Deep Learning vs. Traditional Learning for Radio Frequency Fingerprinting

Andréas OTTO[1], Seani RANANGA[1], Moshe MASONTA[2]
[1]*Data Science for Social Impact, Department of Computer Science, University of Pretoria, South Africa*
Email: u19218525@tuks.co.za, seani.rananga@up.ac.za
[2]*Council for Scientific and Industrial Research, Pretoria, South Africa*
Email: mmasonta@csir.co.za

Abstract: Radio Frequency (RF) fingerprinting is the theory of identifying a wireless device based on its unique transmitting characteristics. RF fingerprinting uses the validated concept that the physical components and configuration of a transmitting device can result in a distinct wireless emission. This research focuses on the application of machine learning algorithms, specifically Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs) for the task of RF fingerprinting. The primary aim of this research paper is to comparatively assess the performance of SVMs and CNNs in RF fingerprinting for wireless device identification, focusing on hyperparameters, accuracy and real-world applicability. The study includes an in-depth implementation and evaluation of the SVMs and CNNs models, considering their performance in a high-dimensional dataset of multiple transmissions and wireless devices. While the CNN model slightly outperformed the SVM in terms of classification accuracy, other metrics such as inference time and training duration made the SVM equally competitive. The high accuracy and competitive inference times affirm the real-world applicability of these models, and their need to be further explored.

Keywords: Radio frequency fingerprinting, Support vector machines, Convolutional neural networks, Physical layer security

1. Introduction

Wireless devices are integral to modern society, serving critical roles in communication and network access. Ensuring secure and authorised use of these devices is paramount. Radio frequency (RF) fingerprinting emerges as a promising technique for enhancing security measures in wireless devices. It not only serves as an additional layer of authentication but also has applications in device localisation, intrusion detection and improved RF spectrum access [1].

1.1 – Radio Frequency Fingerprinting

RF fingerprinting identifies wireless devices based on their unique electromagnetic emissions, which arise from manufacturing inconsistencies in components like power amplifiers and filters. These imperfections lead to unique signal variations, akin to human fingerprints. This is often done by characterising the In-Phase (I) and Quadrature (Q) phase or I/Q offset. The I/Q represents the amplitude and phase of the signal, respectively, and their offset can serve as the unique identifier [2,3]. RF fingerprinting enhances wireless communication security by fortifying authentication mechanisms and strengthening resilience against network attacks. RF fingerprinting ensures that only authorised devices access the network, bolstering overall security measures. This approach aligns with the broader goal of enhancing the security posture of all wireless communication infrastructures,

making them more robust and resistant to emerging threats. By adding this physical layer of security, RF fingerprinting can effectively mitigate threats such as device spoofing [3,4]. The main drive of RF fingerprinting is to add a physical layer of security to wireless communication using improved device identification and authentication [3].

1.2 – Machine Learning in RF Fingerprinting

Supervised machine learning (ML) algorithms have shown efficacy in feature extraction and interpretation for complex tasks like RF fingerprinting. Advances in these algorithms have led to deep learning models, such as Convolutional Neural Networks (CNNs), that mimic neural functions [5,6]. This paper will focus on comparing the performance of CNNs with traditional supervised learning algorithms like Support Vector Machines (SVMs) in the context of RF fingerprinting. The paper addresses the lack of direct comparison between SVMs and CNNs in RF fingerprinting by conducting a comparative analysis. Leveraging a high-dimensional dataset, the paper aims to determine the optimal ML technique for real-world scenarios based on classification accuracy and inference speed, evaluating both SVMs and CNNs with an emphasis on correct and rapid identification.

1.3 – Support Vector Machines

An SVM is a supervised ML algorithm used for classification and regression. They aim to find the hyperplane that best separates two classes by maximising the margin between them. In cases where data cannot be separated linearly, SVMs apply a kernel function to transform it into a higher-dimensional space, enabling separation by a linear hyperplane. SVMs are effective for handling both linearly and nonlinearly separable data [7].

1.4 – Convolutional Neural Network

CNNs represent a class of artificial neural networks comprising convolutional, nonlinear, pooling, and fully connected layers. Within convolutional layers, filters traverse input data, computing a weighted sum via element-wise multiplication with the input's receptive field. These operations are defined by parameters such as stride, filter size, and zero padding. Nonlinear activation functions, like rectified learning unit (ReLU), regulate output characteristics. Pooling layers reduce input dimensionality, typically through operations like max pooling. For multiclass classification tasks such as image recognition, SoftMax layers are often utilised at the output, generating probabilistic class distributions [8].

1.5 – Related Work

1. PARADIS: This system employs five distinct metrics to identify wireless devices using RF fingerprinting. It offers the flexibility of using either SVM or k-NN algorithms for classification. In controlled indoor tests, the SVM-based system achieved an impressively low error rate of 0.0034% [9].
2. I/Q Imbalance SVM: This approach capitalises on the hardware-induced imbalances in quadrature modulation signals to perform RF fingerprinting. When tested on simulated signals, the model demonstrated over 90% accuracy at a signal-to-noise ratio (SNR) of 15dB and higher [10].
3. ORACLE: a CNN-based model that uses raw I/Q samples from a diverse set of over 100 Wi-Fi and 16 software-defined radios. It achieved a median accuracy of 99% for up to 100 devices and 96% for a set of 140 devices, showcasing its robustness [11].
4. A massive experiment: This study compared two CNN architectures, including one based on the ResNet-50 model, across a dataset ranging from 50 to 1 000 devices. The models

showed high accuracy and scalability, with the baseline model performing better on equalised data. The study also noted the impact of channel conditions and SNR on model performance [12].

2. Objectives

Both SVMs and CNNs have individually proven to be effective for the task of RF fingerprinting [9,11]. The primary aim of this research is to comparatively assess the performance of SVMs and CNNs in RF fingerprinting for wireless device identification. The study leverages a high-dimensional dataset and focuses on optimising hyperparameters for both ML models. The overarching goal is to identify the most effective model concerning accuracy, computational time, and inference speed.

2.1 Sub-Objectives:

1. ***Model efficacy***: evaluate the performance of SVM and CNN models in terms of classification accuracy, inference time and training time. This study aims to achieve high accuracy rates, surpassing existing benchmarks in the literature.
2. ***Real-world applicability***: examine the practicality of the models by considering their inference times, thereby evaluating their suitability for real-time applications in security as a service (SaaS) offering.
3. ***Benchmarking and future research***: compare the developed models against existing benchmarks and identify avenues for future research.

3. Methodology

3.1 – Dataset Description

The dataset used in this study was sourced from [13]. It comprises wireless communication data captured in a controlled testbed environment, consisting of multiple nodes organised in a structured arrangement with one-metre spacing. A central receiver node, located near the centre of the arrangement, was used to collect data from 163 surrounding transmitter nodes.

3.2 – Features and Preprocessing

Each Wi-Fi packet in the dataset contains the first 256 I/Q normalised samples (256x2), which were preprocessed to ensure a unity average magnitude. The dataset operates in compliance with Channel 11 of the IEEE 802.11g standard, featuring a central frequency set at 2 462 MHz and offering a 20 MHz bandwidth [13]. For the SVM model, the original 2x256 I/Q points for each sample were flattened into a single 512-dimensional vector. This step was necessary to adapt the dataset for the requirements of the SVM, as SVMs do not inherently handle multi-dimensional data in the same way that CNNs do. For the CNN, the original 2x256 I/Q points were retained in their multi-dimensional form to exploit the ability of the CNN to learn spatial hierarchies of features, potentially improving performance in RF fingerprinting tasks [14].

3.3 – Class Distribution

The dataset is heavily imbalanced, with some classes having as many as 6 000 samples and others as few as 20. It has already been divided into a training set containing 305 289 samples and a test set with 83 806 samples, comprising 163 unique classes. Figure 1 illustrates the class distributions in the training and testing datasets. The distributions emphasise the

extensive class imbalance in the datasets. The predefined data split however does serve as a valuable benchmark for comparing the performance of ML models.

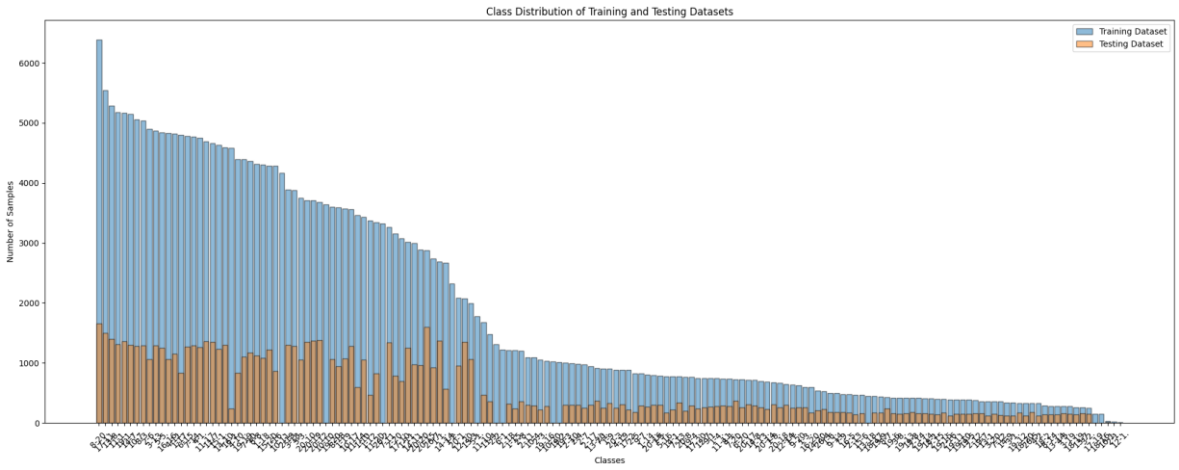

Figure 1 – Class Distributions

3.4 – Hyperparameter Optimization (SVM)

To fine-tune the hyperparameters of the SVM, Randomised Search Cross-Validation (RandomizedSearchCV) was utilised. This method was preferred over Grid Search for its computational efficiency and its capacity to explore a broader hyperparameter space within a constrained time frame. Initially, three kernel functions—linear, radial basis function (RBF), and polynomial—were evaluated. The linear kernel was promptly dismissed due to its inferior performance metrics, while the polynomial kernel demonstrated consistent superiority over the RBF kernel.
A 2-fold cross-validation was employed as an effective choice given the high dimensionality and class imbalance of the dataset. This strategy provided a reliable estimate of the generalisation performance. RandomizedSearchCV was executed with varying sample sizes (1 000, 4 000, 10 000) to investigate the impact on performance and computational time of the model. As the sample size increased, so did the computational time, exponentially. Given that 100 iterations with 2-fold cross-validation would yield nearly 200 trained models, undersampling was adopted to make the hyperparameter search computationally viable. Importantly, the accuracy of the suggested hyperparameters improved as the sample size increased.

The hyperparameter search space encompassed the following parameters:
1. Kernel: linear, radial basis function, polynomial
2. Degree: [2, 3]
3. Gamma: 'scale', 'auto', -0.5, 0.1, 1
4. Regularisation Parameter (C): [0.1, 1, 10, 100]
5. Shrinking: True, False
6. Class Weight: None, Balanced

3.5 – Hyperparameter Optimization (CNN)

3.5.1 - Architecture

Preliminary architectural choices: Two CNN architectures were considered for the task: a simpler model and a more complex model. The simpler model, despite various hyperparameter tuning attempts, was unable to surpass the performance of the SVM model,

achieving a maximum accuracy in the 70% range. This led to the exploration of a more complex architecture.
Final Architecture: the more complex model was designed with additional layers and batch normalisation to improve its learning capabilities. This architecture finally achieved performance metrics comparable to the SVM model. The more sophisticated architecture was designed to capture intricate patterns in the dataset. The architecture comprises several main components: convolutional layers, fully connected layers, and dropout layers for regularisation:

1. **Convolutional layers:** The initial convolutional layer employs 64 filters and utilises a 3x2 kernel dimension. This is succeeded by a layer for batch normalisation. The subsequent convolutional layer incorporates 128 filters with a 3x1 kernel dimension and is also succeeded by a batch normalisation layer. The final convolutional layer includes 256 filters and a 3x1 kernel, followed by another batch normalisation layer.
2. **Fully connected layers:** The initial fully connected layer consists of 512 nodes and accepts a one-dimensional array from the preceding convolutional layer. The following fully connected layer comprises 256 nodes. The terminal fully connected layer contains 163 nodes, aligning with the class count of the dataset.
3. **Regularisation layers:** A pair of dropout layers are incorporated post the fully connected layers to mitigate model overfitting. The dropout ratio is a tuneable hyperparameter during the model training phase.
4. **Activation functions:** After each convolutional and fully connected layer, ReLU activation functions are applied.
5. **Optimization strategy and loss function**: The optimization is conducted using the Adam strategy, and loss calculations are performed using the Cross-Entropy loss metric.

3.5.2 - Hyperparameters

A 2-fold cross-validation was employed, similar to the SVM model, to validate the performance of the CNN. This strategy was effective in providing a reliable measure of the generalisation capabilities of the model. Unlike the SVM model, the CNN benefited from under and over-sampling strategies as well as class weighting. These techniques improved the performance of the model, indicating their suitability for the CNN architecture. Optuna was employed for hyperparameter optimization, focusing on the learning rate, dropout rate, batch size, and optimizer type.

The hyperparameter search space for Optuna was defined as follows:

1. Learning rate: Log-uniform distribution between 1×10^{-5} and 1×10^{-1}
2. Dropout rate: Uniform distribution between 0.0 and 0.7
3. Batch size: [16, 32, 64, 128]
4. Optimizer: Adam, SGD, RMSprop

3.6 – Model Training Stopping Criteria

The methodology distinguishes between the stopping criteria for SVM and CNN models. SVMs optimise a cost function and terminate when changes in this function are below a set threshold or after a maximum number of iterations. CNNs, on the other hand, use epoch-based stopping with early termination based on validation metrics. The Optuna study used a limited 5 epochs for each trial to manage computational time across 200 trials.

4. Technology Description

Experiments were conducted using Python. SVMs utilised the Scikit-learn library, leveraging all six cores and 12 threads of the central processing unit (CPU). CNNs were implemented with PyTorch and CUDA, using an RTX 2070 Super—a graphical processing unit (GPU). For consistency, inference times for both the SVM and the CNN were assessed on the CPU. The CNN was also evaluated on the GPU, where it predictably surpassed the performance of the CPU.

While SVMs can gain from GPU acceleration, the boost might not be significant for high-dimensional data. As per the study titled '*GPU parallel implementation of support vector machines for hyperspectral image classification*' in [15], GPUs yielded an 18.5x speed increase. However, this was with a 13-class dataset, whereas our dataset has 2x256 features and 163 classes, implying potential lesser GPU benefits in our context.

5. Results & Models Developed

5.1 – Final Models and Hyperparameters (SVM)

The optimal SVM model utilised a polynomial kernel of degree 2, a regularisation parameter (C) of 350, and the 'scale' setting for gamma, achieving an accuracy of 86.46% with 85 783 support vectors. It required 3 747 seconds for training and 4 229 seconds for testing. This polynomial kernel indicates non-linear data separability, with a quadratic transformation being effective for classification. The 'scale' gamma setting and the chosen regularisation parameter contributed to the accuracy of the model, while the high number of support vectors underscores the complexity of the decision boundary.

Comparatively, models with different hyperparameters presented varied trade-offs. A model with C = 400 and degree = 3 had an accuracy of 85.81%, longer computational times (5 911 and 5 765 seconds for training and testing), and 116 765 support vectors. Models with C values near 350 displayed similar accuracies but reduced training times. Notably, a 'balanced' class weight model had diminished accuracy, suggesting that this weighting did not benefit the task, possibly due to pronounced class imbalance.

5.2 – Final Models and Hyperparameters (CNN)

The optimal model had a learning rate of 1×10^{-5}, 0.3 dropout rate, and batch size of 128, and used the Adam optimizer, achieving an accuracy of 88.30% in roughly 20 820 seconds. Variations in hyperparameters led to different performance metrics: a 0.35 dropout rate resulted in 87.17% accuracy, while a learning rate of 1×10^{-4} yielded 86.11%.

To counteract class imbalance, a dual sampling strategy was employed, undersampling high-frequency classes and oversampling low-frequency ones. Class weights, calculated using a balanced method, were also applied during training. The models trained for 200 epochs, with loss values indicating convergence by the end. Despite class imbalance challenges, effective hyperparameter tuning enabled high accuracy. Using a predefined data split had limitations but was essential for benchmark comparisons.

5.3 – Results Comparison

5.3.1 - *Accuracy and Hyperparameters*

Hyperparameter space selection was driven by the dataset's complexity and the requirement to accommodate diverse model architectures. Given the high dimensionality of the data, we systematically explored a wide range of hyperparameters, including those typically associated with simpler data structures. This approach proved insightful as it allowed us to

discern that achieving an effective fingerprint required increasing model complexity to adequately capture the intricate features and nuances present in the data. Both the CNN and SVM models were optimised for high classification accuracy. The CNN model achieved an accuracy of 88.30%, slightly outperforming the SVM model, which had an accuracy of 86.46%. Notably, these models not only matched but also surpassed the performance of the models in the paper that created the dataset [13], which reported an accuracy of 85% on unseen data using CNNs and Autoencoders for anomaly detection.

5.3.2 – Computational Time

The SVM model took approximately 3 747 seconds for training and 4 229 seconds for testing. In contrast, the CNN model required about 20 820 seconds for training, significantly higher than its SVM counterpart. This difference in computational time can be attributed to the complex architecture of the CNN.

5.3.3 - Inference Times

CNN on GPU: average inference time of approximately 0.0004 seconds for 1 000 random samples. CNN on CPU: average inference time of approximately 0.1933 seconds for 1 000 random samples. SVM on CPU: average inference time of approximately 0.0322 seconds for 1 000 random samples as seen in Table 1.

Table 1 – Model Comparative Metrics

Model Name	Accuracy	Precision	Recall	F1-Score	Training Time	Inference Time (CPU)	Inference Time (GPU)
CNN	0.88	0.74	0.76	0.74	347 mins	0.1933s	0.0003s
SVM	0.86	0.76	0.75	0.74	62 mins	0.0321s	N/A

6. Business Benefits

The use of advanced ML algorithms such as SVMs and CNNs in RF fingerprinting offers commercial opportunities in the growing field of SaaS. These models demonstrate high accuracy, making them suitable for secure wireless device authentication. In a SaaS context, businesses could deploy these algorithms to provide robust security solutions to other organisations. The low inference times of these models enable real-time applications, enhancing operational efficiency and immediate security. In summary, high accuracy and quick verification create new revenue opportunities like SaaS, supported by real-time security. A recent study [16] confirms the commercial potential of secure wireless device authentication, reinforcing the viability of SaaS models in RF fingerprinting research.

7. Conclusion

This research has made significant contributions to the field of RF fingerprinting by achieving high accuracy rates of 86.46% and 88.30% for SVM and CNN models, respectively while also highlighting inference times. These results not only surpass the existing benchmark of 85% accuracy, found in the original dataset's paper [13] but also indicate room for improvement with different dataset splits or class distribution balancing. These results underscore the potential of machine learning in enhancing wireless device security, particularly in the emerging field of SaaS. The low inference times make real-time authentication feasible, thereby enhancing operational security.

Future work should consider the use of alternative data formats, such as spectrograms [17], which could offer richer representations of RF signals. Additionally, an important consideration for future work is the incorporation of datasets that account for signal-to-noise ratios and other noise factors. Such datasets would provide a more realistic testing environment, thereby increasing the real-world applicability of the models.

Future work should also focus on extending the models to multiple bandwidths and centre frequencies. This would not only improve the generalizability of the models but also make them more adaptable to real-world scenarios. Furthermore, the development of more sophisticated models and perhaps entirely new paradigms tailored for RF fingerprinting is a promising avenue for research. Efforts should also be directed toward reducing inference and training times, thereby making these models more practical for real-time applications.
Rising new studies [16] indicate a growing interest and investment in this specific task, suggesting that RF fingerprinting is on the cusp of becoming a standardised security measure. In summary, this research contributes to the ongoing efforts to bolster wireless security, offering a foundation upon which future studies can build.

References

[1] A. Jagannath, J. Jagannath, and P. S. P. V. Kumar, “A comprehensive survey on radio frequency (RF) fingerprinting: Traditional approaches, deep learning, and open challenges,” Computer Networks, vol. 219, p. 109455, 2022.
[2] O. Ureten and N. Serinken, “Wireless security through rf fingerprinting,” Canadian Journal of Electrical and Computer Engineering, vol. 32, no. 1, pp. 27–33, 2007.
[3] S. U. Rehman, K. W. Sowerby, S. Alam, and I. Ardekani, “Radio frequency fingerprinting and its challenges,” in 2014 IEEE Conference on Communications and Network Security, 2014, pp. 496–497.
[4] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar, “A review of radio frequency fingerprinting techniques,” IEEE Journal of Radio Frequency Identification, vol. 4, no. 3, pp. 222–233, 2020.14
[5] S. Ben-David and S. Shalev-Shwartz, "Understanding machine learning: From theory to algorithms. Cambridge: Cambridge University Press, 2022.
[6] O. F.Y, T. AkinsolaJ.E., O. Awodele, O. HinmikaiyeJ., O. Olakanmi, and J. Akinjobi, “Supervised machine learning algorithms: Classification and comparison,” International Journal of Computer Trends and Technology, vol. 48, pp. 128–138, 2017.
[7] M. Awad and R. Khanna, Support Vector Machines for Classification. Berkeley, CA: Apress, 2015, pp. 39–66.
[8] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a Convolutional Neural Network,” 2017 International Conference on Engineering and Technology (ICET), pp. 1–6, Aug. 2017.
[9] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, “Wireless device identification with radiometric signatures,” 09 2008, pp. 116–127.
[10] Z. Fei, H. Yuanling, and J. chen, “Radio frequency fingerprint extraction of radio emitter based on i/q imbalance,” Procedia Computer Science, vol. 107, pp. 472–477, 12 2017.15
[11] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, “Oracle: Optimized radio classification through convolutional neural networks,” 2018.
[12] T. Jian, B. C. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis, “Deep learning for rf fingerprinting: A massive experimental study,” IEEE Internet of Things Magazine, vol. 3, no. 1, pp. 50–57, 2020.
[13] S. Hanna, S. Karunaratne, and D. Cabric, “Open set wireless transmitter authorization: Deep learning approaches and dataset considerations,” IEEE Transactions on Cognitive Communications and Networking, vol. 7, no. 1, pp. 59–72, 2021.
[14] F. Feng, S. Wang, C. Wang, and J. Zhang, “Learning deep hierarchical spatial–spectral features for hyperspectral image classification based on residual 3d-2d cnn,” Sensors, vol. 19, no. 23, 2019.
[15] K. Tan, J. Zhang, Q. Du, and X. Wang, “Gpu parallel implementation of support vector machines for hyperspectral image classification,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 8, no. 10, pp. 4647–4656, 2015.
[16] D. D. N. Nguyen, K. Sood, Y. Xiang, L. Gao, L. Chi, and S. Yu, “Toward IoT node authentication mechanism in next generation networks,” IEEE Internet of Things Journal, vol. 10, no. 15, pp. 13 333–13 341, 2023.16.
[17] G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang, “Radio frequency fingerprint identification for lora using spectrogram and cnn,” 2020.