Enhancing Adversarial Defense in Behavioral Authentication Systems Through Random Projections

Md Morshedul Islam[a] and Md Khairul Anam[b]
Concordia University of Edmonton, Edmonton, AB., Canada

Keywords: Behavioral Authentication System, Adversarial Attack, Random Projection, Random Noise.

Abstract: Behavioral Authentication (BA) systems employ a verification algorithm to verify users based on their behavior patterns. To eliminate the need for a profile database to store the profiles and to enhance the system's performance, the verification algorithm usually trains a Neural Network (NN) classifier on user profiles. However, like other NN applications, the NN-based BA classifiers are also susceptible to adversarial attacks. To defend against such attacks, we employed a method that adds noise to the training data by using Random Projection (RP) and its reverse process. This approach prevents model overfitting and maintains the model's predictions at an expected level. Our technique has also proven effective against attacks based on adversarial examples. We tested our proposed method on two BA systems, achieving the expected classification accuracy. Furthermore, the attacks based on adversarial examples are significantly less effective against BA classifiers trained with noisy data compared to those trained with plain data. Our approach is general and can be applied to other BA systems to protect their classifiers from similar attacks.

1 INTRODUCTION

There has been a significant technological shift over recent years due to the development and widespread use of smartphones. Different studies have shown that using smartphone sensors, it is possible to collect users' behavioral data, such as touch patterns, keystroke dynamics, and even gait patterns. This collected data can be utilized by Behavioral Authentication (BA) systems (Gupta et al., 2020; Islam and Safavi-Naini, 2020) to verify who is using the device. BA systems are beneficial not only for verifying user identity but also for enhancing the security of systems that use multiple factors for authentication (Ding et al., 2019). One of the key advantage of BA systems is their ability to prevent users from sharing their credentials.

Every BA system incorporates a verifier, usually an online server, that uses a verification algorithm to authenticate users. During the registration phase, the verification algorithm trains a machine learning (ML) classifier with the users' behavioral profiles where a profile is a collection of a user's behavioral data. When a user attempts to log in, the verification algorithm employs this ML classifier to determine if the user's verification profile matches. Recently, Neural Network (NN) based classifiers have been increasingly used to classify various types of profile data, such as mouse movements (Chong et al., 2019), gaits (Jung et al., 2019), and keystrokes (Deng and Zhong, 2015). These NN-based classifiers are designed with data privacy in mind, utilizing techniques to reduce the necessity for long-term storage of sensitive raw data in databases. This approach not only enhances data privacy but also boosts the overall system performance. Additionally, this strategy is also beneficial for continuous authentication (Meng et al., 2014), which keeps verifying a user's identity continuously.

Like other ML models, the NN classifiers of the BA systems can be vulnerable to certain types of misuse, with adversarial examples being one of the most recent and significant threats. Adversarial examples involve making minor adjustments to the input data, such as images, audio, and video, which are crafted to lead highly accurate ML models to confidently make incorrect predictions (Szegedy et al., 2013; Jin et al., 2021; Pacheco and Sun, 2021). These modifications, often imperceptible to humans, can trick the ML models into incorrectly classifying the altered inputs. Therefore, developing robust defense strategies is crucial, especially for the BA classifiers that

[a] https://orcid.org/0000-0002-4215-3174
[b] https://orcid.org/0009-0000-6580-3961

Islam, M. and Anam, M.
Enhancing Adversarial Defense in Behavioral Authentication Systems Through Random Projections.
DOI: 10.5220/0012792800003767
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 21st International Conference on Security and Cryptography (SECRYPT 2024)*, pages 758-763
ISBN: 978-989-758-709-2; ISSN: 2184-7711
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

authenticate users based on their behavioral data.

Research on defending against adversarial attacks is actively exploring diverse strategies to make ML models less prone to these threats. Among them, adversarial training stands out as a crucial technique that involves incorporating adversarial examples into the training dataset, which helps the model learn to identify and deal with these crafted inputs. For example, utilizing Projected Gradient Descent (PGD) (Madry et al., 2017) in adversarial training significantly improves model resilience by systematically introducing adversarial examples generated via PGD. While this method has shown promise, studies indicate it might still be susceptible to other forms of attacks. Additionally, models generating adversarial examples often rely on specific datasets, which raises questions about their applicability across different settings and effectiveness in real-world situations. The network layer augmentation approach proposed in (Yu et al., 2019) enhances the original model by adding multiple auxiliary blocks, akin to a self-ensemble model. However, this method increases computational complexity, making it less practical for devices with limited processing capabilities. In (Byun et al., 2022), the authors advocate for the inclusion of random noise during training to fine-tune the model's parameters, thus mitigating overfitting and enhancing the generalization capabilities of the model. Furthermore, (Dong et al., 2020) highlights that randomization defenses tend to outperform other defense strategies, leading to a growing interest in this field. However, random noise can introduce uncertainty that might reduce model accuracy. Moreover, while most adversarial defense techniques have been developed for computer vision, progress in the behavioral domain has been limited. Nonetheless, behavioral data is crucial not only for user authentication but also for emerging areas like cybersecurity, IoT, and smart cities.

Our Work. We added random noise to the BA profiles during training, aiming to avoid overfitting the model and enhance its generalization capabilities without negatively affecting the system's accuracy. This approach also safeguards the BA classifier against attacks based on adversarial examples. For this, we utilized a technique based on Random Projection (RP) and its inverse. RP is a method that reduces data dimensionality using a random matrix while preserving the pairwise distances among vectors. In contrast, the inverse RP reconstructs the original dimensions of the vectors using a pre-calculated pseudo-inverse matrix, aiming to keep the vectors and their inversely projected versions close to each other. Our strategy involves transforming the BA profiles with RP using a unique sparse random matrix, which serves as a user's secret. We then apply inverse RP to these projected profiles to restore them to their original dimension. These restored profiles, which are close to the original profiles, are considered as noise, thereby not compromising model performance. The closest work to ours is described in (Finlay et al., 2018), where RP and its inverse were utilized to devise regularization parameters to enhance model's resistance against adversarial attacks. We implemented our proposed approach on two existing BA systems to protect their classifier from adversarial attacks.

2 PRELIMINARIES AND RELATED WORKS

BA System. In a BA system, a profile is a collection of m vectors, each with d dimensions, represented as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m\}$. Each dimension represents a feature, and each vector is the measurement of behavioral activity. During registration, the BA system collects N profiles to train a NN classifier $\mathtt{C}(\theta)$, where θ represents the trained network parameters. We aim to train a new classifier $\mathtt{C}(\hat{\theta})$ using noisy profiles to protect against adversarial attacks. A verification request consists of an identity u, and a verification profile $\mathbf{Y}$ which is a collection of n (where $n < m$) behavioral samples. The verification process uses the trained NN classifier to produce prediction vectors that assist in making authentication decisions. A BA system must ensure accuracy and security.

Adversarial Examples. Various algorithms exist for creating adversarial examples in different settings. One such method is the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014), which enables gradient-based attacks by calculating the gradients of loss relative to model inputs and using them to alter input samples toward misclassification. Another technique is the Jacobian-based Saliency Map Attack (JSMA) algorithm (Papernot et al., 2016), which determines the effect of altering specific features on the classification outcome, targeting the most influential features for perturbation. While other algorithms for generating adversarial examples exist, we have opted for FGSM and JSMA due to their proven effectiveness in compromising the robustness of targeted ML models. Considering additional algorithms for future research are planned.

Adversarial Defense. The study by (Dong et al., 2020) highlights that adding random noise serves as an effective defense against adversarial attacks. The Random Self-Ensemble method in (Liu et al., 2018) uses a randomization strategy by injecting Gaussian noise into the input of each convolutional layer, en-

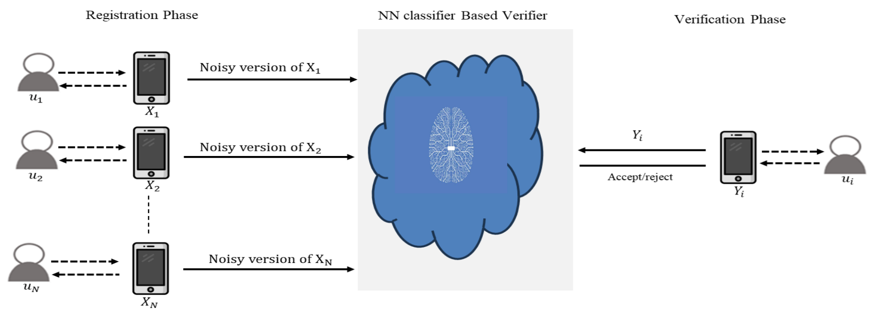


Figure 1: The proposed BA system is robust against attacks based on adversarial examples. In the registration phase, the BA system trains a NN classifier $\mathtt{C}(\hat{\theta})$ using noisy profiles. In the verification phase, the BA verification algorithm decides to accept or reject a verification request based on the outputs of $\mathtt{C}(\hat{\theta})$.

hancing stability through an ensemble of predictions for each image. The Parametric Noise Injection approach in (He et al., 2019) increases NN resilience by adding trainable Gaussian noise to the activations or weights of layers. The technique in (Xie et al., 2017) introduces a random input transformation method that applies random resizing and padding to the input image prior to network analysis, thus introducing noisy gradients to stop adversaries. While this method can be straightforwardly applied to pre-trained models, it requires an increase in computational effort due to the input size expansion. In our case, we generated random noise using RP and inverse RP.

Random Projection (RP). RP is a projection method that uses a function $P(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^k$ to map vectors $\mathbf{x}$ from a higher-dimensional space d to $\mathbf{x}'$ to a low-dimensional space k, through a random matrix $\mathbf{R}$ as $\mathbf{x}' = \frac{1}{\sqrt{k}\sigma_r}\mathbf{R}^T\mathbf{x}$. Here, σ_r denotes the standard deviation of $\mathbf{R}$'s entries and $\mathbf{R}^T$ is the transpose of $\mathbf{R}$. When projecting a profile $\mathbf{X}$, this is denoted as $\mathbf{X}' = \mathbf{R}^T\mathbf{X}$, assuming $\mathbf{X}$ and $\mathbf{X}'$ are collections of vectors. Inverse projection $P^{-1}(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^d$, aims to reverse this mapping. Ideally, applying P^{-1} to $P(\mathbf{x})$ would return the original vector $\mathbf{x}$. However, in practice, the recovered vector $\hat{\mathbf{x}}$ and the original $\mathbf{x}$ are not identical, but rather closely approximate each other. This concept of approximation also applies to sets of vectors, where each vector $\mathbf{x}$ in set $\mathbf{X}$ and its approximate $\hat{\mathbf{x}}$ in set $\hat{\mathbf{X}}$ closely resemble each other.

3 A ROBUST BA SYSTEM

The proposed BA system is outlined in Figure 1. The registration phase comprises four key activities:

Generating Profiles and Random Matrices: The user's device collects behavioral data from the user's activities to create a profile $\mathbf{X}$ and produces a set of random matrices $\{\mathbf{R}_1, \mathbf{R}_2, \cdots, \mathbf{R}_i\}$ to be used in RP.

Performing Random Projection: The device generates a series of projected profiles $\mathbf{X}'_1 = \mathbf{R}_1\mathbf{X}, \mathbf{X}'_2 = \mathbf{R}_2\mathbf{X}, \cdots, \mathbf{X}'_i = \mathbf{R}_i\mathbf{X}$ by applying RP on $\mathbf{X}$ using generated secret random matrices. This process uses a Lipschitz mapping to transition profile vectors from d to $k_1, k_2, \cdots, k_i$ dimension, respectively.

Inverse Random Projection: The device uses inverse RP to project $\{\mathbf{X}'_1, \mathbf{X}'_2, \cdots, \mathbf{X}'_i\}$ to $\{\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \cdots, \hat{\mathbf{X}}_i\}$, treats all of them as noise, and adds them with $\mathbf{X}$ before sending the data to the verifier.

Training a NN Classifier: The verifier, potentially a third-party service provider offering MLaaS, uses noisy profiles from N users to train $\mathtt{C}(\hat{\theta})$ and keep model accuracy above a threshold.

During the verification phase, $\mathtt{C}(\hat{\theta})$ generates m prediction vectors corresponding to the m vectors of $\mathbf{Y}$. The verifier then combines these predictions into a single binary decision (0 to deny and 1 to approve) and returns this decision back to the user.

Adversarial Assumptions. We assume that an attacker has the following knowledge and capabilities: (i) The attacker can access the plain profiles collected by intercepting the communication. (ii) The attacker knows the architecture and input-output dimensions of the trained BA classifier. Attackers can use this information and collected profiles to train a classifier similar to $\mathtt{C}(\theta)$ to generate the adversarial examples. (iii) The attacker knows the distribution and dimensions of $\mathbf{R}$, as this is public information. However, since $\mathbf{R}$ is secret, the attacker cannot train a BA classifier similar to $\mathtt{C}(\hat{\theta})$ to generate adversarial examples.

4 EXPERIMENTAL RESULTS

We applied and tested our method using voice (Gupta et al., 2020) and drawing pattern (Islam and Safavi-Naini, 2020) data. We downloaded all profiles, prepared them for experiments by cleaning, and normalizing. All voice profiles include information from 86 users, each have 120 records and 240 features. The

Table 1: The second and third columns display the accuracy of $\mathtt{C}(\theta)$ and $\mathtt{C}(\hat{\theta})$, trained on regular and noisy profiles, respectively. The final two columns reveal how well adversarial examples performed against both $\mathtt{C}(\theta)$ and $\mathtt{C}(\hat{\theta})$. In both scenarios, the results met our expectations, indicating that the classifier trained on noisy profiles showed improved resistance to adversarial examples, as anticipated.

Data Set	**Accuracy of $\mathtt{C}(\theta)$**		**Accuracy of $\mathtt{C}(\hat{\theta})$**		**Success of FGSM**		**Success of JSMA**	
	Training	**Validation**	**Training**	**Validation**	**On $\mathtt{C}(\theta)$**	**On $\mathtt{C}(\hat{\theta})$**	**On $\mathtt{C}(\theta)$**	**On $\mathtt{C}(\hat{\theta})$**
Voice	98.35	93.85	99.03	99.69	99.08	1.28	99.12	1.77
Drawing	75.41	81.21	98.37	99.88	83.37	1.07	83.46	9.27

drawing profile contains 80 to 240 records with 65 features, totaling 193 users. We used FGSM and JSMA algorithms to generate adversarial examples. Below are the details of our experiments:

Experiment 1: Building and Training $\mathtt{C}(\theta)$. For this experiment, we developed and trained two NN-based BA classifiers using two groups of plain profiles. Both classifiers incorporate dense, batch-normalization, activation (ReLU), and dropout layers, with a softmax function layer. We allocated 80% of the data from each profile for training and the remaining 20% for validation. For both models, we used the Adam optimizer and trained the model for 100 epochs. The results, shown in the second column of Table 1, indicate that the voice data classifier achieved 98.35% training and 93.85% validation accuracy, while the drawing data classifier reached 75.41% training and 81.21% validation accuracy. Our results for the drawing data are slightly lower than those reported in the original study, possibly because we did not perform data oversampling before training, as was done in the original study.

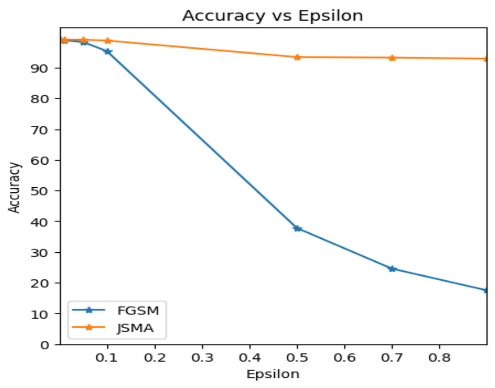
Experiment 2: Generating Adversarial Examples. We generated adversarial examples using FGSM and JSMA algorithms assuming that the attacker could access the training profiles. We also hyper-tuned the epsilon value. Figure 2 (a, b) illustrates how both classifiers' performance varies with different epsilon values. For the voice data, the adversarial examples achieved their highest success rate, 99.08% for FGSM and 99.12% for JSMA, at fooling the NN classifier for epsilon equal to 0.001. However, these success rates decrease at varying rates with epsilon values. Similarly, for drawing data, the highest success rate of 83.37% for FGSM and 83.46% for JSMA was observed with an epsilon of 0.0001, which also decreased with the increase of epsilon. These optimal epsilon settings are used for generating adversarial examples in subsequent experiments. The first component of the third and fourth columns of Table 1 displays how these adversarial examples performed against $\mathtt{C}(\theta)$ at their optimal epsilon settings. The generated adversarial examples effectively fooled both $\mathtt{C}(\theta)$ due to their inability to robustly generalize from the training data, despite having higher classification accuracy.

Experiment 3: Add Random Noise. To apply RP, we created multiple matrices **R** with values of +1, 0, and -1. Using the Johnson-Lindenstrauss (JL) Lemma (Dasgupta and Gupta, 2003), we determined the minimum dimension k_0 necessary for our random matrices $\mathbf{R}^{k\times d}$ to preserve distances within a 0.99 probability. We found that for voice data and drawing data, k needs to be at least 73 and 46, respectively (see Table 2 for detailed calculation along with the values of other parameters of the lemma). Therefore, to ensure the effectiveness of distance preservation, we selected $k = \{75, 80, 85, 90, 95, 100\}$ for voice data and $k = \{50, 52, 54, 56, 58, 60\}$ for drawing data. We then added noise to the profiles using these matrices. Each operation injecting noise proportionally to the profile size. This method introduced a substantial amount of noise to each profile, thereby affecting both the model's performance and the efficiency of potential attacks. The implications of this noise addition on the performance of attacks are further examined in the fifth experiment.

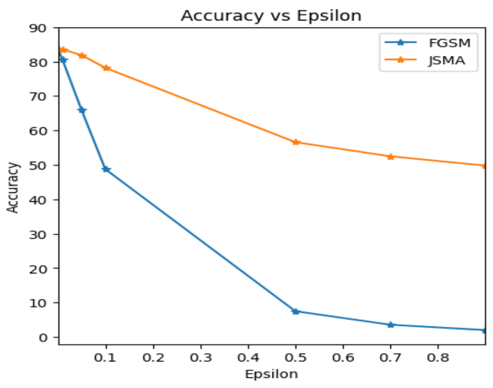
Experiment 4: Building and Training $\mathtt{C}(\hat{\theta})$. We allocated 80% of the noisy data for training $\mathtt{C}(\hat{\theta})$ and the remaining 20% for validation. No architectural modifications were needed for $\mathtt{C}(\hat{\theta})$ and and we used the same network that was designed for $\mathtt{C}(\theta)$. During training, both models reached their acceptable accuracy within just 10 epochs, with the voice data model achieving 99.03% training and 99.69% validation accuracy, and the drawing data model reaching 98.27% training and 99.88% validation accuracy, as detailed in the second column of Table 1. In both cases, incorporating noise into the profiles paradoxically improved models' performances. This improvement is

Table 2: The minimum acceptable value of k in RP, as calculated by the Johnson-Lindenstrauss (JL) Lemma for both the voice and drawing datasets, are provided. A detailed description of the symbols used in the lemma is given in (Dasgupta and Gupta, 2003).

Data Set	k_0	n	ε	β	$1-n^{-\beta}$
Voice data	73.0	200	0.5	1.0	0.99
Drawing data	46.0	300	0.7	1.0	0.99

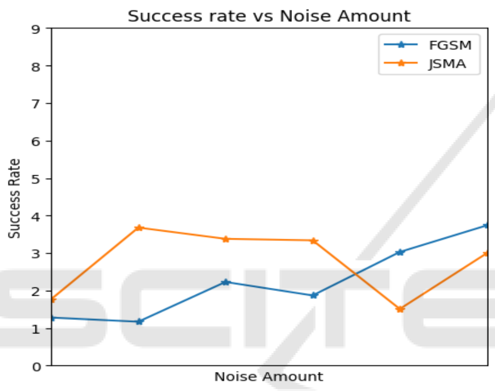


(a) Voice data.

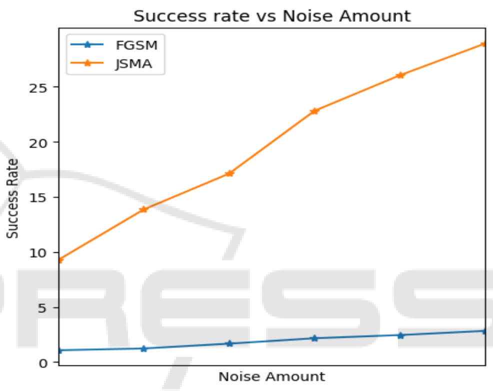


(b) Drawing data.

Figure 2: The performance of FGSM and JSMA algorithms on the classifier $\mathtt{C}(\theta)$, trained with both voice and drawing data, varies with different epsilon values. At the optimal epsilon value, the adversarial attack achieves a success rate of over 99.0% against the voice data and over 83.0% against the drawing data classifier.



(a) Voice data.



(b) Drawing data.

Figure 3: The performance of FGSM and JSMA algorithms on the classifier $\mathtt{C}(\hat{\theta})$, trained with noisy voice and drawing profiles, varies with different noise levels. Introducing noise roughly equal to the profile size resulted in the best defense. However, adding slightly more noise beyond this point did not further improve performance and instead began to compromise the classifier's resilience against both algorithms.

due to generating noise from the profiles and incorporating it, which expands the diversity of the training dataset and increases the models' exposure to a wider range of examples. This, in turn, improved their ability to generalize from the training data to new, unseen data.

Experiment 5: Performance of Adversarial Examples Against $\mathtt{C}(\hat{\theta})$. We evaluated the performance of previously created adversarial examples on $\mathtt{C}(\hat{\theta})$ and observed a significant drop in their effectiveness. For voice data, the success rates ranged only between 1.0% and 4.0% for both FGSM and JSMA algorithms, and for drawing data, they fell between 1.0% and 3.0% for FGSM and 9.0% to 30.0% for the JSMA algorithm. This decrease is due to an increase in models' resilience against adversarial attacks, making it more challenging for attackers to identify precise perturbations. FGSM uniformly alters all input features based on the sign of the gradient and is less effective in both classifiers, where JSMA selectively alters small groups of features based on their calculated importance and is more effective in fooling the drawing classifiers. However, the success rate is still below 10.0% for its best epsilon value.

While adding noise can be beneficial, balancing the amount and type of noise is crucial. Excessive noise can impair the model's performance by obscuring important patterns in the data, whereas the optimal amount can enhance the model's robustness and generalizability. Consequently, we investigated the impact of different amounts of noise on the models' sensitivity to attacks. The results, illustrated in Figure 3 (a, b), showed that adding noise roughly equal to the profile size provided the best defense against attacks. Slightly increasing the volume of noise beyond this optimal amount improved the attackers' chances of success at different rates.

5 CONCLUSION

We generated noises through RP and inverse RP, added them to the BA profiles before using them to train a classifier. This process aimed to increase resistance to attacks based on adversarial examples as well as maintain stable classifier performance. Our approach does not rely on cryptography, thus requiring less computing power, and is suitable for devices with limited processing capabilities. This approach is general and can also be used to protect other behavioral and biometric classifiers. A future improvement of this research work is to compare the performance of our approach with other methods used to avoid adversarial examples. Another future research direction is to analyze the likelihood of adversarial success in deceiving this noisy model, taking into account the attackers' reasonable knowledge and capabilities.

ACKNOWLEDGEMENTS

This research was supported by Concordia University of Edmonton (CUE), Edmonton, AB, Canada, through the Seed Grant program.

REFERENCES

Byun, J., Go, H., and Kim, C. (2022). On the effectiveness of small input noise for defending against query-based black-box attacks. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3051–3060.

Chong, P., Elovici, Y., and Binder, A. (2019). User authentication based on mouse dynamics using deep nn: A comprehensive study. *IEEE Transactions on Information Forensics and Security*, 15:1086–1101.

Dasgupta, S. and Gupta, A. (2003). An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65.

Deng, Y. and Zhong, Y. (2015). Keystroke dynamics advances for mobile devices using deep neural network. *Recent Advances in User Authentication Using Keystroke Dynamics Biometrics*, 2:59–70.

Ding, X., Peng, C., and Ding, e. a. (2019). User identity authentication and identification based on multi-factor behavior features. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE.

Dong, Y., Fu, Q.-A., Yang, X., Pang, T., Su, H., Xiao, Z., and Zhu, J. (2020). Benchmarking adversarial robustness on image classification. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 321–331.

Finlay, C., Oberman, A. M., and Abbasi, B. (2018). Improved robustness to adversarial examples using lipschitz regularization of the loss.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gupta, S., Buriro, A., and Crispo, B. (2020). A chimerical dataset combining physiological and behavioral biometric traits for reliable user authentication on smart devices and ecosystems. *Data in brief*, 28:104924.

He, Z., Rakin, A. S., and Fan, D. (2019). Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 588–597.

Islam, M. M. and Safavi-Naini, R. (2020). Scalable behavioral authentication systems. IEEE Access. manuscript submitted for review.

Jin, W., Li, Y., Xu, H., Wang, Y., Ji, S., Aggarwal, C., and Tang, J. (2021). Adversarial attacks and defenses on graphs. *ACM SIGKDD Explorations Newsletter*, 22(2):19–34.

Jung, D., Nguyen, M. D., Han, J., Park, M., Lee, K., Yoo, S., Kim, J., and Mun, K.-R. (2019). Deep neural network-based gait classification using wearable inertial sensor data. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3624–3628. IEEE.

Liu, X., Cheng, M., Zhang, H., and Hsieh, C.-J. (2018). Towards robust neural networks via random self-ensemble. In *Proceedings of the european conference on computer vision (ECCV)*, pages 369–385.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Meng, W., Wong, D. S., Furnell, S., and Zhou, J. (2014). Surveying the development of biometric user authentication on mobile phones. *IEEE Communications Surveys & Tutorials*, 17(3):1268–1293.

Pacheco, Y. and Sun, W. (2021). Adversarial machine learning: A comparative study on contemporary intrusion detection datasets. In *ICISSP*, pages 160–171.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016). The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. (2017). Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*.

Yu, Y., Yu, P., and Li, W. (2019). Auxblocks: defense adversarial examples via auxiliary blocks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.