GPS Jamming Signal Classification with CNN Feature Extraction in low Signal-to-Noise Environments

Carolyn J. Swinney[1 2] and John C. Woods[1]
[1] Department of Computer Science and Electronic Engineering, University of Essex, Colchester, UK.
[2] Air and Space Warfare Centre, Royal Air Force, UK

ABSTRACT

The Global Positioning System (GPS) is a satellite constellation which gives users access to position, navigation and timing services. Many industries not only benefit from this but are reliant on it. Although illegal, GPS jamming devices have the power to cause major disruption to many services including financial, power distribution and communication systems. Recent testing assesses Global Navigation Satellite System (GNSS) jammers as being very dangerous to aircraft and Unmanned Aerial Vehicles (UAVs) especially those flying at low height. GNSS is also critical to the safe operation of Connected and Autonomous Vehicles (CAV) such as driverless cars. Timely detection of an attack is deemed to be enough to ensure the safety of the vehicle. Detection and classification of GNSS jamming signals is necessary to enable this. This paper considers feature extraction using a Convolutional Neural Network (CNN) when representing the signal as a graphical image. The JamDetect dataset is produced containing 6 different types of commercial jamming signals. Features are extracted using a CNN before a machine learning classifier is trained for classification. Results show that representing the signal in the graphical form of Power Spectral Density (PSD) is the least susceptible to noise. CNN feature extraction with machine learning classifier Logistic Regression using PSD produces 82.7% (+/-0.7%) at -20dB SNR and 100% accuracy at -10dB SNR. The results using PSD graphical signal representation are significant for when it is necessary to detect and classify GPS jamming signals in low SNR environments.

Keyword: *Convolutional Neural Network; Deep Learning; GNSS Jamming; Machine Learning; Classification; RF Signal Analysis; Transfer Learning; Feature Extraction; JamDetect.*

1. INTRODUCTION

For many sectors, the Global Navigation Satellite System (GNSS) provides vital position, navigation and timing services but are extremely vulnerable to interference due to the signals being weak at the receiver. Space (communications) in the UK are deemed critical to national security and therefore classed as Critical National Infrastructure (Centre for the Protection of National Infrastructure 2021). In 2017 the potential economic impact from a loss of services for 5 days was assessed at £5.2bn (London Economics 2017). Although this study has not been repeated it can assume the figure is much higher today. In 2021 society is highly dependent on GNSS for applications from power distribution, emergency services, travel and even 5G performance for a precise timing signals (Colard 2020). A further concern is the cascading effect on secondary and tertiary sectors from GNSS disruption (Pescaroli et al. 2019). Intentional interference known as jamming occurs when a jamming signal is transmitted at high power in a GNSS band, disrupting the GNSS service. Jamming equipment is illegal in many countries. In the UK the use of a jamming device is an offence under the Wireless Telegraphy Act but it is not illegal to buy or own the equipment (Ofcom 2021). Their availability on the open market from as little as £10.99 (Amazon n.d.) combined with an increase in the use of drones causing public annoyance, has created fears of a stark increase in usage, making them a serious and credible threat to satellite navigation. An increased use of UAVs by law enforcement, hobbyists and commercial, has uncovered a potential motivation for civilians to purchase jamming devices to illegally combat their use. Morong *et al.* (Morong, Puričer, and Kovář 2019) carry out a study of GNSS jamming in a real world environment. They assess that GNSS jammers are very dangerous to aircraft and UAVs, especially those that are flying at low height. The IET recently did a study to look at how easy it would be to purchase a Drone Jammer Gun from Asia revealing a straightforward practise (Heubl 2021).

However, a high powered jamming gun is not required to create a significant and critical effect. Today, with low cost Software Defined Radios (SDR) on the market it is possible to re-produce many different jamming signals, Lineswala and Shah show this in (Lineswala and Shah 2018) for jamming the Indian Regional Navigation Satellite System. Ferreira et al. (Ferreira et al.

2020) show the use of GNURadio software to produce jamming signals which are then transmitted on a BladeRF SDR to jam GPS signals for the disruption of UAV operation. Glomsvoll and Bonenberg (Glomsvoll and Bonenberg 2017) showed maritime GPS receivers to be affected by even low power jamming signals from 1600 metres away. Positional accuracy was affected from under 1000m and showing up to 10m discrepancy in position. A real world example of the detrimental effect of a low cost GNSS jammer was shown when a truck driver used a device to obscure his movements from his employer. As part of his daily route he passed Newark International Airport and caused disruption to the Air Traffic Control System. It took 3 months to identify the issue with the system (Borio et al. 2016). The potential harm which could result from the use of a low cost jamming device becomes more critical when future reliance of Connected and Autonomous Vehicles (CAV) such as driverless cars on GNSS is considered. Pham and Xiong (Pham and Xiong 2020) present a survey of 184 papers considering state of the art attacks on CAVs. With regards to GNSS jamming they determine that the timely detection of a jamming incident in enough to ensure CAV safety and can be a pre-cursor to filtering out the attack signal so that the CAV can continue its operation in certain circumstances.

BACKGROUND

Jamming types have been classified into various categories in previous studies. A graphical representation of this can be seen in Fig.1 below.

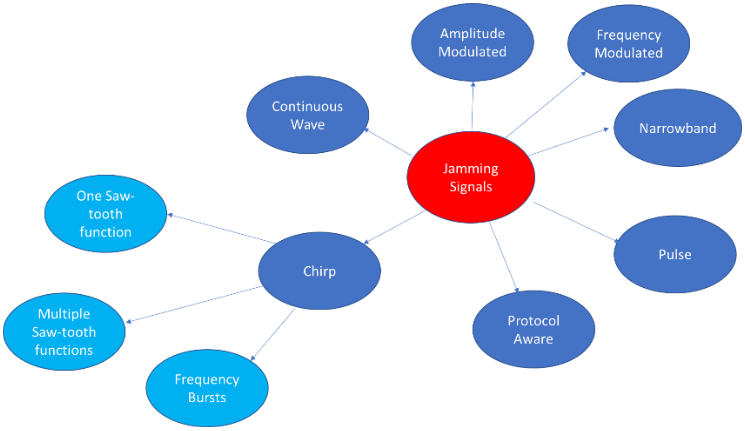


Figure 1.Graphical Representation of Jamming Signal Types.

Kraus et al. (Kraus, Bauemfeind, and Eissfeller 2011) define four classes to include continuous wave jamming and three variations of chirp jamming signals. Ferre et al. (Ferre, Fuente, and Lohan 2019) increase these definitions to 5 classes; Amplitude Modulated, Chirp jammers; Frequency modulated; Pulse or Distance Measurement Equipment (DME) jammers and Narrowband (NB) jammers. Other studies are less discriminative, Glomsvoll and Bonenberg (Glomsvoll and Bonenberg 2017) consider narrow and wideband jamming signals on high end receivers for GPS L1 and GLONASS L1 frequency bands. Their work concluded that the impact to GPS from jamming was more critical than jamming GLONASS. Lee et al. (Lee, Kim, and Won 2018) consider a cloud based solution for the detection of GNSS jamming but rely on multiple receivers in the area of the jammer. They prove the detection of jammer type and estimate localisation based on the 2 dimensional time-frequency correlation between receivers. Kim et al. (Kim, Jin, and Won 2020) again show the validity of considering the jamming signal as a 2 dimensional correlation in the time-frequency domain for detection of jamming type and estimated localisation. As in (Lee, Kim, and Won 2018) this solution is based on generating a network of receivers. Xu et al. (Xu et al. 2020) propose the use of a Deep Neural Network (DNN) with time, frequency and transform domain features from the signal for jamming recognition. The DNN was able to detect 12 jammer types with over 99% accuracy and outperformed traditional machine learning classifiers. Lineswala and Shah (Lineswala and Shah 2019) use power spectral density to represent the signals but only consider jamming detection and not classification. Wu et al. (Wu et al. 2018) classify jamming signals for space communication links by extracting features through a CNN. In particular they consider the combination of jamming signals and various levels of jamming signal to noise ratios showing features to be robust. Ferre et al. (Ferre, Fuente, and Lohan 2019) use spectrograms and treat jamming classification as an image classification problem. They achieve 94.90% accuracy using Support Vector Machine and 91.36% using CNN. By utilising transfer learning with a pre-trained CNN on ImageNet, Swinney and Woods (Swinney and Woods 2019) are able to increase this accuracy to 98% by feeding the CNN various representations of the signal at once. This work does not consider specifically altering the SNR to set levels to observe the effect. However, signals are normally corrupted by the jamming signal and Additive White Gaussian Noise (AWGN) due to background noise over wireless communication links (Wu et al. 2018). Although a large increase in AWGN will interfere with the reception of GNSS signals itself, it is important to know if a jamming signal is present in these situations for reasons of timely attribution and intelligence. In other fields, Swinney and Woods show the effectiveness of the deep learning technique presented in this paper utilising transfer learning with a pre-trained CNN when considering the detection and classification of UAV signals in (Swinney and Woods 2021a)(Swinney and Woods 2020)(Swinney and Woods 2021b)(Swinney 2021).

Evaluating signals in low SNR environments is significant for organisations who require to know whether an adversary is attempting a jamming attack for reasons of timely attribution and enabling an intelligence picture of attacks. Therefore in this paper previous work by Swinney and Woods (Swinney and Woods 2019) is extended by producing the JamDetect dataset to evaluate the ability of transfer learning via a pre-trained VGG-16 CNN to maintain accuracy in detecting and classifying GNSS jamming signals in low SNR environments. The purpose of this work is to understand whether the technique presented in (Swinney and Woods 2019) will be valuable in low SNR environments.

2 METHODOLOGY

Dataset Generation

Now the generation of the dataset will be discussed. First, $s(t)$ is represented as the signal at the GPS receiver shown in equation (1).

$$s(t) = g(t) + j(t) + w(t) \qquad (1)$$

Whereby $g(t)$ is the GPS signal coming from the satellite, $j(t)$ is our jamming signal and $w(t)$ is the noise generated by AWGN across the wireless channel.

GPS Signal

First of all $g(t)$ is considered, the GPS signal coming from the satellite constellation. GPS is specifically be considered as the GNSS system. GPS is a US system consisting of 24 satellites across 6 orbits which operate over 12 hr periods. GPS has two carrier frequencies L1 and L2 which broadcast Binary Phase-Shift Keying (BPSK). The L1 band is the focus which operates at 1575.42MHz across a bandwidth of 24MHz. However, it has been proven due to the design of the L1 signal that a bandwidth of 9.66MHz is enough for tracking and navigation (LabSat 2021). A bandwidth of 10MHz is used within our experiments.

To try and create a realistic signal a real GPS signal is generated using GPS-SDR-SIM (Ebinuma 2018). GPS-SDR-SIM is python based code which generates GPS baseband data for the intention of being broadcast by an SDR. It allows you to define a static location, for our experiments the New York New York Piano Bar in Las Vegas, GPS co-ordinates 36° 10' 11.7876" N 115° 8' 23.3952" W which corresponds to Latitude Longitude (36.169941, -115.139832) are used. The GPS Satellite constellation is then specified using a daily GPS broadcast ephemeris file from the NASA Earth Data site (NASA 2021). For a date the 20 Dec 2014 is used for testing but ephemeris data is available up to 24hrs into the future, so applications can predict orbits. GPS-SDR-SIM uses the data to generate the pseudo range and Doppler for simulated GPS satellites in view at that time which is then converted to I/Q baseband samples.

Jamming Signals

Next the jamming signals $j(t)$ are considered and the details of the production of the JamDetect dataset. GNURadio was used to produce the jamming signals. GNURadio is an open source framework which includes a free toolkit for the development of SDRs in a PC environment (Nutaq n.d.). The signals defined in (Ferreira et al. 2020) are used with GNURadio as a starting point and narrowband jamming is added to the jamming classes already defined.

Chirp Jammer

Chirp signals were generated by increasing frequency over time, also known as sweep jamming. Signals are constructed across a bandwidth of 10MHz with a fast sweep rate of 10KHz, the stop and start frequencies are calculated using equation (2).

$$\begin{aligned} f_{min} &= f_c - \frac{BW}{2} = 1575.42\,x\,10^6 - \frac{10\,x\,10^6}{2} = 1570.42MHz \\ f_{max} &= f_c + \frac{BW}{2} = 1575.42\,x\,10^6 + \frac{10\,x\,10^6}{2} = 1580.42MHz \end{aligned} \tag{2}$$

The implementation used to produce the chirp signal in GNURadio is sourced from (Markowski 2021).

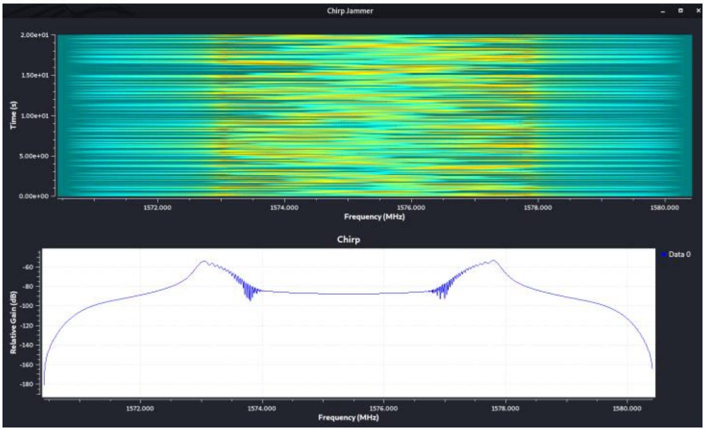
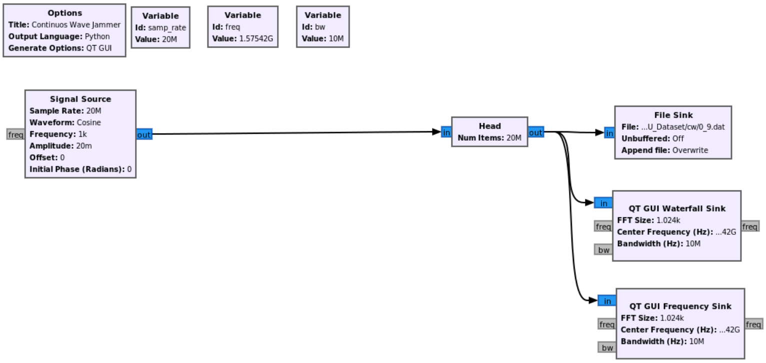


Figure 2. Chirp Jamming Signal Spectrogram (top) PSD (bottom).

Fig.2 shows the Spectrogram and PSD of the signal. The PSD shows the signal at its highest point is approximately -65dB/Hz.

Continuous Wave (CW) Jammer

The CW jamming signal is produced in GNURadio using a 1KHz cosine signal source. Fig.3 shows the implementation of the jamming signal generation in GNURadio.



Options
Title: Continuos Wave Jammer
Output Language: Python
Generate Options: QT GUI

Variable
Id: samp_rate
Value: 20M

Variable
Id: freq
Value: 1.57542G

Variable
Id: bw
Value: 10M

Signal Source
Sample Rate: 20M
Waveform: Cosine
Frequency: 1k
Amplitude: 20m
Offset: 0
Initial Phase (Radians): 0

Head
Num Items: 20M

File Sink
File: ...U_Dataset/cw/0_9.dat
Unbuffered: Off
Append file: Overwrite

QT GUI Waterfall Sink
FFT Size: 1.024k
Center Frequency (Hz): ...42G
Bandwidth (Hz): 10M

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): ...42G
Bandwidth (Hz): 10M

Figure 3. CW Generation GNURadio.

In Fig.4 it is observed that the CW jamming signal only occupies a small part of the GPS bandwidth.

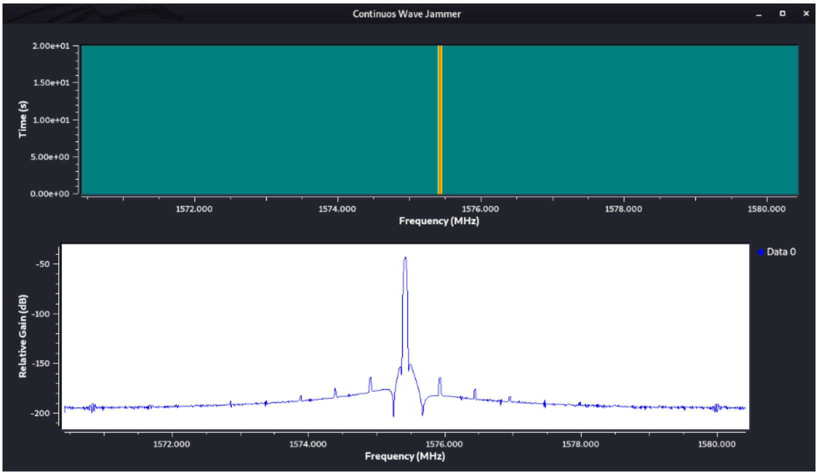
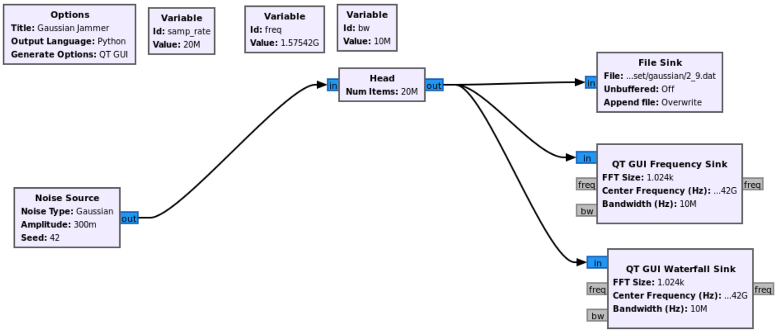


Figure 4. CW Jamming Signal Spectrogram (top) PSD (bottom).

This can be seen clearly in Fig.4 which also shows the highest peak of the signal at approximately -55db/Hz on the PSD.

Barrage Jammer

The barrage jamming signal is produced using a gaussian noise source in GNURadio. Fig.5 shows the implementation in GNURadio.


Options
Title: Gaussian Jammer
Output Language: Python
Generate Options: QT GUI

Variable
Id: samp_rate
Value: 20M

Variable
Id: freq
Value: 1.57542G

Variable
Id: bw
Value: 10M

Noise Source
Noise Type: Gaussian
Amplitude: 300m
Seed: 42

Head
Num Items: 20M

File Sink
File: ...set/gaussian/2_9.dat
Unbuffered: Off
Append file: Overwrite

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): ...42G
Bandwidth (Hz): 10M

QT GUI Waterfall Sink
FFT Size: 1.024k
Center Frequency (Hz): ...42G
Bandwidth (Hz): 10M

Figure 5. Barrage Jamming Generation GNURadio.

As opposed to the CW jamming signal, the barrage jammer occupies all the bandwidth. It is not a discrete choice of jammer but potentially more effective against GPS due to the spread spectrum nature of a GPS signal.

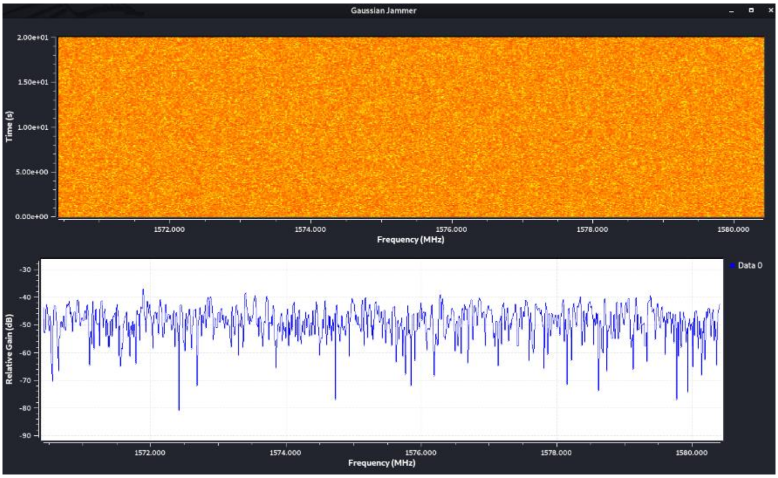


Figure 6. Barrage Jamming Signal Spectrogram (top) PSD (bottom).

The difference can be seen clearly when comparing Fig.4 for the CW jammer to Fig.6 for the barrage jamming.

Narrowband Jammer

The narrowband (NB) jammer is constructed in GNURadio by generating a QPSK signal which covers a bandwidth of 1.6MHz. Generation of the QPSK constellation was produced using the information in (Steve and Patel 2016).

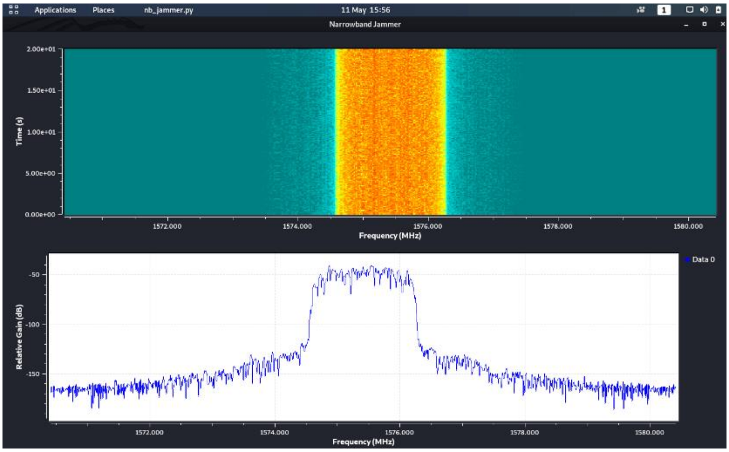
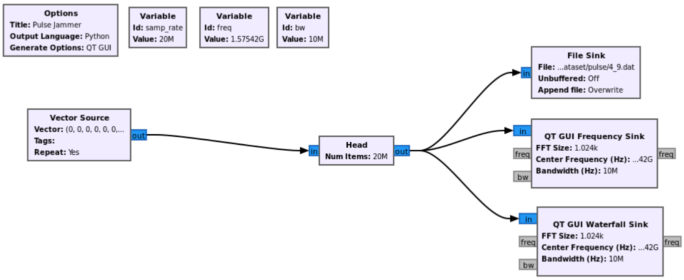


Figure 7. NB Jamming Signal Spectrogram (top) PSD (bottom).

Fig.7 shows the signal represented in both Spectrogram and PSD form. It is observed that the signal peaking at approximately -55dB/Hz.

Pulse Jammer

The pulse jamming signal was created using a vector source with a low duty cycle of 2% in GNURadio (Ferreira et al. 2020). The implementation of the jamming signal generation is seen in Fig.8 below.



Options
Title: Pulse Jammer
Output Language: Python
Generate Options: QT GUI

Variable
Id: samp_rate
Value: 20M

Variable
Id: freq
Value: 1.57542G

Variable
Id: bw
Value: 10M

Vector Source
Vector: (0, 0, 0, 0, 0, 0,...
Tags:
Repeat: Yes

Head
Num Items: 20M

File Sink
File: ...ataset/pulse/4_9.dat
Unbuffered: Off
Append file: Overwrite

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): ...42G
Bandwidth (Hz): 10M

QT GUI Waterfall Sink
FFT Size: 1.024k
Center Frequency (Hz): ...42G
Bandwidth (Hz): 10M

Figure 8. Pulse Jamming Generation GNURadio.

The sequences of pulses occupy the full bandwidth. Fig.9 shows the Spectrogram and PSD for the pulse jamming signal.

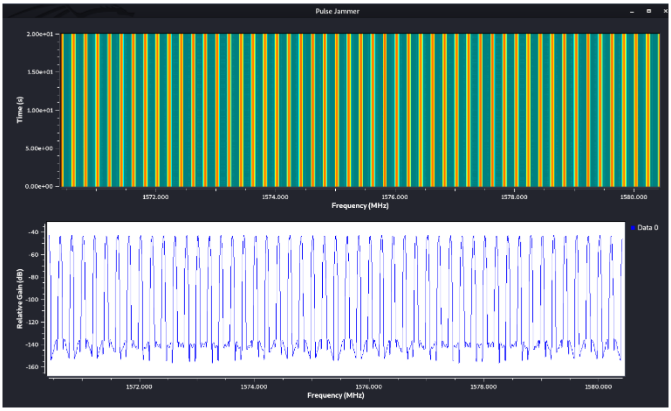


Figure 9. Pulse Jamming Signal Spectrogram (top) PSD (bottom).

Protocol Aware Jammer

The protocol aware jammer is constructed in GNURadio by generating a BPSK signal (GNU Radio 2021) to represent the structure of a GPS signal. Fig.10 shows the spectrogram and PSD representations of the protocol aware jamming signal.

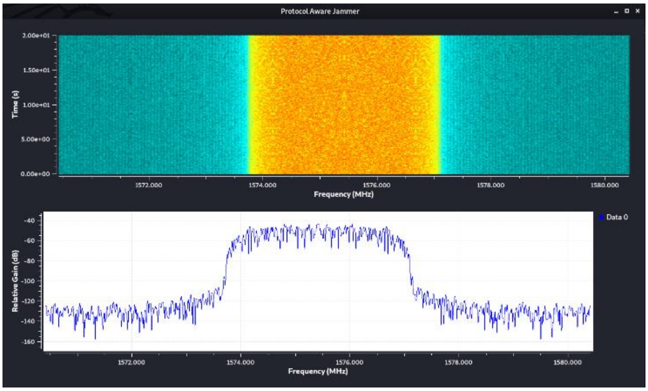


Figure 10. Protocol Jamming Signal Spectrogram (top) PSD (bottom).

The GPS signal and the jamming signals were combined to achieve a JSR of 40dB.

Additive Gaussian White Noise

Lastly $w(t)$ modelled by Additive Gaussian White Noise (AWGN) to corrupt the signal and test classification accuracy for various levels of SNR is considered. AWGN was generated in Python 3 and added to the signal as shown in Fig.11.

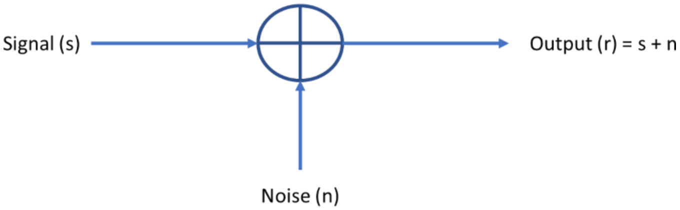


Figure 11. AWGN Channel Process.

SNR is defined in (3) where P_r is the power in the signal, B the bandwidth and PSD of the noise $\frac{N_0}{2}$ (Mathuranathan 2015).

$$SNR = \frac{P_r}{N_0 B} \tag{3}$$

To add the noise the power in our signal is first calculated. This is seen in (4) where N is the length of signal s.

$$P = \frac{1}{N}\sum_{i=0}^{N-1}|s_i|^2 \tag{4}$$

P is then used with our desired SNR to calculate the noise spectral density N_0 as seen in (5).

$$N_0 = \frac{P}{SNR} \tag{5}$$

Lastly the noise power is calculated which is required to generate Gaussian random noise in (6).

$$\sigma^2 = \frac{N_0}{2} \tag{6}$$

The calculations for generating AWGN and its implementation in python are referenced (Viswanathan 2019).

Graphical Signal Representations

Datasets were created for SNR 50dB, 30dB, 10dB, -10dB and -20dB to understand the effect on classification accuracy. A Power Spectral Density (PSD) was used to represent the signal in the frequency domain, displaying the power distribution over the frequency range. A spectrogram was used to consider the signal in the time domain. Both representations were plotted in Matplotlib Python 3 with an FFT size of 1024 and a Hanning windowing function. A raw constellation is plotted with real and imaginary parts of the signal on the x and y axis respectively and a histogram representation of the real part of the signal over 500 bins is also considered. Lastly a concatenated representation onto one image is created using python to include a dataset which contains all 4 graphical signal representations. Datasets were created of 1000 images of size 224x224 for use with 5-fold cross validation. 5-fold cross validation was chosen due to the work of (James, Gareth Witten, Daniela Hastie, Trevor Tibshirani 2017) who found that either k=5 or k=10 both showed empirically errors rates which displayed neither high variance or high bias. K=5 is less computationally heavy than k=10 so it was the choice for the experiments.

Next the graphical representations of s(t) at varying levels of SNR are visualised. First the CW jamming signal at a SNR of 50dB are observed in Fig.12 (left). The jamming signal is clearly present on the raw constellation graphical representation.

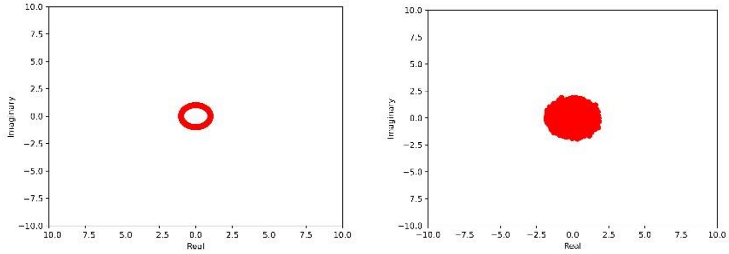


Figure 12. CW Jamming Signal SNR 50dB (left) SNR 10dB (right).

Fig.12 (right) shows the same CW jamming signal but after dropping the SNR to 10dB. The constellation in red fills in and increases in size compared to Fig.12 (left).

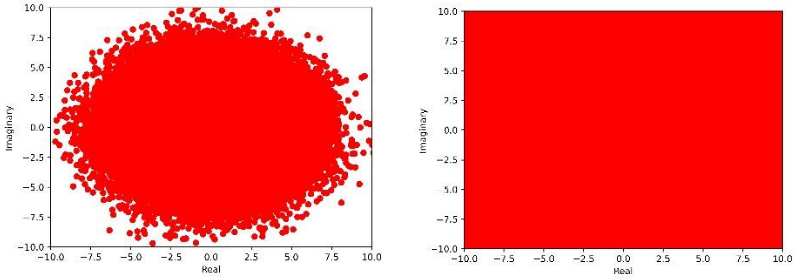


Figure 13. CW Jamming Signal SNR -10dB (left) SNR -20dB (right).

In Fig.13 (left) the SNR ratio is dropped to -10dB and again the red constellation increases in size. In Fig.13 (right) the CW jamming signal with a SNR of -20dB is observed and the whole plot is covered in red due to the increase of noise.

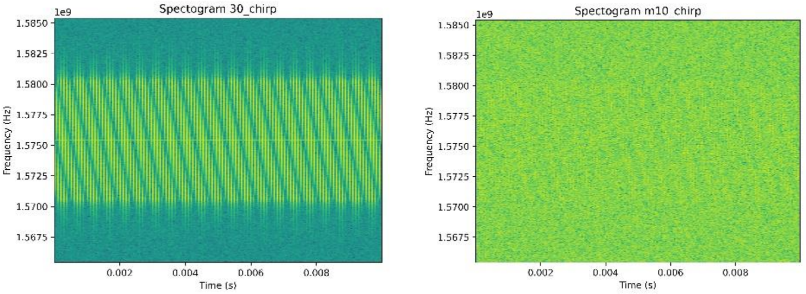


Figure 14. Chirp Jamming Signal SNR 30dB (left) SNR -10dB (right).

Fig.14 (left) shows the chirp jamming signal represented in spectrogram form with a SNR of 30dB. The sweep nature of the signal in yellow is clearly observed. As the SNR decreases to -10dB we start to lose the signal in the noise as seen in Fig.14 (right), however the pattern is still visible if it is considered closely. When the SNR is further decreased to -20dB the signal cannot be visually observed at all on the spectrogram.

CNN Feature Extraction and Machine Learning Classifiers

CNN Feature Extraction

CNNs are often used for object detection but through a process called transfer learning, a pre-trained CNN can be used for other purposes such as detecting medical condition through brain scan or eye scan images (Thota and Umma Reddy 2020), (Kaur and Gandhi 2019). A VGG-16 is a type of CNN with 16 layers, produced by Oxford Visual Geometry Group and commonly utilised in a pre-trained manner using a 14 million image database called ImageNet (Simonyan and Zisserman 2015). To utilise the VGG-16 in our research for feature extraction forward propagation is stopped at the last pooling layer to enable features to be saved. Table 1 shows the structure of the CNN used.

Layer Type	Shape
Input Layer)	224x224x3
Convolutional 2D Layer	112x112x128
Convolutional 2D Layer	112x112x128
Max Pooling 2D Layer	112x112x128
Convolutional 2D Layer	56x56x256
Convolutional 2D Layer	56x56x256
Max Pooling 2D Layer	56x56x256
Convolutional 2D Layer	28x28x512
Convolutional 2D Layer	28x28x512
Convolutional 2D Layer	28x28x512
Max Pooling 2D Layer	28x28x512
Convolutional 2D Layer	14x14x512
Convolutional 2D Layer	14x14x512
Convolutional 2D Layer	14x14x512
Max Pooling 2D Layer	7x7x512
Convolutional 2D Layer	7x7x512
Convolutional 2D Layer	7x7x512
Convolutional 2D Layer	7x7x512
Max Pooling 2D Layer	7x7x512

Table 1. VGG-16 Architecture.

The last layer of shape 7x7x512 produces a feature vector of 25,088 values when flattened. A batch size of 16 was used for the CNN.

Machine Learning Classifier Logistic Regression

The machine learning model LR has a number of fixed parameters which are set depending on the number of features given to the model as the input. The LR output produces a sigmoidal curve as it is categorical and can be seen in equation (7).

$$h = \frac{e^{x}}{1 + e^{-x}} \tag{7}$$

Input features are represented by x and x is initialised using a random value θ. Equation (8) shows the algorithm for when there are multiple features and as it is updated a relationship is formed between the features and the output.

$$h = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + .. \tag{8}$$

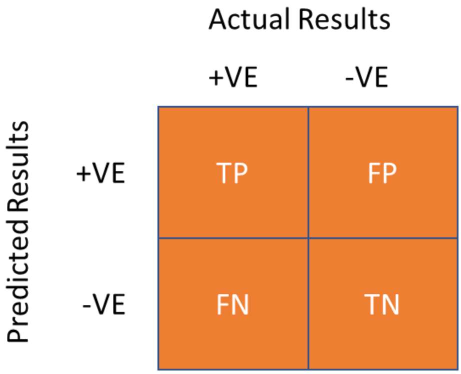
Multiple classes means that the sigmoid is generalised and this is known as the Softmax function which plots the input feature vector to a probability distribution which lies somewhere between 0 and 1. Equation (9) shows a softmax function for a feature vector z with k number of classes (Daniel and Martin 2019).

$$softmax(z_{\text{i}}) = \frac{e^{z_{\text{i}}}}{\sum_{j=1}^{k} e^{z\text{j}}} \tag{9}$$

For this work Python3 library SKlearn is utilised to implement the LR with Limited-memory Broyden–Fletcher–Goldfarb–Shanno (LGBFS) as the solver and ridge Regression as the penalty for the loss function.

Performance Evaluation

Accuracy was the metric chosen to evaluate performance, but the confusion matrix is explained so that accuracy can be suitably defined below. A confusion matrix is seen in Fig. 15.



Actual Results

Predicted Results	+VE	-VE
+VE	TP	FP
-VE	FN	TN

Figure 15. Confusion Matrix

A True Positive (TP) indicates that the algorithm prediction was correct, a True Negative (TN) is when the prediction was incorrect, but the actual result was also incorrect. A False Positive (FP) shows a correct prediction but an incorrect actual result and a False Negative (FN) shows the prediction incorrect but the actual result correct. The definitions are used to produce equations for accuracy as show in equation 10 below.

$$Accuracy \; = \frac{\mathrm{TP} + \mathrm{TN}}{TP + TN + FP + FN} \qquad (10)$$

3 Results

Table 2 shows the training/test 5-fold cross validation results. PSD produces the highest accuracy when SNR is reduced. It can be seen that as the SNR reduces from 50 downwards, but a decrease in accuracy is not seen until the SNR drops below 10dB. This allows us to make the assumption that evaluating the signal in the frequency domain is less susceptible to noise and able to still identify and classify the signal. When the time domain via the Spectrogram graphical representation is considered, it can be seen that it maintains accuracy levels up to -10dB SNR but significantly decreases below this, this is a significant drop when compared to the PSD graphical signal representation. The raw constellation is shown to be the most susceptible to noise and has the lowest performing accuracy scores. This indicates that the raw constellation data may perform better with some pre-processing which could include processes such as filtering. Concatenating the different signal representations seems to also inherit the low accuracy levels seen for the raw constellation and histogram representations at lower SNR levels which is logical.

SNR (dB)	PSD Accuracy (%)	Spectrogram Accuracy (%)	Raw Accuracy (%)	Hist Accuracy (%)	Concat Accuracy (%)
50	100	100	83.3 (+/- 0.1)	100	100
30	100	100	66.7 (+/- 0.1)	99.9 (+/- 0.1)	100
10	100	100	76.8 (+/- 1.5)	94.3 (+/- 0.2)	100
-10	100	99.6 (+/- 0.3)	50.2 (+/- 1.2)	50.7 (+/- 0.7)	100
-20	82.7 (+/- 0.7)	40.0 (+/-0.6)	25.8 (+/- 1.3)	22.6 (+/- 0.8)	74.0 (+/- 1.5)

Table 2. Results of 5 Fold Cross Validation - Accuracy (\%).

Table 3 shows the validation results. The most significant finding from the validation results is with respect to the concatenated signal representations. Although effective at SNR levels of 10dB and above, the validation scores show that the concatenation of the signal representation presents overfitting at -10dB and below. This means that the model was learning the training data so well that the model did not generalise when it was given new samples in the validation set. A slight increase at the mid SNR level is also seen which indicates that the model performs better with some level of noise present. This indicates that the model can generalise better with a certain amount of noise present.

The validation results also confirm the training and test results that the PSD produces the highest accuracy scores in low SNR environments and therefore is the least susceptible to noise.

SNR (dB)	PSD Accuracy (%)	Spectrogram Accuracy (%)	Raw Accuracy (%)	Hist Accuracy (%)	Concat Accuracy (%)
50	100	100	83.4	94.1	96.1
30	100	100	66.8	100	84.1
10	100	100	78.5	94.2	100
-10	100	99.7	51.1	50.1	55.8
-20	81.7	40.1	27.1	23.9	19.4

Table 3. Results of 5 Fold Cross Validation - Accuracy (\%).

4 CONCLUSION

Overall, our results have shown that the PSD graphical signal representation is the least susceptible to noise and produces the highest accuracy in low SNR environments. While our previous work showed that concatenating various graphical signal representations together as the input for the CNN, this extension of that work has shown this to be effective only in environments with SNR levels higher than -10dB. This is significant for congested environments where it is vital to know GPS jamming is being attempted. In low SNR environments PSD should be utilised for GPS jamming classification.

5 FURTHER RESEARCH DIRECTIONS

Further work could include exploring deeper architectures and types of neural networks for feature extraction. Experimentations with real GPS jamming signals which includes distance of detection and classification should also be considered for future exploration. Overall, this work has shown that PSD graphical signal representation provides the highest accuracy for GPS jamming classification in low SNR environments with CNN feature extraction and machine learning classifier logistic regression.

6 REFERENCES

Amazon. n.d. "Generic GPS Signal Interference Blocker." *Amazon.Co.Uk*. Accessed September 16, 2020. https://www.amazon.co.uk/Grneric-interference-anti-tracking-tracking-anti-positioning/.

Borio, Daniele, Fabio Dovis, Heidi Kuusniemi, and Letizia Lo Presti. 2016. "Impact and Detection of GNSS Jammers on Consumer Grade Satellite Navigation Receivers." *Proceedings of the IEEE* 104 (6): 1233–45. https://doi.org/10.1109/JPROC.2016.2543266.

Centre for the Protection of National Infrastructure. 2021. "Critical National Infrastructure." Accessed April 14, 2021, *CPNI*. https://www.cpni.gov.uk/critical-national-infrastructure-0.

Colard, Eric. 2020. "Distributing High-Precision Time over Optical Networks in the 5G World." Accessed September 14, 2020, *GPS World*, April. https://www.gpsworld.com/distributing-high-precision-time-over-optical-networks-in-the-5g-world/.

Daniel, Jurafsky, and James H Martin. 2019. "Logistic Regression." Accessed December 10, 2020. *Speech and Language Processing*, 1–19.

Ebinuma, Takuji. 2018. "Software-Defined GPS Signal Simulator." Accessed May 12, 2021. *GitHub - Osqzss/Gps-Sdr-Sim*. https://github.com/osqzss/gps-sdr-sim.

Ferre, Ruben Morales, Alberto De La Fuente, and Elena Simona Lohan. 2019. "Jammer Classification in GNSS Bands via Machine Learning Algorithms." *Sensors (Switzerland)* 19 (22). https://doi.org/10.3390/s19224841.

Ferreira, Renato, João Gaspar, Pedro Sebastião, and Nuno Souto. 2020. "Effective GPS

Jamming Techniques for UAVs Using Low-Cost SDR Platforms." *Wireless Personal Communications* 115 (4): 2705–27. https://doi.org/10.1007/s11277-020-07212-6.

Glomsvoll, Oeystein, and Lukasz K. Bonenberg. 2017. "GNSS Jamming Resilience for Close to Shore Navigation in the Northern Sea." *Journal of Navigation* 70 (1): 33–48. https://doi.org/10.1017/S0373463316000473.

GNU Radio. 2021. "Simulation Example: BPSK Demodulation." Accessed May 12, 2021. GNU Radio. 2021. https://wiki.gnuradio.org/index.php/Simulation_example:_BPSK_Demodulation

Heubl, Ben. 2021. "How Illegal Drone Jammers Are Sold to Europe." Accessed April 14, 2021. *IET E&T*, March. https://eandt.theiet.org/content/articles/2021/03/how-drone-jammers-are-sold-to-europe-and-the-uk/.

James, Gareth Witten, Daniela Hastie, Trevor Tibshirani, Robert. 2017. "An Introduction to Statistical Learning," Page 184.

Kaur, Taranjit, and Tapan Kumar Gandhi. 2019. "Automated Brain Image Classification Based on VGG-16 and Transfer Learning." *Proceedings - 2019 International Conference on Information Technology, ICIT 2019*, December, 94–98. https://doi.org/10.1109/ICIT48102.2019.00023.

Kim, Hyeong-Pil, Gwon-Gyu Jin, and Jong-Hoon Won. 2020. "GNSS Cloud-data Processing Technique for Jamming Detection, Identification, and Localisation." *IET Radar, Sonar & Navigation* 14 (8): 1143–49. https://doi.org/10.1049/iet-rsn.2019.0518.

Kraus, Thomas, Roland Bauemfeind, and Bernd Eissfeller. 2011. "Survey of In-Car Jammers - Analysis and Modeling of the RF Signals and IF Samples (Suitable for Active Signal Cancellation)." *24th International Technical Meeting of the Satellite Division of the Institute of Navigation 2011, ION GNSS 2011* 1 (September): 430–35.

LabSat. 2021. "GNSS Frequency Guide." Accessed May 12, 2021. *LabSat Frequency Guide*. https://www.labsat.co.uk/index.php/en/gnss-frequency-guide.

Lee, Jung-Hoon, Hyeong-Pil Kim, and Jong-Hoon Won. 2018. "GNSS Cloud-Data Processing Technique For Jamming Detection And Localization." *International Technical Symposium on Navigation and Timing, Toulouse, France*, October. https://doi.org/10.31701/itsnt2018.23ï.

Lineswala, Priyanka L., and Shweta N. Shah. 2018. "Designing of SDR Based Malicious Act: IRNSS Jammer." *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST* 220 LNICST: 237–46. https://doi.org/10.1007/978-3-319-73712-6_25.

———. 2019. "Performance Analysis of Different Interference Detection Techniques for Navigation with Indian Constellation." *IET Radar, Sonar and Navigation* 13 (8): 1207–13. https://doi.org/10.1049/iet-rsn.2019.0091.

London Economics. 2017. "Economic Impact to the UK of a Disruption to GNSS Showcase Report," Accessed September 14, 2020. *London Economics.* https://www.gov.uk/government/publications/the-economic-impact-on-the-uk-of-a-disruption-to-gnss

Markowski, Mike. 2021. "Gnuradio Mini Projects." Accessed May 12, 2021. *Udel*. http://udel.edu/~mm/gr/.

Mathuranathan. 2015. "Simulate Additive White Gaussian Noise (AWGN) Channel." Accessed February 26, 2021. *Gaussian Waves*, June. https://www.gaussianwaves.com/2015/06/how-to-generate-awgn-noise-in-matlaboctave-without-using-in-built-awgn-function/.

Morong, Tomáš, Pavel Puričer, and Pavel Kovář. 2019. "Study of the GNSS Jamming in Real Environment." *International Journal of Electronics and Telecommunications* 65 (1): 65–70. https://doi.org/10.24425/ijet.2019.126284.

NASA. 2021. "Earthdata." Accessed May 12, 2021. *NASA Earth Data*. https://urs.earthdata.nasa.gov/.

Nutaq. n.d. "A Short History of Software-Defined Radio (SDR) Technology." *Nutaq*. Accessed November 7, 2019. https://www.nutaq.com/blog/short-history-software-defined-radio-sdr-technology.

Ofcom. 2021. Accessed April 14, 2021. "Radio Frequency Jammers." https://www.ofcom.org.uk/spectrum/interference-enforcement/spectrum-offences/jammers.

Pescaroli, G, L M Green, R T Wicks, S Bhattarai, S Turner, L Green, and Turner. 2019. “Cascading Effects of Global Positioning and Navigation Satellite Service Failures.” Institute for Risk and Disaster Reduection Mullard Space Science Laboratory Cascading Effects of Global Positioning and Navigation Satellite Service Failures. https://doi.org/10.14324/000.rp.10076568.

Pham, Minh, and Kaiqi Xiong. 2020. “A Survey on Security Attacks and Defense Techniques for Connected and Autonomous Vehicles.” *ArXiv*, 1–24.

Simonyan, Karen, and Andrew Zisserman. 2015. “Very Deep Convolutional Networks for Large Scale Image Recognition.” *ICLR 2015*. http://www.robots.ox.ac.uk/.

Steve, Prepared, and Jordan-Bhaumil Patel. 2016. “Image Transfer and Software Defined Radio Using USRP and GNU Radio.” Accessed May 12, 2020. *Project Report*. http://academic.csuohio.edu/yuc/mobile/mcproj/3p-PatelJordan.pdf.

Swinney, Carolyn J., and John C. Woods. 2019. “GNSS Jamming Classification via CNN , Transfer Learning & the Novel Concatenation of Signal Representations.” *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, June, 1–9. https://doi.org/10.1109/CYBERSA52016.2021.9478250.

———. 2021a. “Unmanned Aerial Vehicle Operating Mode Classification Using Deep Residual Learning Feature Extraction.” *Aerospace* 8 (3): 79. https://doi.org/10.3390/aerospace8030079.

Swinney, Carolyn J. 2021. “RF Detection and Classification of Unmanned Aerial Vehicles in Environments with Wireless Interference.” *2021 International Conference on Unmanned Aircraft Systems (ICUAS) Athens, Greece. June 15-18, 2021*, 1494–98.

Swinney, Carolyn J, and John C Woods. 2020. “Unmanned Aerial Vehicle Flight Mode Classification Using Convolutional Neural Network and Transfer Learning.” *2020 16th International Computer Engineering Conference (ICENCO)*, 83–87.

———. 2021b. “The Effect of Real-World Interference on CNN Feature Extraction and Machine Learning Classification of Unmanned Aerial Systems.” *Aerospace* 8(7) (179): 1–18. https://doi.org/https://doi.org/10.3390/aerospace8070179.

Thota, Narayana Bhagirath, and Doshna Umma Reddy. 2020. “Improving the Accuracy of Diabetic Retinopathy Severity Classification with Transfer Learning.” *Midwest Symposium on Circuits and Systems* 2020-Augus (August): 1003–6. https://doi.org/10.1109/MWSCAS48704.2020.9184473.

Viswanathan, Mathuranathan. 2019. “Digital Modulations Using Python.” Edited by Varsha Srinivasan. https://www.amazon.co.uk/Digital-Modulations-using-Python-Color/dp/1712321633/ref=sr_1_3?dchild=1&keywords=digital+modulation+python&qid=1614176971&sr=8-3.

Wu, Zhilu, Yanlong Zhao, Zhendong Yin, and Haochen Luo. 2018. “Jamming Signals Classification Using Convolutional Neural Network.” *2017 IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2017*, June, 62–67. https://doi.org/10.1109/ISSPIT.2017.8388320.

Xu, Hao, Yufan Cheng, Jindi Liang, and Pengyu Wang. 2020. “A Jamming Recognition Algorithm Based on Deep Neural Network in Satellite Navigation System.” *China Satellite Navigation Conference 2020* 652 LNEE (May): 701–11. https://doi.org/10.1007/978-981-15-3715-8_63.

BIOGRAPHICAL NOTES

Carolyn J. Swinney received a B.Eng.(hons.) degree (first class) in 2007 and a M.Sc.(dist.) in Electronics Engineering from the University of Essex, Colchester, UK in 2013. She graduated as a Communications and Electronics Engineering Officer in the Royal Air Force in 2014. She currently works within the Air and Space Warfare Centre and is working towards a Ph.D. degree in Electronic Systems Engineering at the University of Essex, Colchester, UK. Her main research interests are signal processing, unmanned aerial vehicles, neural networks, machine learning and cyber security.

John C. Woods was born in a small fishing village near Colchester, U.K., in 1964. He received the B.Eng. (hons.) degree (first class) in 1996 and the Ph.D. degree in 1999 from the University of Essex, Colchester, UK. He has been a Lecturer in the Department of Computer Science and Electronic Systems Engineering, University of Essex, since 1999. Although his field of expertise is image processing, he has a wide range of interests including telecommunications, autonomous vehicles and robotics.

REFERENCE

Reference to this paper should be made as follows: Swinney, Carolyn J., Woods John C. (2021). GPS Jamming Signal Classification using CNN Feature Extraction in low Signal to Noise Environments. *International Journal on Cyber Situational Awareness*, Vol. 6, No. 1, pp1-21.