Adversarial Evasion Attacks to Deep Neural Networks in ECR Models

Shota Nemoto[1], Subhash Rajapaksha[2] and Despoina Perouli[2]
[1]*Case Western Reserve University, 10900 Euclid Avenue, Cleveland, Ohio, U.S.A.*
[2]*Marquette University, 1250 West Wisconsin Avenue, Milwaukee, Wisconsin, U.S.A.*

Keywords: Neural Networks, Adversarial Examples, Evasion Attacks, Security, Electrocardiogram, ECR.

Abstract: Evasion attacks produce adversarial examples by adding human imperceptible perturbations and causing a machine learning model to label the input incorrectly. These black box attacks do not require knowledge of the internal workings of the model or access to inputs. Although such adversarial attacks have been shown to be successful in image classification problems, they have not been adequately explored in health care models. In this paper, we produce adversarial examples based on successful algorithms in the literature and attack a deep neural network that classifies heart rhythms in electrocardiograms (ECGs). Several batches of adversarial examples were produced, with each batch having a different limit on the number of queries. The adversarial ECGs with the median distance to their original counterparts were found to have slight but noticeable perturbations when compared side-by-side with the original. However, the adversarial ECGs with the minimum distance in the batches were practically indistinguishable from the originals.

1 INTRODUCTION

Machine learning and neural networks in particular are capable of approximating complex functions, which allows them to accomplish traditionally difficult tasks in fields such as natural language processing and computer vision. Additional advances such as the rectified linear activation function and residual networks allow much deeper and more complex networks to be trained by helping to avoid the problem of vanishing gradients and slow training processes. In healthcare, neural networks could be used to diagnose diseases in patients bringing more automation in checkups. Ideally, they can help save time while reducing monetary costs related to the number of medical personnel required to examine test results.

Before widespread use of machine learning techniques in healthcare, security concerns must be addressed as a number of sophisticated attacks on neural networks are being produced. For example, dataset poisoning refers to attacks where an adversary tampers with the training dataset in order to compromise the final model's performance or even add a "backdoor" into a network (Yao et al., 2019). The backdoor is a trigger that forces the network to always make a certain decision when it is present. However, these attacks require the adversary to have access to the training data of the model. In terms of privacy, certain attacks can attempt to obtain data from the training set using the model's output predictions, or infer a particular property about the entire training set. These attacks are known as model inversion attacks (Yang et al., 2019). Finally, more practical attacks can take a correctly classified input and add human imperceptible perturbations that alter the label the model ends up applying to the input. These attacks are known as evasion attacks and the inputs with imperceptible perturbations added are known as adversarial examples.

The practicality of evasion attacks comes from recent studies on adversarial examples which demonstrate that it is possible to perform a black-box attack on neural networks. This means that no tampering needs to be done to the training data, and no information about the training data is necessary at all. The adversary also does not need to know the internals of the model's architecture, such as how many layers there are, if there are pooling convolutional layers, if there are residual blocks, etc. Black-box attacks only require the final output scores of the model or the final decision that was made.

In this paper, we examine whether a black-box evasion attack could successfully create an adversarial example to a neural network intended for use in healthcare. The evasion attack algorithm is HopSkipJumpAttack (Chen et al., 2020) and the chosen

Nemoto, S., Rajapaksha, S. and Perouli, D.
Adversarial Evasion Attacks to Deep Neural Networks in ECR Models.
DOI: 10.5220/0010848700003123
In *Proceedings of the 15th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2022) - Volume 5: HEALTHINF*, pages 135-141
ISBN: 978-989-758-552-4; ISSN: 2184-4305

network to be attacked is a deep neural network designed to classify heart rhythms using electrocardiograms (Hannun et al., 2019).

2 RELATED WORK

Adversarial examples, initially introduced in the context of computer vision, are images or other input vectors containing perturbations that alter the label assigned to them by a target classifier from their true label. These perturbations do not alter a human's original classification of the image and are often considered imperceptible. Introduced in 2014 (Szegedy et al., 2014), a number of different methods for creating adversarial examples have arisen in recent years. The original attack used the L-BFGS optimization method to minimize a cost function. This cost function represents the distance of the adversarial example to the original input vector and if its outputted label differs from the true label. The L-BFGS optimization method requires the ability to calculate the gradient of the cost function, or how much the cost function changes with respect to each element of the input vector. An advantage of the L-BFGS method is that it does not require calculations of the second derivative, or the Hessian matrix, of the cost function. Methods like this are known as quasi-Newton methods and can save a large number of calculations.

A second, faster attack known as the Fast Gradient Sign Method was introduced the following year (Goodfellow et al., 2015). It only used the sign from the gradient and a chosen step size to update the adversarial image. These first adversarial examples are known as white-box attacks and require full knowledge of the internal workings of the network. The practicality of these attacks is limited since they require knowledge of the derivative of the cost function with respect to each input and thus full knowledge of the network architecture.

The core issue that research on black-box attacks addresses is the estimation of a network's gradient from only the input and output vectors. One of the first black-box attacks (Papernot et al., 2017), where the adversary has no knowledge of the internals of the network, creates a substitute network using a training set of images labelled by the target network. Then, white-box adversarial attacks are used to generate adversarial examples on the substitute network. These examples have been found to be capable of fooling the target network, thus proving the viability of transfer attacks.

Another black-box adversarial attack relies on the scores or probabilities the model assigns to the input image. The attack (Narodytska and Kasiviswanathan, 2017) uses the scores to numerically approximate the gradient of the network, then finds a subset of pixels to perturb in order to place the adversarial example in one of the network's "blind spots". However, this can also be thought of as a partial knowledge attack, since the adversary may not always get access to the full list of probabilities and scores for inputs, but only to the final decision.

A more recent class of attacks are decision-based adversarial attacks, which rely solely on the final output or the highest probability labels predicted by the classifier. These are the most practical attacks, as most publicly available classifiers will only give users a single, final decision. One decision-based attack known as Boundary Attack (Brendel et al., 2018) starts with a large adversarial perturbation. This perturbation is then minimized while still remaining adversarial, essentially estimating the location of the boundary between an adversarial input and a correctly labelled input, then finding the closest point on that boundary to the original image.

Building off of Boundary Attack, Chen et al. (Chen et al., 2020) introduced an improvement to boundary attack that uses a new technique to estimate the gradient and requires fewer queries to the model. This attack was named Boundary Attack++ or HopSkipJumpAttack. A reduction in the number of queries is important as publicly available models may have some cost associated with each query, such as a time or monetary cost. Thus, practical evasion attacks in the future will likely need to reduce the required number of queries as much as possible or else they reduce their probability of success.

Another aspect of these evasion attacks is that they have all mostly been tested in the computer vision field. Very little research on evasion attacks has attempted to attack models unrelated to image recognition. One study (Zhao et al., 2019) applied adversarial examples to object recognition and found that, while the attack was successful, object detectors posed an extra challenge. Object detectors had to accomplish two tasks: predicting the existence of an object as well as the label of the object. Their inputs were also typically video feeds instead of image vectors, so constantly changing backgrounds, distances, viewing angles, and illumination added to the difficulty of creating adversarial examples. This seems to imply that it may not be a given that all neural networks are vulnerable to evasion attacks. It is possible that some applications of neural networks may be naturally more robust to adversarial examples. This paper seeks to investigate whether ECG models are vulnerable to evasion attacks.

3 HOP SKIP JUMP ATTACK

Our goal is to implement an evasion attack algorithm called HopSkipJumpAttack (Chen et al., 2020) and apply it to a deep neural network developed to classify electrocardiograms. We then evaluate the success of the resulting adversarial examples by measuring their distances to the original electrocardiograms. In this section we summarize the HopSkipJumpAttack algorithm (Chen et al., 2020).

HopSkipJumpAttack focuses on being query efficient, as accessing publicly available models might have some cost associated with each query. The cost could be monetary, time, risk of arousing suspicion, etc. At its core, HopSkipJumpAttack follows the major steps listed below and illustrated in Figure 1.

1. An adversarial example is initialized using a sample image from the target class. The class must be different from the original, else the solution will be trivial.
2. The boundary between adversarial images and correctly classified images is then estimated using binary search on a spectrum of blended images of the original and current adversarial images.
3. The gradient at the boundary is estimated using a weighted sum of random perturbations.
4. A step size for the current iteration is calculated, and a perturbation is added to the current adversarial example using the estimated gradient direction and the step size.
5. The process is repeated using the current adversarial example in the binary search.

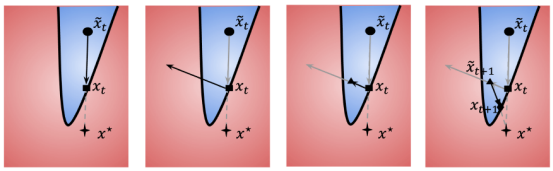

Figure 1: Visualization of Hop Skip Jump Attack in a 2D space as shown in (Chen et al., 2020). The blue region is the space where an image is given the adversarial label by the classifier. The red region is where an image is given any non-adversarial label. The first image shows a boundary search between the current adversarial example and the original image. The second image shows the gradient estimate at the boundary. The third image shows an appropriate step size being calculated. The fourth image shows the boundary search for the next iteration.

Throughout HopSkipJumpAttack, an indicator function is used. This indicator function will take an input image, then output 1 if the adversary's desired outcome is achieved and 0 otherwise. If the attack is untargeted, the desired outcome is for the image to be classified as any class other than the correct one. If the attack is targeted, the desired outcome is for the image to be classified as the target class.

Estimating the boundary uses two images, one that is classified as the target class and the original image. Blending these, a spectrum between the two is created, where images have varying proportions of the adversarial and original image specified by a parameter $\alpha \in [0, 1]$. When using L_2 distance, the images on the spectrum can be created using the following equation:

$$\alpha x^* + (1-\alpha)x \tag{1}$$

The equation can be thought of as projecting a point x onto a sphere of radius α centered at x^*. In this case, x and x^* are input vectors to a neural network.

A region on this spectrum is classified as the original class, and a region on this spectrum is classified as the target class. The boundary between these two regions is estimated using binary search. The boundary image is taken as the current adversarial image.

At the boundary, a batch of random unit vectors is sampled. These unit vectors are individually added to the adversarial image creating new adversarial images. Then, an indicator function is evaluated for each of these perturbed images. The size of these random perturbations is a function of the dimension of the space and the distance of the current image to the original. The average of these indicator values is then saved as the baseline value. A normalizing constant proportional to the batch size is also calculated. For each vector, a coefficient is calculated by subtracting the baseline from their indicator value. This is done in order to reduce the variance in the estimate. Then, a weighted sum of the unit vectors is taken, with their corresponding coefficients as their weighting. Finally, the normalizing constant is applied.

The step size is designed to become smaller as more iterations are completed in order to prevent overshooting the minimum. It is also a function of the distance of the current adversarial image to the original image. However, if the calculated step size yields a non- adversarial image, the step size is divided by factors of 2 until it does produce an adversarial image. The function to calculate the step size is shown below:

$$\xi_t = \frac{||x_t - x^*||_p}{\sqrt{t}} \tag{2}$$

where p is the norm order (2 or infinite).

4 ELECTROCARDIODIAGRAM MODEL

Electrocardiodiagrams (ECGs) are recordings of the electrical signals in a person's heart. They are used to detect abnormal heart rhythms, known as arrythmias, in patients. Hannun et al (Hannun et al., 2019) developed a deep neural network (DNN) for classifying 10 different classes of arrythmias, normal sinus rhythms, and noise from these ECGs. This model achieved an area under the Receiver Operating Characteristic (ROC) curve of 0.97 for all but one class, and an area of 0.91 for the last class. The model makes these classifications using only the raw ECG data, and does not use any other patient information. The model architecture consists of 16 residual blocks with two convolutional layers per block. The residual blocks help speed up the training and optimization process for such a deep network.

To demonstrate the generalizability of their network, Hannun et al. applied their network to the 2017 PhysioNet Computing in Cardiology Challenge dataset, which required classification of ECGs into four different classes:

- Normal Sinus Rhythm
- Atrial Fibrillation
- Other
- Noisy

When applied to the PhysioNet public dataset, the network had a class average F_1 score of 0.83, which was among the best performers in the challenge. The F_1 score measures the test's accuracy by calculating the harmonic mean of its precision ($\frac{numberOfCorrectlyLabelledPositives}{totalNumberOfPositiveLabels}$) and recall ($\frac{numberOfCorrectlyLabelledPositives}{totalNumberOfTruePositiveElements}$).

5 METHODS

We replicated the DNN model for identifying arrythmias in ECGs (Hannun et al., 2019) and applied the HopSkipJumpAttack (Chen et al., 2020) to it. The code for the DNN was pulled from the corresponding repository (https://github.com/awni/ecg), and the dataset of ECGs used for training and evaluation of the DNN were pulled from the PhysioNet 2017 website (https://physionet.org/content/challenge-2017/1.0.0/). The model being attacked had about 91% accuracy. To keep the environment consistent with the environment used for the DNN, the HopSkipJumpAttack was implemented in Python 2.7 on a Linux system (Ubuntu 18.04 on Windows Subsystem for Linux 2). The distance metric used for this paper was the Euclidean norm, or L_2 norm. The initial batch size was chosen to be 100 random unit vectors, generated by sampling a uniform random distribution using the NumPy library. The attack algorithm was also implemented to halt and return the current adversarial ECG, if the model query limit would be exceeded during an iteration.

ECGs for the original and target sample in the HopSkipJumpAttack algorithm were chosen uniformly randomly from the PhysioNet dataset, and selected so that they would have differing labels assigned to them by the DNN. Original and Target pairs were chosen in batches of approximately 100, with each batch having a different query limit. The chosen query limits were 2500, 5000, 10000, and 15000 queries.

6 RESULTS

The distances of the produced adversarial examples to their original counterpart ECGs are shown in Table 1. The trend of distances as the number of queries used increases is shown in Figure 2.

Table 1: L_2 Distances of Adversarial Examples.

Queries	Sample Size	Mean	Median	Min
2500	88	53.3	42.8	6.4
5000	97	53.3	40.0	5.5
10000	19	43.9	24.0	3.8
15000	92	27.0	19.7	1.7

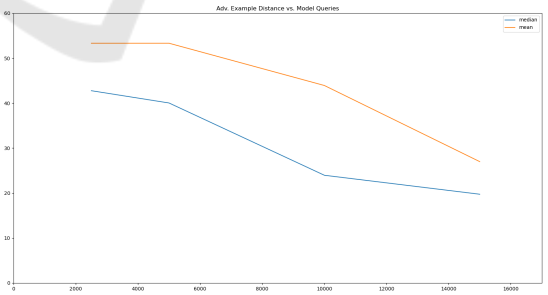

Figure 2: L_2 distances between adversarial examples and original ECGs as a function of model queries. The bottom blue line plots the median distances of each batch. The top orange line plots the mean distances of each batch.

The minimum distance examples produced appear to be practically indistinguishable with the human eye as shown in Figures 3 and 4. The L_2 distance between the original and adversarial ECG is less than 10. The median examples in the batches that used

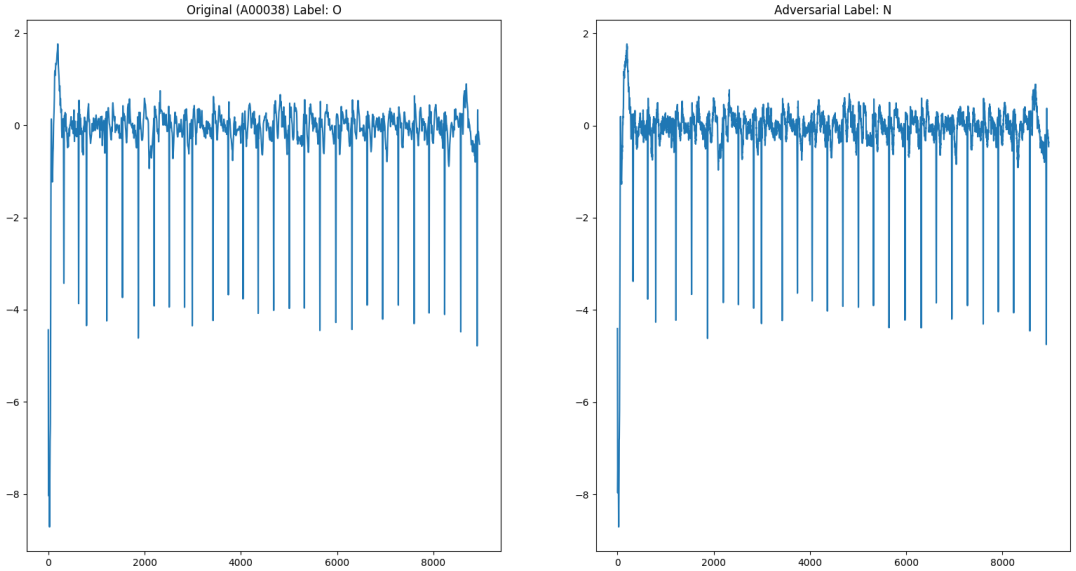

Figure 3: Minimum distance adversarial example for 10000 queries. The original ECG on the left was labeled as "Other Arrhythmia". The adversarial ECG on the right was labeled as "Normal Sinus Rhythm".

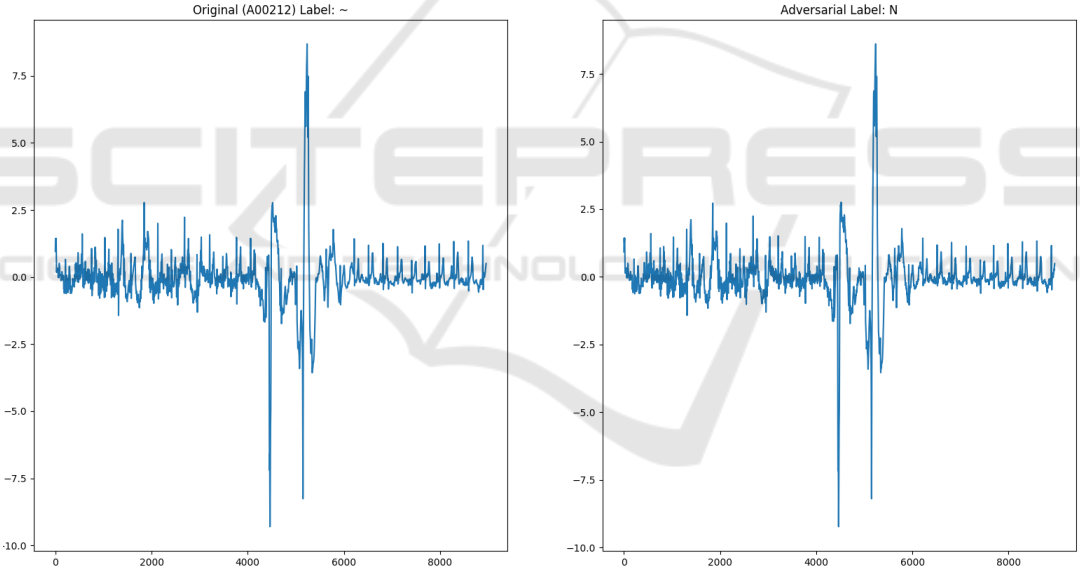

Figure 4: Minimum distance adversarial example for 15000 queries. The original ECG on the left was labeled as "Too Noisy". The adversarial ECG on the right was labeled as "Normal Sinus Rhythm".

2500 and 5000 queries have some noticeable perturbations when compared side-by-side as shown in Figures 3 and 4. In the median cases for the 10000 and 15000 query batches, the noise could be written off as simple noise, and would likely not be easily noticeable unless compared while directly adjacent.

7 CONCLUSION

If no defenses against adversarial attacks are utilized and a model is made publicly available so that making thousands of queries is possible, an evasion attack will be able create an adversarial ECG indistinguishable from an original ECG by a human eye. As for the adversarial examples with L_2 distances greater than 10, it is uncertain if they would be able to go unnoticed if not compared side-by-side with the original. The total

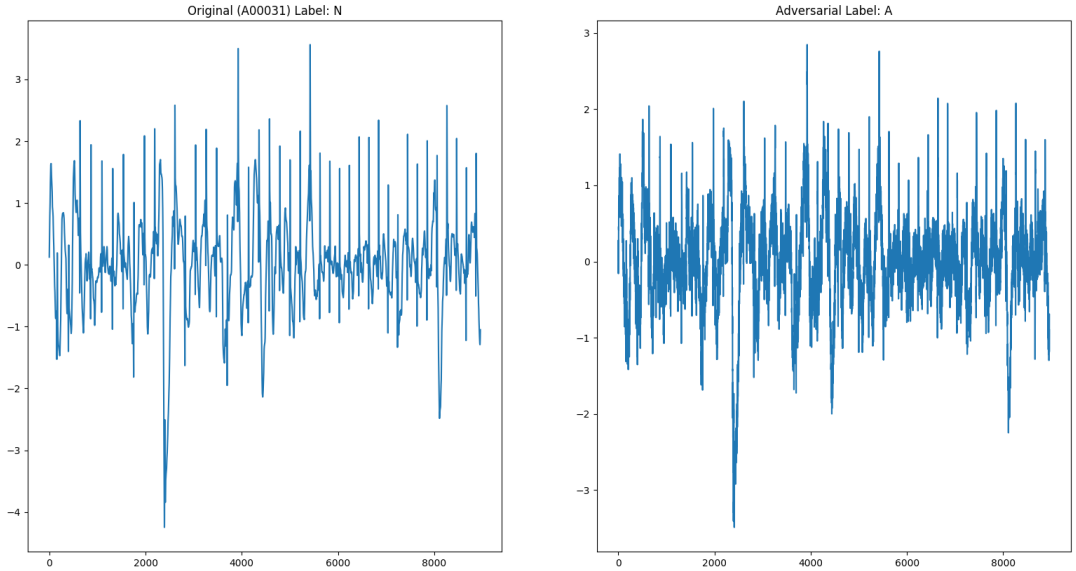

Figure 5: Median distance adversarial example for 10000 queries. The original ECG on the left was labeled as “Normal Sinus Rhythm”. The adversarial ECG on the right was labeled as “Atrial Fibrillation”.

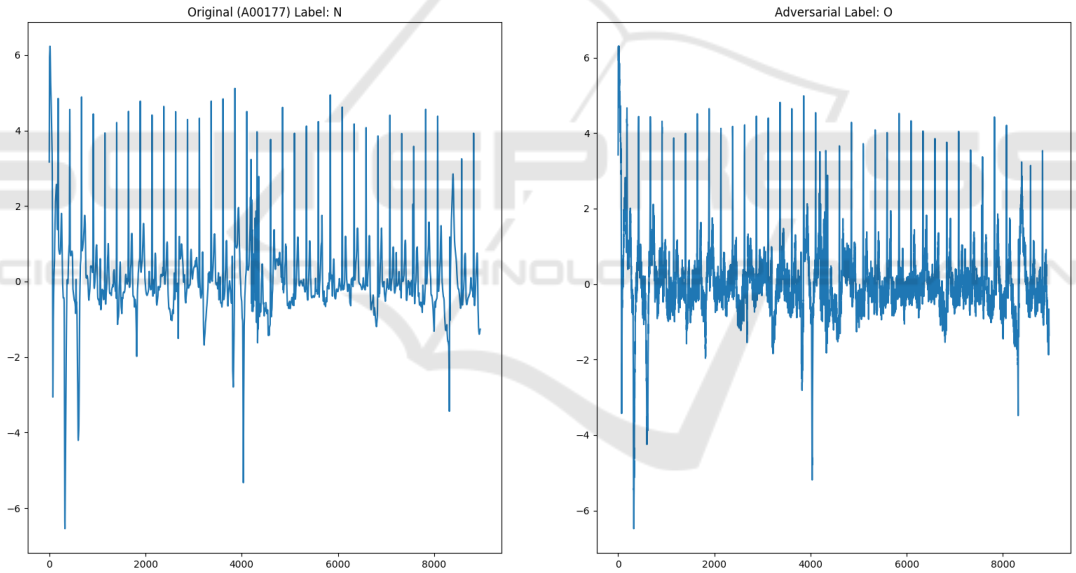

Figure 6: Median distance adversarial example for 15000 queries. The original ECG on the left was labeled as “Normal Sinus Rhythm”. The adversarial ECG on the right was labeled as “Other Arrhythmia”.

results indicate that healthcare models can be vulnerable to evasion attacks. Thus, even if neural networks used in healthcare manage to obtain accuracy, precision, and recall greater than 0.99, they should not be taken as a complete replacement for an opinion from a medical professional until adequate defenses against adversarial attacks, such as evasion attacks, are implemented.

A possible method for defending against evasion attacks would be monitoring or controlling access to the model. HopSkipJumpAttack is very efficient in number of queries, but the experiments run by Chen et al (Chen et al., 2020) show that their attacks against most models required upwards of one thousand queries to generate a single adversarial example. Policies that control access to the model could perhaps require some patient identification and limit the frequency of queries per patient or possibly add a monetary requirement. However, such policies should not become so prohibitive such that patients in need are unable to access the model, as that would defeat the original purpose of making access to

an expert-level diagnosis widely available. This becomes a much larger concern if a monetary requirement is added.

8 FUTURE WORK

It will be helpful to consult a panel of expert cardiologists to evaluate the success of these attacks and determine if they would be noticed by professionals. Additionally, it may be possible that certain classes of arrhythmias are easier to target and create adversarial examples of. In the same vein, certain samples of a target class may serve as a better target sample to initialize the HopSkipJumpAttack algorithm with.

It is also worth investigating possible defenses against black-box evasion attacks. One option would be to add the correctly labelled adversarial examples to the training set in order to reduce the sensitivity of the model to these perturbations. A second option would be to limit the number of model queries permitted per user; even though HopSkipJumpAttack is a query-efficient algorithm, generating human imperceptible adversarial examples still requires thousands of queries. Defense mechanisms should take into account that malicious actors could use multiple user accounts to gain access to bypass protections.

Finally, adversarial attacks such as HopSkipJumpAttack should be applied to more models in healthcare.

ACKNOWLEDGEMENTS

Thanks to Dr. Debbie Perouli for guiding and serving as the mentor for this project. Additional thanks to Dr. Praveen Madiraju and Dr. Dennis Brylow for running the Research Experience for Undergraduates program at Marquette University.

This material is based upon work supported by the National Science Foundation under Grant #1950826.

REFERENCES

Brendel, W., Rauber, J., and Bethge, M. (2018). Decision-based adversarial attacks: Reliable attacks against black-box machine learning models.

Chen, J., Jordan, M. I., and Wainwright, M. J. (2020). Hopskipjumpattack: A query-efficient decision-based attack.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples.

Hannun, A. Y., Rajpurkar, P., Haghpanahi, M., Tison, G. H., Bourn, C., Turakhia, M. P., and Ng, A. Y. (2019). Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature medicine*, 25(1):65–69.

Narodytska, N. and Kasiviswanathan, S. (2017). Simple black-box adversarial attacks on deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1310–1318.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. (2017). Practical black-box attacks against machine learning.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks.

Yang, Z., Zhang, J., Chang, E.-C., and Liang, Z. (2019). Neural network inversion in adversarial setting via background knowledge alignment. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 225–240, New York, NY, USA. Association for Computing Machinery.

Yao, Y., Li, H., Zheng, H., and Zhao, B. Y. (2019). Latent backdoor attacks on deep neural networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2041–2055, New York, NY, USA. Association for Computing Machinery.

Zhao, Y., Zhu, H., Liang, R., Shen, Q., Zhang, S., and Chen, K. (2019). Seeing isn't believing: Towards more robust adversarial attack against real world object detectors. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 1989–2004, New York, NY, USA. Association for Computing Machinery.