider the range of points starting from $i - 10$ to $i + 10$ in the new projection to determine the new w_0. For a visual illustration, see our toy search problem shown in Figure 2.2.

The w_{inc} parameter corresponds to the learning rate η in gradient descent optimizers. We implement a simple adaptive procedure that considers values of w_{inc} from the set $\{\pm 10^2, \pm 10^4, \pm 10^6\}$ and picks the one with the greatest decrease in our objective Objective 2.1.

To understand why, we have such large learning rates consider the toy example shown below in Figure 2.3. With small and constant step sizes, we would be perpetually stuck in difficult local minima since 01 loss is non-unique and can have infinite solutions.



(a)



(b)

Figure 2.2 Illustration of our coordinate search on a toy example. In (a), we show a hyperplane with an initial random normalized w. The dotted lines show where the projected points would lie on w. The optimal w_0 that minimizes our objective lies just after the fourth projected point. In (b), we increase the x-coordinate of w thus modifying the orientation of the plane (we renormalize w after the orientation). In the new projection, the optimal w_0 is also after the fourth projected point. Thus, we don't need to perform a full $O(n)$ search after modifying w, but instead considering just a few projected points around the previous w_0 is sufficient as a heuristic.

Algorithm 1 Coordinate descent (Continued)

Input: Training data $x_i \in R^d$ for $i = 0..n-1$ with labels $y_i \in \{+1, -1\}$, and $w_{inc} \in R$ (set to 100 by default)

Output: Vector $w \in R^d$ and $w_0 \in R$

Procedure:

1. Let each feature w_i of w be randomly drawn from $[-1, 1]$. Set $\|w\| = 1$. Throughout our search we ensure that $\|w\| = 1$ by renormalizing each time w changes.
2. Compute data projection $w^T x_i, \forall i = 0..n-1$ and determine the optimal w_0. Determining w_0 takes $O(n)$ time because we consider mid points between all projected points $w^T x_i$ and $w^T x_{i+1}$ as potential candidates.
3. Compute value of objective 2.1

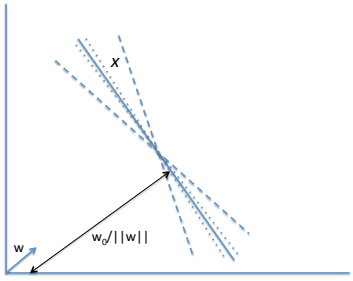


Figure 2.3 The hyperplane in solid line is given by w and w_0 and it misclassifies the point x. The dotted hyperplanes are given by a small step size in the two coordinates of w and insufficient to cross over point x. The dashed hyperplanes are given by a larger step size that is sufficient to cross over x and give a potentially lower 0/1 loss.

2.2.2 Stochastic coordinate descent

There is no guarantee our local search algorithm will return the global solution. The global solution may not even be unique. Once we reach a local minima, we may choose the random restart approach and run the search again. An alternative is rely on random batches of the training data across many iterations of the coordinate descent above so as to better explore the search space. We call this stochastic coordinate descent since this is essentially stochastic gradient descent [10] with the gradient

Algorithm 1 (Continued) Coordinate descent

4. Set $prevobj = \infty$.

while $prevobj - obj > .01$ **do**

Consider a random permutation of the d feature indices.

for $i = 0$ to $d - 1$ **do**

if adding w_{inc} to w_i (the i^{th} component of w) improves the objective **then**

$sign = 1$

else if subtracting w_{inc} from w_i improves the objective **then**

$sign = -1$

else

skip the next loop

end if

for $j = 1$ to 10 **do**

w_i += $sign \times w_{inc}$

Determine the optimal w_0. Since we are making local changes to w the new value of w_0 is likely to be not very far from the previous one. Based on this intuition we avoid expensive $O(n)$ searches and use a constant time heuristic instead.

if $prevobj - obj > .01$ **then**

update the variable obj

Set $prevobj = obj$.

else

Set $j = 10$ to exit this loop

end if

end for

end for

end while

descent replaced by our coordinate descent above. We describe this in detail in Algorithm 2. For a single run of our heuristic, we save the best solution of w, w_0 across all random restarts and use that as the model parameters.

Algorithm 2 Stochastic coordinate descent

Input: Feature vectors $x_i \in R^d$ with labels $y_i \in \{+1, -1\}$, number of random restarts $rr \in N$ (Natural numbers), number of iterations per random restart $it \in N$ (Natural numbers), batch size as a percent of training data $p \in [1, 100]$, and $w_{inc} \in R$ (set to 100 by default)

Output: Total of rr pairs of $bestw \in R^d$,$bestw_0 \in R$ after each random restart

Procedure:

Set $j = 0$

while $j < rr$ **do**

Set $bestw = w, bestw_0 = w_0, bestloss = objective(w, w_0)$

Run our coordinate descent (Algorithm 1) and output local minimum w and w_0

for $i = 0$ to it **do**

Randomly pick p percent of rows as input training data to Algorithm 1 and run it to completion starting with the values of w and w_0 from the previous call to it.

In the next step we calculate objectives on the full input training set

if $objective(w, w_0) < objective(bestw, bestw_0)$ **then**

Set $bestw = w$, $bestw_0 = w_0$, and $bestloss = objective(w, w_0)$

end if

end for

Output $bestw$ and $bestw_0$

Set $j = j + 1$.

end while

We output the best $(bestw, bestw_0)$ pair across all random restarts.

2.2.3 Related work

Our coordinate descent and that of [37] differ in how the coordinates are optimized. In their case the authors project the data onto the current hyperplane (that is initially random), scale each projected value by the inverse of its projection on a random vector r, sort the projected values to determine the value that optimizes the 0/1 loss (call it α), and update the solution w by adding $\alpha \times r$. This is repeated for a fixed number of iterations. In our case, we focus on optimizing objective 2.1: we make an incremental change to a coordinate at a time, project the data onto the hyperplane, determine the optimal threshold w_0 for our objective, and repeat until the objective converges.

2.2.4 Experimental performance study on UCI datasets

In order to evaluate our stochastic coordinate descent (SCD) algorithm, we study it on the below datasets in an experimental performance study. Our purpose here is to demonstrate that our SCD 01 loss works on real data and is comparable to the popular linear SVM in accuracy. While we omit runtimes here, our program is quite fast in practice and finishes in tractable times of seconds and minutes as opposed to hours and days.

Datasets: We obtained 52 datasets from the UCI repository. The datasets include data from different sources such as biological, medical, robotics, and business. Some of the datasets are multi-class and since we are studying only binary classification in this paper we convert them to binary. We label the largest class to be -1 and remaining as +1. We trim down excessively large datasets and ignore instances with missing values across the datasets. Thus, the number of instances in some of our datasets are different from that given in the UCI website `https://archive.ics.uci.edu/ml/`. For example the SUSY dataset originally has 5 million entries but we choose the first

5000 for our study. We provide our cleaned data with labels, splits, and a README file on the website `http://web.njit.edu/~usman/scd01oss`.

Programs compared: We compare our SCD 01 loss with random restarts $rr = 100$, number of iterations per random restart $it = 100$ and $p = 75\%$ against a cross-validated linear SVM. For cross-validating linear SVM we select values of C from the set $\{100, 10, 1, .1, .01, .001, .0001, .00001, .000001\}$.

Experimental platform: We run our experiments on a cluster of computing nodes equipped with Intel Xeon E5-2660v2 2.27GHz processors with one method and dataset exclusively on a processor core.

Train and test splits: For each dataset we create ten random partitions into training and test datasets in the ratio of 90% to 10%. We run all programs on each training dataset and predict the labels in the corresponding test set.

Measure of accuracy: We use the number of misclassifications divided by the number of test datapoints as the measure of error throughout in our study.

2.2.5 Experimental performance study on image benchmarks CIFAR10 and STL10

In order to evaluate the adversarial sensitivity of SCD 01 loss to adversarial attack we expose it to a black box attack method that we describe below in detail.

Black box adversarial attack A black box adversarial attack approximates the model by giving it inputs and recording its outputs. It then uses the predictions as labels to train itself and then use its gradient to produce adversarial examples. We implement a double layer neural network with 200 hidden nodes in each layer and ten

nodes in the output one as the adversarial attacker (B). We fully describe our attack algorithm in Algorithm 3.

Algorithm 3 Shown here is a basic outline of the black box attack method [51]

Input: Model M to be attacked, Adversarial attacker B, Feature vectors $x_i \in R^d$ with labels $y_i \in \{+1, -1\}$, number of epochs $ep \in N$ (Natural numbers)

Procedure:

Set data $D = \{x_i, y_i\}$

for $i = 0$ to ep **do**

Obtain predictions y_i' of D from black box model M

Set adversarial training data A to be D except we replace each y_i with the predicted label y_i'.

Train attacker B with A as input training data

With B's gradient we produce adversarial examples.

For each sample a_i in A create adversary $a_i = a_i + \lambda \nabla f$ where ∇f is the gradient of B and λ is randomly chosen from $[-.1, .1]$.

Add new adversarial samples $\{a_i, y_i\}$ to D

This effectively doubles the number of adversarial samples after each iteration. If we instead select 100 random samples from A then we increase the adversarial size by 100 as opposed to doubling.

end for

Datasets: We study adversarial attacks on two image benchmarks:

- CIFAR10[33]: Object recognition from ten classes in 32×32 color images, training size of 50,000, test size of 10,000
- STL10 [19]: Object recognition from ten classes in 96×96 color images, training size of 5000, and test size of 8000

Programs compared:

- Multi-class SCD 01 loss: We implement a one-vs-one [4] multi-class classification method on top of our SCD 01 loss. In the SCD 01 loss we use the same parameters as above.
- Multi-class linear SVM: We implement one-vs-all on top of the linear SVM with C=1.

Experimental platform: We run our experiments on a cluster of computing nodes equipped with Intel Xeon E5-2660v2 2.27GHz processors with one method and dataset exclusively on a processor core.

Train and test splits: For the image benchmarks both train and test datasets are provided in advance.

Measure of accuracy: We use the number of misclassifications divided by the number of test datapoints as the measure of error throughout in our study.

2.3 Results

2.3.1 Classification on UCI datasets

In Table 2.1, we see the average and median error of SCD 01 loss and the cross-validated linear SVM. We see the linear SVM is better in both mean and median but the difference between them is not statistically significant. According to a simple t-test, the p-values between their errors across the 52 datasets is 0.53 which is far from significant. This is more evident when we see their errors across all 52 datasets in Table 2.2.

2.3.2 Black box adversarial attacks

CIFAR10 We generate adversarial samples on the CIFAR10 dataset and study them on both the multi-class SCD 01 loss and multi-class linear SVM. We also study

Table 2.1 Mean and Median Error of SCD 01 Loss and Linear SVM

	SCD 01 loss	***Linear SVM***
Mean	13.5	13.2
Median	11.8	10.3

a bootstrapped version of both: instead of using a single classifier we take the majority vote output of each classifier on ten bootstraps. Bootstrapping is a simple powerful method to boost model accuracy [4, 11]. We study it here to see if bootstrapping can strengthen defense capabilities of SCD 01 loss.

In Figure 2.4, we see the accuracy of CIFAR10 adversarial samples in both SCD 01 loss and linear SVM as the number of epochs progresses. We see both methods start losing accuracy as the black box method progresses, but interestingly we find SCD 01 loss to be relatively less sensitive. Both methods have about 40% accuracy on CIFAR10 and after epoch 10 both linear SVM and its bootstrapped version are at 0% accuracy. The SCD 01 loss also loses accuracy but the bootstrapped one demonstrates a greater defense.

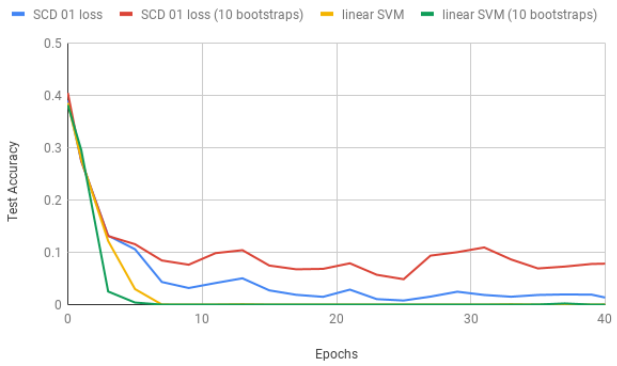


Figure 2.4 CIFAR10 black box attack described in Algorithm 3. We double the number of adversarial samples per epoch.

Table 2.2 Mean and Median Error of SCD 01 Loss and Linear SVM (Continued)

	Rows (columns)	***SCD 01 loss***	***Linear SVM***
Antivirus	373(531)	1.54	1.03
Bach Choral	5665(148)	3.13	2.66
Banknote	1371(4)	0.72	1.16
Breast-cancer	569(30)	3.1	3.28
Chronic kidney	400(30)	0.75	2.5
Climate	540(18)	5.45	4.73
CNAE9*	1080(856)	2.69	1.85
Default credit card	10000(23)	19.19	20.7
Diabetic retinopathy	1150(19)	25	25.1
EEG eye state	10000(14)	30.9	44.4
Fertility	99(9)	21.82	18.18
Forest*	522(27)	12.08	10.94
Gas-sensor*	6953(128)	1.22	0.45
Gesture*	1743(32)	11.53	11.37
Grammatical facial	4225(300)	9.65	9.76
Heart	267(44)	20.36	23.93
Hepmass	10000(28)	16.34	16.15
Hill-valley	606(100)	6.45	8.55
Indian-liver-patient	579(10)	26.61	28.47
Insurance	5822(85)	6.28	6.00
Ionosphere	351(34)	15.28	13.06
Isolet*	1559(617)	1.67	0.83
Libras*	360(90)	2.43	1.35

Table 2.2 (Continued) Mean and Median Error of SCD 01 Loss and Linear SVM

	Rows (columns)	***SCD 01 loss***	***Linear SVM***
LSVT*	126(310)	22.14	10.00
MFEAT*	2000(649)	0.6	0.25
Mhealth	10000(23)	20.55	22.05
Micromass	931(1300)	8.19	4.57
Musk	476(166)	21.25	13.54
Occupancy	10000(5)	1.42	1.43
Online news popularity	10000(59)	36.26	35.7
Ozone	1847(72)	6.92	6.65
Parkinsons	195(22)	13.5	14.5
Parkinson-speech	1040(26)	37.6	38.56
Phishing websites	2455(30)	5.63	5.38
Planning relax	182(12)	36.84	31.58
Qsar	1055(41)	13.77	14.72
Secom	1567(590)	7.72	7.72
Seeds	210(7)	8.57	7.62
Seismic	2538(18)	6.98	6.56
Smartphone*	7352(561)	0.04	0.05
Sonar	208(60)	22.27	22.27
Spambase	4601(57)	6.29	6.96
Steel-faults*	1941(27)	26.4	26.3
Student alcohol	649(30)	26.06	26.97
SUSY	5000(18)	21.38	21.74

Table 2.2 (Continued) Mean and Median Error of SCD 01 Loss and Linear SVM

	Rows (columns)	***SCD 01 loss***	***Linear SVM***
Theorem-proving*	6118(51)	27.44	27.7
Thoraric	470(16)	17.02	14.89
TV news channel	10000(122)	12.47	10.56
Urban-land	675(147)	9.28	8.55
Vertebral	310(6)	16.77	16.77
Wall-follow*	5455(24)	21.54	24.49
Wilt	4339(5)	1.13	1.31
Average		13.47	13.19

In Figure 2.5, we see the adversarial attack with 100 new adversaries per epoch. Here too the SCD 01 loss bootstrapped version shows a greater resilience than the linear SVM.

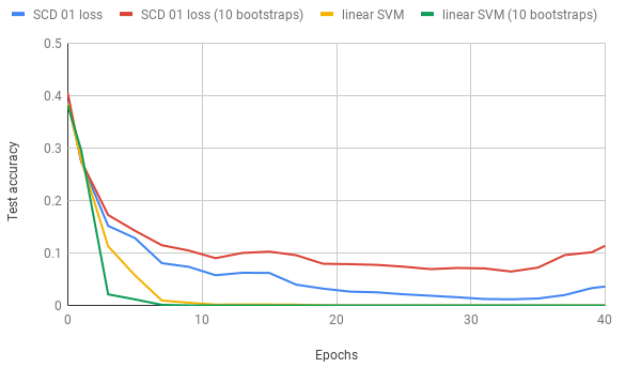


Figure 2.5 CIFAR10 black box attack: we generate a 100 new adversarial samples per epoch.

STL10 We generate adversarial samples on the STL10 dataset and study their accuracy on SCD 01 loss and linear SVM. In Figure 2.6, we see that the linear SVM reaches 0% accuracy after the fifth epoch while SCD 01 loss remains above 10% at

that epoch. However, by epoch 15 SCD 01 loss is close to 0% accuracy but the bootstrapped version is still at 10%.

Figure 2.6 STL10 black box attack: we double the number of adversarial samples per epoch.

Since these images are larger, all components of the black box are slower and thus we have fewer epochs if we want to double the adversarial sample size. Thus we study accuracy on adversaries if we increase them by 100 after each epoch. In this way we can run the attack for more epochs to see if the accuracy becomes 0 for SCD 01 loss at some point.

In this setting, we see that the bootstrapped SCD 01 loss remains well above 10% accuracy. Thus, for STL10 we need to double adversarial samples after each epoch if we want to bring down bootstrapped SCD 01 loss, but this requires more time and computation.

If we increase the number of adversarial samples by only 50 after each epoch, we see an even smaller effect on SCD 01 loss. In fact in this setting even the linear SVM does not reach 0% and stays above 5% accuracy. The bootstrapped SCD 01 loss, however, is above 20% accuracy here.

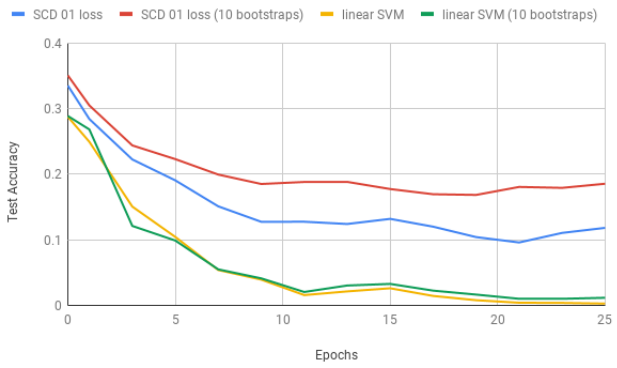


Figure 2.7 STL10 black box attack: we generate a 100 new adversarial samples per epoch.

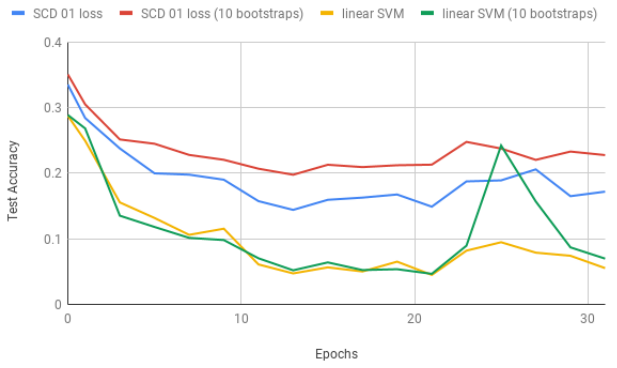


Figure 2.8 STL10 black box attack: we generate 50 new adversarial samples per epoch.

2.4 Conclusion

We present a stochastic coordinate descent heuristic that performs comparably to a trained cross-validated linear support vector machine but demonstrates greater defense against a black box adversarial attack on two image benchmarks. We conjecture this may be due to 01 loss's non-unique solutions and discrete loss and further work is required.

CHAPTER 3

BOUNDARY BLACKBOX ATTACK

3.1 Background and Related Work

State of the art machine learning algorithms can achieve high accuracies in classification tasks but misclassify minor perturbations in the data known as adversarial attacks [27, 52, 35, 13, 12]. Adversarial examples have been shown to transfer across models which makes it possible to perform transfer-based (substitute model) black box attacks [50]. To counter adversarial attacks many defense methods have been proposed with adversarial training being the most popular [66, 70]. However, this tends to lower accuracy on clean test data that has no perturbations [57, 80] and can still be attacked with better transfer based methods [73, 74, 21]. Many previously proposed defenses have also been shown to be vulnerable [13, 6, 25] thus leaving adversarial robustness an open problem in machine learning.

A more lethal and practical attack than substitute model training is a boundary based one that requires only the prediction of the model [12]. These attacks are aimed at finding the minimum distortion to an image such that it will fool a classifier. This is in fact an NP-hard problem for ReLu activated neural networks [32, 61] and tree ensemble classifiers [30]. Even approximating the minimum distortion for ReLu activated neural networks is NP-hard [72]. Boundary based black box attacks such as HopSkipJump [15], Boundary Attack [12] and RayS [14] give an upper bound on the minimum adversarial distortion.

Binary neural networks that have sign activation and binary weights were originally proposed as lightweight models. These are trained with gradient descent by approximating the sign activation. Recent work has shown that they tend to be more

adversarially robust than full precision networks but the improvements are marginal (see Tables 4 and 5 in [23] and Table 8 in [48]).

In this research, we propose a gradient free stochastic coordinate descent algorithm for training sign activation networks with and without binary weights similar to recent work [79, 78, 76]. While our original intention was to study the accuracy of a sign activation network trained directly without any approximation we make an interesting finding on the adversarial robustness of our model. We find that ensembling our model gives a high minimum distortion (as measured by HopSkipJump) compared to full precision, binary, and convolutional neural networks. We explain this phenomena by measuring the transferability between networks in an ensemble.

In summary, we make the following observations in our paper:

- Our single hidden layer sign activation network has higher minimum distortion than ensembles of full precision and binary neural networks, than random forests that have the advantage of bootstrapping and random feature selection, and than ensembles of convolutional networks that have the advantage of convolutions and several layers.
- Our model's robustness stems from non-transferability of adversarial examples between networks in our ensemble and its robustness increases as we add more networks to the ensemble.

3.2 Methods

3.2.1 Gradient-free stochastic coordinate decent

Suppose we are given binary class data $x_i \in R^d$ and $y_i \in \{-1, +1\}$ for $i = 0...n-1$. Consider the objective function of a single hidden layer neural network with sign activation and 01 loss given below. We employ a stochastic coordinate descent algorithm shown in Algorithm 4 (similar to recent work [79, 78, 76]) to minimize this objective.

$$\frac{1}{2n} \underset{W,W_0,w,w_0}{\arg\min} \sum_i (1 - sign(y_i(w^T(sign(W^T x_i + W_0)) + w_0))) \tag{3.1}$$

Algorithm 4 Stochastic Coordinate Descent for Single Hidden Layer Network

Procedure:

1. Initialize all network weights W, w to random values from the Normal distribution $N(0, 1)$.

2. Set network thresholds W_0 to the median projection value on their corresponding weight vectors and w_0 to the projection value that minimizes our network objective.

while $i < epochs$ **do**

1. Randomly sample a batch of data equally from each class. (We set this to 75% of the training data in image and text data experiments and 25% in the ECG data.)

2. Perform coordinate descent separately first on the final node w and then a randomly selected hidden node u (a random column from the hidden layer weight matrix W)

3. Suppose we are performing coordinate descent on node w. We select a random set of features (coordinates) from w called F. For each feature $w_i \in F$ we add/subtract a learning rate η and then determine the w_0 that optimizes the loss (done in parallel on a GPU). We consider all possible values of $w_0 = \frac{w^T x_i + w^T x_{i+1}}{2}$ for $i = 0...n-2$ and select the one that minimizes the loss (also performed in parallel on a GPU).

4. After making the update above we evaluate the loss on the full dataset (performed on a GPU for parallel speedups) and accept the change if it improves the loss.

end while

We can train sign activation networks with and without binary weights using our SCD training procedure above. In the case of binary weights we do not need a learning rate.

3.2.2 Implementation, test accuracy, and runtime

We implement our training procedure in Python, numpy, and Pytorch [54] and make our code freely available from `https://github.com/zero-one-loss/scd_github`. We train three types of sign activation networks with our algorithm: (1) SCD01: 01-loss in the final node, (2) SCDCE: cross-entropy loss in the final node, and (3) SCDCEBNN: cross-entropy in the final node with binary weights throughout the model. Since sign activation is non-convex and our training starts from a different random initialization, we run it 100 times and output the majority vote.

To illustrate our real runtimes and clean test accuracies, we compare our models with a single hidden layer of 20 nodes to the equivalent network with sigmoid activation and logistic loss (denoted as MLP) and the binary neural network (denoted as BNN) [29]. We used the MLPClassifier in scikit-learn [55] to implement MLP and the Larq library [24] with the *approx* approximation to the sign activation. This has shown to achieve a higher test accuracy than the original straight through estimator (STE) of the sign activation [41].

We perform 1000 iterations of SCD01 and SCDCE and 10000 of SCDCEBNN. In Table 3.1, we show the runtimes of a single run of all models on CIFAR10 [33] ($32 \times 32 \times 3$, 10K train, 2K test), CelebA facial attributes black hair vs brown hair [40] ($96 \times 96 \times 3$, 1K train, 1K test), GTSRB street sign recognition 60 vs 120 speed limit signs [63] ($48 \times 48 \times 3$, 2816 train, 900 test), and ImageNet class 0 vs. 1 [59] ($256 \times 256 \times 3$, 2580 train, 100 test). Our training runtimes are comparable to gradient descent in MLP and BNN and thus practically usable. We can trivially parallelize training an ensemble by doing multiple runs on CPU and GPU cores at the same

time. We also show test accuracies of 100 vote ensembles of all models and find our model accuracies to be comparable to MLP and BNN.

Table 3.1 Training Runtimes of Single Run in Seconds and Test Accuracies of 100 Vote Ensembles in Parenthesis for Binary Classification

	SCD01	**SCDCE**	**SCDCEBNN**	**MLP**	**BNN**
CIFAR10	64 (87%)	56 (88%)	422 (87%)	13 (90%)	106 (83%)
CelebA	20 (79%)	18 (81%)	111 (72%)	41 (78%)	32 (76%)
GTSRB	22 (97%)	22 (97%)	92 (98%)	8 (99%)	42 (96%)
ImageNet	77 (72%)	54 (73%)	338 (71%)	115 (72%)	78 (66%)

3.3 Results

Going forward, we compare the adversarial robustness of ensembles of our three models SCD01, SCDCE, and SCDCEBNN, their full precision and binary gradient descent trained equivalent counterparts MLP and BNN, two convolutional neural networks: LeNet [36] and ResNet50 [28], and random forests [11] (denoted as RF). For each model, we use the majority vote output of 100 votes each with different initial parameters except for ResNet50 where we use ten votes. In random forest, we use an ensemble of 100 trees.

We use a single hidden layer of 20 nodes in our three models and in MLP and BNN throughout the paper. The convolutional networks and random forest are not a fair comparison to our model since it has fewer parameters and does not perform bootstrapping or random feature selection as random forest. We include them nevertheless since convolutional neural networks serve as state of the art references and random forest serves as an alternative ensemble method.

3.3.1 Adversarial distortion on image data

The minimum distortion required to make a datapoint adversarial is an indicator of a model's adversarial and even corruption/general robustness [26]. We consider ten randomly selected datapoints from the CIFAR10 benchmark [33] and report their minimal adversarial distortion as given by HopSkipJump [15], Boundary Attack [12] and RayS [14].

We use the HopSkipJump and Boundary Attack implementation in the IBM Adversarial Robustness Toolkit (ART) [47] In order to obtain as accurate an estimate as possible, we run both methods ten times each with an initial pool size of 1000 random datapoints and maximum iterations of 100 and report the minimum value. For a single datapoint, this typically takes several hours to finish and thus we are able to report the distortion of only ten random points in this study. We use the RayS implementation from their GitHub site `https://github.com/uclaml/RayS` and run it with default parameters of 40,000 queries to obtain a distortion estimate.

In Table 3.2 first row, we show the clean test accuracy of all models on CIFAR10 class 0 vs. 1. The convolutional networks LeNet and ResNet50 have higher accuracies since they have the advantage of convolutions. In the following four rows of Table 3.2, we see the minimum adversarial distortion of models as estimated by four boundary attack methods. We were unable to attack some models with HopSkipJump 500 due to time constraints and mark them as NA. We see that HopSkipJump gives the lowest distortion for each model except for SCD01 and SCDCE where it is comparable to RayS.

Amongst HopSkipJump distortions, our sign activation trained models have the highest adversarial distortion with the binary weights cross-entropy variant as the winner. All other neural networks lag far behind and have distortion even lower than random forest. Even though BNN also has sign activations, its distortions are similar to MLP possibly due to its approximation of the sign activation and gradient

descent search. If we use the the straight through estimator and swish approximations [20], the distortions remain similar to what we report here.

Table 3.2 Mean Minimum L_2 Distortion of Ten Random Test Images from CIFAR10 Class 0 vs. 1 as Estimated by Four Different Boundary Attack Methods. Highest Distortion by HopSkipJump Shown in Bold

	SCD01	**SCDCE**	**SCDCEBNN**	**MLP**	**BNN**	**ResNet50**	**LeNet**	**RF**
Clean acc	87	88	88	90	83	98	96	88
HSJ	3.2	3.36	**3.6**	0.77	0.76	0.76	1.73	1.91
HSJ500	1.47	1.36	1.8					
Boundary	7.69	8.23	8.81	2.47	2.94	3.44	7.29	6.63
RayS	3.14	3.08	3.57	0.99	1.03	0.92*	2.54	6.77

To further validate the distortions above, we run HopSkipJump with ten maximum iterations on the first 100 CIFAR10 test datapoints. We used a fixed image as the initial one in these experiments. In Table 3.3, we see that SCDCEBNN distortions are the highest and the relative ranking is the same as we saw for the ten images above with 100 maximum iterations of HopSkipJump.

Table 3.3 Mean Minimum L_2 Distortion of First 100 Test Images from CIFAR10 Class 0 vs. 1 as Estimated by HopSkipJump with Ten Maximum Iterations Starting from Random Initial Images. Highest Distortion in Bold

	SCD01	**SCDCE**	**SCDCEBNN**	**MLP**	**BNN**	**ResNet50**	**LeNet**	**RF**
Clean acc	87	88	88	90	83	98	96	88
HSJ	7.35	7.45	**7.69**	1.3	NA	2.79	5.8	5.44

In Table 3.4, below we show HopSkipJump distortions (min of 10 runs 100 max iterations each) on a single random image from CelebA, GTSRB, and ImageNet datasets. We find our SCD models to have a higher distortion on both CelebA and GTSRB but comparable to MLP on ImageNet.

Table 3.4 Minimum L_2 Adversarial Distortion of A Single Random Test Image from CelebA, GTSRB, and ImageNet Class 0 vs. 1. In Bold Are The Largest Distortion Values for Each Dataset

			Celeba					
	SCD01	SCDCE	SCDCEBNN	MLP	BNN	ResNet50	LeNet	RF
Image 0	8.77	8.6	**14.13**	1.02	.22	1.68	3.3	2.82
			GTSRB					
	SCD01	SCDCE	SCDCEBNN	MLP	BNN	LeNet	RF	
Image 0	.6	.82	**1.24**	.62	.87	.33	.01	
			ImageNet					
	SCD01	SCDCE	SCDCEBNN	MLP	BNN	ResNet50	RF	
Image 0	20.9	16.17	3.26	**24.1**	5.68	2.01	5.78	

3.3.2 Transferability within ensembles and effect of ensemble size

To understand the above phenomena, we estimate the probability that an adversarial example targeting a single model in the ensemble will also be adversarial to other models in the ensemble. We can estimate this by first performing a HopSkipJump attack on each model in the ensemble separately. Let x_i' be the adversary obtained by targeting model m_i in the ensemble. Let k_i be the number of models in the ensemble that are also misclassified by the adversary x_i' (thus transferable). We sum k_i for $i = 0...n-1$ and divide by 9900 which is the maximum value of this sum (obtained when the adversary attacks all models in the ensemble excluding the target of course).

We average this probability for Images 0 through 7 for each method. In Table 3.5, we see that this probability is lowest for our models and highest for MLP and BNN. The fact that this probability is very low for our models indicates that for several of the networks in our ensemble the adversary targeting a fixed network does not transfer to other ones. The low transferability of our models indicates that a greater distortion is required for an image to be adversarial.

Table 3.5 Estimated Probability That An Adversarial Image Targeting A Single Model in The Ensemble (of 100 Models) Will Transfer to Other Models. Lowest Probability in Bold

	SCD01	**SCDCE**	**SCDCEBNN**	**MLP**	**BNN**	**ResNet50**	**LeNet**	**RF**
Prob	.006	.004	**.002**	.39	.2	.02	.01	.07

In fact, as we see in Figure 3.1, the robustness of our models increases as we increase the ensemble size to a much larger degree than ensembles of MLP and BNN, and than RF. We use ensemble sizes of 100 in this study but the figure suggests that increasing our ensemble size is likely to further increase robustness.

3.3.3 Discussion

Using ensembles of neural networks and promoting diverse ensembles has been previously proposed as a defense against adversarial attacks. Studies using ensembles with different initializations (like we do), bootstrapping, and Gaussian noise have shown robustness but only in the white box setting [65] (which is somewhat unrealistic since it assumes the attacker has full knowledge of the model and its parameters). Other studies combine the loss of all models in the classifier and add a regularizer that promotes diversity.

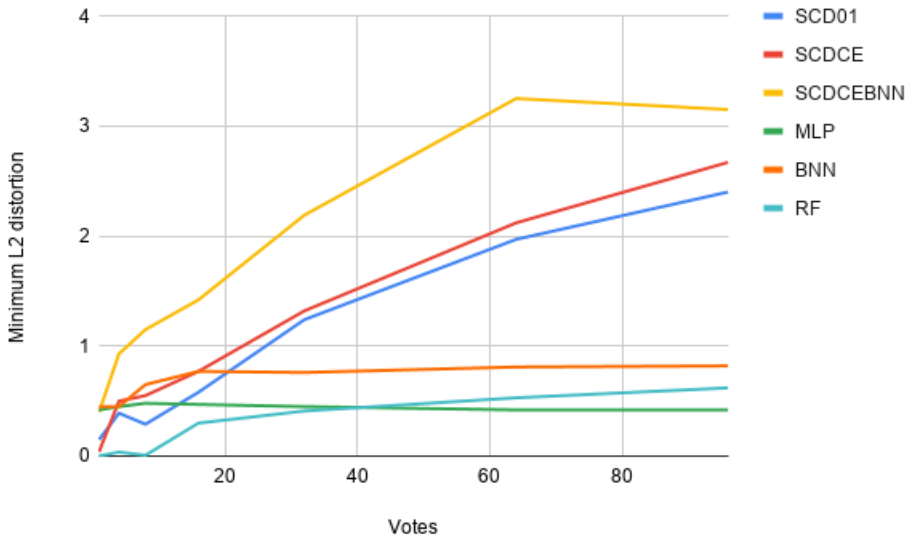


Figure 3.1 Minimum L_2 image distortion as a function of ensemble size.

For example, we could try to maximize the angle between gradients of models in the ensemble [31] to make them *misaligned*. In their diversity training, they use a Gaussian noise augmented dataset which raises concerns about the effectiveness of their method since augmentation alone has been shown to be effective in ensemble training [65]. Another study maximizes diversity between classes [49] and thus does not apply to our work here that focuses on binary classes only. Even for multiple classes their method is computationally expensive as it uses a joint loss function. Other methods inject noise to models in the ensemble [39]; but their evaluation is only in the white box setting. Various measures for ensemble diversity have been previously proposed for deep networks [38] and evaluated in the white-box setting.

We can apply all of the above diversity training methods to our ensemble of sign networks. Our work, however, is not explicitly aimed at enhancing diversity. As we show it is naturally diverse; and we conjecture this is due to the non-convexity of sign activation and our randomized training method. Even sigmoid activation networks have a non-convex search space but we can imagine that sign activation gives a greater

degree of freedom. This can easily be seen in the case of a linear classifier with logistic or hinge loss vs. 01 loss [78].

It is hard to make a general claim of robustness with only 100 images from CIFAR10. We would need to show more images from CIFAR10 and other image benchmarks as well but our preliminary experiments on CelebA, GTSRB and ImageNet (shown in Table 3.4) suggest higher distortion on other image data as well.

3.3.4 Conclusion

We show that our ensemble of gradient-free sign activation networks are harder to attack than ensembles of several other networks and random forests on images.

CHAPTER 4

VESSEL SEGMENTATION IN INTERNAL CAROTID ARTERY ULTRASOUNDS IMAGES

4.1 Background and Related Work

Carotid ultrasound is a screening modality used by physicians to direct treatment in the prevention of ischemic stroke in high-risk patients. It is a time intensive process that requires highly trained technicians and physicians. Evaluation of a carotid ultrasound requires identification of the vessel wall, lumen, and plaque of the carotid artery. Prior work in automated approaches to evaluating carotid ultrasounds is highly limited and does not use modern deep learning methods. Previously vessel identification in carotid ultrasounds with preprocessing and marker-controlled watershed transform has been explored [68]. Deep learning solutions have been proposed for vessel segmentation in liver ultrasounds [44] and for vessel detection in femoral regions [62]. In the latter study, authors also evaluate their method on carotid ultrasound images from two individuals, however, their target is detection as opposed to segmentation that we seek. DeepVesselNet [69] is another deep learning model designed for vessel detection but in 3D angiographic volumes. A patch-based deep learning solution has also been proposed for 3D ultrasounds [82]. None of these are end-to-end systems that are simple to train and implement and none address vessel lumen segmentation in internal carotid ultrasounds images that we seek here.

We present here a study to evaluate using a basic convolutional U-network to accurately identify the vessel lumen of internal carotid artery in vascular ultrasound images. Our network is a simple end-to-end solution and addresses for the first time the problem of vessel lumen segmentation in internal carotid artery 2D ultrasound images. These ultrasound images are more affordable and common than 3D ones.

Such ultrasound images are broadly used in vascular diagnosis and thus our solution has a broader impact than previous work.

4.2 Methods

4.2.1 Data collection

We obtained IRB approval from Robert Wood Johnson Medical School to use de-identified images from the Department of Vascular Surgery for this research. We obtained the carotid ultrasound study of 98 patients. We utilized an automated script to crop all patient identifiers from the ultrasound images and manually verified this de-identification. For this study, we focused exclusively on the left and right internal carotid artery ultrasound images.

We cropped each image to obtain just the ultrasound removing all text and annotations on the image. Each images was resized to 224×224 pixels. This gave us a total of 302 images that we then manually segmented. Using RectLabel software (`https://rectlabel.com/`), we manually segmented the vessel lumen for each image to serve as ground truth for training and validation.

4.2.2 Convolutional neural networks

Convolutional neural networks are the current state of the art in machine learning for image recognition [36, 34], including for MRI [9]. They are typically composed of alternating layers for convolution and pooling, followed by a final flattened layer. A convolution layer is specified by a filter size and the number of filters in the layer. Briefly, the convolution layer performs a moving dot product against pixels given by a fixed filter of size $k \times k$ (usually 3×3 or 5×5). The dot product is made non-linear by passing the output to an activation function such as a sigmoid or rectified linear unit (also called relu or hinge) function. Both are differentiable and thus fit into the standard gradient descent framework for optimizing neural networks during training.

The output of applying a $k \times k$ convolution against a $p \times p$ image is an image of size $(p-k+1) \times (p-k+1)$. In a CNN, the convolution layers just described are typically alternated with pooling layers. The pooling layers serve to reduce dimensionality, making it easier to train the network.

4.2.3 Convolutional U-network

After applying a series of convolutional filters, the final layer dimension is usually much smaller than that of the input images. For the current problem of determining whether a given pixel in the input image is part of a vessel, the output must be of the same dimension as the input. This dimensionality problem was initially solved by taking each pixel in the input image and a localized region around it as input to a convolutional neural network instead of the entire image [18].

A more powerful recent solution is the Convolutional U-Net (U-Net) [58]. This has two main features that separate it from traditional CNNs: (a) deconvolution (upsampling) layers to increase image dimensionality, and (b) connections between convolution and deconvolution layers.

4.2.4 Basic U-Net for vessel segmentation

We implemented a basic U-Net [58] in the Pytorch library [53] as shown in Figure 4.1. The U-Net is a popular choice for medical artificial intelligence work and has proven to be a successful baseline that can be built upon. The input to the model is an ultrasound image and output is an image of the same dimensions with 0 and 1 pixel values indicating background and vessel lumen.

Roughly speaking, in our model we first extract features with a series of convolutional kernels and then apply transpose convolutions to increase the dimensionality of the image up to the original. Thus, we have an end-to-end network that is much

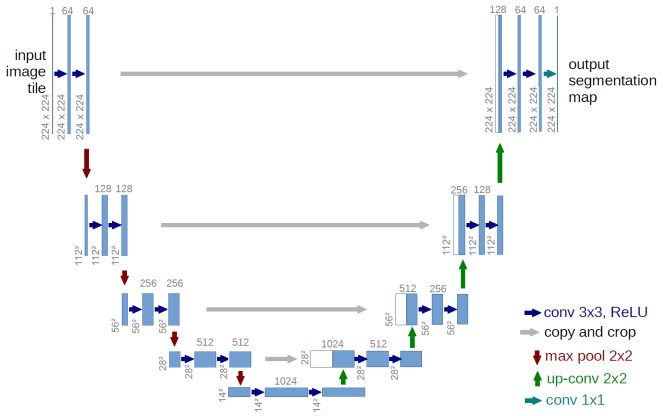


Figure 4.1 Basic U-Net architecture [58] that we use in our preliminary work. Shown here are dimensions of our images in each layer and the number of convolutional and transposed convolutions per layer.

simpler to train than otherwise patch-based approaches that have previously been used for segmentation.

Inspired by our success in synthetic modalities for brain MRI systems [77] we utilized a dual-encoder model as well. We modified the original input ultrasound image by flipping it along the y-axis (called flip) and modifying the brightness and contrast separately. These modified images were then used the as a second modality. In Figure 4.2, we see our dual-path encoder model that has a feature fusion for combining features from the two encoders.

Our feature fusion is a simple concatenation of features from the two encoders. In order to maintain the correct dimensions for the decoder, we reduced each downsampling layer's output channels by half. In this way, the concatenation restores the original dimensionality that is required by the decoder layer.

4.2.5 Dice loss

The final output from our network is a two-dimensional (2D) predicted image of dimensions 224×224. We convert each pixel value into probabilities with softmax [4] and call the resulting image p. The target ground truth r is also of the same

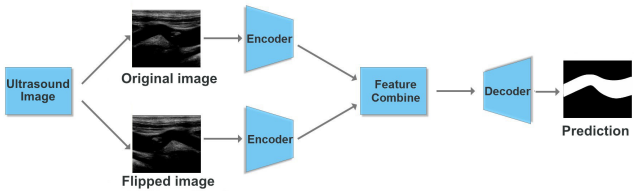


Figure 4.2 Dual-encoder network that treats a modification of the input image as a synthetic modality.

dimensions as p and contains a 1 if the pixel is within the vessel lumen and 0 otherwise. We then use the Dice loss to train our model. This is defined to be $1 - D$ where

$$D(p) = \frac{2\sum_i p_i r_i}{\sum_i p_i^2 + \sum_i r_i^2}$$

and p_i and r_i are the i^{th} pixel values of p and r respectively.

4.2.6 Model implementation and training

We implemented our system using Pytorch [53] and ran it on NVIDIA Pascal P100 and NVIDIA Titan RTX GPUs. We trained our model with 20 epochs of stochastic gradient descent [10], a learning rate of 0.03, decay step of 15, and a batch size of 1. We did not perform any normalization on the input images.

4.2.7 Post processing

We applied a simple post processing procedure to reduce potential false positives. In the final predicted segmentation, we remove all disconnected components except for the largest one that is meant to be the vessel lumen. We found that this improved accuracy by a moderate margin.

4.2.8 Measure of accuracy: Dice coefficient

The Dice coefficient is typically used to measure the accuracy of predicted segmentations in medical images [84]. We convert the output image of our network into a binary mask by setting each pixel value to 1 if its softmax output is at least 0.5 and

0 otherwise. Thus, we use 0.5 as the probability threshold that a pixel value is part of the vessel lumen or outside it. Starting with the human binary mask as ground truth, each predicted pixel is determined to be either a true positive (TP, also one in true mask), false positive (FP, predicted as one but zero in the true mask), or false negative (FN, predicted as zero but one in the true mask). The Dice coefficient is formally defined as

$$DICE = \frac{2TP}{2TP + FP + FN} \tag{4.1}$$

4.2.9 10-fold cross-validation

We performed 10-fold cross-validation experiments on our data. We randomly split our dataset into ten equal parts and selected one part for validation while the remaining nine parts were used to train the model. We then rotated the validation part across the other nine parts giving us a total of ten pairs of training validation splits. We trained the model on each split and reported the average validation and training accuracy below.

4.3 Result

4.3.1 Cross-validation on all images

In Table 4.1, we show the average 10-fold Dice values of vessel lumen segmentations separately for both training and validation samples. The training Dice is typically higher than validation as we see below and the validation is not far behind which suggests that our model is generalizing. We ran the 10-fold three times on our model to check for stability and found that each time our model gives a high training and validation accuracy.

In Figure 4.3, we see ultrasound images with their ground truth and predicted segmentations In (a) we see a vessel with no plaque or calcified walls. In (b) we see

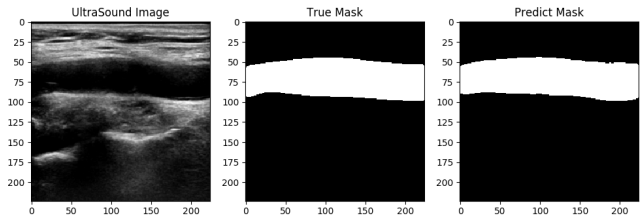
Table 4.1 Average Dice Coefficients of Train and Validation Splits in Our 10-fold Cross-validation Across Three Different Runs of Our Model

Model train run	Vessel train	Vessel validation
Sample size of 234 images		
1	97.95%	93.96%
2	98.00%	93.73%
3	97.99%	93.67%
Sample size of all 302 images		
1	97.83%	94.63%

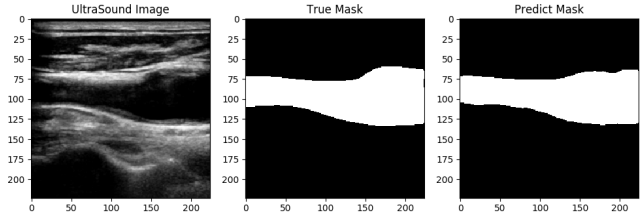
a vessel with thick calcified walls, in (c) we have a vessel with plaque, and in (d) we have a vessel with plaque and other regions above that could be mistaken for vessel lumen. In all four cases our model predicts the vessel lumen accurately as clear from the predicted segmentations. We found that the plaque and calcified wall does not affect the model and are contained within and outside the lumen respectively as we desire. From Figure 4.3(d) we also see that the model can tell the true vessel lumen from other regions that appear to be the lumen when in fact they are not.

4.3.2 Effect of sample size

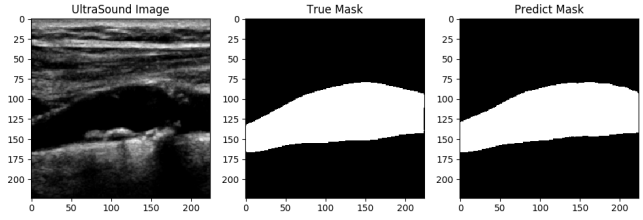
We ran the training model with five different sample sizes (25, 50, 100, 234, and all 302 images). Figure 4.4 demonstrates the average Dice coefficient in our 10-fold experiment with the respective different sample sizes. We see that both training and validation accuracies increase as we add more samples. In fact, the validation accuracy approaches the training one as the sample size increases.



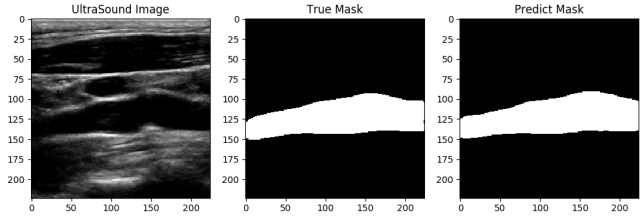
(a)



(b)



(c)



(d)

Figure 4.3 Examples of ultrasound images with their manual ground truth and predicted segmentations.

4.3.3 Effect of dual-path encoder

To study the performance of the synthetic modality of our dual-encoder model, we modified the original input image by flipping it along the y-axis (called flip) and

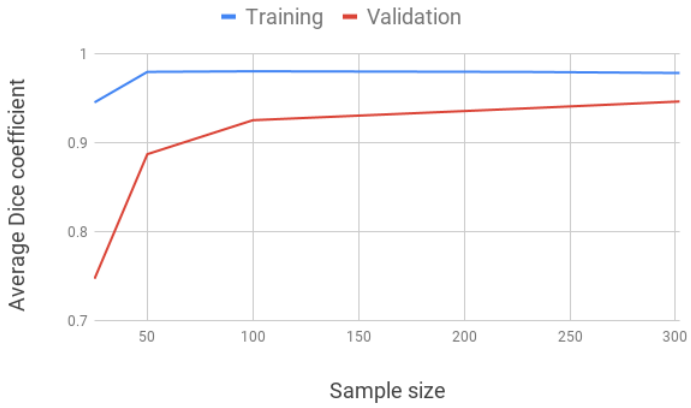


Figure 4.4 As our sample size increases we see both training and validation accuracy in 10-fold cross-validation increases.

adjusting the brightness and contrast separately by a factor of 2 (using functions implemented in Pytorch and Torchvision). In Table 4.2, we show that all four synthetic modalities don't improve the validation accuracy of the single-path model.

Table 4.2 Training and Validation Dice Accuracy of Our Dual-encoder Model with Different Synthetic Modalities

Synthetic modality	Vessel train	Vessel validation
Flip	98.46%	91.76%
Brightness	98.48%	93.83%
Contrast	98.39%	93.7%

4.3.4 Effect of modified Dice loss to emphasize recall

While our overall validation accuracy is at 95%, our model still has difficulty in some cases. In Figure 4.5(a), we see that the predicted segmentation is incomplete for this image even though the image does not appear to be hard. While there are no false positives the true positive rate (recall) is low. To fix this, we try a modified Dice loss below that upweights the recall component by a factor of 4.

$$D(p) = \frac{5\sum_i p_i r_i}{\sum_i p_i^2 + 4 \times \sum_i r_i^2}$$

We evaluate our model with the modified Dice loss on a subset of 234 samples on which we also report the original model's accuracy above. In Figure 4.5(b), we see that the modified loss improves the segmentation of this particular image. We see it also improves segmentation of the image shown in Figure 4.5(c) with our original Dice loss to cover the entire vessel with the modified loss in Figure 4.5(d).

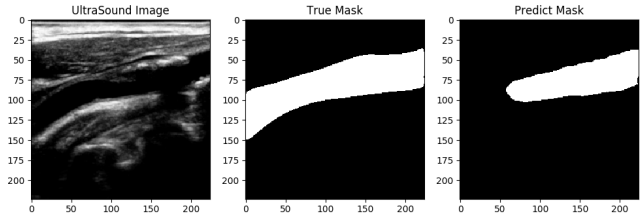
While we see an improvement in individual images, our average accuracy with the modified loss on train and validation is the same as the original one on 234 samples (see Table 4.3 below). This suggests that the new loss lowers the accuracy of other images and thus may not be the best solution to improve our recall.

Table 4.3 Average Dice Coefficients of Train and Validation Splits in Our 10-fold Cross-validation with Our Modified Dice Loss Across Three Different Runs of Our Model

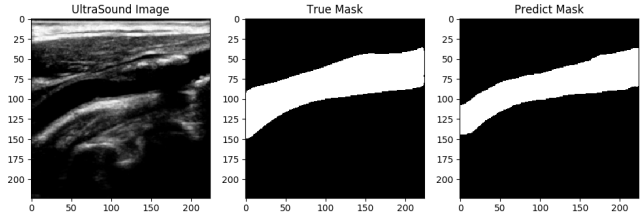
Model train run	Vessel train	Vessel validation
1	97.75%	93.7%
2	97.77 %	93.51%
3	97.77 %	93.24%

4.4 Discussion

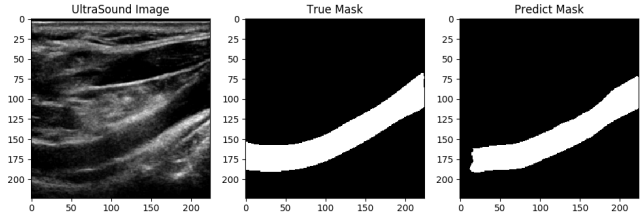
One of the difficulties that automated systems face in the evaluation of vascular ultrasounds is that image output and quality is highly dependent upon both technician technique and the patient's body habitus and individual anatomy. Plaque and calcium in the vessel wall can result in acoustic shadows that make it difficult to visualize the posterior wall. At 95% accuracy, the basic U-Net system that we have developed is able to adjust for these issues. In Figure 4.6, we present four different images where the posterior vessel wall is difficult to image and in each of these cases our system is



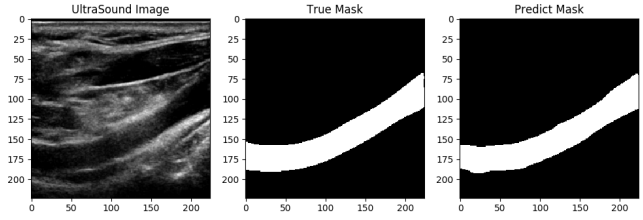
(a)



(b)



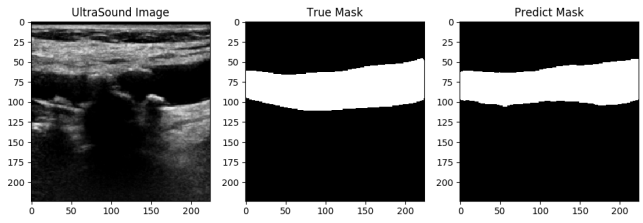
(c)



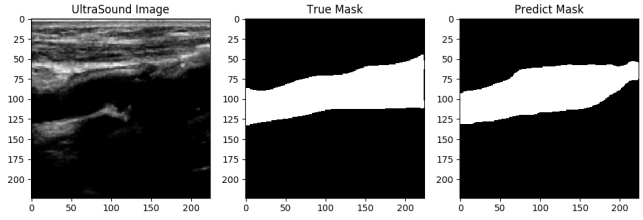
(d)

Figure 4.5 Examples of ultrasound images and their manual ground truth and predicted segmentations obtained from the original Dice loss in (a) and (c) and the modified loss in (b) and (d).

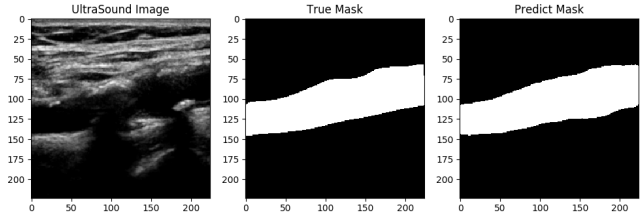
able to accurately predict the correct vessel segmentation except for minor errors as in Figure 4.6(b).



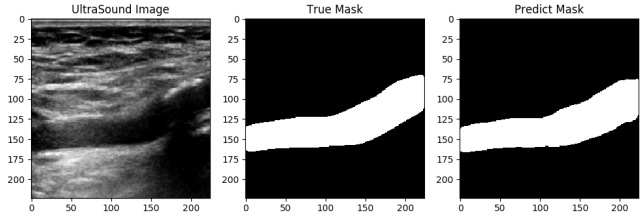
(a)



(b)



(c)



(d)

Figure 4.6 Examples of ultrasound images and their manual ground truth and predicted segmentations that have part of the vessel missing from the image due to ultrasound shadowing.

We believe that the strengths of our model will be beneficial as we expand the scope of our work. While initially we focused on imaging just the internal carotid artery we next plan to expand this to the entire carotid artery study looking at

the common carotid artery, external carotid artery and the carotid bifurcation. We chose the internal carotid artery initially due to the significant amount of plaque and atherosclerotic disease, making it one of the more difficult areas to get accurate predictions of the vessel lumen. Due to our successes with the internal carotid artery, we expect an overall high accuracy on ultrasounds of the entire carotid artery system.

Accurate visualization of the vessel lumen is only the first step in creating a valid clinical tool that can evaluate vascular ultrasounds. Creating a system that can also identify and accurately segment atherosclerotic plaque, identify the vessel wall and accurately measure its width and identify calcification within the wall will be required. Identifying several regions of the vessel within the ultrasound is more challenging and falls under multi-class segmentation. Our encouraging results here suggest we should achieve high accuracy there as well but may need to add more images from patients just because multi-class classification typically requires more data than the binary case.

4.5 Conclusion

We evaluated a single and dual path convolutional neural network for vessel lumen segmentation in carotid artery vascular ultrasounds. In 10-fold cross-validation on 302 images from 98 patients, we obtained 95% accuracy and expect this to rise as we add more images. Our work shows that vessel lumen segmentation can be achieved with high accuracy.

CHAPTER 5

VESSEL SEGMENTATION IN CAROTID ARTERY ULTRASOUNDS IMAGES

5.1 Background and Related Work

Carotid ultrasound is a screening modality used by physicians to direct treatment in the prevention of ischemic stroke in high-risk patients. It is a time intensive process that requires highly trained technicians and physicians. Evaluation of a carotid ultrasound requires identification of the vessel wall, lumen, and plaque of the carotid artery. Prior work in automated approaches to evaluating carotid ultrasounds is highly limited and there are no prior methods for vessel segmentation in carotid ultrasounds of the entire carotid system. A convolutional U-Net for 2D ultrasounds like ours was explored in previous work [75] but only for internal carotid artery ultrasounds. In this work, we explore a U-Net for the entire carotid artery system that includes internal, external, and common carotid arteries. We also explore images containing bifurcations, longitudinal images, and images with ultrasound shadowing, plaque, and gray shading, all of which make vessel segmentation even harder. Thus our work has a much broader scope than the previous study.

Vessel identification in carotid ultrasounds with preprocessing and marker-controlled watershed transform has been explored previously [68]. DeepVesselNet [69] is a deep learning model designed for vessel detection but in 3D magnetic resonance angiography data unlike the 2D ultrasounds that we consider here. A patch-based deep learning solution has also been proposed segmenting and measuring plaque for 3D ultrasounds [82]. Of note, 3D ultrasound is available only in research studies and is not commonly utilized clinically. In contrast, in our study is a full end-to-end trainable convolutional network that allows for the segmentation of 2D ultrasounds, the most widely utilized modality.

5.2 Methods

5.2.1 Data collection

We obtained IRB approval from Robert Wood Johnson Medical School to use de-identified images from the Department of Vascular Surgery for this research. We manually downloaded B-mode carotid ultrasound examinations of 226 patients. We utilized an automated script to crop all patient identities from the ultrasound images and manually verified this de-identification.

We then cropped each image to obtain just the ultrasound removing all text and annotations on the image. Each image was resized to 224x224 pixels. This gave us a total of 2156 images that we then manually segmented. Using RectLabel software (`https://rectlabel.com/`) we manually segmented the vessel lumen for each image to serve as ground truth for training and validation

5.2.2 Background

Convolutional neural networks Convolutional neural networks are the current state of the art in machine learning for image recognition [36, 34], including for MRI [9]. They are typically composed of alternating layers for convolution and pooling, followed by a final flattened layer. A convolution layer is specified by a filter size and the number of filters in the layer. Briefly, the convolution layer performs a moving dot product against pixels given by a fixed filter of size $k \times k$ (usually 3×3 or 5×5). The dot product is made non-linear by passing the output to an activation function such as a sigmoid or rectified linear unit (also called relu or hinge) function. Both are differentiable and thus fit into the standard gradient descent framework for optimizing neural networks during training. The output of applying a $k \times k$ convolution against a $p \times p$ image is an image of size $(p-k+1) \times (p-k+1)$. In a CNN, the convolution layers just described are typically alternated with pooling layers. The pooling layers serve to reduce dimensionality, making it easier to train the network.

Convolutional U-Net After applying a series of convolutional filters, the final layer dimension is usually much smaller than that of the input images. For the current problem of determining whether a given pixel in the input image is part of a vessel or plaque, the output must be of the same dimension as the input. This dimensionality problem was initially solved by taking each pixel in the input image and a localized region around it as input to a convolutional neural network instead of the entire image [18].

A more powerful recent solution is the Convolutional U-Net (U-Net) [58]. This has two main features that separate it from traditional CNNs: (a) deconvolution (upsampling) layers to increase image dimensionality, and (b) connections between convolution and deconvolution layers.

U-Net for vessel segmentation We implemented a basic U-Net [58] in the Pytorch library [53] as shown in Figure 5.1. The U-Net is a popular choice for medical artificial intelligence work and has proven to be a successful baseline that can be built upon. The input to the model is an ultrasound image and output is an image of the same dimensions with 0 and 1 pixel values indicating background and vessel lumen.

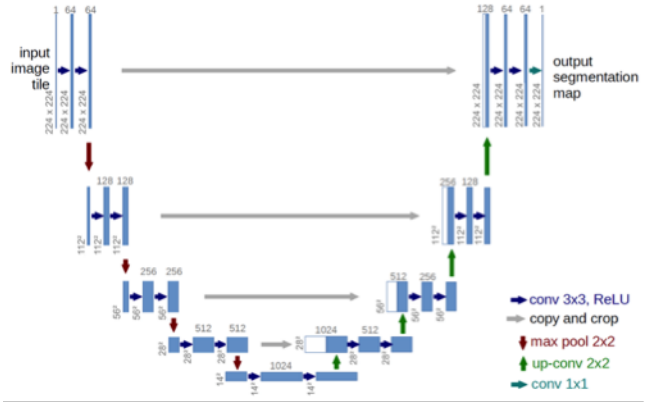


Figure 5.1 U-Net architecture [58] that we use in our preliminary work. Shown here are dimensions of our images in each layer and the number of convolutional and transposed convolutions per layer.

Roughly speaking, in this model we first extract features with a series of convolutional kernels and then apply transpose convolutions to increase the dimensionality of the image up to the original. Thus, we have an end-to-end network that is much simpler to train than otherwise patch-based approaches that have previously been used for segmentation.

Dice loss The final output from the each of our models is a 2D predicted image of dimensions 224×224. We convert each pixel value into probabilities with softmax [4] and call the resulting image p. The target ground truth r is also of the same dimensions as p and contains a 1 if the pixel is within the vessel lumen and 0 otherwise. We then use the Dice loss to train our model. This is defined to be $1 - D$ where

$$D(p) = \frac{2\sum_i p_i r_i}{\sum_i p_i^2 + \sum_i r_i^2}$$

and p_i and r_i are the i^{th} pixel values of p and r respectively.

5.2.3 Implementation, accuracy, and validation

Implementation We implemented our models using Pytorch [53] and ran them on NVIDIA Pascal P100 and NVIDIA Titan RTX GPUs. We trained our models with 20 epochs of stochastic gradient descent [10], a learning rate of 0.03, decay step of 15 (with $\gamma = .1$), and a batch size of 4. We did not perform any normalization on the input images.

Post processing We applied a simple post processing procedure to reduce potential false positives. In the final predicted segmentation, we remove all disconnected components except for the largest one that is meant to be the vessel lumen. We found that this improved accuracy by a moderate margin.

Measure of accuracy: Dice coefficient The Dice coefficient is typically used to measure the accuracy of predicted segmentations in medical images [84]. We convert

the output image of our network into a binary mask by setting each pixel value to 1 if its softmax output is at least 0.5 and 0 otherwise. Thus we use 0.5 as the probability threshold that a pixel value is part of the vessel lumen or outside it.

Starting with the human binary mask as ground truth, each predicted pixel is determined to be either a true positive (TP, also one in true mask), false positive (FP, predicted as one but zero in the true mask), or false negative (FN, predicted as zero but one in the true mask). The Dice coefficient is formally defined as

$$DICE = \frac{2TP}{2TP + FP + FN} \tag{5.1}$$

10-fold cross-validation We performed 10-fold cross-validation experiments on our data. We randomly split our dataset into ten equal parts and selected one part for validation while the remaining nine parts were used to train the model. We then rotated the validation part across the other nine parts giving us a total of ten pairs of training validation splits. We trained the model on each split and reported the average validation and training accuracy below.

5.3 Results

We first perform a 10-fold cross-validation on the entire set of images. In Table 5.1, we see that we achieve a high training and validation accuracy of 95.1% and 94.3% respectively. The small difference between our training and validation accuracies indicates our model is not overfitting and instead is generalizing well.

When we train and test on internal (ICA), external (ECA), and the common (CCA) carotid artery ultrasounds alone we see varying degrees of accuracy. Both ICA and ECA images achieve similar and lower train and validation accuracies than CCA which alone has a 96.6% accuracy (Table 5.1).

Table 5.1 Average Accuracy of Training and Validation Splits in Our 10-fold Experiment

	Training	Validation
All	95.1%	94.3%
ICA	94.9%	92.7%
ECA	96.2%	91.9%
CCA	97.9%	96.6%

In Figure 5.2, we see that adding more samples increases both the training and validation accuracy of our model. This is overall encouraging, however, the increase in accuracy is by small margins and is plateauing at 95% as we add more samples.

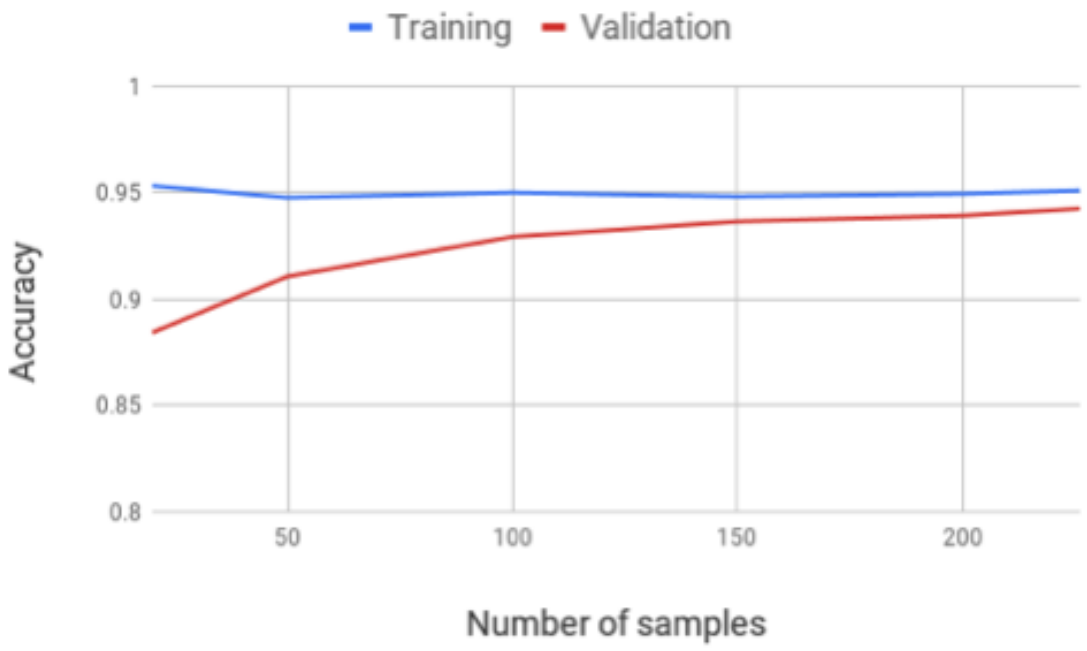


Figure 5.2 Training and validation accuracy of our training and validation folds as a function of sample size.

In Figure 5.3, we see examples of some images with their true and predicted segmentations (also known as masks). Both Figure 5.3 (a) and (b) show examples with significant plaque and shadowing that could obfuscate the untrained eye but

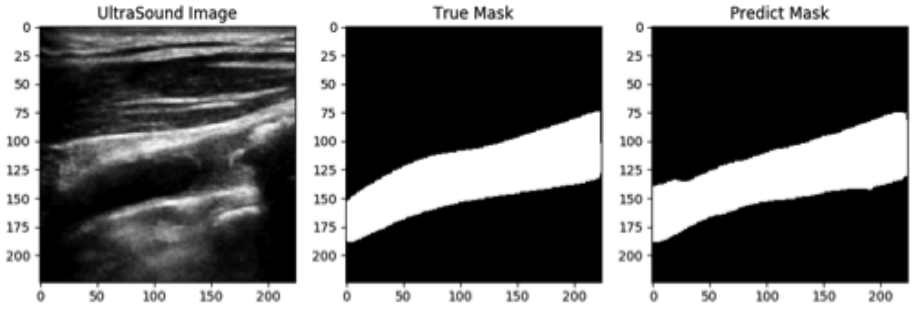
our model gives a highly correct segmentation. In Figure 5.3 (c) and (d), we have examples of bifurcated and gray shaded vessels that that are also correctly segmented by our model.

5.4 Discussion

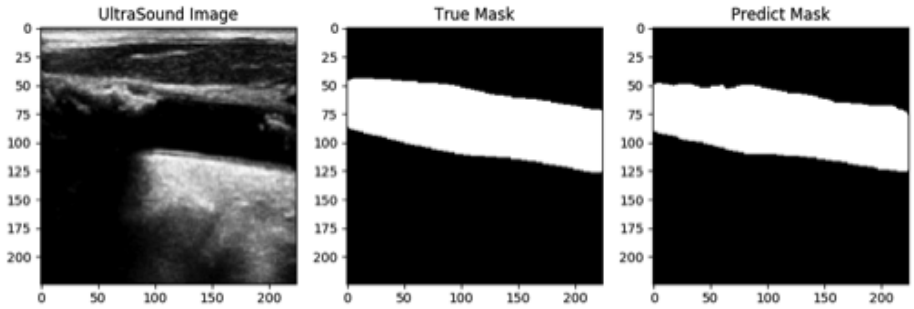
Medical imaging has become an essential component in modern medicine. It aids in diagnosis, tracks progression of disease and can be utilized to screen individuals for cancer and for the prevention of stroke. Numerous studies are looking at using deep learning methods to increase accuracy of diagnosis and aid in the interpretation of these studies [43]. As of yet, there are few studies that look at utilizing deep learning for vessel identification and evaluation with ultrasound images specifically in the carotid artery system.

Ultrasound images provide a distinct challenge that is different than other medical imaging modalities. Computed tomography and magnetic resonance imaging (MRI) have set protocols that control formatting and orientation. For example, MRI images are typically aligned to a standard reference brain template such as the Montreal Neurological Institute reference space [22] that makes it easier to compare different MRI images.

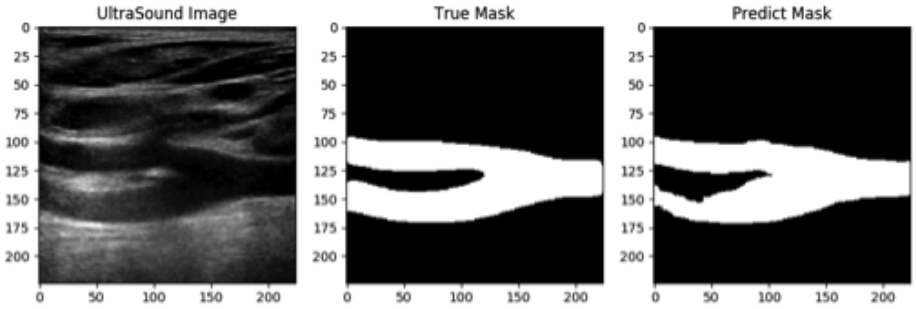
A convolutional neural network was previously proposed for vessel detection in ultrasounds of femoral regions and also applied to carotid artery ultrasounds [62]. There are several key differences between our study and this previous one. In the previous study, authors evaluate their method on transverse images of the common carotid artery. Specifically, they identify the center of the vessel and outline the vessel with an ellipse that approximates the vessel. To do this, they are using a simplified version of the AlexNet [34] convolutional neural network. They reduced it to two convolutional layers, one normalization, two max pooling, and three fully connected. Their modified network outputs the center and two radii of the ellipse enclosing the



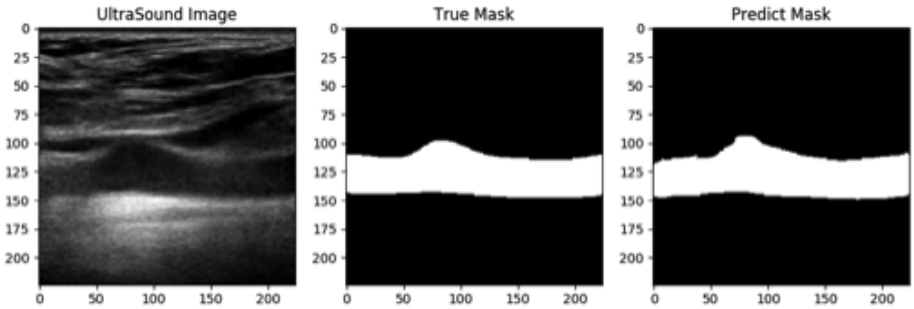
(a) Shadow and plaque I



(b) Shadow and plaque II



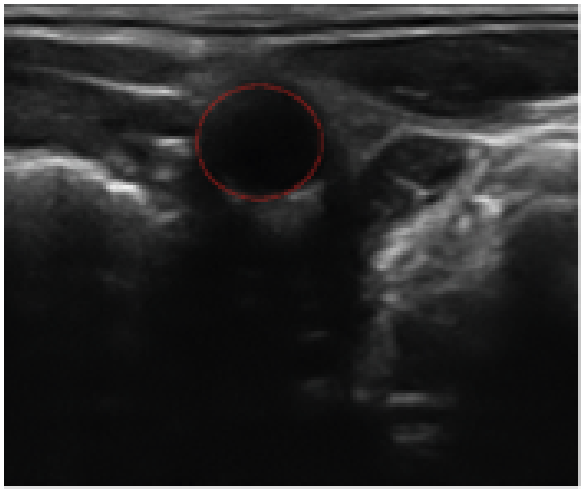
(c) Bifurcation



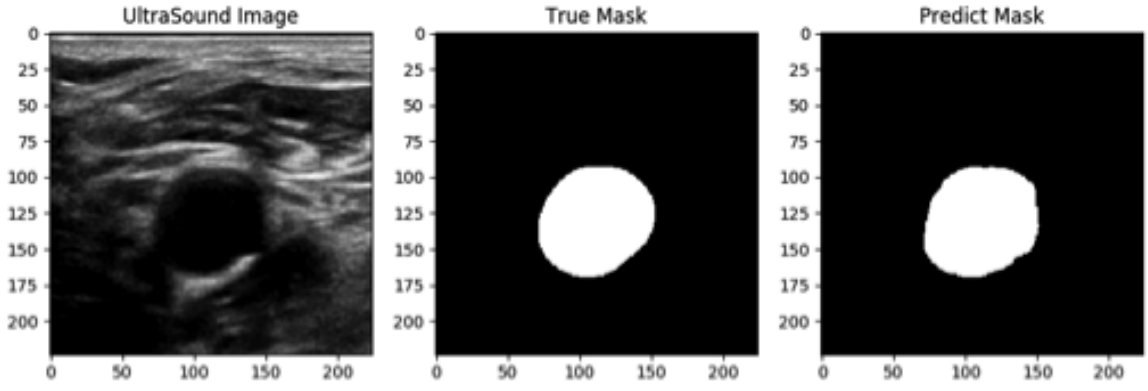
(d) Gray shading

Figure 5.3 Non-trivial examples of vessels in ultrasound image.

circular vessel. In contrast, the U-Net that we use outputs a full segmentation of the vessel that can segment both transverse and longitudinal images of the vessel (Figure 5.4). Their study also only evaluates the common carotid artery, whereas our model can also be used to evaluate the internal and external carotid arteries, which is important because assessment of the carotid bifurcation and internal carotid artery has the most clinical relevance to stroke prevention. The previous study purely helps to identify that a vessel is present, it provides little additional input to aid in the interpretation of the ultrasound.



(a) Vessel identification from a previous study



(b) Our model performs circular segmentation that includes longitudinal images

Figure 5.4 Comparison of vessel identification from previous work to vessel segmentation in our work.

5.5 Conclusion

The work that we present above is entirely novel in scope. It is the first step in attempting to create and implement a neural network that can independently and accurately identify and segment the lumen of the carotid artery in a vascular ultrasound. Further studies will be required to advance this model so that it can handle segmentation of the vessel wall, atherosclerotic plaque and evaluate the direction of flow and flow velocity within the lumen, before it can provide clinically relevant interpretations.The model has the potential to be the first step in creating a complete end-to-end solution for the evaluation of vascular ultrasound images.

CHAPTER 6

VESSEL AND PLAQUE SEGMENTATION IN CAROTID ARTERY ULTRASOUNDS IMAGES

6.1 Background and Related Work

Carotid ultrasound is a screening modality used by physicians to direct treatment in the prevention of ischemic stroke in high-risk patients. It is a time intensive process that requires highly trained technicians and physicians. Evaluation of a carotid ultrasound requires identification of the vessel wall, lumen, and plaque of the carotid artery. Most prior work in automatic vessel and plaque segmentation consider 3-dimensional (3D) ultrasound images with narrowed regions of interest or MRI images, and few are based on deep or machine learning methods. For example convolutional U-Nets have been explored previously on 3D ultrasound images from the common carotid artery but their data is narrowed to a region of interest and they examined only images that contain plaque [83]. Traditional machine learning methods have been applied for plaque segmentation from B-mode ultrasounds of common carotid arteries [56]. Another study applied basic machine learning methods for plaque segmentation but on MRI images [81].

Other than the above there are plaque segmentation methods limited to video and common carotid arteries [42], histogram based methods on combined B-mode and contrast enhanced ultrasounds of common, internal, and external carotid arteries [3], and parametric and geometric deformable models followed by Bayesian classifiers on intravascular ultrasounds [67]. Previous work also includes image intensity and structure based methods on 3D ultrasounds of common carotid arteries [17], fuzzy clustering on MRI images for plaque detection only [1, 2], 3D volume-based level-set method on 3D ultrasounds of common, internal, and external carotid arteries [16],

and a slice-based semi-automatic method on CTA images of common and internal carotid arteries [71].

In contrast to the previous work, our work considers raw ultrasound images without any pre-processing or narrowing the region of interest. These are taken directly from the hospital ward and are the same images that a trained physician would be looking at. Of note, 3D ultrasound is available only in research studies and is not commonly utilized clinically. Our proposed models are full end-to-end trainable convolutional U-Nets that allow for the segmentation of 2D ultrasounds, the most widely utilized modality. Deep learning models for vessel segmentation alone have been proposed previously on 2D and 3D carotid ultrasounds [75, 69, 62]; but in this work, we consider segmentation of both vessel and plaque.

6.2 Methods

6.2.1 Data collection

We obtained IRB approval from Robert Wood Johnson Medical School to use de-identified images from the Department of Vascular Surgery for this research. We manually downloaded B-mode carotid ultrasound examinations of 226 patients. We utilized an automated script to crop all patient identities from the ultrasound images and manually verified this de-identification.

We then cropped each image to obtain just the ultrasound removing all text and annotations on the image. Each images was resized to 224x224 pixels. We manually segmented the vessel lumen and plaque of 500 images using the RectLabel software (https://rectlabel.com/). These serve as ground truth for training and validation

6.2.2 Background

Convolutional neural networks Convolutional neural networks (CNN) are the current state of the art in machine learning for image recognition [36, 34], including

for MRI [9]. They are typically composed of alternating layers for convolution and pooling, followed by a final flattened layer. A convolution layer is specified by a filter size and the number of filters in the layer. Briefly, the convolution layer performs a moving dot product against pixels given by a fixed filter of size $k \times k$ (usually 3×3 or 5×5). The dot product is made non-linear by passing the output to an activation function such as a sigmoid or rectified linear unit (also called relu or hinge) function. Both are differentiable and thus fit into the standard gradient descent framework for optimizing neural networks during training. The output of applying a $k \times k$ convolution against a $p \times p$ image is an image of size $(p - k + 1) \times (p - k + 1)$. In a CNN, the convolution layers just described are typically alternated with pooling layers. The pooling layers serve to reduce dimensionality, making it easier to train the network.

Basic convolutional U-Net After applying a series of convolutional filters, the final layer dimension is usually much smaller than that of the input images. For the current problem of determining whether a given pixel in the input image is part of a vessel or plaque, the output must be of the same dimension as the input. This dimensionality problem was initially solved by taking each pixel in the input image and a localized region around it as input to a convolutional neural network instead of the entire image [18].

A more powerful recent solution is the Convolutional U-Net (U-Net) [58]. This has two main features that separate it from traditional CNNs: (a) deconvolution (upsampling) layers to increase image dimensionality, and (b) connections between convolution and deconvolution layers.

We implemented a basic U-Net [58] in the Pytorch library [53] as shown in Figure 6.1. The U-Net is a popular choice for medical artificial intelligence work and has proven to be a successful baseline that can be built upon. The input to the model

is an ultrasound image and output is an image of the same dimensions with 0, 1, and 2 pixel values indicating background, vessel lumen, and plaque.

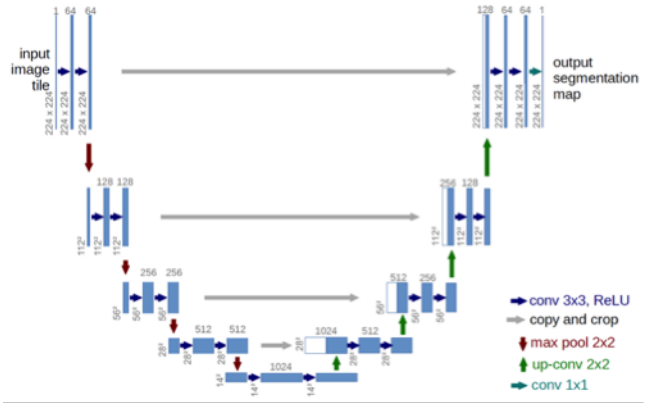


Figure 6.1 Basic U-Net architecture [58] that we use as a baseline for our work. Shown here are dimensions of our images in each layer and the number of convolutional and transposed convolutions per layer.

Roughly speaking, in this model we first extract features with a series of convolutional kernels and then apply transpose convolutions to increase the dimensionality of the image up to the original. Thus, we have an end-to-end network that is much simpler to train than otherwise patch-based approaches that have previously been used for segmentation.

6.2.3 Our proposed U-Net models

Aside from the basic U-net that we use as a baseline, we investigate two extensions: a cascaded model with two networks and a dual decoder network with separate decoders for vessel and plaque.

Two-stage convolutional U-Net In the two-stage approach, we have a cascaded model of two convolutional U-Nets (as shown in Figure 6.2). In the first model, we segment the vessel lumen from which we then segment the plaque with a second U-Net. For training the first network, we manually segment the vessel of an additional 1661 images from our cohort of patients so as to have it be as accurate as possible.

In fact, we see later that the two-stage model indeed gives a better segmentation of the vessel lumen than basic U-Net and the dual decoder in the next subsection.

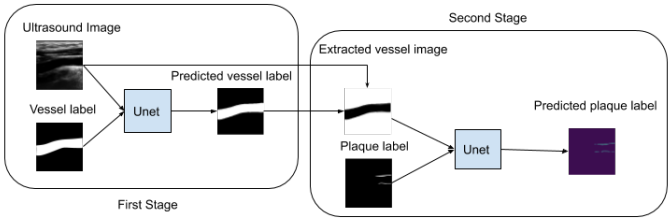
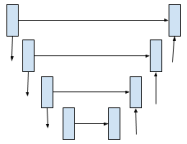
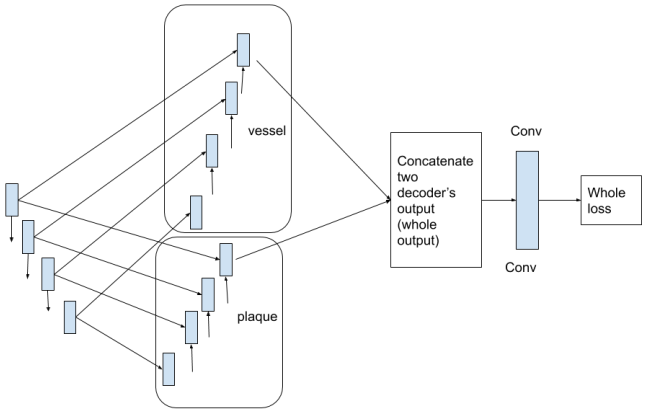


Figure 6.2 Two-stage cascaded model containing two convolutional U-Nets.

Dual decoder convolutional U-Net The basic U-Net is a series of encoders and decoders with connections between them. In our dual-decoder, we have a pair decoders in each step of the decoding (see Figure 6.3). In each pair, one decoder is for segmenting the vessel and the other is for the plaque.



(a)



(b)

Figure 6.3 Dual decoder convolutional U-Net with separate decoders for vessel and plaque. In (a) is the basic U-Net and in (b) is the dual decoder version.

Ensemble models and confidence scores We obtain confidence estimates by running each model 39 times starting with different seeds for the random number generator. From the outputs of each model, we obtain a confidence estimate based on pixel frequencies as shown in Figure 6.4. We also obtain a majority vote prediction from the ensemble output as shown in *final prediction* in Figure 6.4.

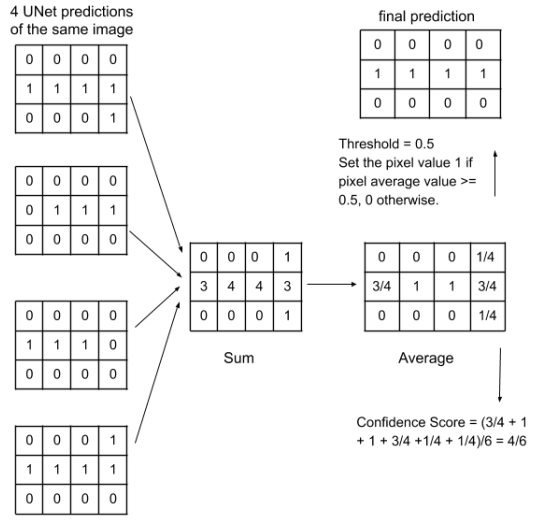


Figure 6.4 Obtaining a confidence score and a final prediction from the outputs of four U-nets.

Dice loss The final output from the each of our models is a 2D predicted image of dimensions 224×224. We convert each pixel value into probabilities with softmax [4] and call the resulting image p. The target ground truth r is also of the same dimensions as p.

In the basic U-Net and our dual decoder, the output is a segmentation of both the vessel and plaque. In the ground truth pixels, we assign 1 if the pixel is within the vessel lumen, 2 if it in a plaque region, and 0 otherwise. In the two stage model, we first output a vessel segmentation followed by a plaque. The ground truth for each network contains 1 if the pixel is in the vessel or plaque and 0 otherwise.

We then use the Dice loss to train our model. This is defined to be $1 - D$ where

$$D(p) = \frac{2\sum_i p_i r_i}{\sum_i p_i^2 + \sum_i r_i^2}$$

and p_i and r_i are the i^{th} pixel values of p and r respectively.

6.2.4 Implementation, accuracy, and validation

Implementation We implemented our models using Pytorch [53] and ran them on NVIDIA Pascal P100 and NVIDIA Titan RTX GPUs. We trained our models with 20 epochs of stochastic gradient descent [10], a learning rate of 0.03, decay step of 15 (with $\gamma = .1$), and a batch size of 4. We did not perform any normalization on the input images.

Measure of accuracy: Dice coefficient The Dice coefficient is typically used to measure the accuracy of predicted segmentations in medical images [84]. The output from our network is a set of probabilities for each pixel indicating the class it belongs to. For example, for a given pixel we have probabilities that it is background, vessel, or plaque. We assign 0, 1, or 2 depending upon the maximum probability. This then gives us an image with the same pixel values as the ground truth and allows us to calculate the Dice coefficient separately for vessel and plaque.

For vessel Dice, each predicted pixel is determined to be either a true positive (TP, also one in ground truth), false positive (FP, predicted as one but zero or two in the ground truth), or false negative (FN, predicted as zero or two but one in ground truth). To calculate the plaque Dice coefficient, we follow the same formula except that positive predictions have value two and negative are zero or one. After calculating these values, the Dice coefficient is then formally defined as

$$DICE = \frac{2TP}{2TP + FP + FN} \tag{6.1}$$

10-fold cross-validation We performed 10-fold cross-validation experiments on our data. We randomly split our dataset into ten equal parts and selected one part

for validation while the remaining nine parts were used to train the model. We then rotated the validation part across the other nine parts giving us a total of ten pairs of training validation splits. We trained the model on each split and reported the average validation and training accuracy below.

6.3 Results

6.3.1 Vessel and plaque segmentation

In Table 6.1, we see the vessel and plaque segmentation Dice accuracies of our models. Our dual decoder has the highest plaque accuracy followed by the basic model and then the two-stage. Interestingly if we were to use the true vessel as input to the second network in the two-stage model the plaque Dice accuracies increases to 0.82. This suggests there is room for improvement in this model: if we can get the first network to produce more accurate vessel segmentations, it would in turn improve the plaque Dice accuracy of the second network.

Table 6.1 10-fold Dice Coefficients of The Basic U-Net and Our Two Models

	Basic U-Net	Two stage	Dual decoder
Vessel Dice	.9	.95	.91
Plaque Dice	.67	.65	.69

In Figure 6.5, we see several ultrasound images, their ground truth segmentations, and their predicted segmentations by our dual decoder model. These are handpicked images where our model produces a visually correct vessel and plaque segmentation.

In Figure 6.6, we show handpicked images with poorer segmentations. We see that in some cases our model produces false positives. Even though these images have low plaque segmentation accuracies, in some cases, the segmentations are still useful than otherwise.

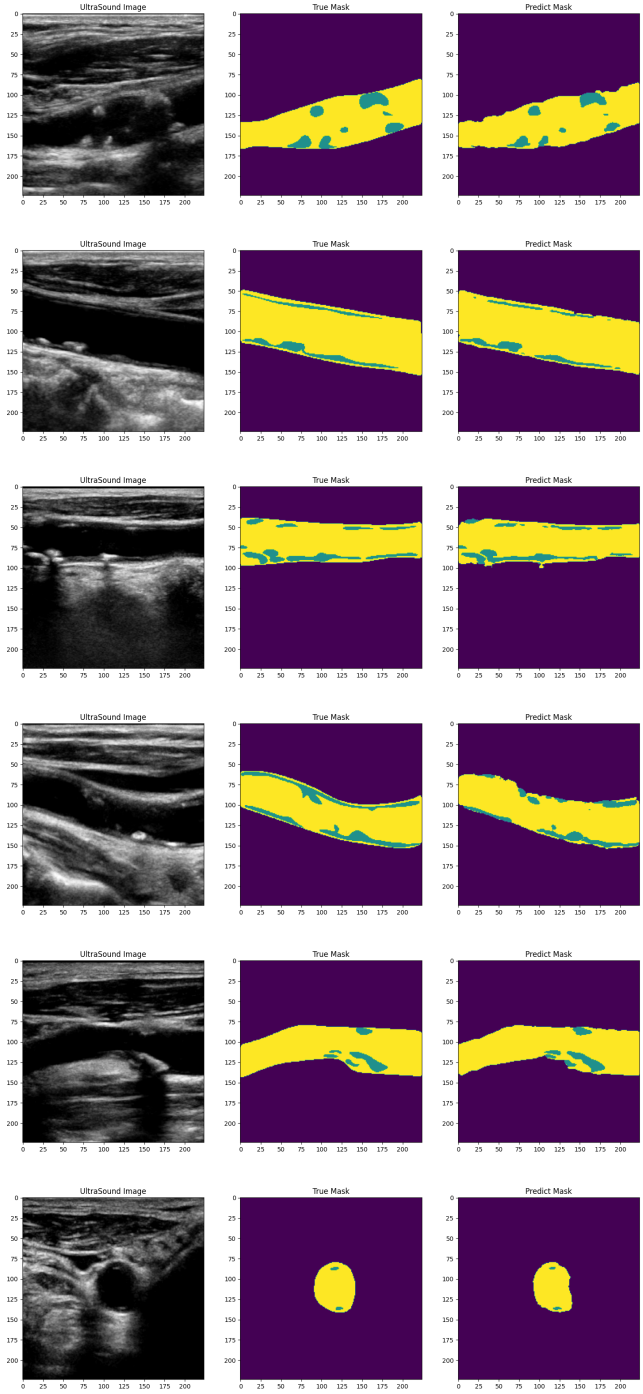


Figure 6.5 Examples of ultrasound images and their true and predicted segmentations. In yellow is the vessel and green is the plaque. Our model performs very well in these handpicked images.

6.3.2 Ensemble and confidence scores

By ensembling, we can obtain confidence scores for each model as described earlier.

In Table 6.2 below we show the plaque Dice coefficients for different confidence score

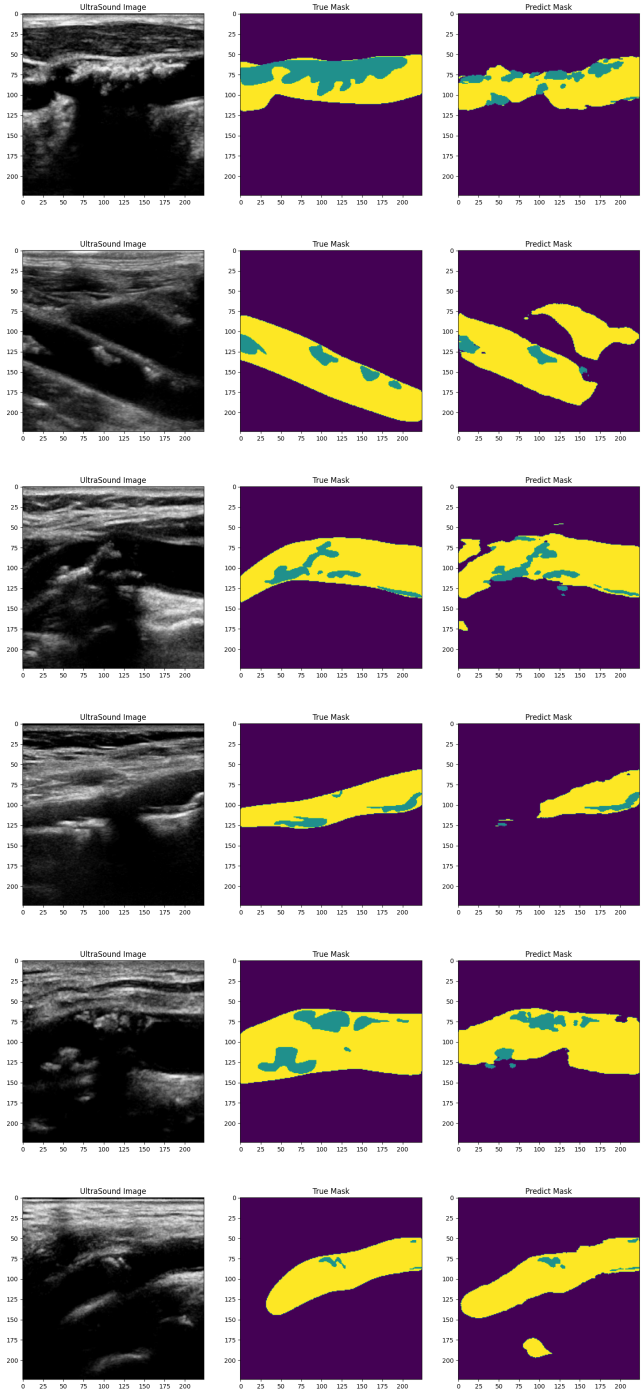


Figure 6.6 Examples of ultrasound images and their true and predicted segmentations. In yellow is the vessel and green is the plaque. These are handpicked images where our model performs relatively poorly compared to images in Figure 6.5.

thresholds. As we raise the confidence threshold, the plaque Dice coefficient increases

with the dual decoder performing slightly better than the basic model. We get fewer images at higher thresholds but these are likely to be highly accurate as we see below.

Table 6.2 10-fold Plaque Dice Coefficients of The Basic U-Net and The Dual-decoder Model for Different Confidence Thresholds

Confidence threshold	Basic U-Net	Dual decoder
.5	.728 (34.7)	.732 (35.6)
.6	.753 (27.3)	.752 (27.9)
.7	.792 (16.2)	.795 (15.8)
.8	.859 (6.2)	.873 (5.1)

Note: In parenthesis are the average number of images with confidence above the given threshold across the ten folds.

In an attempt to improve the vessel segmentation in our two-stage model, we explored ensembling. We used the majority vote output (see Figure 6.4) of 39 models for vessel segmentation. This improved the plaque segmentation from .65 to .67 but not near the .82 accuracy we obtain with the true vessel image as input. Upon closer examination of our predicted vessel segmentations, we may have found the source of our problem.

Even though vessel segmentations in the two-stage model are highly accurate (.95 Dice), if the segmentation is missing part of the vessel that contains the plaque it severely affects the plaque segmentation accuracy (see Figure 6.7). One way to address this is to expand the vessel segmentation to nearby pixels so that the plaque is included in the image for the plaque segmentation model. We plan to explore this in future work.

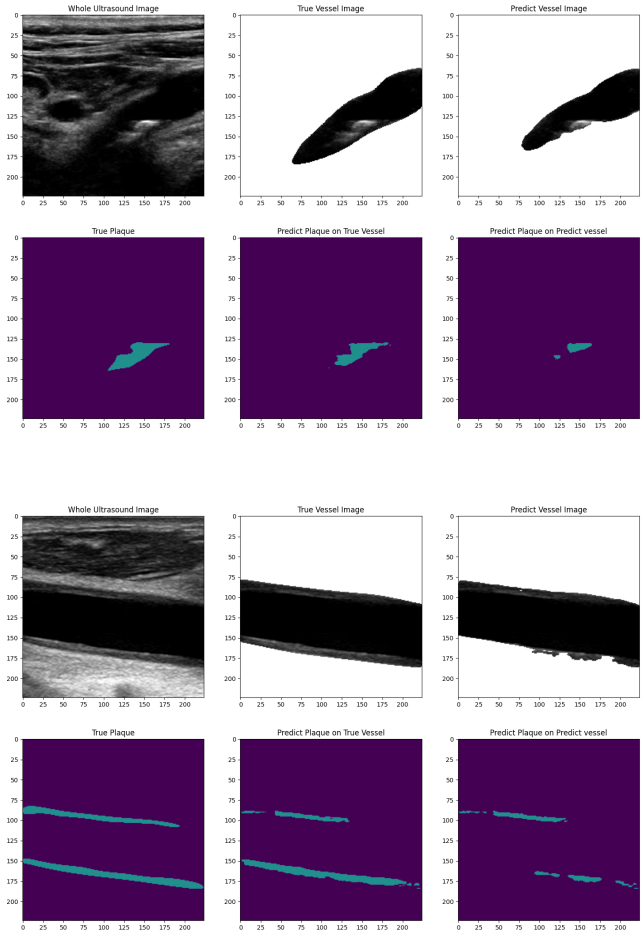


Figure 6.7 Examples of ultrasound images and their true and predicted vessel segmentations from the two-stage model. Even though the overall vessel is correctly segmented some parts near the plaque are left out. The plaque segmentation model can thus never identify the plaque since it is missing altogether from the input. As a result the plaque segmentation in our two-stage model is lower than the base and dual decoder model.

6.4 Conclusion

Our work here shows the potential of dual and two-stage methods for vessel and plaque segmentation in carotid artery ultrasound images. This is an important first step in creating a system that can independently evaluate carotid ultrasounds.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

This dissertation investigates adversarial robustness with 01 loss models and investigates novel convolutional neural net system for vascular ultrasound images. First the dissertation presents SCD for 01 loss and its sensitivity to adversarial attacks. Second, the dissertation presents a sign activation networks with SCD and its ensembling model, and their sensitivity to boundary blackbox attack. Third, this dissertation tackles three important segmentation problems for vascular ultrasound images: (1) vessel segmentation in internal carotid artery, (2) vessel segmentation in the entire carotid system, and (3) vessel and plaque segmentation in the entire carotid system.

In the future work, further experimental work for SCD 01 loss is required, such as more experiments on other image benchmarks and more bootstraps. This dissertation shows the potential of dual and two-stage methods for vessel and plaque segmentation in carotid artery ultrasound images. In the future, more works will be investigated, especially the two-stage methods. This dissertation tackles the problems on vascular ultrasound images. In the future, other medical data will be studied, such as vascular ultrasound video.

REFERENCES

[1] I. M. Adame, R. J. van der Geest, B. A. Wasserman, M. A. Mohamed, J. H. C. Reiber, and B. P. F. Lelieveldt. Automatic plaque characterization and vessel wall segmentation in magnetic resonance images of atherosclerotic carotid arteries. In *Proceedings of Medical Imaging: Image Processing*, volume 5370, pages 265–273, 2004.

[2] I. M. Adame, R. J. Van Der Geest, B. A. Wasserman, M. A. Mohamed, J. H. C. Reiber, and B. P. F. Lelieveldt. Automatic segmentation and plaque characterization in atherosclerotic carotid artery mr images. *Magnetic Resonance Materials in Physics, Biology and Medicine*, 16(5):227–234, 2004.

[3] Z. Akkus, N. de Jong, A. F. van der Steen, J. G. Bosch, S. C. van den Oord, A. F. Schinkel, D. D. Carvalho, W. J. Niessen, and S. Klein. Fully automated carotid plaque segmentation in combined b-mode and contrast enhanced ultrasound. In *Proceedings of IEEE International Ultrasonics Symposium*, pages 911–914, 2014.

[4] E. Alpaydin. *Machine Learning*. MIT Press, Cambridge, MA, 2004.

[5] A. Asuncion and D. J. Newman. UCI machine learning repository, 2007.

[6] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of International Conference on Machine Learning*, pages 274–283, 2018.

[7] P. L. Bartlett, M. I. Jordan, and J. D. Mcauliffe. Large margin classifiers: Convex loss, low noise, and convergence rates. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 1173–1180. MIT Press, Cambridge, MA, 2004.

[8] S. Ben-David, N. Eiron, and P. M. Long. On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514, 2003.

[9] J. Bernal, K. Kushibar, D. S. Asfaw, S. Valverde, A. Oliver, R. Martí, and X. Lladó. Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. *Artificial Intelligence in Medicine*, 2018.

[10] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of the 19th International Conference on Computational Statistics*, pages 177–186. Springer, 2010.

[11] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[12] W. Brendel, J. Rauber, and M. Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018.

[13] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Proceedings of IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[14] J. Chen and Q. Gu. Rays: A ray searching method for hard-label adversarial attack. In *Proceedings of the 13th ACM Special Interest Group on Knowledge Discovery and Data Mining*, pages 1739–1747, 2020.

[15] J. Chen, M. Jordan, and M. Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *Proceedings of IEEE Symposium on Security and Privacy (SP)*, pages 668–685.

[16] J. Cheng, Y. Chen, Y. Yu, and B. Chiu. Carotid plaque segmentation from three-dimensional ultrasound images by direct three-dimensional sparse field level-set optimization. *Computers in Biology and Medicine*, 94:27–40, 2018.

[17] J. Cheng, H. Li, F. Xiao, A. Fenster, X. Zhang, X. He, L. Li, and M. Ding. Fully automatic plaque segmentation in 3-d carotid ultrasound images. *Ultrasound in Medicine and Biology*, 39(12):2431–2446, 2013.

[18] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2843–2851, 2012.

[19] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

[20] S. Darabi, M. Belbahri, M. Courbariaux, and V. P. Nia. Bnn+: Improved binary network training. *arXiv preprint arXiv:1812.11800*, 2018.

[21] Y. Dong, T. Pang, H. Su, and J. Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4312–4321, 2019.

[22] V. Fonov, A. C. Evans, K. Botteron, C. R. Almli, R. C. McKinstry, D. L. Collins, B. D. C. Group, et al. Unbiased average age-appropriate atlases for pediatric studies. *Neuroimage*, 54(1):313–327, 2011.

[23] A. Galloway, G. W. Taylor, and M. Moussa. Attacking binarized neural networks. In *International Conference on Learning Representations*, 2018.

[24] L. Geiger and P. Team. Larq: An open-source library for training binarized neural networks. *Journal of Open Source Software*, 5(45):1746, 2020.

[25] A. Ghiasi, A. Shafahi, and T. Goldstein. Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates. *arXiv preprint arXiv:2003.08937*, 2020.

[26] J. Gilmer, N. Ford, N. Carlini, and E. Cubuk. Adversarial examples are a natural consequence of test error in noise. In *Proceedings of International Conference on Machine Learning*, pages 2280–2289, 2019.

[27] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[29] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *Proceedings of Advances in Neural Information Processing Systems*, pages 4107–4115, 2016.

[30] A. Kantchelian, J. D. Tygar, and A. Joseph. Evasion and hardening of tree ensemble classifiers. In *Proceedings of International Conference on Machine Learning*, pages 2387–2396, 2016.

[31] S. Kariyappa and M. K. Qureshi. Improving adversarial robustness of ensembles with diversity training. *arXiv preprint arXiv:1901.09981*, 2019.

[32] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *Proceedings of International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[33] A. Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto, 2009.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[35] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[37] L. Li and H.-T. Lin. Optimizing 0/1 loss for perceptrons by random coordinate descent. In *Proceedings of Neural Networks, IJCNN, International Joint Conference on Neural Networks*, pages 749–754, 2007.

[38] L. Liu, W. Wei, K.-H. Chow, M. Loper, E. Gursoy, S. Truex, and Y. Wu. Deep neural network ensembles against deception: Ensemble diversity, accuracy and robustness. In *Proceedings of IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 274–282, 2019.

[39] X. Liu, M. Cheng, H. Zhang, and C.-J. Hsieh. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 369–385, 2018.

[40] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

[41] Z. Liu, B. Wu, W. Luo, X. Yang, W. Liu, and K.-T. Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 722–737, 2018.

[42] C. P. Loizou, S. Petroudi, C. S. Pattichis, M. Pantziaris, T. Kasparis, and A. Nicolaides. Segmentation of atherosclerotic carotid plaque in ultrasound video. In *Proceedings of Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 53–56, 2012.

[43] A. S. Lundervold and A. Lundervold. An overview of deep learning in medical imaging focusing on mri. *Zeitschrift für Medizinische Physik*, 29(2):102–127, 2019.

[44] D. Mishra, S. Chaudhury, M. Sarkar, S. Manohar, and A. S. Soin. Segmentation of vascular regions in ultrasound images: A deep learning approach. In *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2018.

[45] D. Mozaffarian, E. J. Benjamin, A. S. Go, D. K. Arnett, M. J. Blaha, M. Cushman, S. R. Das, S. De Ferranti, J. P. Després, H. J. Fullerton, et al. Heart disease and stroke statistics-2016 update a report from the american heart association. *Circulation*, 133(4):e38–e48, 2016.

[46] T. Nguyen and S. Sanner. Algorithms for direct 0–1 loss optimization in binary classification. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1085–1093, 2013.

[47] M.-I. Nicolae, M. Sinn, M. N. Tran, B. Buesser, A. Rawat, M. Wistuba, V. Zantedeschi, N. Baracaldo, B. Chen, H. Ludwig, et al. Adversarial robustness toolbox v1. 0.0. *arXiv preprint arXiv:1807.01069*, 2018.

[48] P. Panda, I. Chakraborty, and K. Roy. Discretization based solutions for secure machine learning against adversarial attacks. *IEEE Access*, 7:70157–70168, 2019.

[49] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu. Improving adversarial robustness via promoting ensemble diversity. In *Proceedings of International Conference on Machine Learning*, pages 4970–4979, 2019.

[50] N. Papernot, P. McDaniel, and I. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[51] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[52] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Proceedings of IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387, 2016.

[53] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NeurIPS-W*, 2017.

[54] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[56] C. Qian and X. Yang. An integrated method for atherosclerotic carotid plaque segmentation in ultrasound image. *Computer Methods and Programs in Biomedicine*, 153:19–32, 2018.

[57] A. Raghunathan, S. M. Xie, F. Yang, J. C. Duchi, and P. Liang. Adversarial training can hurt generalization. In *Identifying and Understanding Deep Learning Phenomena ICML Workshop*, 2019.

[58] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Preceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[59] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[60] S.-S. Shai, O. Shamir, and K. Sridharan. Learning linear and kernel predictors with the 0-1 loss function. *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, 22(3), 2011.

[61] A. Sinha, H. Namkoong, and J. Duchi. Certifiable distributional robustness with principled adversarial training. In *International Conference on Learning Representations*, 2018.

[62] E. Smistad and L. Løvstakken. Vessel detection in ultrasound images using deep convolutional neural networks. In *Proceedings of Deep Learning and Data Labeling for Medical Applications*, pages 30–38. 2016.

[63] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *Proceedings of IEEE International Joint Conference on Neural Networks*, pages 1453–1460, 2011.

[64] J. H. Stein, C. E. Korcarz, R. T. Hurst, E. Lonn, C. B. Kendall, E. R. Mohler, S. S. Najjar, C. M. Rembold, and W. S. Post. Use of carotid ultrasound to identify subclinical vascular disease and evaluate cardiovascular disease risk: a consensus statement from the american society of echocardiography carotid intima-media thickness task force endorsed by the society for vascular medicine. *Journal of the American Society of Echocardiography*, 21(2):93–111, 2008.

[65] T. Strauss, M. Hanselmann, A. Junginger, and H. Ulmer. Ensemble methods as a defense to adversarial perturbations against deep neural networks. *arXiv preprint arXiv:1709.03423*, 2017.

[66] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[67] A. Taki, Z. Najafi, A. Roodaki, S. K. Setarehdan, R. A. Zoroofi, A. Konig, and N. Navab. Automatic segmentation of calcified plaques and vessel borders in ivus images. *International Journal of Computer Assisted Radiology and Surgery*, 3(3-4):347–354, 2008.

[68] P. Tamimi-Sarnikowski, A. Brink-Kjær, R. Moshavegh, and J. A. Jensen. Automatic segmentation of vessels in in-vivo ultrasound scans. In *Proceedings of Medical Imaging: Biomedical Applications in Molecular, Structural, and Functional Imaging*, volume 10137, page 101371P, 2017.

[69] G. Tetteh, V. Efremov, N. D. Forkert, M. Schneider, J. Kirschke, B. Weber, C. Zimmer, M. Piraud, and B. H. Menze. Deepvesselnet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-d angiographic volumes. *arXiv preprint arXiv:1803.09340*, 2018.

[70] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. D. McDaniel. Ensemble adversarial training: Attacks and defenses. In *Proceedings of 6th International Conference on Learning Representations, ICLR*, 2018.

[71] D. Vukadinovic, T. van Walsum, S. Rozie, T. de Weert, R. Manniesing, A. van der Lugt, and W. Niessen. Carotid artery segmentation and plaque quantification in cta. In *Proceedings of IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 835–838, 2009.

[72] L. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, L. Daniel, D. Boning, and I. Dhillon. Towards fast computation of certified robustness for relu networks. In *Proceedings of International Conference on Machine Learning*, pages 5276–5285, 2018.

[73] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, and X. Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. *arXiv preprint arXiv:2002.05990*, 2020.

[74] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019.

[75] M. Xie, Y. Li, Y. Xue, R. Shafritz, S. A. Rahimi, J. W. Ady, and U. W. Roshan. Vessel lumen segmentation in internal carotid artery ultrasounds with deep convolutional neural networks. In *Proceedings of IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2393–2398, 2019.

[76] M. Xie, Y. Xue, and U. Roshan. Stochastic coordinate descent for 0/1 loss and its sensitivity to adversarial attacks. In *Proceedings of 18th IEEE International Conference on Machine Learning and Applications - ICMLA 2019*, pages 299–304, 2019.

[77] Y. Xue, M. Xie, F. G. Farhat, O. Boukrina, A. Barrett, J. R. Binder, U. W. Roshan, and W. W. Graves. A fully 3d multi-path convolutional neural network with feature fusion and feature weighting for automatic lesion identification in brain mri images. *Submitted*, 2019.

[78] Y. Xue, M. Xie, and U. Roshan. On the transferability of adversarial examples between convex and 01 loss models. In *IEEE International Conference on Machine Learning and Applications*, 2020.

[79] Y. Xue, M. Xie, and U. Roshan. Towards adversarial robustness with 01 loss neural networks. In *IEEE International Conference on Machine Learning and Applications*, 2020.

[80] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of International Conference on Machine Learning*, pages 7472–7482, 2019.

[81] Q. Zhang, H. Qiao, J. Dou, B. Sui, X. Zhao, Z. Chen, Y. Wang, S. Chen, M. Lin, B. Chiu, et al. Plaque components segmentation in carotid artery on simultaneous non-contrast angiography and intraplaque hemorrhage imaging using machine learning. *Magnetic Resonance Imaging*, 60:93–100, 2019.

[82] R. Zhou, A. Fenster, Y. Xia, J. D. Spence, and M. Ding. Deep learning-based carotid media-adventitia and lumen-intima boundary segmentation from three-dimensional ultrasound images. *Medical Physics*, 2019.

[83] R. Zhou, W. Ma, A. Fenster, and M. Ding. U-net based automatic carotid plaque segmentation from 3d ultrasound images. In *Medical Imaging 2019: Computer-Aided Diagnosis*, volume 10950, page 109504F. International Society for Optics and Photonics, 2019.

[84] A. P. Zijdenbos, B. M. Dawant, R. A. Margolin, and A. C. Palmer. Morphometric analysis of white matter lesions in mr images: method and validation. *IEEE Transactions on Medical Imaging*, 13(4):716–724, 1994.