Received: 12 October 2021 | Revised: 8 January 2022 | Accepted: 10 January 2022

DOI: 10.1002/nme.6927

RESEARCH ARTICLE

WILEY

Identification of metallic objects using spectral magnetic polarizability tensor signatures: Object classification

Ben A. Wilson[1] | **Paul D. Ledger**[2] | **William R. B. Lionheart**[3]

[1]Zienkiewicz Centre for Computational Engineering, Swansea University, Swansea, UK

[2]School of Computing & Mathematics, Keele University, Keele, UK

[3]Department for Mathematics, The University of Manchester, Manchester, UK

Correspondence
Paul D. Ledger, School of Computing & Mathematics, Keele University, Keele, UK.
Email: p.d.ledger@keele.ac.uk

Funding information
Engineering and Physical Sciences Research Council, Grant/Award Numbers: EP/R002134/2, EP/R002177/1, EP/V009028/1, EP/V009109/1, EP/V049453/1, EP/V049496/1; Royal Society

Abstract

The early detection of terrorist threat objects, such as guns and knives, through improved metal detection, has the potential to reduce the number of attacks and improve public safety and security. To achieve this, there is considerable potential to use the fields applied and measured by a metal detector to discriminate between different shapes and different metals since, hidden within the field perturbation, is object characterization information. The magnetic polarizability tensor (MPT) offers an economical characterization of metallic objects and its spectral signature provides additional object characterization information. The MPT spectral signature can be determined from measurements of the induced voltage over a range of frequencies in a metal signature for a hidden object. With classification in mind, it can also be computed in advance for different threat and non-threat objects. In this article, we evaluate the performance of probabilistic and non-probabilistic machine learning algorithms, trained using a dictionary of computed MPT spectral signatures, to classify objects for metal detection. We discuss the importance of using appropriate features and selecting an appropriate algorithm depending on the classification problem being solved, and we present numerical results for a range of practically motivated metal detection classification problems.

KEYWORDS

finite element method, machine learning, magnetic polarizability tensor, metal detection, object classification, reduced order model

1 | INTRODUCTION

The purpose of this article is to compare the performance of probabilistic and non-probabilistic machine learning (ML) algorithms, trained using object characterization information, to classify objects for metal detection. Key applications include in the discrimination between threat and non-threat objects in security screening, whereby the early detection of knives and guns has the potential to reduce the number of attacks and improve safety and security. Further applications include distinguishing between metallic clutter (e.g., ring-pulls, coins, shrapnel) and threat items for the improved identification of hidden anti-personnel mines and unexploded ordinance (UXO) in areas of former conflict, improving identification of metallic objects of significance in archaeological searches and treasure hunts, improving

non-destructive testing as well as discriminating between real and fake coins in vending machines and automatic checkouts.

This article builds on our earlier work[1] and our recent `MPT-Library` open data set,[2] where we have established an extensive suite of simulated threat and non-threat object characterizations using their magnetic polarizability tensor (MPT) spectral signatures. The complex symmetric MPT,

$$\begin{aligned}\mathcal{M}[\alpha B, \omega, \sigma_*, \mu_r] &= (\mathcal{M}[\alpha B, \omega, \sigma_*, \mu_r])_{ij}\boldsymbol{e}_i \otimes \boldsymbol{e}_j \\ &= (\tilde{\mathcal{R}}[\alpha B, \omega, \sigma_*, \mu_r] + \mathrm{i}\mathcal{I}[\alpha B, \omega, \sigma_*, \mu_r])_{ij}\boldsymbol{e}_i \otimes \boldsymbol{e}_j, \end{aligned} \tag{1}$$

has been shown to offer an economical characterization of conducting permeable objects and explicit formulae for the computation of its 6 independent complex coefficients, $\mathcal{M}\ [\alpha B, \omega, \sigma_*, \mu_r])_{ij}$, which are a function of the exciting angular frequency, ω, the object's size, α, its shape, B, as well as its conductivity, σ_*, and relative permeability, μ_r, have been obtained.[3-6] In the above, $\tilde{\mathcal{R}}[\alpha B, \omega, \sigma_*, \mu_r]$ and $\mathcal{I}[\alpha B, \omega, \sigma_*, \mu_r]$ denote the MPT's real imaginary parts, $\mathrm{i} := \sqrt{-1}$ and $\boldsymbol{e}_i$ is the ith orthonormal unit coordinate vector. The MPT spectral signature refers to the fact that the MPT is captured as a function of frequency, which has been shown to provide important additional object characterization information.[7] Our suite of object characterizations has been obtained using our `MPT-Calculator` software,[8] which employs a reduced order model based on proper orthogonal decomposition (POD) for efficient calculations and an a posteriori error estimate to certify its predictions with respect to full order model solutions provided by the open source finite element library `NGSolve`.[9-12] Our computational MPT spectral signature simulations are in excellent agreement with measured MPT spectral signatures[13,14] and, thus, they provide realistic object characterizations.

Measured MPT spectral signatures have previously been used in conjunction with a k nearest neighbors (KNN) classification algorithm[15] and other ML approaches.[16] In addition, existing examples of practical MPT classification of objects include in airport security screening,[15,17] waste sorting,[18] and anti-personal landmine detection.[19] In such situations, induced voltages are measured over a range of frequencies by a metal detector from which the MPT spectral signature of the hidden object is obtained and then a classifier applied.[20-22] However, dictionaries of measured MPT coefficients contain unavoidable errors if the object is placed in a non-uniform background field that varies significantly over the object as well as other errors and noise associated with capacitive coupling with other low-conducting objects or soil in the background. There will also be other general noise (e.g., from amplifiers, parasitic voltages and filtering).[23] This means the accuracy of the measured MPT coefficients is about 1–5 %,[15,23,24] depending on the application. Furthermore, obtaining a large dictionary is time-consuming and there may be practical challenges in acquiring a sufficiently large selection of objects.

Simulated MPT spectral signatures and a simple dictionary classifier were employed for homogenous[25] and inhomogeneous[6] objects, respectively, but the full potential of using a dictionary of simulated MPT spectral signatures and ML classifiers has yet to be investigated. Dictionaries of simulated MPT spectral signatures allow noise at an appropriate level to the desired application to be added and the richness of the dataset allows them to be applied to a large range of different metal detectors operating at different frequencies. Furthermore, our `MPT-Library` can be further extended to characterize other objects of different sizes and with different conductivities using simple scaling results at negligible computational cost,[26] which make it ideal for creating (very) large datasets needed for training ML classifiers that would be impractical with measured MPT spectral signatures.

In this work, we present the first systematic review of the performance of a wide range of probabilistic and non-probabilistic ML classifiers for discriminating between different classes of metallic objects using a dictionary of simulated MPT spectral signatures. The practical classification problems we consider are relevant for discriminating between threat and non-threat objects in security screening and discriminating between real and fake coins in vending machines and automatic checkouts. However, our article is not only of interest to metal detection experts, but is also of educational value to engineers interested in learning about ML classification algorithms. In particular, the classification problem we present is of a medium size, which cannot be trivially solved by hand, is different to well-known image classification problems, and, therefore, presents an interesting test case for investigating the performance of different ML algorithms.

The novelties of our article are as follows: We present a novel approach to training ML classifiers using our simulated MPT-Library enhanced by simple scaling results to create large dictionaries of objects. We choose tensor invariants of MPT spectral signatures as novel object features and explain the benefits of our approach over eigenvalues of MPT used in all previous ML classifiers for this problem. We include a novel well-reasoned approach for justifying the performance of different ML classifiers for practical classification problems using uncertainty quantification, statistical analysis, and

ML metrics. Furthermore, we explore the ability of our classification approaches to classify unseen threat objects and also discuss the limitations of our current approach.

The presentation of the material proceeds as follows: In Section 2, we briefly review the use of invariants of MPT spectral signatures as ML features and describe the creation of our dictionary for training ML algorithms. Then, in Section 3, we review a range of probabilistic and non-probabilistic classifiers that will be considered in this work and describe a simple approach for investigating uncertainty. Section 4 discusses a range of ML metrics for assessing the performance of classifiers. This is followed in Section 5 by the application of the classifiers to a series of practically motivated classification problems, where each dataset is analyzed and the suitability of ML classifiers investigated and their subsequent performance critically analyzed. We finish with some concluding remarks.

2 | MPT SPECTRAL SIGNATURE INVARIANTS FOR OBJECT CLASSIFICATION

Bishop[27] describes the process of classification as taking an input vector $\mathbf{x}$ and assigning it to one of K discrete classes C_k, $k = 1, \dots, K$. For example, in security screening, the simplest form of classification with $K = 2$ involves only the classes *threat object* (C_1) and *non-threat object* (C_2), and one with a higher level of fidelity might include the classes of metallic objects such as key (C_1), coin (C_2), gun (C_3), knife (C_4), … where the class numbers are assigned as desired. He recommends that it is convenient to use a 1-of-K coding system in which the entries in a vector $\mathbf{t} \in \mathbb{R}^K$ take the form

$$\mathrm{t}_i := \begin{cases} 1 & \text{if } i = k, \\ 0 & \text{otherwise,} \end{cases}$$

if the correct class is C_k. Requiring that we always have $\sum_{k=1}^{K} \mathrm{t}_k = 1$, then this approach has the advantage that t_k can be interpreted as the probability that the correct class is C_k.

In Reference 1, we have considered different choices for the F features in the input vector $\mathbf{x} \in \mathbb{R}^F$, which are associated with either the eigenvalues, principal invariants or deviatoric invariants of $\tilde{\mathcal{R}}[\alpha B, \omega, \sigma_*, \mu_r]$ and $\mathcal{I}[\alpha B, \omega, \sigma_*, \mu_r]$, respectively, evaluated at different frequencies $\omega = \omega_m$, $m = 1, \dots, M$. In this article, we restrict our focus to the situation where

$$\mathrm{x}_i = \begin{cases} I_j(\tilde{\mathcal{R}}[\alpha B, \omega_m, \sigma_*, \mu_r]), & i = j + (m-1)M, \\ I_j(\mathcal{I}[\alpha B, \omega_m, \sigma_*, \mu_r]), & i = j + (m+2)M, \end{cases} \tag{2}$$

with $j = 1, 2, 3$, $m = 1, \dots, M$. For exact arithmetic, and a rank 2 tensor $\mathcal{A}$,

$$I_1(\mathcal{A}) := \mathrm{tr}(\mathcal{A}) = \lambda_1(\mathcal{A}) + \lambda_2(\mathcal{A}) + \lambda_3(\mathcal{A}), \tag{3a}$$

$$I_2(\mathcal{A}) := \frac{1}{2}\left(\mathrm{tr}(\mathcal{A})^2 - \mathrm{tr}(\mathcal{A}^2)\right) = \lambda_1(\mathcal{A})\lambda_2(\mathcal{A}) + \lambda_1(\mathcal{A})\lambda_3(\mathcal{A}) + \lambda_2(\mathcal{A})\lambda_3(\mathcal{A}), \tag{3b}$$

$$I_3(\mathcal{A}) := \det(\mathcal{A}) = \lambda_1(\mathcal{A})\lambda_2(\mathcal{A})\lambda_3(\mathcal{A}), \tag{3c}$$

are the principal invariants. In the above, $\mathrm{tr}(\cdot)$ denotes the trace, $\det(\cdot)$ the determinate and we assume the entries of $\mathcal{A}$ are arranged as a symmetric 3×3 matrix, which we also call $\mathcal{A}$. We define $\mathbf{x}$ in this way for the following reasons:

1. Using features that are invariant to object rotation is important as both a hidden object's shape and its orientation are unknown. Using either eigenvalues $\lambda_i(\tilde{\mathcal{R}})$, $\lambda_i(\mathcal{I})$, $i = 1, 2, 3$ or the principal tensor invariants overcomes this issue as both are invariant to an object's unknown orientation and, hence, simplifies the classification problem.
2. Invariants overcome the ordering issue that is associated with assigning the eigenvalues as the invariants are independent of how the eigenvalues are assigned.
3. The invariants can be computed as either products or sums of the entries of $\mathcal{A}$ without first calculating $\lambda_i(\mathcal{A})$. Hence, they are smooth functions of the tensor coefficients. Rather than a sub-determinant method, an alternative approach for finding I_3 follows by first converting $\mathcal{A}$ to (upper) triangular form and the determinant follows by the product of its diagonal entries. However, while the eigenvalues of a triangular matrix are its diagonal entries, the eigenvalues

of $\mathcal{A}$ are not preserved when it is converted and further computation is needed. There are many iterative computational alternatives for finding $\lambda_i(\mathcal{A})$ (see, e.g., Reference 28), which are preferred to directly finding the roots of $\det(\mathcal{A} - \lambda\mathbb{I}) = 0$, especially for large matrices. Although, even a simple low-cost approach for directly determining the eigenvalues of a symmetric 3×3 matrix still requires determining an inverse cosine of a non-linear function of the matrix's trace and determinant[29] and, hence, involves non-smooth operations. Thus, the finding $\lambda_i(\mathcal{A})$ may result in a loss of accuracy in practical numerics compared to using the simple sums or products to find $I_i(\mathcal{A})$.

4. The (probabilistic) ML classification algorithms we will consider are better at capturing an underlying relationship between features and the likelihood of class that is smooth, albeit, with noisy data. The processes involved in finding eigenvalues may lead to a greater entanglement between class and features that a classifier might need to unravel compared to using invariants.

As an example, Figure 1 shows a comparison of the principal tensor invariants for a selection of 4 different metallic watch styles computed by the `MPT-Calculator` software. These calculations were performed in a similar manner to those for other geometries in Reference 1. The object dimensions are in mm so $\alpha = 0.001$ m and the results shown are for the case where the material is gold, so that $\sigma_* = 4.25 \times 10^7$ S/m and $\mu_r = 1$ (`MPT-Library` also includes MPT spectral signatures for watches made of platinum and silver). An unstructured mesh of tetrahedra is used to discretize each object and the truncated unbounded region which surrounds it, resulting in meshes ranging from 14,935 to 175,217 elements. In each case, the truncated boundary for the non-dimensional transmission problem is $[-1000, 1000]^3$. Order $p = 4$ elements were applied on the meshes and snapshot solutions obtained at 13 logarithmically spaced frequencies over the range $1 \leq \omega \leq 1 \times 10^{10}$ rad/s. The MPT spectral signature for each object was produced using the projected proper orthogonal decomposition (known as PODP) method[26] using a relative singular value truncation of 10^{-4}. Also shown is a vertical line, which indicates the value of ω that the eddy current model assumption is likely to become inaccurate for this geometry.[1,30] Finally, we include a grey window corresponding to the frequency range $5.02 \times 10^4 \leq \omega \leq 8.67 \times 10^4$ rad/s, where measurements taken by a commercial walk through metal detector,[15] and the greater range $7.53 \times 10^2 \leq \omega \leq 5.99 \times 10^5$ rad/s, where measurements are taken using recent MPT measurement system,[13] the latter being able to capture more information from the signature. These spectral signatures will form part of the dictionary for object classification, which will be discussed later in Section 5.2.

2.1 | Construction of the dictionary

Each class C_k may be comprised of $G^{(k)}$ geometries and, in addition, we consider $V^{(k)}$ variations in object size and object materials so that each class is comprised of $P^{(k)}$ different samples. In total, over all the classes, we have $P = \sum_{k=1}^{K} P^{(k)}$ samples.

Given the information α, B, σ_*, μ_r the MPT spectral signature described by $\tilde{\mathcal{R}}[\alpha B, \omega, \sigma_*, \mu_r]$ and $\mathcal{I}[\alpha B, \omega, \sigma_*, \mu_r]$ can be obtained, as described above, and then the invariants follow from (2). We can repeat this process for each of the geometries $B^{(g_k)}$, $g_k = 1, \ldots, G^{(k)}$ that makes up the class. To take account of the $V^{(k)}$ different object sizes and materials, we draw physically motivated samples $\alpha \sim N(m_\alpha, s_\alpha)$ and $\sigma_* \sim N(m_{\sigma_*}, s_{\sigma_*})$, where $N(m, s)$ denotes a normal distribution with mean m and standard deviation s. While it would be possible to also obtain the MPT spectral signature using the `MPT-Calculator` software for each sample, instead, we reduce the computational cost of obtaining these spectral signatures by using the simple scaling results we have derived in Lemmas 2 and 3 of Reference 26. Given an MPT spectral signature of an object for a given α, B, μ_r, and σ_*, these results predict the MPT spectral signature of another object with the same B and μ_r, but different σ_* and α, at negligible computational cost. The results hold for objects with homogeneous materials and there are no restrictions on ω (up to the limit of the eddy current model), but assume that a broader band MPT spectral signature is available for the original object compared to the gray windows highlighted in Figure 1. The invariants then again follow from (3). Finally, by setting the class labels, the pairs $(\mathbf{x}_p \in \mathbb{R}^F, \mathbf{t}_p \in \mathbb{R}^K)$, $p = 1, \ldots, P^{(k)*}$ for all the samples that make up the class C_k are obtained. Repeating this process for each of the classes gives rise to the general dictionary

$$D = ((\mathbf{x}_1, \mathbf{t}_1), (\mathbf{x}_2, \mathbf{t}_2), \ldots, (\mathbf{x}_P, \mathbf{t}_P)), \tag{4}$$

or, alternatively,

$$D = (D^{(1)}, D^{(2)}, \ldots, D^{(K)}), \tag{5}$$

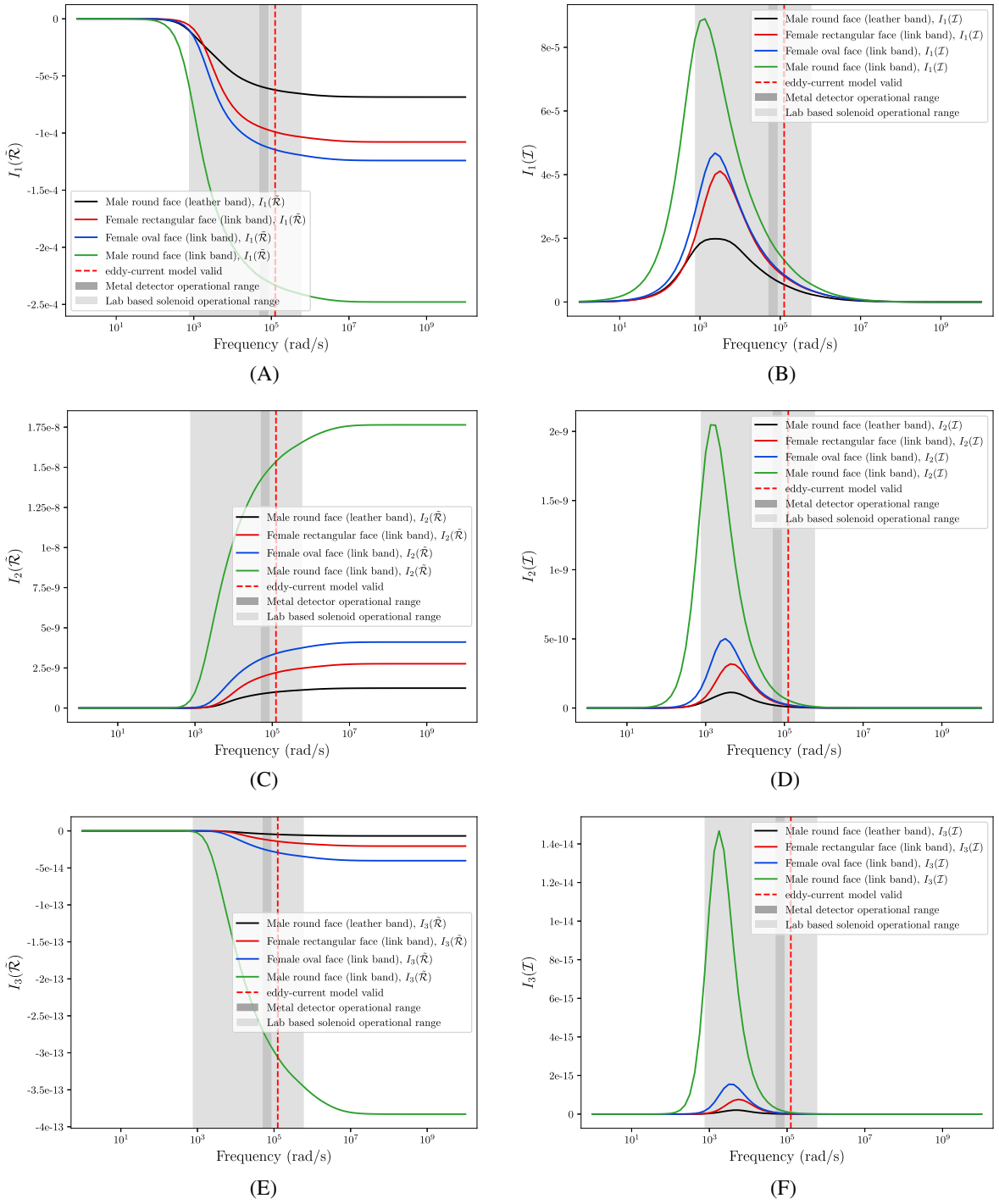

FIGURE 1 Set of watches: Comparison of tensor invariants. (A) $I_1(\tilde{\mathcal{R}})$, (B) $I_1(\mathcal{I})$, (C) $I_2(\tilde{\mathcal{R}})$, (D) $I_2(\mathcal{I})$, (E) $I_3(\tilde{\mathcal{R}})$, and (F) $I_3(\mathcal{I})$

where

$$D^{(k)} = ((\mathbf{x}_1, \mathbf{t}_1), (\mathbf{x}_2, \mathbf{t}_2), \ldots, (\mathbf{x}_{P^{(k)}}, \mathbf{t}_{P^{(k)}})), \tag{6}$$

is the dictionary associated with class C_k and consists of $P^{(k)}$ observations.

In practice, we split the dictionary stated in (4) as $D = (D^{(\text{train})}, D^{(\text{test})})$ where $D^{(\text{train})}$ is the training and $D^{(\text{test})}$ is the testing dataset, respectively. The purpose of this splitting is to enable the classifier to be trained on given data $D^{(\text{train})}$ and then tested on previously unseen data $D^{(\text{test})}$. Although there is no optimal choice for the ratio of training to testing, we

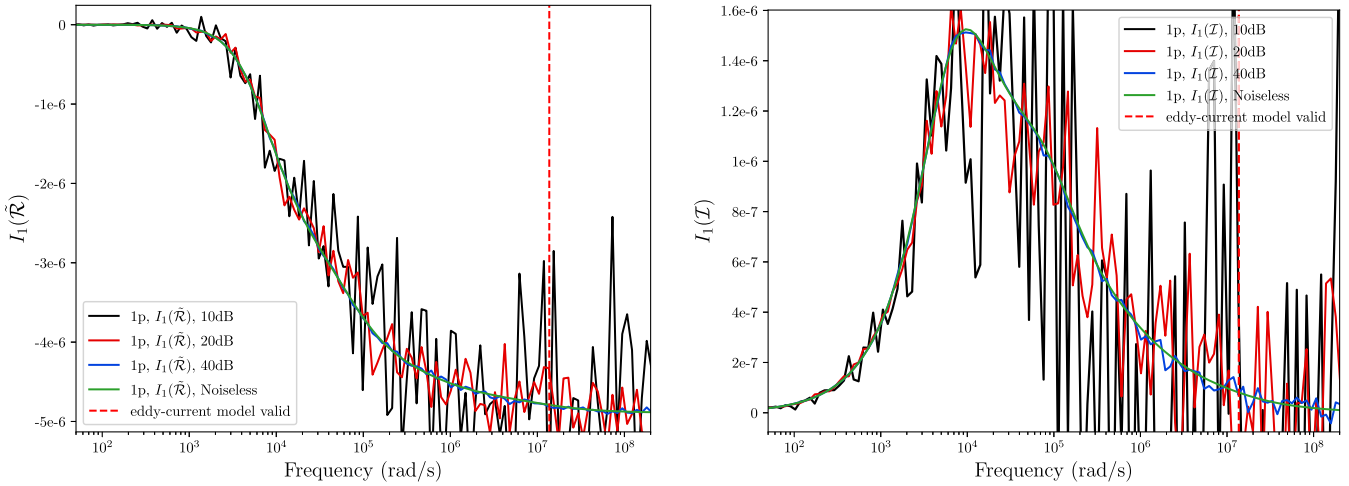

FIGURE 2 Set of British coins: 1 p coin αB, $\alpha = 0.001$ m, $\mu_r = 1$, and $\sigma^* = 4.03 \times 10^7$ S/m showing the spectral signatures for noiseless and realizations of noise with SNRs of 40, 20, and 10 dB. (A) $I_1(\tilde{\mathcal{R}}[\alpha B, \omega, \sigma_*, \mu_r])$ and (B) $I_1(\mathcal{I}[\alpha B, \omega, \sigma_*, \mu_r])$

will employ a ratio of 3:1 throughout, which is commonly used in ML classification and is in the range of 3:1 to 4:1 that Kuhn and Johnson[31(p71)] suggest.

2.2 | Noise

When MPT spectral signatures for hidden objects are measured by a metal detector they will contain un-avoidable errors, as pointed out in Reference 26. For example, if an object is placed in a non-uniform background magnetic field that varies significantly over the object there is a modeling error since the background field in the rank 2 MPT model assumes the field over the object is uniform. There are other errors and noise associated with capacitive coupling with other low-conducting objects or soil, if the object is buried, as well as other generals noise (e.g., from amplifiers, parasitic voltages, and filtering).[23] The accuracy of the signature can be improved by repeating the measurements and applying averaging filters, at the cost of spending more time to take the measurements. However, in a practical setting, there is trade-off to be made in terms of improving the accuracy against the measurement time and, consequently, the accuracy of the measured MPT coefficients is about 1–5 %,[15,23,24] depending on the application.

The MPT spectral signature coefficients we will use have been produced numerically using our `MPT-Calculator` software tool.[1,26] This means that the MPT coefficients are obtained with higher accuracy than can currently be achieved from practical measurements, since the spectral signature is accurately computed for a large frequency range (up to the limit of the eddy current model) rather than noisy measurements being taken at a small number of discrete frequencies. The advantage of this is it allows a much larger library of objects and variations of materials to be considered, which is all highly desirable for achieving greater fidelity and accuracy when training an ML classifier. But, for practical classification, noise appropriate to the system must be added.

After filtering and averaging, the noise remaining in an MPT spectral signature measured by a metal detector can be well approximated by Gaussian additive noise, with a level dependent on the factors identified above. Hence, in this work, we add noise to our simulated MPT spectral signatures in the following way: We specify a signal noise ratio (SNR) in decibels and use this to determine the amount of noise to add to each of the complex tensor coefficients $(\mathcal{M}[\alpha B^{(p)}, \omega, \sigma_*, \mu_r])_{ij}$ as a function of frequency for each object $\alpha B^{(p)}$ in the dictionary. Considering each of the i,jth MPT coefficients individually, we introduce

$$\mathrm{v} := (\mathcal{M}[\alpha B^{(p)}, \omega_m, \sigma_*, \mu_r])_{ij},$$

and calculate a noise-power measure as

$$\text{Noise} = \frac{\bar{\text{v}}\text{v}}{10^{\text{SNR}/10}}.$$

The noisy coefficients are then specified as

$$(\mathcal{M}[\alpha B^{(p)}, \omega_m, \sigma_*, \mu_r])_{ij} + e_{ij}, \quad \text{where } e_{ij} = \sqrt{\frac{\text{noise}}{2}}(u + \text{i}v),$$

with $u, v \sim N(0, 1)$. The above process is repeated for the 6 independent coefficients of the complex symmetric MPT and for each frequency in the spectral signature. SNR values of 40, 20, and 10 dB lead to values of $|\frac{e_{ij}}{\mathcal{M}_{ij}}| = 0.01, 0.10$, and 0.32 on average, which is equivalent to 1 %, 10 %, and 32 % noise, respectively. One realization of the effect of the added noise on $I_1(\tilde{\mathcal{R}})$ and $I_1(\mathcal{I})$ for a British one penny coin can be seen in Figure 2. Further details about the MPT spectral signature of British coins are provided in Section 5.1.1. Note that our model for noise results in a noise power measure that varies over the MPT spectral signature according to $\bar{\text{v}}\text{v}$. If the physical system behaves differently, this can be taken in to account by applying an appropriate model for the noise at this stage. Once the noisy $\mathcal{M}$ at each ω_m are found, the principal invariants of the real and imaginary parts of $\mathcal{M}$ at each ω_m easily follow. Hence, an entry $(\mathbf{x}, \mathbf{t}) \in D$ is replaced with $(\mathbf{x}^{\text{noise}}, \mathbf{t})$. By repeating this for all objects leads to the updated dictionary D.

3 | CLASSIFICATION

In this section, we provide a quick hands-on review of ML classification. Readers who are familiar with this subject should skip this section as this material can be found in the references we cite below.

3.1 | Probabilistic versus non-probabilistic classification

Applied to classification problems, Bayes' theorem can be expressed in the form[27]

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})}, \quad k = 1, \dots, K, \tag{7}$$

which relates the posterior probability density function $p(C_k|\mathbf{x})$ to the likelihood probability density function $p(\mathbf{x}|C_k)$ and the prior probability density function $p(C_k)$ where, for classification,

$$p(\mathbf{x}) = \sum_{n=1}^{K} p(\mathbf{x}|C_n)p(C_n),$$

is easily explicitly obtained as the normalizing constant.

In the inference stage of probabilistic classification, one seeks to design a classifier $\gamma_k(\mathbf{x})$ that provides a probabilistic output, which approximates $p(C_k|\mathbf{x})$. On the other hand, non-probabilistic classifiers either predict a class C_k with certainty or, more commonly, have a statistical interpretation that provides a frequentist approximation $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to $p(C_k|\mathbf{x})$. One measure of accuracy of classification is the mean squared error (MSE)

$$\text{MSE}(\gamma_k) = E_{\mathbf{x}}[\gamma_k(\mathbf{x}) - p(C_k|\mathbf{x})]^2, \tag{8}$$

where $E_{\mathbf{x}}$ is the expectation with respect to $p(\mathbf{x})$.[32(p309)] If desired, this can be summed over the classes $k = 1, \dots, K$ or considered for each class. Other metrics are considered in Section 4.

Given approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ for classes C_k, $k = 1, \dots, K$, the class decision is typically achieved using the maximum a posteriori (MAP) estimate $\arg\ \max_{k\in K}(p(C_k|\mathbf{x})) \approx \arg\ \max_{k\in K}(\gamma_k(\mathbf{x}))$, that is, the MAP estimate corresponds to the class C_n with n such that

$$n = \arg\max_{k\in K}(\gamma_k(\mathbf{x})). \tag{9}$$

However, the MAP may have drawbacks if their are several similar probabilities and/or if the data is noisy. Hence, understanding the uncertainty in the approximations of $p(C_k|\mathbf{x})$ are also important. We consider this in Section 3.5.

3.2 | Bias and variance

The classifiers we will consider are based on ML algorithms. For an ML method Γ, which takes $\mathbb{D} = D^{(\text{train})}$ as the input and returns a learned classifier γ_k, $k = 1, \dots, K$, Manning et al.[32(pp309–312])] define the Learning-error(Γ) $= E_{\mathbb{D}}[\text{MSE}(\Gamma_{\mathbb{D}})]$ as a measure of accuracy of the classifier, which is to be minimized, and they show that it can be expressed as

$$\begin{aligned}\text{Learning-error}(\Gamma) &:= E_{\mathbf{x}}[\text{bias}(\Gamma, \mathbf{x}) + \text{variance}(\Gamma, \mathbf{x})],\\ \text{bias}(\Gamma, \mathbf{x}) &:= [p(C_k|\mathbf{x}) - E_{\mathbb{D}}\Gamma_{\mathbb{D}}(\mathbf{x})]^2,\\ \text{variance}(\Gamma, \mathbf{x}) &:= E_{\mathbb{D}}[\Gamma_{\mathbb{D}}(\mathbf{x}) - E_{\mathbb{D}}\Gamma_{\mathbb{D}}(\mathbf{x})]^2,\end{aligned}$$

where $\Gamma_{\mathbb{D}}(\mathbf{x})$ implies that the ML method is applied to data set $\mathbb{D}$ and outputs approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to $p(C_k|\mathbf{x})$, $k = 1, \dots, K$, for data $\mathbf{x}$.

Bias measures the difference between the true $p(C_k|\mathbf{x})$ and the prediction $\Gamma_{\mathbb{D}}(\mathbf{x})$ averaged over the training sets.[32(p311)] Bias is large if the classifier is consistently wrong, which may stem from erroneous assumptions. A small bias may indicate several things and not just that the classifier is consistently correct, for further details, see Reference 32. Related to high bias is underfitting, which refers to a classifier that is unable to capture the relationship between the input and output variables correctly and produces large errors on both the training and testing data sets.

Variance is the variation of the prediction of learned classifiers and is the average difference between $\Gamma_{\mathbb{D}}(\mathbf{x})$ and its average $E_{\mathbb{D}}\Gamma_{\mathbb{D}}(\mathbf{x})$.[32(p311)] Variance is large if different training sets give rise to different classifiers and is small if the choice of training set has only a small influence on the classification decisions, for further details, see Reference 32. Related to high variance is overfitting, which is the opposite of underfitting, and refers to a classifier that has too much complexity and also learns from the noise resulting in high errors on the test data.

The ideal classifier would be a classifier having low bias and low variance, however, the two are inextricably linked and there is therefore a trade off between two.[27,32] While the performance of classifiers is very application dependent, linear classifiers tend to have a high bias and low variance and non-linear classifiers tend to have a low bias and high variance.[32] The ML methods Γ that we will consider are commonly found in established ML libraries such as `scikit-learn`, which is the implementation that we will use. The finer details of the different methods can be found in References 27, 33, and 34, among many others, although we give a brief summary of the methodologies for those less familiar with the approaches and to set notation. We start with non-probabilistic methods and then proceed to probabilistic classifiers.

3.3 | Non-probabilistic classifiers

3.3.1 | Decision trees

Tree based algorithms can simply be interpreted as making a series of (binary) decisions that ultimately lead to the prediction of a class. For $F = 2$, this results in the partition of the feature space into a series of K rectangular regions corresponding to the different classes. To establish the regions, and, hence the classes, a tree is constructed where, at each node, a binary decisions about a component of $\mathbf{x}$, and the process is terminated with a decision at the leaf-node (for example, if $x_1 \le t_1$ and $x_2 \le t_2$ then the class is C_1, whereas if $x_1 \le t_1$ and $x_2 > t_2$ the class is C_2, etc.). In order to grow a tree, a greedy algorithm is applied to decide how to split the variables, the split points and the topology of the tree.[33(p308)] Growing a tree that is too large may overfit the data, while a small tree may not capture the structure. To overcome this, a larger than needed tree is usually grown and is then pruned. To understand how this works, consider a tree, then by applying the rules of the tree, a subset of the training data for its m node is obtained. It then follows that $\hat{p}_{mk}$ is the proportion of this data that has class C_k and the associated class is determined as $k(m) = \arg\max_k \hat{p}_{mk}$. The pruning is then achieved by applying a cost-complexity optimization based on the node impurity measures *misclassification error*, *Gini index*, or *cross-entropy*, which can each be written in terms of [(non-linear) functions and/or sums of] $\hat{p}_{mk}$. For further details, see Reference 27 (p666) and Reference 33 (p309). Decision trees are often used as a base classifier for more advanced ensemble methods such as gradient boost and random forests discussed below.

3.3.2 | Random forests

The random forests classifier is an example of what is known as "a bootstrap aggregating (bagging) algorithm" that attempts to reduce the variance of the typically high-variance low-bias decision tree algorithm.[33(p587)] As Hastie et al.[33(p587)] continue to describe, the idea behind bagging is to average many noisy, but unbiased models to reduce their variance. Trees are notoriously noisy, and hence benefit from averaging. Since each tree generated by bagging is identically distributed then expectation of an average of such trees is the same as the expectation of any of them. This means that bias unchanged, but random forests offer the hope of improvement by variance reduction. For details of their implementation, we refer to Reference 33 (Chap15).

3.3.3 | Support vector machine

A support vector machine (SVM) classifier generalizes simple linear classifiers (e.g., Fisher's linear discriminant analysis) by producing non-linear, rather than linear, classification boundaries. These are obtained by constructing a linear boundary in a transformed version of the feature space, which becomes nonlinear in the feature space.[27(p325),33(Chap12),35] So, after making a transformation $\boldsymbol{\psi}(\mathbf{x}) : \mathbb{R}^F \to \mathbb{R}^{\tilde{F}}$ with $\tilde{F} \geq F$ being possibly infinite dimensional,[†] the goal of the classifier is to learn how to determine $\mathbf{w}$ and w_0 in

$$G(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}) + \mathrm{w}_0, \tag{10}$$

with $\|\mathbf{w}\| = 1$ such that it predicts C_1(*say*) if $G(\mathbf{x}) < 0$ and C_2 if $G(\mathbf{x}) > 0$. For the separable case, the idea behind SVM is to find the hyperplane that creates the biggest margin, defined by $2M$, between the training data describing the two classes. Given N training points $(\mathbf{x}_1, \mathrm{y}_1), (\mathbf{x}_2, \mathrm{y}_2), \ldots, (\mathbf{x}_N, \mathrm{y}_N)$ with $\mathrm{y}_i \in \{-1, 1\}$ indicating the class label, then this problem can be framed as the optimization problem

$$\begin{aligned} &\max_{\mathbf{w}, \mathrm{w}_0, \|\mathbf{w}\|=1} M, \\ &\text{subject to } \mathrm{y}_i(\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i) + \mathrm{w}_0) \geq M, \quad i = 1, \ldots, N, \end{aligned} \tag{11}$$

which can also be rephrased as a convex optimization problem (quadratic criterion and linear constraints). In the non-separable case, slack variables $s_1, \ldots, s_N$ are introduced to deal with points that lie on the wrong side of the margin and the linear constraint is replaced by $\mathrm{y}_i(\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i) + \mathrm{w}_0) \geq M(1 - s_i)$. For further details, and its computational implementation using Lagrange multipliers, see Reference 33 (p420). This practical implementation involves the introduction of symmetric positive definite or symmetric positive semi-definite kernel functions $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\psi}(\mathbf{x}')^T \boldsymbol{\psi}(\mathbf{x})$,[33(p424)] which avoids the introduction of $\boldsymbol{\psi}(\mathbf{x})$ itself, but imposes limitations on their choice in order that optimization problem remains convex. Typically kernel types include polynomial, Gaussian, and radial basis function kernels, however, we limit our investigation to the latter in this article.

To apply SVM to multi-class problems, the problem is reduced into a series of binary classification problems. This is done by employing either an *ovo* (one vs. one) or an *ovr* (one vs. rest) strategy, we have chosen the former, where an SVM is trained for each possible binary classification. This leads to $\frac{K(K-1)}{2}$ models being trained, for example in a 3-class problem ($K = 3$) we would train a classifier to separate the pairs (C_1, C_2), (C_1, C_3), and (C_2, C_3), we then create a voting scheme based on these classifiers.

3.4 | Probabilistic classification

We will consider problems with $K > 2$ and, in this case, it is beneficial for the practical probabilistic classifiers we describe below to write (7) in the form of the softmax function (also known as the normalized exponential)

$$p(C_k|\mathbf{x}) = \sigma(\mathbf{x}) := \frac{\exp a_k}{\sum_{k=1}^{K} \exp a_k}, \tag{12}$$

where $a_k = \ln(p(\mathbf{x}|C_k)p(C_k))$.

If $p(\mathbf{x}|C_k)$ has a simple Gaussian form, the evaluation of $p(C_k|\mathbf{x})$ for given $\mathbf{x}$ becomes explicit. However, in many practical cases, $p(\mathbf{x}|C_k)$ will have a complicated form and this will dictate the use of classifier $\gamma_k(\mathbf{x})$ that approximates $p(C_k|\mathbf{x})$ instead. Nonetheless, all probabilistic classifiers benefits from establishing (approximations of) the likelihood of each of the classes C_k, $k = 1, \dots, K$ rather than just a single output.

We explore some alternative ML methods Γ, which provide probabilistic classifiers, below.

3.4.1 | Logistic regression

In the case that $p(\mathbf{x}|C_k)$ has a simple Gaussian form, a suitable linear classifier is logistic regression.[27(Chap4)] This is based on the following assumptions

1. The likelihood probability distribution is Gaussian:

$$p(\mathbf{x}|C_k) = \frac{1}{(2\pi)^{F/2}|\mathbf{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{m}_k)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{m}_k)\right),$$

   where $\mathbf{m}_k$ is the mean of all $\mathbf{x}_i$, $(\mathbf{x}_i, \mathbf{t}_i) \in D^{(k)}$, that are associated with class C_k and $\mathbf{\Sigma}$ is a covariance matrix.
2. The covariance matrix $\mathbf{\Sigma}$ is common to all the classes.

In this case, the evaluation of $p(C_k|\mathbf{x})$ is explicit with a_k in (12) replaced with the rescaled $\tilde{a}_k$ for $K > 2$[27]

$$\tilde{a}_k = \tilde{\mathbf{w}}_k^T\mathbf{x} + \tilde{\mathrm{w}}_{k0},$$

where

$$\tilde{\mathbf{w}}_k = \mathbf{\Sigma}^{-1}\mathbf{m}_k,$$
$$\tilde{\mathrm{w}}_{k0} = -\frac{1}{2}\mathbf{m}_k\mathbf{\Sigma}^{-1}\mathbf{m}_k + \ln p(C_k).$$

In the generative approach, the learning involves first computing $\mathbf{m}_k$ and $\mathbf{\Sigma}$ directly from the training data $D^{(\text{train})}$, while in the discriminative approach, the $(K-1)(F+1)$ coefficients of $\tilde{\mathbf{w}}_k$ and w_{k0} compared to the $KF + F^2/2$ coefficients needed otherwise are found by numerical optimization from $D^{(\text{train})}$. When applied to other data sets, this results in an approximation $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to $p(C_k|\mathbf{x})$. Note, that only $\tilde{\mathbf{w}}_k$ and $\tilde{\mathrm{w}}_{k0}$ for $k = 1, \dots, K-1$ need to be determined since we know $\sum_{k=1}^{K} p(C_k|\mathbf{x}) = 1$, which allows the approximation $p(C_K|\mathbf{x}) \approx \gamma_K(\mathbf{x})$ to be found from $\gamma_k(\mathbf{x})$, $k = 1, \dots, K-1$.

3.4.2 | Multi-layer perceptron

If $p(\mathbf{x}|C_k)$ does not have a simple form, and $p(C_k|\mathbf{x})$ is not explicit, then the multi-layer perceptron (MLP) neural network can be applied in an attempt to approximate $p(C_k|\mathbf{x})$.[27(Chap5)] For example, for a $K > 2$ class problem using 3 layers (with 1 input, 1 hidden, and 1 output) and J internal variables (neurons) in the hidden layer, the approximation to $p(C_k|\mathbf{x})$ takes the form

$$p(C_k|\mathbf{x}) \approx \ \gamma_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{J} \mathrm{w}_{kj}^{(2)}\sigma\left(\sum_{i=1}^{F} \mathrm{w}_{ji}^{(1)}(\mathbf{x})_i + \mathrm{w}_{j0}^{(1)}\right) + \mathrm{w}_{k0}^{(2)}\right), \tag{13}$$

where $\mathrm{w}_{ji}^{(1)}$, $\mathrm{w}_{kj}^{(2)}$ are the $J(F+1+K)+K$ coefficients of $\mathbf{w}$ to be found from network training and $\sigma(\cdot)$ is the softmax activation function defined in (12). In our case, the input layer comprises of the features $\mathbf{x} \in \mathbb{R}^F$ and output layer are the approximate posterior probabilities $\gamma_k(\mathbf{x}, \mathbf{w})$, $k = 1 \dots, K$. If the number of internal variables in each hidden layer is fixed at J, and there are L hidden layers, then the total number of parameters, $\mathrm{w}_{ji}^{(1)}$, $\mathrm{w}_{ji}^{(2)}$, …, which describe the network, that need to be found are $J^2(L-1) + J(F+L+K) + K$. Given N training points $(\mathbf{x}_1, \mathbf{t}_1), (\mathbf{x}_2, \mathbf{t}_2), \dots, (\mathbf{x}_N, \mathbf{t}_N)$, and following a maximum likelihood[27(p232)] approach, the parameters are found by optimizing a logloss error function evaluated over the training data set

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K}(\mathbf{t}_n)_k \ln \gamma_k(\mathbf{x}_n, \mathbf{w}), \tag{14}$$

or, alternatively, they can be found by a Bayesian approach.[27(p277)]

As remarked by Richard and Lippmann,[36] MLP can provide good estimates of $p(C_k|\mathbf{x})$ if sufficient training data are available, if the network is complex enough and if the classes are sampled with the correct a priori class probabilities in the training data. Nonetheless, designing appropriate networks, with the correct number of hidden layers and neurons, can be challenging. Furthermore, a complex network with a large number of neurons can require a large amount of training data to avoid overfitting.

3.4.3 | Gradient boost

Gradient boost is an example of what is known as a "boosting algorithm."[37,38] Boosting attempts to build a stronger classifier by combining the results of weaker base classifiers through a weighted majority vote[33(Chap10)] and, in the case of gradient boost, this is achieved through optimization using steepest descent. As described by Friedman,[37] gradient boost can be applied to approximating $p(C_k|\mathbf{x})$ in probabilistic classification. By again considering (14), with $\gamma_k(\mathbf{x}_n, \mathbf{w})$ replaced by (12) with $\mathbf{x} = \mathbf{x}_n$, and choosing the parameters $\mathbf{w} = \mathbf{w}(\mathbf{x})$ as $(\mathbf{w}(\mathbf{x}))_k = \mathrm{w}_k(\mathbf{x}) = a_k(\mathbf{x})$, then the kth component of the negative gradient of the loss function is

$$r_k = -\frac{\partial E(\mathbf{w})}{\partial w_k} = \sum_{n=1}^{N}(\mathbf{t}_n)_k - \gamma_k(\mathbf{x}_n, \mathbf{w}).$$

Starting with an initial guess $a_k^{[0]}(\mathbf{x}) = 0$, $k = 1, \ldots, K$, then, for a given $\mathbf{x}$, an iterative procedure is used to improve the estimate $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x}, \mathbf{w}^{[m]})$ of $p(C_k|\mathbf{x})$ at iteration m. In this procedure, K decision trees are trained at each iteration to predict r_k, $k = 1, \ldots, K$, and the leaf nodes of the tree are then used to update $a_k^{[m]}(\mathbf{x})$ until a convergence criteria is reached. For details of the practical implementation, see Reference 37.

3.5 | Understanding uncertainty in classification

For the majority of the classifiers we will consider, an ML algorithm $\Gamma_{\mathbb{D}}$ trained on dictionary $\mathbb{D}$ produces a classifier $\Gamma_{\mathbb{D}}(\mathbf{x}) = \gamma_k(\mathbf{x})$, which approximates $p(C_k|\mathbf{x})$ and provides an indication of the likelihood of the class C_k being correct. We then base the decision as to the correct class based on the MAP estimate. When this process is repeated for different pairs $(\mathbf{x}_i, \mathbf{t}_i) \in \mathbb{T} = D^{(\text{test})}$, $\gamma_k(\mathbf{x}_i)$ may be different for each $\mathbf{x}_i$. It is useful to explore how sensitive $\gamma_k(\mathbf{x}_i)$ is to changes in $\mathbf{x}_i$ when it is evaluated for different $(\mathbf{x}_i, \mathbf{t}_i) \in \mathbb{T}_\ell = D^{(\text{test},(\ell))}$ associated with the test data for one class C_ℓ. To do this, we consider confidence intervals for the average $E_{\mathbb{T}_\ell}\gamma_k(\mathbf{x})$.

A first approach might be to use the sample mean and sample variance

$$\overline{\gamma_k} = \frac{1}{P^{(\text{test},(\ell))}}\sum_{i=1}^{P^{(\text{test},(\ell))}}\gamma_k(\mathbf{x}_i) \quad \text{and} \quad S_k = \sqrt{\frac{\sum_{i=1}^{P^{(\text{test},(\ell))}}(\gamma_k(\mathbf{x}_i) - \overline{\gamma_k})^2}{P^{(\text{test},(\ell))}}},$$

to construct an interval in the form

$$\overline{\gamma_k} - CV\frac{S_k}{\sqrt{P^{(\text{test},(\ell))}}} \le E_{\mathbb{T}_\ell}\gamma_k(\mathbf{x}) \le \overline{\gamma_k} + CV\frac{S_k}{\sqrt{P^{(\text{test},(\ell))}}}. \tag{15}$$

In the above, CV is a critical value based on a t-test and the confidence level chosen. However, in practice, if $\overline{\gamma}_k \to 0.5$ as the sample size $P^{(\text{test},(\ell))} \to \infty$ and, we have small confidence bounds, we might wrongly conclude that $\gamma_k(\mathbf{x}) = 0.5$ with a high degree of confidence. Instead, this may also indicate that half the observations are predicting $\gamma_k(\mathbf{x}_i) \approx 0$ and the half are predicting $\gamma_k(\mathbf{x}_i) \approx 1$, which has the same sample mean. This can occur, since at most $S_k = 0.5$, and, for large $P^{(\text{test},(\ell))}$, we find the confidence bounds produced by (15) are narrow due to division by this quality in computation of the bounds. Hence, $\overline{\gamma_k}$ and (15) do not give any insight into the variation within the different observations $\gamma_k(\mathbf{x}_i)$.

Instead, ordering $\gamma_k(\mathbf{x}_i)$ as

$$O(C_k) = (\gamma_k \mathbf{x}_1), \gamma_k(\mathbf{x}_2), \dots, \gamma_k(\mathbf{x}_{P^{(\text{test},(\ell))}})), \quad \text{such that} \quad \gamma_k(\mathbf{x}_i) \leq \gamma_k(\mathbf{x}_{i+1}), \tag{16}$$

we define the yth-percentile

$$\gamma_{k,y} = (O(C_k))_{\frac{y}{100}P^{(\text{test},(\ell))}}, \tag{17}$$

and use interpolation between neighboring values if $\frac{y}{100}$ is not an integer. We then consider the median value of $\gamma_k(\mathbf{x})$, given by $\gamma_{k,50}$, as the average and use the percentiles corresponding to $Q_1 \equiv \gamma_{k,25}$, $Q_3 \equiv \gamma_{k,75}$ and $\gamma_{k,5}$, $\gamma_{k,95}$ to understand uncertainty in the predictions.

4 | EVALUATING THE PERFORMANCE OF CLASSIFIERS

We describe metrics for assessing performance are applicable to both probabilistic and non-probabilistic classifiers given $D^{(\text{train})}$ and $D^{(\text{test})}$ data sets.

4.1 | Metrics

4.1.1 | Confusion matrices, precision, sensitivity, and specificity

We begin by recalling the definitions of true positive, false positive, true negative, and false negative for a given class C_k (see, e.g., Reference 39).

- True positive (TP), the case where the classifier predicts $\mathbf{x}$ *belongs* to C_k and is *correct* in its prediction.
- False positive (FP) (type 1 error), the case where the classifier predicts $\mathbf{x}$ *belongs* to C_k and is *incorrect* in its prediction.
- True negative (TN), the case where the classifier predicts $\mathbf{x}$ *does not belong* to C_k and is *correct* in its prediction.
- False negative (FN) (type 2 error), the case where the classifier predicts $\mathbf{x}$ *does not belong* to C_k and is *incorrect* in its prediction.

Following the training of a classifier, its performance can be evaluated on the test data set $D^{(\text{test})}$. Applying the classifier to each sample $(\mathbf{x}_n, \mathbf{t}_n) \in D^{(\text{test},(i))}$, where the true class label is C_i, the number of predictions of each class $C_j, j = 1, \dots, K$ can be counted and the result recorded in the $(\mathbf{C})_{ij}$th element of a confusion matrix $\mathbf{C} \in \mathbb{R}^{K \times K}$. Repeating this process for $i = 1, \dots, K$ leads to the complete matrix.[‡] The 4 cases (TP, FP, TN, FN) for each class C_k can be defined in terms of $(\mathbf{C})_{ij}$ as[34,39]

$$\text{TP}(C_k) := (\mathbf{C})_{kk}, \qquad \text{FN}(C_k) := \sum_{\substack{j=1 \\ j \neq k}}^{K} (\mathbf{C})_{kj},$$

$$\text{FP}(C_k) := \sum_{\substack{i=1 \\ i \neq k}}^{K} (\mathbf{C})_{ik}, \quad \text{TN}(C_k) := \sum_{\substack{i=1 \\ i \neq k}}^{K} \sum_{\substack{j=1 \\ j \neq k}}^{K} (\mathbf{C})_{ij},$$

and the precision, sensitivity, and specificity for each of the classes C_k using[39]

$$\text{Precision}(C_k) := \frac{\text{TP}(C_k)}{\text{TP}(C_k) + \text{FP}(C_k)} = \frac{\text{TP}(C_k)}{\#\text{predicted positives for } C_k},$$

$$\text{Sensitivity}(C_k) := \frac{\text{TP}(C_k)}{\text{TP}(C_k) + \text{FN}(C_k)} = \frac{\text{TP}(C_k)}{\#\text{actual positives for } C_k},$$

$$\text{Specificity}(C_k) := \frac{\text{TN}(C_k)}{\text{TN}(C_k) + \text{FP}(C_k)} = \frac{\text{TN}(C_k)}{\#\text{predicted negatives for } C_k}.$$

The precision and sensitivity (also known as the true positive rate or recall) are measures of the proportion of positives that are correctly identified and specificity (also called the true negative rate) measures the proportion of negatives that are correctly identified.

The entries in confusion matrices are often presented as frequentist probabilities (i.e., $(\mathbf{C})_{ij}$ is normalized by $\sum_{p=1}^{K}\sum_{q=1}^{K}(\mathbf{C})_{pq}$), which, as the sample size $P^{(k)}$ becomes large, provides an approximation to $p(C_j|\mathbf{x})$ with $(\mathbf{x},\mathbf{t}) \in D^{(\text{test},(i))}$. We will also present confusion matrices in this way.

4.1.2 | κ score

Possible choices for a metric, which provides an overall score of the performance of the classifier, include accuracy, the F_1 score and Cohen's κ score.[39-43] Variants of the commonly used F_1 score include the macro-averaged F_1 score (or macro F_1 score), the weighted-average F_1 score (or weighted F_1 score), and the micro-averaged F_1 score (micro F_1 score). However, the F_1 score can sometimes lead to an incorrect comparison of classifiers.[40,44] As Powers[44] notes, the macro F_1 score is not normalized, which is overcome by the weighed F_1 score and the F_1 score is not symmetric with respect to positive and negative cases. Some of these drawbacks are taken in to account by using the micro F_1 score, however, the κ score also takes into account chance agreement.[41,42] This is useful when comparing problems with both, differing numbers of instances per class and differing numbers of classes as it takes the chance a naive classifier has into account. For these reasons, we will use the κ score defined as

$$\kappa := \frac{\text{Accuracy} - \text{Random accuracy}}{1 - \text{Random accuracy}}, \tag{18}$$

for comparing classifiers where

$$\text{Accuracy} := \frac{\sum_{k=1}^{K}\text{TP}(C_k)}{\sum_{k=1}^{K}\text{TP}(C_k) + \text{FN}(C_k)},$$

$$\text{Random accuracy} := \sum_{k=1}^{K}\frac{(\text{TP}(C_k) + \text{FN}(C_k)) \cdot (\text{TP}(C_k) + \text{FP}(C_k))}{(\text{TP}(C_k) + \text{FP}(C_k) + \text{TN}(C_k) + \text{FN}(C_k))^2}.$$

4.2 | Validation methods

Evaluating the performance of different classifiers can be considerably enhanced by employing cross validation.[31] This is particularly important if $D^{(\text{test})}$ is small and, otherwise, may lead to inaccurate predictions of a classifier's performance. We have chosen to employ the Monte Carlo cross validation (MCCV) technique (also known as "leave group out cross validation").[31(p71)] This involves performing ℓ iterations where, for each iteration, the dataset D is split into training and testing $D = (D^{(\text{test})}, D^{(\text{train})})$ with $D^{(\text{train})}$ and $D^{(\text{test})}$ being drawn differently from D each time, irrespective of the splittings in previous iterations. Other variants of cross validation include k-fold cross validation, repeated k-fold cross validation and bootstrapping, for further details, see Kuhn and Johnson.[31] Kuhn and Johnson explain that no resampling method is uniformly better than another and that the differences between the different methodologies is small for larger samples sizes, which further motivates that actually performing cross validation is more important than the method chosen for doing so.

5 | RESULTS

5.1 | Classification of British coins

5.1.1 | Construction of the coin dictionary

To create the coin dictionary, we follow the approach described in Section 2.1. We choose the kth class C_k, $k = 1, \dots, K = 8$, to correspond to the kth British coin denomination one penny (1 p), two pence (2 p), five pence (5 p), ten pence (10 p),

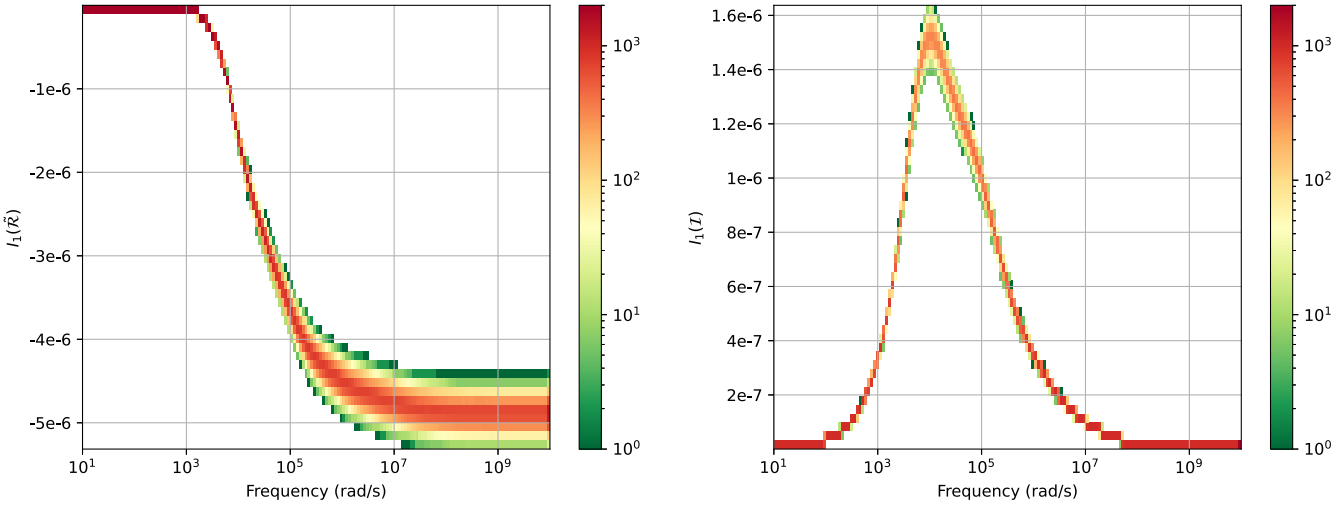

FIGURE 3 Set of British coins: 1 p coin (class C_1) for $P^{(k)} = P/K = 2000$, with $\alpha \sim N(0.001, 8.4 \times 10^{-6})$ m and $\sigma_* \sim N(4.03 \times 10^7, 9.52 \times 10^5)$ S/m showing the histograms for the distribution of the spectral signatures (A) $I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega, \sigma_*, \mu_r])$ and (B) $I_1(\mathcal{I}[\alpha B^{(1)}, \omega, \sigma_*, \mu_r])$

twenty pence (20 p), fifty pence (50 p), one pound (£ 1), two pounds (£ 2), respectively, in order of increasing value. The coins have different geometries, and, in some cases different materials,[1] although, within each class, we restrict ourselves to a single geometry $G^{(k)} = 1$ and consider a fixed $P^{(k)} = P/K$ number of samples for each class so that $P^{(k)} = V^{(k)}$ in this case. We use $P^{(k)} = 2000$ for coin classification unless otherwise stated. The shape $B^{(k)}$ of the kth coin class is as described in the specification of Table 1 in Reference 1 and in our `MPT-Library`[2] we have previously obtained the MPT spectral signatures for each coin geometry. From these, we choose the signatures evaluated at M equally spaced ω_m such that $5.02 \times 10^4 \leq \omega_m \leq 8.67 \times 10^4$ rad/s, although we have also considered the larger frequency range of $7.53 \times 10^2 \leq \omega_m \leq 5.99 \times 10^5$ rad/s, which leads to very comparable results[45] to those presented here. To account for the fact that the measured MPT coefficients will be noisy, we illustrate in Figure 2 realizations of noise being added to the spectral signatures of $I_1(\tilde{\mathcal{R}})$ and $I_1(\mathcal{I})$ for the 1 p coin. The curves in this figure correspond to the cases of no noise and noise with SNR values of 40, 20, and 10 dB.

The $V^{(k)}$ variations account for the fact that within each class there can be:

1. Variation in the object size α, such that the volume in the different observations of coins of a certain denomination can change. While it is expected that a coins size may be fairly uniform when they leave the mint, they are likely to become increasingly bashed and dented once they enter circulation, hence their object size may change over time. Hence we choose the object size to be ±2.52 % of the coins specification. We set $m_\alpha = 0.001$ m for each coin and set $s_\alpha = 0.001(1.0252 - 1)/3 = 8.4 \times 10^{-6}$ m, that is, to be 1/3 the difference of the upper limit and the respective mean.
2. Variation in the object conductivity/conductivities $\sigma_*(\boldsymbol{x})$ to account for variations in the manufacturing process, such that the conductivity in the different observations of coins of a certain denomination can change. In most cases, the coins are assumed to be homogeneous conductors, but for £ 1 and £ 2 coin denominations the objects are each an annulus. As the coins dominant composition material is copper, using[46,47] we find an upper limit for conductivity to be ±7.09 % of the coins specification. We set m_{σ_*} corresponding to the conductivities of each coin denomination, so for a 1 p coin, for example, $m_{\sigma_*} = 4.307 \times 10^7$ S/m and set s_{σ_*} in a similar way to s_α, so that $s_{\sigma_*} = 4.307 \times 10^7(1.0709 - 1)/3 = 9.52 \times 10^5$ S/m.
3. Note that the object's permeability will be fixed as $\mu_r = 1$ as we assume all the coins considered are non-magnetic.[1]

Given that $p\left(-3 \leq \frac{\alpha - m_\alpha}{s_\alpha} \leq 3\right) = 0.9974$, we expect 99.74 % of the object sizes generated to fall within our prescribed variation in object sizes due to being bashed and dented in circulation. Similarly, we expect 99.74 % of the σ_* values generated to fall within our prescribed variation in σ_*. Overall, this means that $0.9974 \cdot 0.9974 = 0.9948$ or 99.48 % of the values generated are representative of genuine currency.

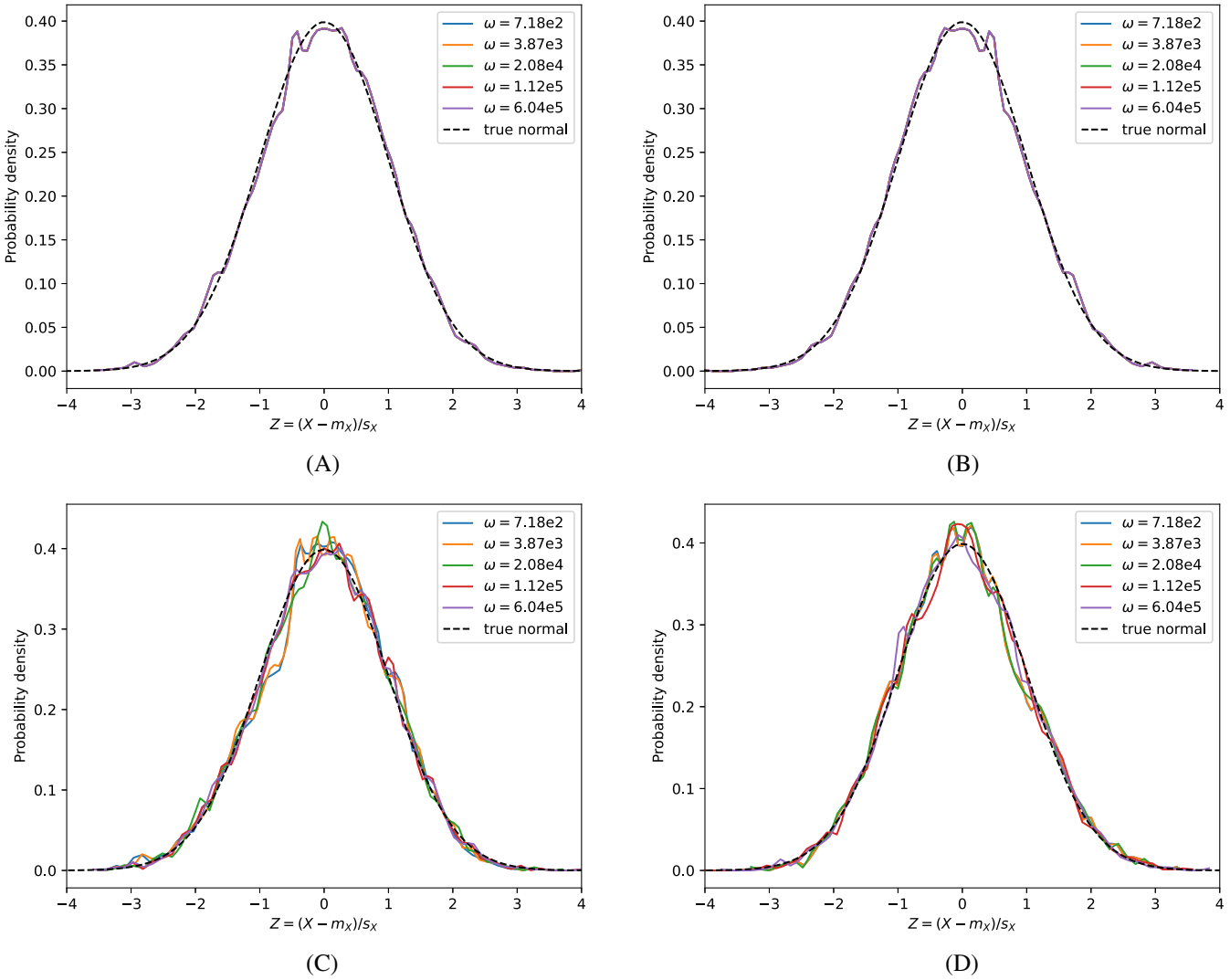

FIGURE 4 Set of British coins: 1 p coin (class C_1) for $P^{(k)} = P/K = 2000$, with $\alpha \sim N(0.001, 8.4 \times 10^{-6})$ m and $\sigma_* \sim N(4.03 \times 10^7, 9.52 \times 10^5)$ S/m showing the histograms of $(Z - m_X)/s_X$, presented in the form of probability densities, where X is instances of the following (A) $I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$, (B) $I_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$, (C) $\lambda_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$, and (D) $\lambda_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ at distinct frequencies ω_m

The effect of these samples on the MPT spectral signature is illustrated in Figure 3 for the 1 p coin class (C_1) and noiseless data. We now explore this further: Given $\alpha \sim N(0.001, 8.4 \times 10^{-6})$ m, $\sigma_* \sim N(4.03 \times 10^7, 9.52 \times 10^5)$ S/m, drawing $P^{(k)}$ samples of α and σ_* and applying the scaling results in Lemmas 2 and 3 of Reference 26, we obtain the histograms of the random variables $X = I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r]) \sim p(\mathrm{x}_{1+(m-1)M}|C_1)$ and $X = I_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r]) \sim p(\mathrm{x}_{1+(m+2)M}|C_1)$ shown in Figure 3. Then, by taking cross sections at selected frequencies ω_m we obtain the histograms shown in Figure 4, which are presented in the form of probability densities. The corresponding distributions obtained by sampling $X = \lambda_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ and $X = \lambda_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ are also included in this figure. In each case, these distributions have been normalized using the transformation $Z = (X - m_X)/s_X$, where m_X and s_X indicates the mean and standard deviation of X[§] and a curve of best fit made through the histogram. As a comparison, the standard normal distribution is included. Notice that the normalized sample distributions of $X = I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ for different ω_m are identical, as are those for $X = I_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ for different ω_m, and have a close fit to the standard normal distribution. The conclusion is similar for the other invariants and other coins. This outcome can be explained by the central limit theorem, which implies, given a large enough sample size, we expect the samples of $I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ and $I_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ to follow a normal distribution even if the parent distribution is not normal.[¶] The normalized sample distributions of $X = \lambda_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ and $X = \lambda_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ also follow a normal distribution, but the fit is not as good as for the invariants. The results are similar for other eigenvalues and other coins. For

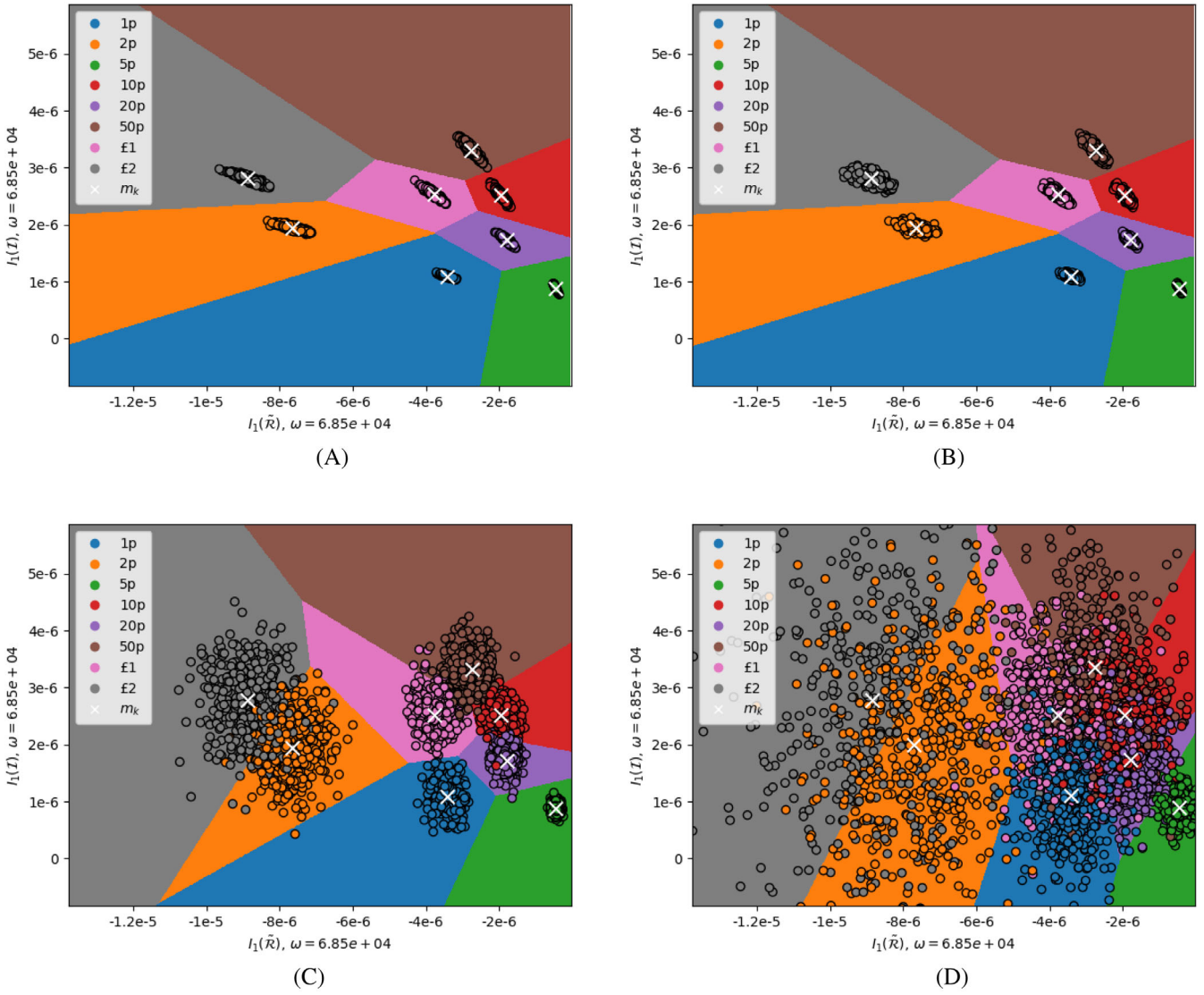

FIGURE 5 Set of British coins: linear feature space splitting for the classes C_k, $k = 1, \ldots, K$, for a simplified case of $F = 2$ features based on $\max_k \gamma_k(\mathbf{x})$ for logistic regression and $P^{(k)} = P/K = 2000$ for (A) noiseless and SNR of (B) 40 dB, (C) 20 dB, (D) 10 dB

our chosen $\alpha \sim N(0.001, 8.4 \times 10^{-6})$ m, $\sigma_* \sim N(4.03 \times 10^7, 9.52 \times 10^5)$ S/m and smaller sample sizes, the distributions of $X = I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$, $X = I_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$, $X = \lambda_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$, and $X = \lambda_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ still approximately follow a normal distribution with the fit being superior for the invariants. By considering different instances of noise, similar histograms to those shown in Figure 4 can be obtained and again similar conclusions about the resulting distributions of the eigenvalues and invariants at each ω_m apply.

5.1.2 | Classification results

For the coin classification problem, we restrict consideration to the logistic regression classifier and the default settings of `scikit-learn`, as Figure 4 indicates that a normal distribution is a good approximation for the sample distributions of $I_i(\tilde{\mathcal{R}}[\alpha B^{(k)}, \omega_m, \sigma_*, \mu_r])$ and $I_i(\mathcal{I}[\alpha B^{(k)}, \omega_m, \sigma_*, \mu_r])$, $i = 1, 2, 3$, for a sufficiently large sample size. We have also observed that the feature space can be separated linearly. To illustrate this, we begin by examining the simplest case of just $F = 2$ features, $I_1(\tilde{\mathcal{R}}[\alpha B^{(k)}, \omega_1, \sigma_*, \mu_r])$ and $I_1(\mathcal{I}[\alpha B^{(k)}, \omega_1, \sigma_*, \mu_r])$, and $M = 1$ with $\omega_1 = 6.85 \times 10^4$ rad/s. Figure 5 shows the class boundaries when the MAP estimate (9) is applied for different levels of noise, the crosses indicate the locations of the means $\mathbf{m}_k$ for each class obtained from $D^{(\text{train})}$ and the circles indicate the samples from $D^{(\text{test})}$ assuming a 3:1 training–testing $D = (D^{(\text{train})}, D^{(\text{test})})$ splitting and MCCV with $\ell = 100$ (as described in Section 4.2), which we employ

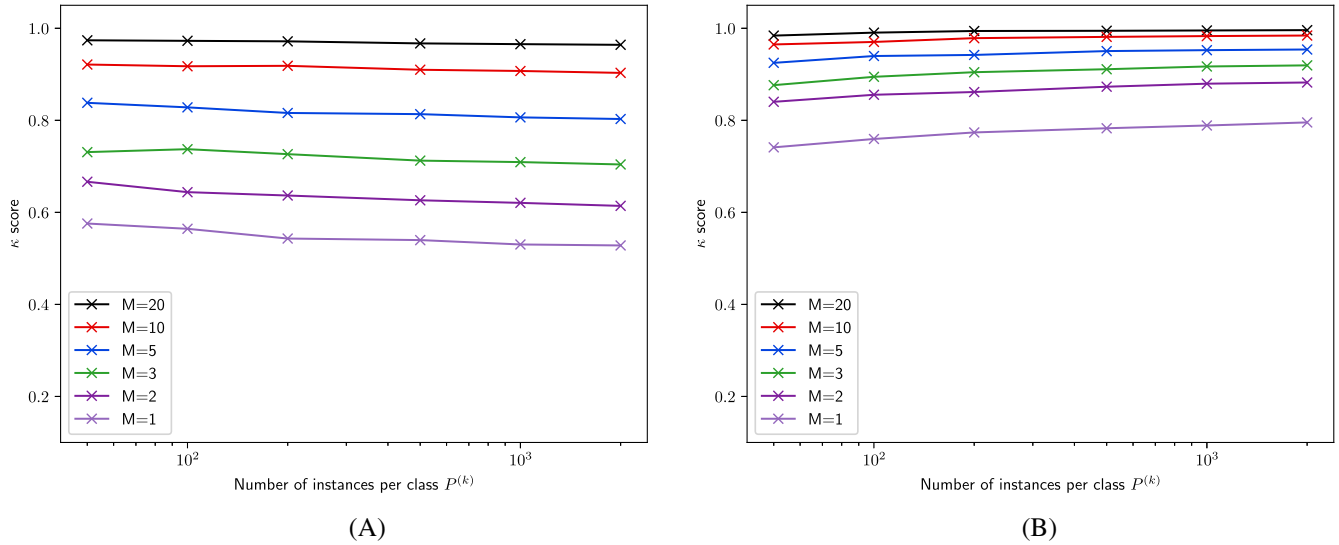

FIGURE 6 Set of British coins: Overall performance of logistic regression classifier as a function of $P^{(k)}$ and different numbers of frequencies M using the κ score (18) for a testing noise SNR = 10 dB, for (A) noiseless training data and (B) training data with SNR = 10 dB

throughout. From this figure, we observe the class boundaries change only slightly if noise with SNR of 40 dB is added and, with greater noise, the changes to the boundaries are only moderate. It is also possible to see that the number of misclassifications is very small for SNR = 40 and 20 dB and still modest for 10 dB, which has a $\kappa = 0.66$ score using (18). Furthermore, and importantly, the locations of the means $\mathbf{m}_k$ for each class do not change significantly for $P^{(k)} = P/K = 2000$ since using this large number of instance per class has the effect of largely averaging out the effects of noise and the object variations that we previously illustrated in Figures 2 and 3. While this figure indicates that the samples form a $p(\mathbf{x}|C_k)$ that is normally distributed, especially for noiseless and noisy data with SNR = 40 and 20 dB, which is consistent with the assumptions of this classifier, described in Section 3.4.1, we observe that the assumption of a common covariance matrix between the classes does not hold for coin data set. The variance between the features is anisotropic for each cluster, as indicated by different sized and different orientated ellipses, which also becomes increasingly apparent for increased noise levels. While logistic regression typically has a high-bias and low-variance, we expect its bias to be lower for this problem than others given the above.

This behavior also carries over when we use $F = 6M$ and greater M. In Figure 6, we illustrate the overall performance of the classifier using the κ score (18) as a function of $P^{(k)}$ for test data with SNR = 10 dB noise and (A) for noiseless training data and (B) noisy training data with SNR = 10 dB. The different curves correspond to $M = 1, 2, 3, 5, 10, 20$ frequencies. The curve for $M = 1$ corresponds to the same frequency considered in Figure 5, but has $F = 6$ features instead of $F = 2$. Increasing M also increases F and, for either noiseless or noisy training data, the classifier's performance is improved for fixed $P^{(k)}$ as more feature information is available in $\mathbf{x} \in \mathbb{R}^F$ for each $(\mathbf{x}, \mathbf{t}) \in D^{(\text{train})}$ and, hence, it becomes easier for the classifier to find relationship between the features and classes and, in the decision stage, partitioning according to (9) becomes easier for larger F. On the other hand, increasing $P^{(k)}$, for a fixed M and noiseless training data, reduces the κ score and increases the variability as the classifier becomes increasingly overfitted to the training data and experiences more misclassifications as $P^{(k)}$ is increased. For noisy training data, the classifier is exposed to more noisy data in $D^{(\text{train})}$ as $P^{(k)}$ is increased and, hence, its performance improves and its variability decreases. The relatively high accuracy of logistic regression for the coin classification problem, even with an SNR of 10 dB, can in part be attributed to how well the assumptions of logistic regression hold in practice for this problem and the normalization of the data that is performed prior to training.

We consider further the relationship between noise level, number of frequencies and classifier performance in Figure 7. This figure shows the noise level against M, with the contours indicating the resulting κ score for fixed $P^{(k)}$. The concentric curves correspond to all the systems with a M and a noise level that achieve the same accuracy. As is to be expected, results for the classifier can be improved by increasing both M and SNR (or either). This figure is of practical value as it allows practitioners to choose M, given an SNR, in order to achieve a desired level of accuracy.

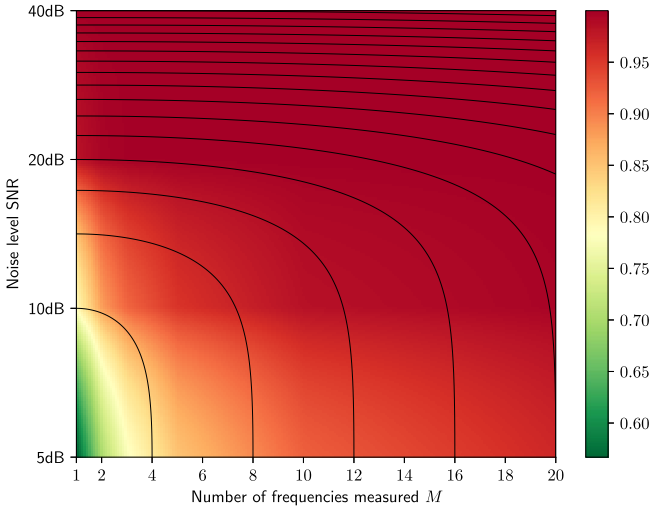

FIGURE 7 Set of British coins: Overall performance of logistic regression classifier for $P^{(k)} = 2000$, comparing M and SNR using the κ score (18)

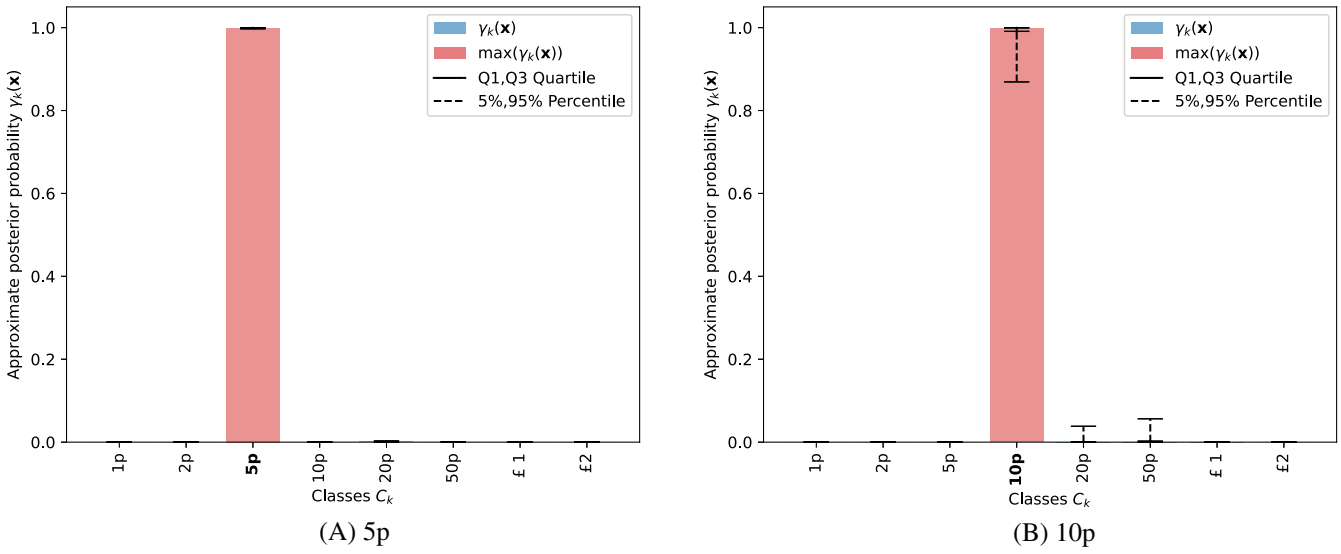

FIGURE 8 Set of British coins: Approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, using the logistic regression classifier for $P^{(k)} = 2000$ where (A) $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(3))}$ and (B) $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(4))}$

Some illustrative approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, that are obtained for SNR = 10 dB are illustrated in Figure 8. For each $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test})}$, a potentially different distribution can be expected and, in the cases shown, we have chosen $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(3)}$ and $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(4))}$ so that correct classifications should be C_3 (a 5 p coin) and C_4 (a 10 p coin), respectively. Additionally, the bars we show are for the median value $\gamma_{k,50}$, obtained by considering all the samples $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(3))}$ and $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(4))}$, respectively, and we also indicate the Q_1, Q_3 quartiles as well as $\gamma_{k,5}$ and $\gamma_{k,95}$ percentiles, which have been obtained using (17). The cases shown correspond to the best and worst cases among all $D^{(\text{test},(k))}$ for this level of noise. A common trait of logistic regression is that it gives a strong $\gamma_k(\mathbf{x})$ for one class and low values for the other classes and the results we obtain also exhibit this. Comparing $\gamma_k(\mathbf{x})$, $k = 1, \ldots, K$, for $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(3)}$ and $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(4))}$, we find the most likely classes correspond to the 5 p and 10 p coins, respectively. For the 5 p coin, the inter quartile and inter percentile ranges are small and so we have high confidence in this prediction and a low variability. For the 10 p coin, they are larger indicating we have less confidence in the prediction and a higher variability.

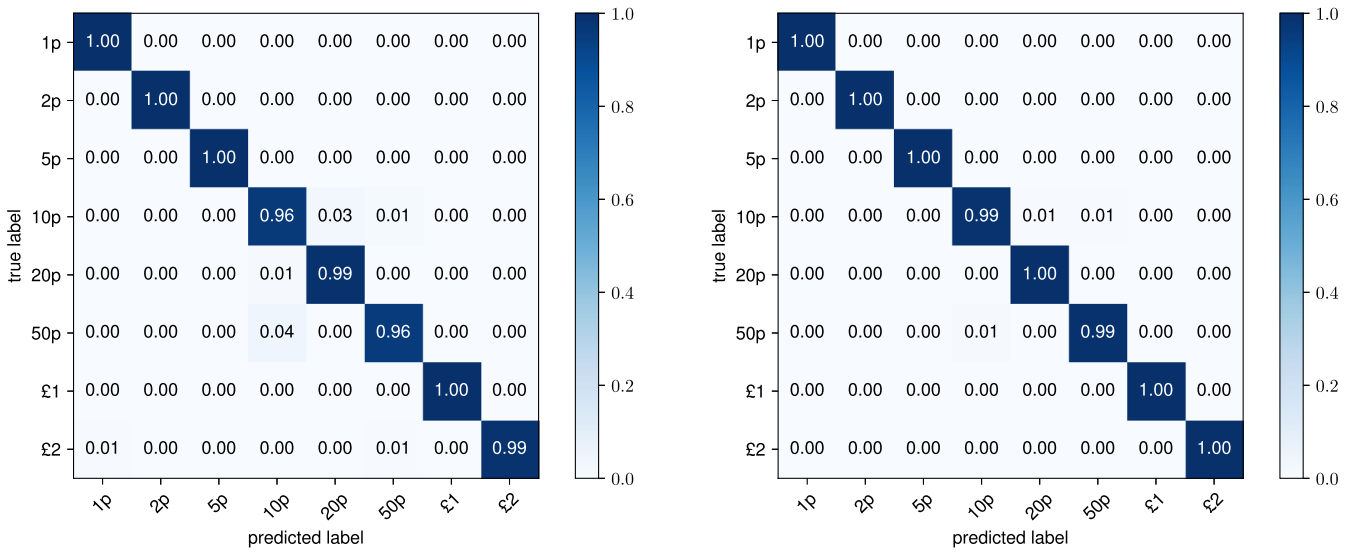

FIGURE 9 Set of British coins: Confusion matrices for noise corresponding to SNR = 10 dB showing the effect of different numbers of instance per class (A) $P^{(k)} = 50$ and (B) $P^{(k)} = 2000$

Next, we consider the frequentist approximations to $p(C_j|\mathbf{x})$ for $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(i))}$ presented in the form of a confusion matrix with entries $(\mathbf{C})_{ij}$, $i, j = 1, \ldots, K$, for the cases of SNR = 20 dB and SNR = 10 dB in Figure 9, using the approach described in Section 4.1.1. We consider the case with SNR = 10 dB and compare the performance of the classifier using $P^{(k)} = 50$ and $P^{(k)} = 2000$ instances per class. There are only a small number of misclassifications for the $P^{(k)} = 50$ case and these are further reduced by using $P^{(k)} = 2000$.

5.2 | Multi-class problem

5.2.1 | Construction of the multi-class dictionary

To create the multi-class dictionary, we follow the approach described in Section 2.1 where, in the most general setting, we choose the classes C_k, $k = 1, \ldots, K$, to correspond to the different threat and non-threat type objects listed in Table 1. Unlike the coins, each class is comprised of objects of different geometries, as well as different sizes and materials, so that $G^{(k)} \neq 1$ in general. However, when creating the classes, we have assembled geometries that have (physical) similarities. For example, the coins class C_1 includes the $G^{(1)} = 8$ different denomination of British coins described in the previous section. Furthermore, in Figures 10 and 11, we illustrate the surface finite element meshes corresponding to exemplar threat and non-threat object geometries, respectively, within each of these different classes and Table 2 gives an overall summary of the materials and object sizes. In the case of the coins, guns, keys, and knives, the simulated MPT spectral signatures are those presented in Reference 1 and we provide the complete set of MPT spectral signatures for all objects in our `MPT-Library` dataset.[2] These simulated spectral signatures were generated in a similar way to those described in Reference 1 and in Figure 12 we show an illustration of some the contours of $\boldsymbol{J}_i^e = \mathrm{i}\omega\sigma_*\boldsymbol{\theta}_i$ at $\omega = 1 \times 10^4$ rad/s that are obtained as part of this process. In total, we have $\sum_{k=1}^{K} G^{(k)} = 67$ different distinct geometries and 158 including different material variations. In Table 1, we give the relationship between $P^{(k)}$, $V^{(k)}$, and $G^{(k)}$ for each class C_k and choose $V^{(k)}$ so that we have an approximately equal number of samples $P^{(k)} \approx P/K$ for each class of object. We employ $P^{(k)} = 5000$ in the following unless otherwise stated. While m_{σ_*} is object specific, we set $m_\alpha = 0.001$ m, to fix the object size, and choose $s_\alpha = 0.0084 m_\alpha$ and $s_{\sigma_*} = 0.024 m_{\sigma_*}$, to account for manufacturing imperfections. We consider a fixed number of $M = 28$ linearly spaced frequencies, such that $7.53 \times 10^2 \leq \omega_m \leq 5.99 \times 10^5$ rad/s, although we also give some comments about the performance using $5.02 \times 10^4 \leq \omega_m \leq 8.67 \times 10^4$ rad/s. In a similar manner to the coin classification problem, noise corresponding to SNR values of 40 and 20 dB is added. We do not consider an SNR of 10 dB as this represents a very high level of 32 % noise, which, of course, performs worse than 20 dB noise. Using the information above, two different types

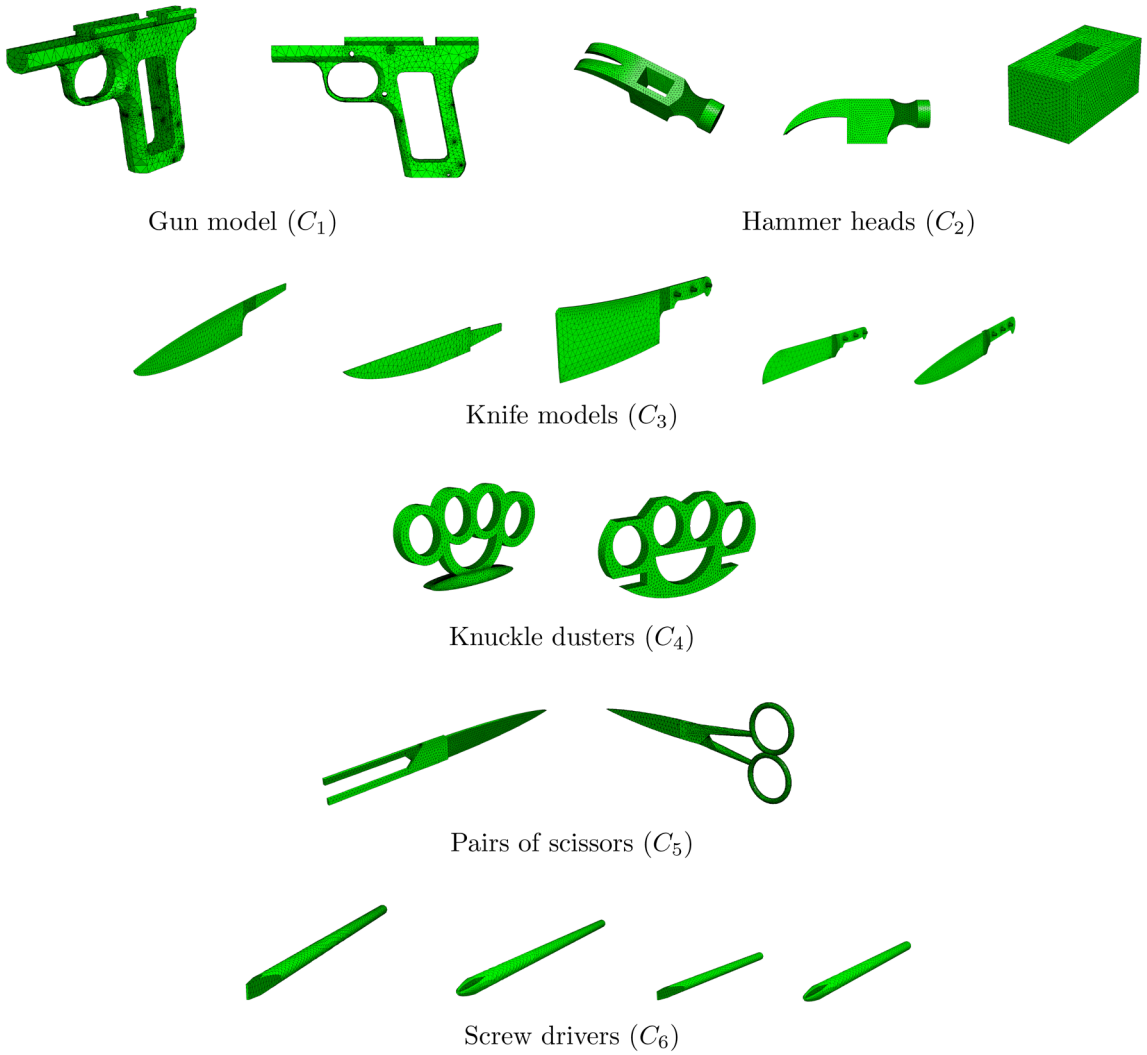

FIGURE 10 Set of multiple threat and non-threat objects: Sample illustrations of some of the different threat object geometries considered (not to scale)

of dictionaries were formed. First, D_{15} for the complete set of $K = 15$ classes and, second, D_8 comprising of $K = 8$ different classes. The grouped classes for the D_8 are described in Table 3.

For these dictionaries, we have $G^{(k)} \neq 1$ in the majority of cases. Considering $\alpha \sim N(m_\alpha, s_\alpha)$, $\sigma_* \sim N(m_{\sigma_*}, s_{\sigma_*})$, and $g_k = 1, \ldots, G^{(k)}$, the parent distributions of the variables $X = I_i(\tilde{\mathcal{R}}[\alpha B^{(g_k)}, \omega_m, \sigma_*, \mu_r]) \sim p(x_{i+(m-1)M}|C_k)$ and $X = I_i(\mathcal{I}[\alpha B^{(g_k)}, \omega_m, \sigma_*, \mu_r]) \sim p(x_{i+(m-1)M}|C_k)$ will be far from normal. For $P^{(k)} = 5000$ samples and the class C_1, comprised of the $G^{(1)} = 8$ different denominations of British coins, the probability density distributions are shown in Figure 13. Even with a sample size of $P^{(k)} = 5000$ the sample distributions are also far from normal and a very large sample is expected to be needed in order for the central limit theorem to apply in this case.

In the following, we will start with classification using the dictionary D_8 and then proceed to present results for D_{15}.

5.2.2 | Classification results using D_8

From the observations in Figure 13, we do not expect logistic regression to perform well using the D_8 dictionary and for it to have a high bias. Instead, we will consider the full range of probabilistic and non-probabilistic classifiers described in Sections 3.3 and 3.4 and retain logistic regression for comparison. An optimized set of hyper parameters for each classifier were obtained as follows: A grid based search was performed to maximize the κ score (18) for the relevant hyperparameters

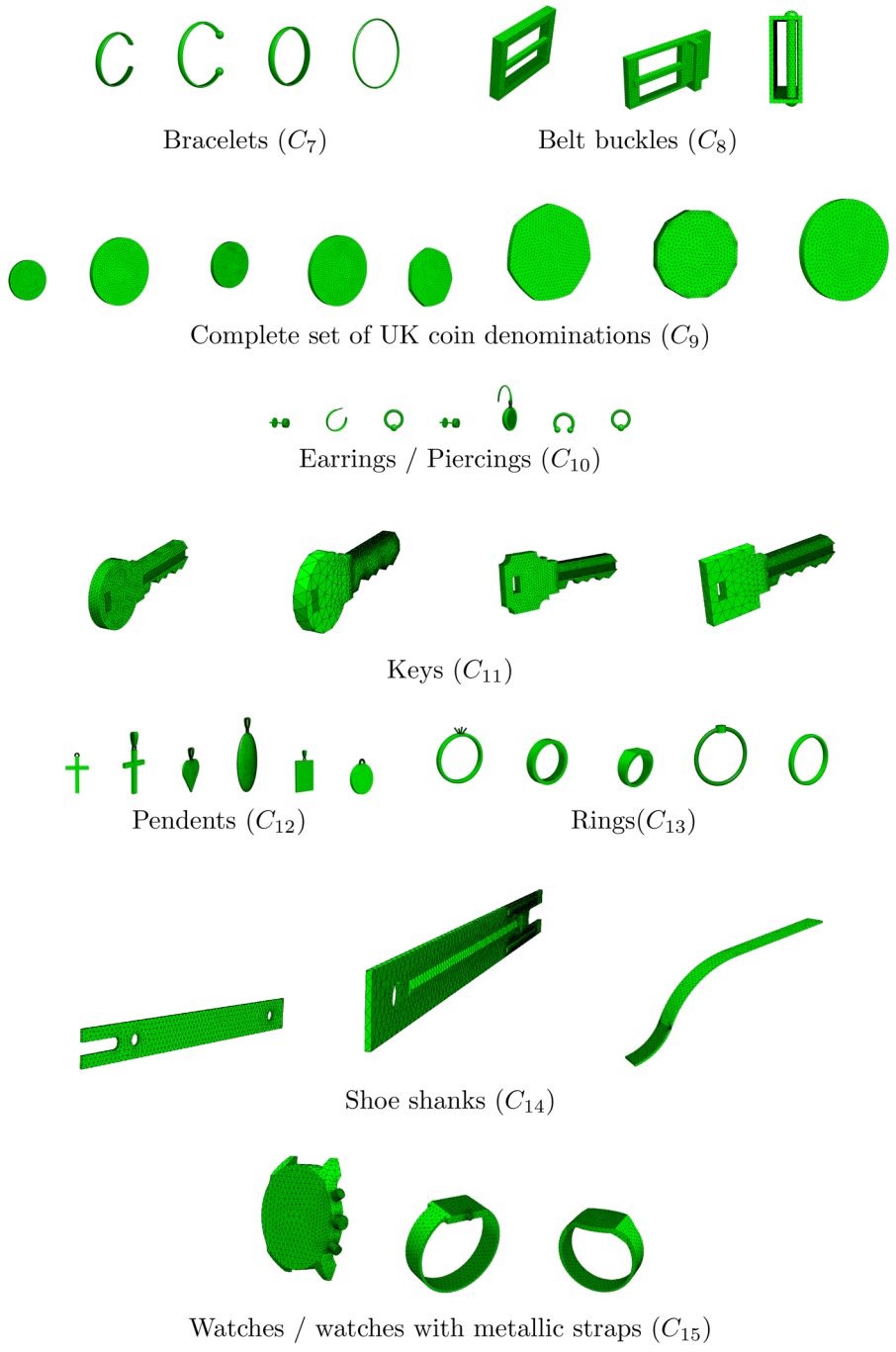

FIGURE 11 Set of multiple threat and non-threat objects: Sample illustrations of some of the different non-threat object geometries considered (not to scale)

for each classifier for the dictionary corresponding to $P^{(k)} = 5000$ and SNR = 40 dB and these were then adopted for the simulations presented in this section. To illustrate our hyperparameter grid-based optimization, we investigate the κ score for different choices of L and J for MLP in Figure 14. For this result, we have assumed the same number of neurons in each layer and used `max_iter = 300` rather than `max_iter = 200` to allow an increased number of iterations to be performed to ensure convergence. From this figure, we observe there are a range of different L and J that lead to a network with a similar level of accuracy. As remarked in Section 3.4.2, for the type of network we are considering, the number of variables grows quadratically with J and linearly with L. Hence, from a computational cost perspective, choosing a network with a small J and a large L is generally preferable to a network with a large J and a small L, if the cost of computing each variable is assumed the same. For this reason, we adopt a network with $L = 3$ and $J = 50$ as MLP

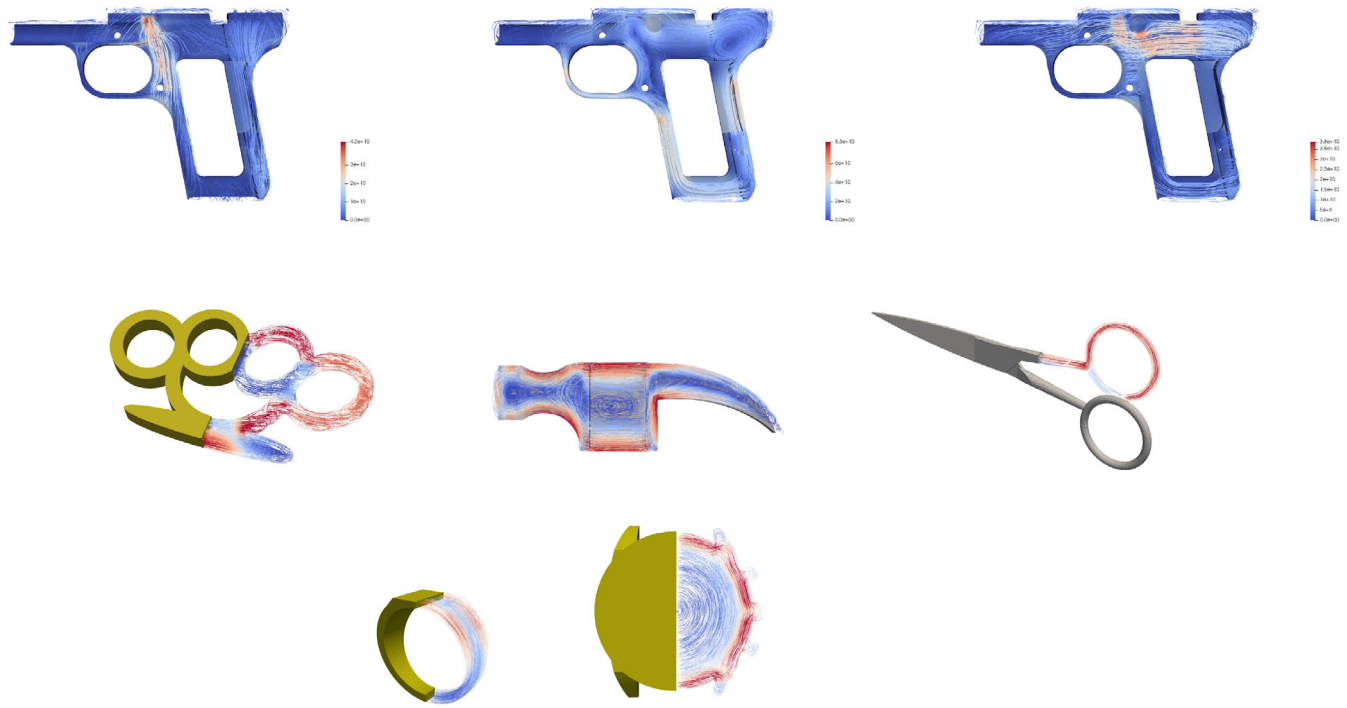

FIGURE 12 Set of multiple threat and non-threat objects: Illustrative contours of $\boldsymbol{J}_i^e = \mathrm{i}\omega\sigma_*\boldsymbol{\theta}_i$ at $\omega = 1 \times 10^4$ rad/s for a sample of the threat and non-threat objects

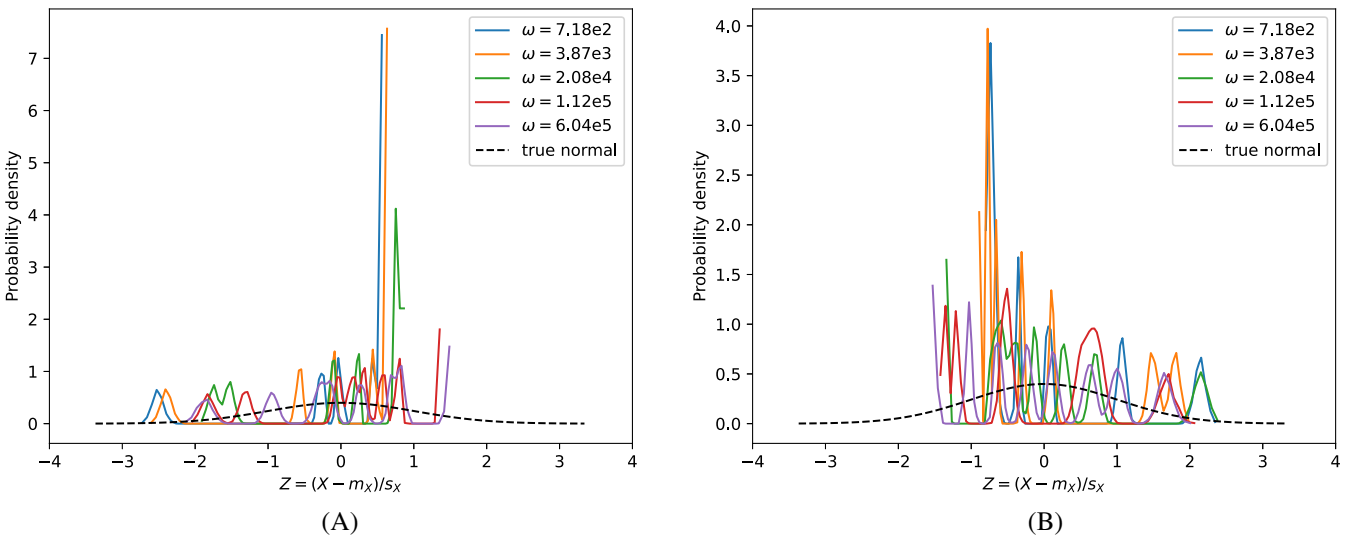

FIGURE 13 Set of multiple threat and non-threat objects: British coins (class C_1) for $P^{(k)} = 5000$, with $\alpha \sim N(0.001, 8.4 \times 10^{-6})$ m and $\sigma_* \sim N(m_{\sigma_*}, 0.024 m_{\sigma_*})$ S/m, where m_{σ_*} is determined by the material $B^{(k)}$, showing the histograms of $(Z - m_X)/s_X$, presented in the form of probability densities, where X is instances of the following (A) $I_1(\tilde{\mathcal{R}}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ and (B) $I_1(\mathcal{I}[\alpha B^{(1)}, \omega_m, \sigma_*, \mu_r])$ at distinct frequencies ω_m

TABLE 1 Set of multiple threat and non-threat objects: Full list of 15 threat and non-threat object classes detailing the number of geometries in each class, $G^{(k)}$, the number of materials per geometry, the number of additional variations to account for geometrical and material variations, and the total number in each class, $P^{(k)}$

Class	# Geometries ($G^{(k)}$)	# Materials per geometry	# Additional variations ($V^{(k)}$)	Total ($P^{(k)}$)
Guns (C_1)	1	1	$V^{(1)}$	$V^{(1)}$
Hammers (C_2)	2	3	$V^{(2)}$	$6V^{(2)}$
Knives (C_3)	5	1	$V^{(3)}$	$5V^{(3)}$
Knuckle dusters (C_4)	2	1	$V^{(4)}$	$2V^{(4)}$
Screw drivers (C_5)	6	3	$V^{(5)}$	$18V^{(5)}$
Scissors (C_6)	2	3	$V^{(6)}$	$6V^{(6)}$
Bracelets (C_7)	4	3	$V^{(7)}$	$12V^{(7)}$
Belt buckles (C_8)	3	4	$V^{(8)}$	$12V^{(8)}$
Coins (C_9)	8	1	$V^{(9)}$	$8V^{(9)}$
Earrings (C_{10})	9	3	$V^{(10)}$	$18V^{(10)}$
Keys (C_{11})	4	1	$V^{(11)}$	$4V^{(11)}$
Pendents (C_{12})	7	3	$V^{(12)}$	$21V^{(12)}$
Rings (C_{13})	7	3	$V^{(13)}$	$21V^{(13)}$
Shoe shanks (C_{14})	3	1	$V^{(14)}$	$3V^{(14)}$
Watches (C_{15})	4	3	$V^{(15)}$	$12V^{(15)}$

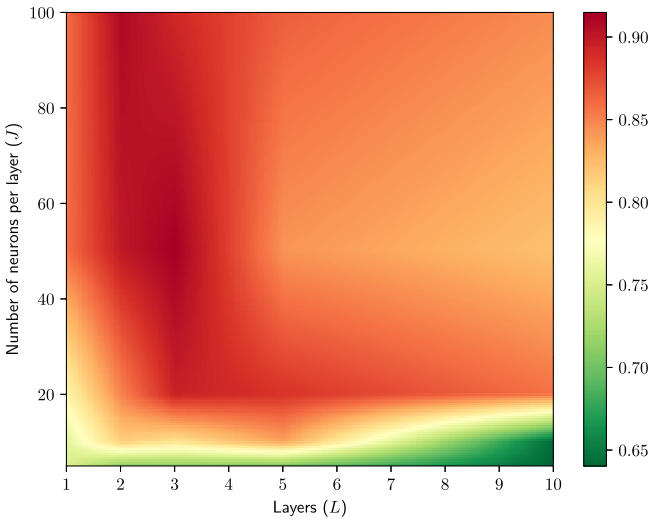

FIGURE 14 Set of multiple threat and non-threat objects: Overall performance of MLP for different uniform network architectures, for $P^{(k)} = 5000$ when $K = 8$ and SNR = 40 dB, showing κ score for different numbers of hidden layers L and numbers of neurons per layer J

architectures in this range result in high κ score, while minimizing computational cost. Also, if desired, J could be further reduced without comprising accuracy.

For SVM, rather than the default *ovr* strategy, we employ `decision_function_shape = 'ovo'`, this is due to the performance of kernel based methods not scaling in proportion with the size of the training dataset. The grid-based optimization led to a significant variation in performance for SVM, with the optimum values being a regularization parameter `C = 10e6` and a Kernel coefficient `gamma = 1`. For the random forest classifier, the grid based optimization resulted in the choice of `max_depth = 100` and `n_estimators = 100`. For gradient boost, the grid-based optimization led to choice of `n_estimators = 50` and `max_depth = 2` and later, in the case of an unseen object, we show

TABLE 2 Set of multiple threat and non-threat objects: Full list of 15 threat and non-threat object classes detailing composition of different materials and different object sizes in each class

Class	$\min(\sigma_*)$ (S/m)	$\max(\sigma_*)$ (S/m)	$\min(\mu_r)$	$\max(\mu_r)$	$\min(\alpha^3\lvert B\rvert)$ (m^3)	$\max(\alpha^3\lvert B\rvert)$ (m^3)
Guns (C_1)	6.2×10^6	6.2×10^6	5	5	3.3×10^{-5}	3.3×10^{-5}
Hammers (C_2)	1.3×10^6	1.6×10^6	1.02	5	4.3×10^{-5}	2.0×10^{-4}
Knives (C_3)	1.5×10^5	5.8×10^7	1	5	3.3×10^{-6}	6.5×10^{-5}
Knuckle dusters (C_4)	1.5×10^7	1.5×10^6	1	1	1.7×10^{-5}	1.8×10^{-5}
Screw drivers (C_5)	1.3×10^6	1.6×10^6	1.02	5	1.1×10^{-6}	3.4×10^{-6}
Scissors (C_6)	1.3×10^6	1.6×10^6	1.02	5	2.4×10^{-6}	9.5×10^{-6}
Bracelets (C_7)	9.4×10^6	6.3×10^7	1	1	5.5×10^{-7}	2.1×10^{-6}
Belt buckles (C_8)	5.6×10^5	1.5×10^7	1	5	1.5×10^{-5}	2.1×10^{-5}
Coins (C_9)	2.9×10^6	4.0×10^7	1	1	4.3×10^{-7}	1.9×10^{-6}
Earrings (C_{10})	9.4×10^6	6.3×10^7	1	1	1.0×10^{-8}	2.3×10^{-7}
Keys (C_{11})	2×10^7	1.5×10^7	1	1	6.3×10^{-7}	6.7×10^{-7}
Pendents (C_{12})	9.4×10^6	6.3×10^7	1	1	8.0×10^{-8}	1.6×10^{-6}
Rings (C_{13})	9.4×10^6	6.3×10^7	1	1	7.0×10^{-8}	9.2×10^{-7}
Shoe shanks (C_{14})	6.2×10^6	6.2×10^6	5	5	8.9×10^{-7}	1.4×10^{-6}
Watches (C_{15})	9.4×10^6	6.3×10^7	1	1	4.7×10^{-6}	3.3×10^{-5}

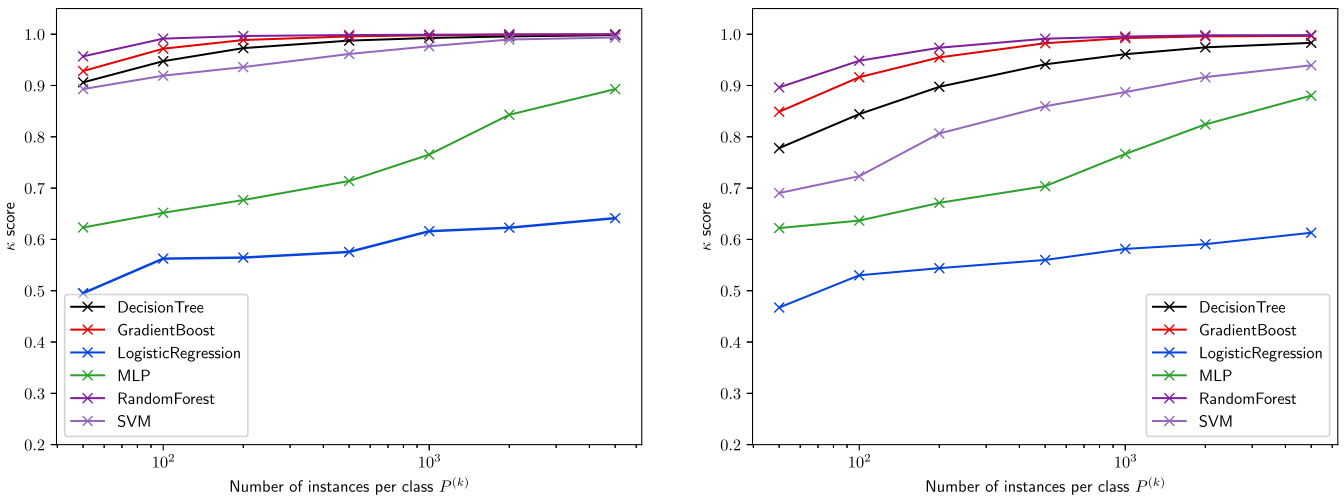

FIGURE 15 Set of multiple threat and non-threat objects: Overall performance of different classifiers as a function of $P^{(k)}$ when $K = 8$ using the κ score (18) showing (A) SNR = 40 dB and (B) SNR = 20 dB

the effects of varying the number of trees within the ensemble and the maximum depth of each tree. Finally, for the decision tree classifier, the grid based optimization led to the choice `max_depth = 100`. Of course, our hyperparameter choices for each classifier have been optimized for $P^{(k)} = 5000$, SNR = 40 dB and this $K = 8$ class problem, for different $P^{(k)}$ and SNR levels, as well as other classification problems, this choice may no longer be optimum. More sophisticated alternatives to our simple grid-based optimization include using a Bayesian optimization,[48] a hyperband optimization,[49] or simulated annealing.[50]

In Figure 15, we show the overall performance of the classifiers with different levels of noise. We use the κ score to assess the performance of the classification due to the variations within the classes. In each case, we observe that increasing $P^{(k)}$ generally leads to an improved performance of the classification in all cases, since the classifier is exposed to more noisy data in $D_8^{(\text{train})}$ and its variability decreases. The figure shows that, in both noise cases, the best performing

TABLE 3 Set of multiple threat and non-threat objects: Amalgamated list of $K = 8$ threat and non-threat object classes detailing their composition and total number in each class $P^{(k)}$

Class	Composition	Total $P^{(k)}$
Tools (C_1)	Hammers	$6V^{(2)} + 6V^{(6)} + 18V^{(5)}$
	Scissors	
	Screwdrivers	
Weapons (C_2)	Guns	$V^{(1)} + 2V^{(4)} + 5V^{(3)}$
	Knuckle dusters	
	Knives	
Clothing (C_3)	Belt buckles	$12V^{(8)} + 3V^{(14)}$
	Shoe shanks	
Earrings (C_4)	Earrings	$18V^{(10)}$
Pendants (C_5)	Pendants	$21V^{(12)}$
Pocket items (C_6)	Coins	$8V^{(9)} + 4V^{(11)}$
	Keys	
Rings (C_7)	Rings	$21V^{(13)}$
Wrist items (C_8)	Bracelets	$21V^{(7)} + 12V^{(15)}$
	Watches	

classifier is random forests, although, for large $P^{(k)}$, the performance of random forest, gradient boost, decision trees, and SVM (particularly for SNR = 40 dB) are all very similar with $\kappa \approx 1$ indicating a low bias and low variance. As random forest is a bagging algorithm and gradient boost is a boosting algorithm we expect them to perform well. However, the good performance of decision trees is surprising. While the performance of SVM is good, it is less robust as (small) changes in hyperparameters can have a significant affect on its performance. The second best probabilistic classifier is MLP, which shows a significant benefit for large $P^{(k)}$. Comparing SNR = 40 dB and SNR = 20 dB, we see a slight reduction in accuracy for a given $P^{(k)}$ using SNR = 20 dB, although, by increasing $P^{(k)}$, the effects of noise can be overcome. In particular, SVM suffers noticeably more with SNR = 20 dB compared to random forest, gradient boost, and decision trees, but its performance for large $P^{(k)}$ is still good and may be improved further by additional hyperparameter optimization. Also, although not included, the corresponding results for $5.02 \times 10^4 \leq \omega_m \leq 8.67 \times 10^4$ rad/s using $M = 20$ are not as good as those for $7.53 \times 10^2 \leq \omega_m \leq 5.99 \times 10^5$ rad/s using $M = 28$, with those shown offering at least a 5 % improvement for the best performing classifiers, small $P^{(k)}$ and SNR = 20 dB. Interestingly, logistic regression improves by 25 % when the larger frequency range is used. We focus on the two best performing probabilistic classifiers, gradient boost and MLP, in the following.

The approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, we obtain for gradient boost and MLP are shown in Figure 16. We have chosen $(\mathbf{x}, \mathbf{t}) \in D_8^{(\text{test},(6))}$ so that the correct classification should be C_6 (ie a pocket item: a coin or a key). Additionally, the bars we show are for the median value $\gamma_{k,50}$ of $\gamma_k(\mathbf{x})$, obtained by considering all the samples $(\mathbf{x}, \mathbf{t}) \in D_8^{(\text{test},(6))}$, and we also indicate the Q_1, Q_3 quartiles as well as $\gamma_{k,5}$ and $\gamma_{k,95}$, for different SNR, which have been obtained using (17). The results for SNR = 40 dB strongly indicate that the most likely class is a pocket item for both classifiers, since $\gamma_{6,50} \approx 1$. For the gradient boost classifier, the inter quartile and inter percentile ranges are small and, so, we have high confidence in this prediction. However, we have less confidence in the corresponding prediction for the MLP as both the inter quartile and inter percentile ranges are larger. For SNR = 20 dB, we see the median value $\gamma_{6,50}$ fall for both classifiers and we also have much greater uncertainty in the classification over the samples, as illustrated by the larger inter percentile ranges for the different object classes. Comparing the two classifiers, we have less confidence in the prediction with MLP than for gradient boost.

Next, we consider the frequentist approximations to $p(C_j|\mathbf{x})$ for $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(i))}$ presented in the form of a confusion matrix with entries $(\mathbf{C})_{ij}$, $i,j = 1, \ldots, K$, for the cases of SNR = 40 dB and SNR = 20 dB and the gradient boost and MLP classifiers in Figure 17. As expected, for SNR = 20 dB, we see increased misclassification among the classes compared to SNR = 40 dB with situation being worse for the MLP classifier compared to the gradient boost. Looking at the row corresponding to the true label for the C_6 (pocket items) class, for both SNR = 40 dB and SNR = 20 dB, we can see that

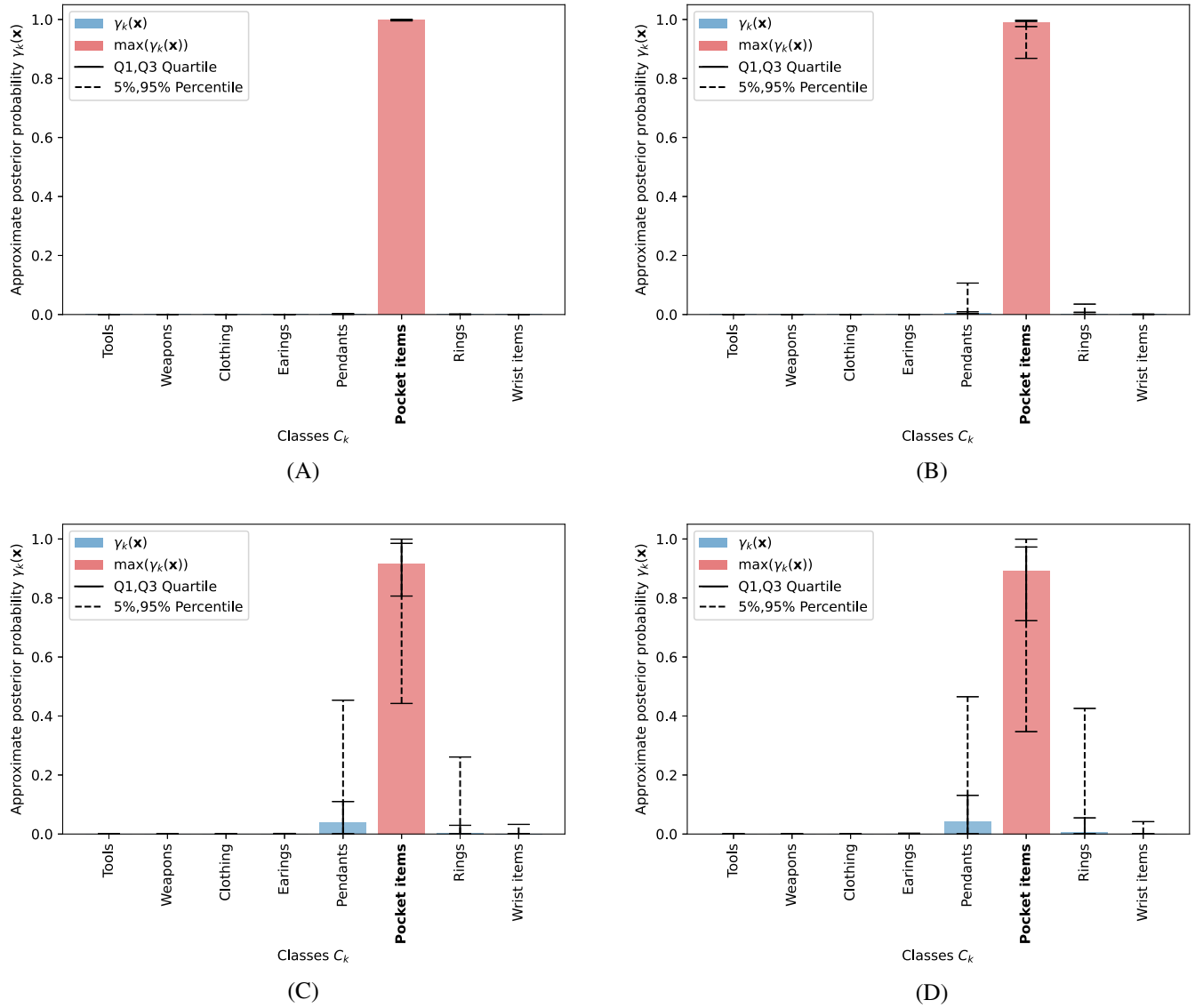

FIGURE 16 Set of multiple threat and non-threat objects: Approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, where $(\mathbf{x}, \mathbf{t}) \in D_8^{(\text{test},(2))}$ for $P^{(k)} = 5000$ when $K = 8$ showing the classifiers (A) gradient boost SNR = 40 dB, (B) gradient boost SNR = 20 dB, (C) MLP SNR = 40 dB, and (D) MLP SNR = 20 dB

the frequentist probability in column j are approximately similar to the median approximate posterior probability $\gamma_j(\mathbf{x})$ shown in Figure 16. Also, while the gradient boost exhibits near perfect classification for SNR = 40 dB (and SNR = 20 dB), MLP does not perform as well, particularly among the earrings and pendents.

In Table 4, we show the precision, sensitivity, and specificity for each of the different object classes $C_k, k = 1, \ldots, K$, for the case of SNR = 20 dB and the MLP classifier. In general, we see the proportion of negatives that are correctly identified is very high (as indicated by the specificity) and is close to 1 in all cases, whereas the proportions of positives correctly identified (indicated by the precision and sensitivity) varies among the different object classes, the best case being C_2 (weapons) and worst case C_5 (pendents). The corresponding results for gradient boost are all close to 1.

5.2.3 | Classification results using D_{15}

Figure 18 repeats the investigation shown in Figure 15 for D_{15}, instead of D_8, and, given the relationship between the multi-class dictionaries, uses the same classifier hyperparameters. The trends described previously again apply, except, with a further significant gain in the performance for all classifiers for the increased fidelity $K = 15$ class problem compared to the previous $K = 8$ class problem. This is because each class, for $K = 15$, is comprised of objects

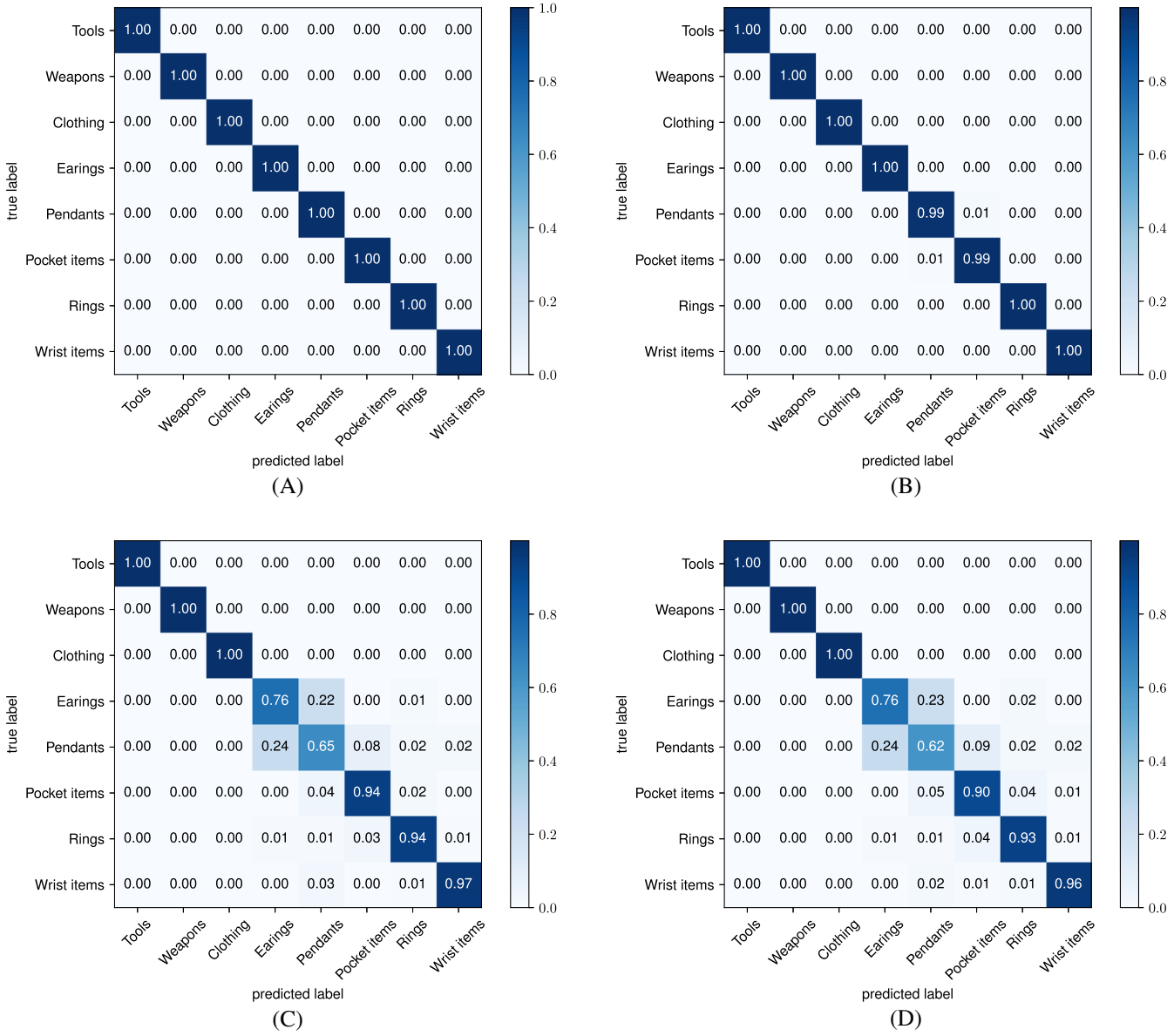

FIGURE 17 Set of multiple threat and non-threat objects: Comparison of confusion matrices for $P^{(k)} = 5000$ when $K = 8$ showing the classifiers (A) gradient boost SNR = 40 dB, (B) gradient boost SNR = 20 dB, (C) MLP SNR = 40 dB, and (D) MLP SNR = 20 dB

that have increased similarity between their volumes, shapes and materials, and, hence, their MPT spectral signatures, compared to the $K = 8$ problem. This, in turn, reduces each classifier's bias as it becomes easier to establish the relationship between the features and class. Nonetheless, $X = I_i(\tilde{\mathcal{R}}[\alpha B^{(g_k)}, \omega_m, \sigma_*, \mu_r]) \sim p(\mathrm{x}_{i+(m-1)M}|C_k)$ and $X = I_i(\mathcal{I}[\alpha B^{(g_k)}, \omega_m, \sigma_*, \mu_r]) \sim p(\mathrm{x}_{i+(m-1)M}|C_k)$ are still far from normal and, so, logistic regression does not perform well. The best performance being again given by random forests, gradient boost, decision trees and SVM (particularly for SNR = 40 dB). We focus on the gradient boost and MLP, which are the best two performing probabilistic classifiers in the following.

The approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, we obtain for gradient boost and MLP are shown in Figure 19. We have chosen $(\mathbf{x}, \mathbf{t}) \in D_{15}^{(\mathrm{test},(9))}$ so that the correct classification should be C_9. The bars are for $\gamma_{k,50}$, obtained by considering all the samples $(\mathbf{x}, \mathbf{t}) \in D_{15}^{(\mathrm{test},(9))}$, and we also indicate the Q_1, Q_3 quartiles as well as $\gamma_{k,5}$ and $\gamma_{k,95}$, for different SNR, which have been obtained using (17). The results for SNR = 40 dB strongly indicate that the most likely class is a pocket item for both classifiers, since $\gamma_{9,50} \approx 1$. For the gradient boost classifier, the inter quartile and inter percentile ranges are very small and so we have very high confidence in this prediction; the MLP classifier has larger

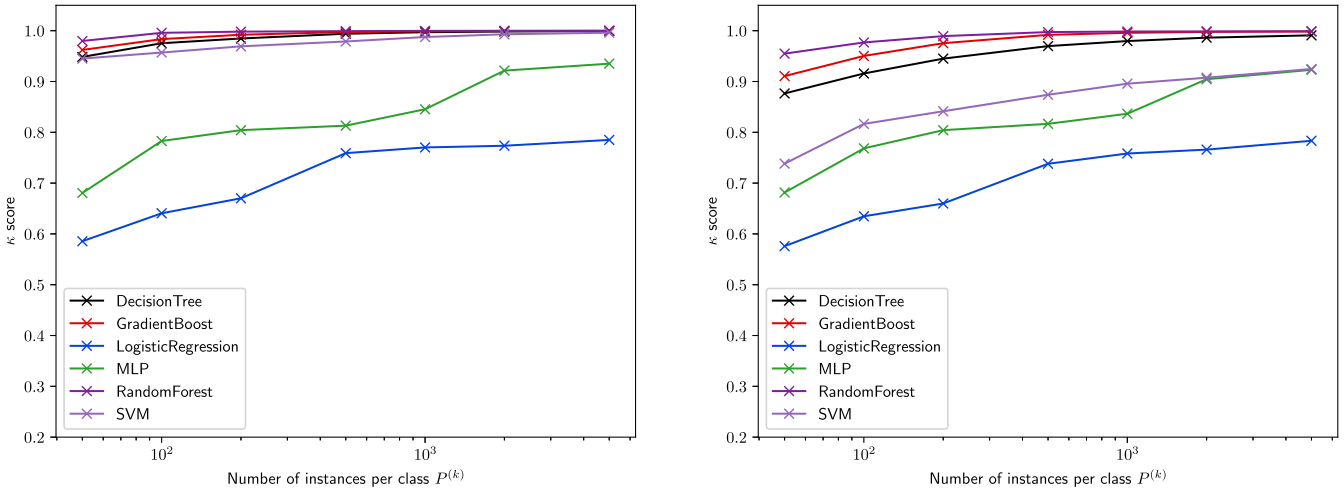

FIGURE 18 Set of multiple threat and non-threat objects: Overall performance of different classifiers as a function of $P^{(k)}$ when $K = 15$ using the κ score (18) showing (A) SNR = 40 dB and (B) SNR = 20 dB

TABLE 4 Set of multiple threat and non-threat objects: Precision, sensitivity, and specificity measures (to 2d.p.) for each of the classes C_k when SNR = 20 dB and $P^{(k)} = 5000$ for the MLP classifier

C_k	Precision	Sensitivity	Specificity
Tools (C_1)	0.97	0.98	1.00
Weapons (C_2)	0.99	0.99	1.00
Clothing (C_3)	0.98	0.98	1.00
Earrings (C_4)	0.64	0.84	0.93
Pendants (C_5)	0.60	0.34	0.97
Pocket items (C_6)	0.75	0.73	0.96
Rings (C_7)	0.70	0.82	0.95
Wrist items (C_8)	0.90	0.87	0.99

ranges and less confidence. For SNR = 20 dB, we see $\gamma_{9,50}$ fall slightly for gradient boost and by a larger amount for MLP. The gradient boost still shows a high degree of confidence in the prediction, but the MLP is more uncertain. Compared to the results shown in Figure 16 for D_8, the performance in Figure 19 for D_{15} is improved for MLP and remains excellent for gradient boost (when considering the amalgamated pocket item class and the split coin and keys classes).

Next, we consider the frequentist approximations to $p(C_j|\mathbf{x})$ for $(\mathbf{x}, \mathbf{t}) \in D^{(\text{test},(i))}$ presented in the form of a confusion matrix with entries $(\mathbf{C})_{ij}, i, j = 1, \ldots, K$, for the cases of SNR = 40 dB and SNR = 20 dB and the MLP classifier, in Figure 20. We do not show the results for the gradient boost as it has a near perfect identity confusion matrix on this scale for these noise levels. Compared to the corresponding results shown in Figure 17 for D_8, the results for D_{15} show the ability of the classifier to better discriminate between different objects. However, MLP still shows significant misclassifications for pendents whereas gradient boost does not.

In Table 5, we show the precision, sensitivity, and specificity for each of the different object classes $C_k, k = 1, \ldots, K$, for the case of SNR = 20 dB and the MLP classifier. In general, we see the proportion of negatives that are correctly identified (as indicated by the specificity) is very high and is close to 1 in all cases. The proportions of positives correctly identified (indicated by the precision and sensitivity) varies among the different object classes, but is generally much closer to 1 than shown in Table 4 for D_8. The classes C_1, C_3, and C_4 (guns, knives, and knuckle dusters in $D^{(15)}$), which make up the amalgamated weapons class C_2 in D_8, all perform very well, but the worst case still remains C_{12} (the pendents). The corresponding results for precision, sensitivity, and specificity for the gradient boost classifier are all close to 1.

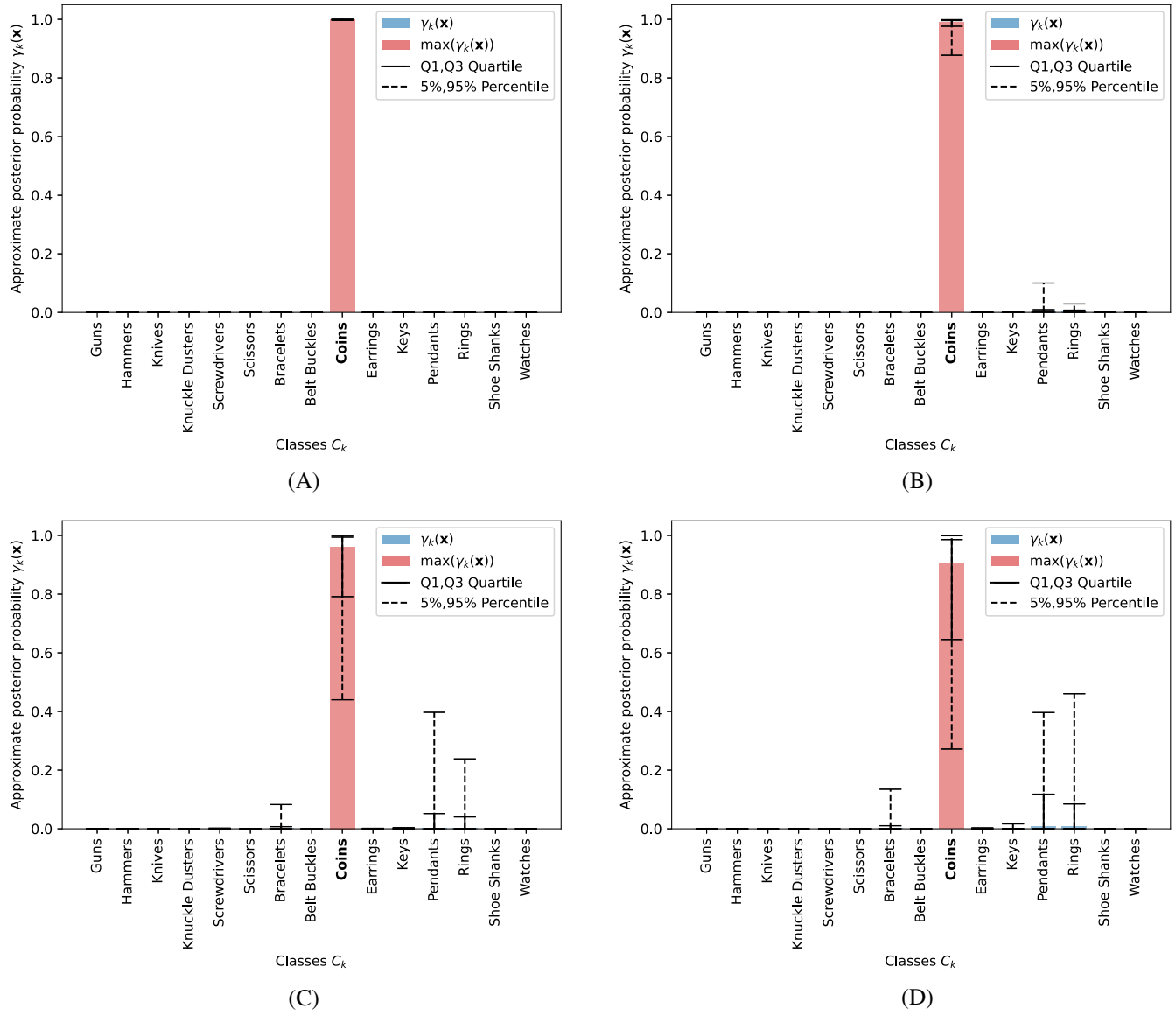

FIGURE 19 Set of multiple threat and non-threat objects: Approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, where $(\mathbf{x}, \mathbf{t}) \in D_{15}^{(\text{test},(9))}$ (Coins) for $P^{(k)} = 5000$ when $K = 15$ showing (A) gradient boost SNR = 40 dB, (B) gradient boost SNR = 20 dB, (C) MLP SNR = 40 dB, and (D) MLP SNR = 20 dB

5.2.4 | Classification of unseen objects using D_8

When testing the performance of classifiers in the previous sections, the construction of the dictionary, described in Section 5.2.1, means that $D^{(\text{train})}$ and $D^{(\text{test})}$ are both comprised of samples that have MPT spectral signatures associated with objects that share the same geometry and have similar object sizes and material parameters. To illustrate the ability of a classifier to recognize an unseen threat object, we construct $D_8^{(\text{train})}$, as in Section 5.2.1, except, for one class C_k, where we replace $D_8^{(\text{train},(k))}$ with data that is obtained from $G^{(k)} - 1$ (instead of $G^{(k)}$) geometries and $V^{(k)}$ samples. Also, we use $P^{(k)} = 2000$, instead of $P^{(k)} = 5000$, due to the higher computational cost of the investigation presented in the following. We proceed to test the classifier using a sample that is constructed only from $V^{(k)}$ samples of the unseen $G^{(k)}$th geometry.

We focus on removing a geometry from the C_2 class of weapons, which originally has $G^{(2)} = 8$ geometries, and we vary the unseen geometry to be one of the chef, cutlet, meat cleaver, Santoku, and Wusthof knives, shown in Figure 10, where the naming convention from Section 6.4 of Reference 1 is adopted. We apply the gradient boost classifier to this problem, as it was seen to perform best for both the $K = 8$ and $K = 15$ class problems. Previously, the optimized hyperparameters

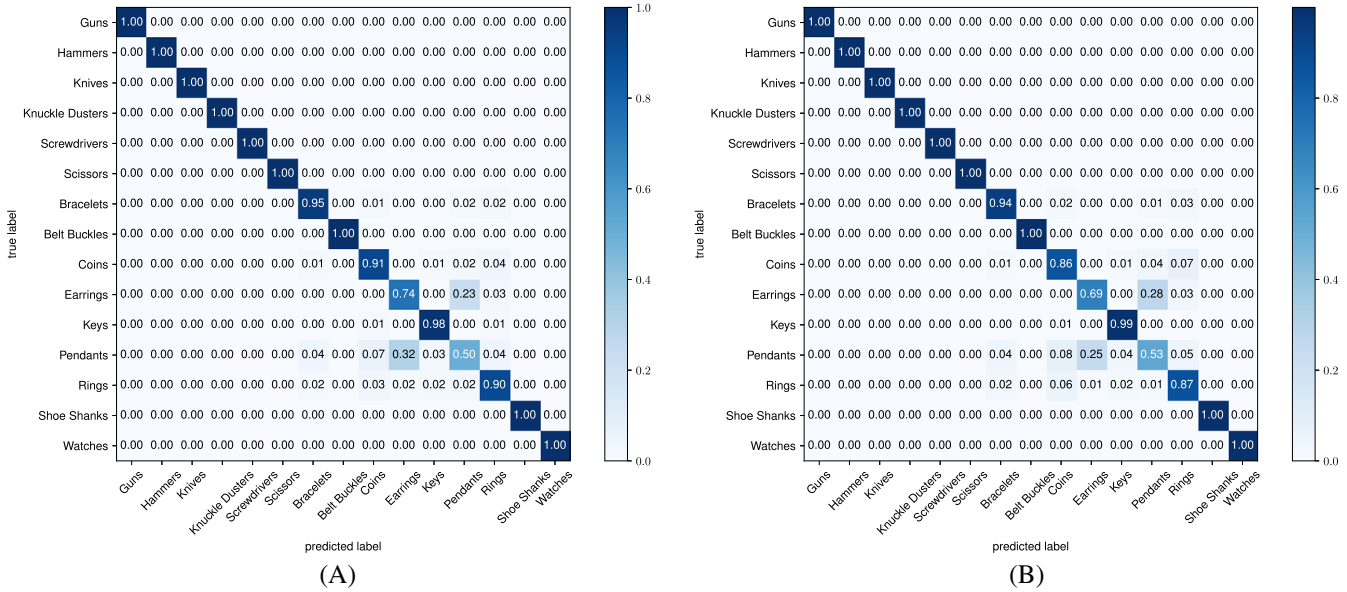

FIGURE 20 Set of multiple threat and non-threat objects: Comparison of confusion matrices for $P^{(k)} = 5000$ when $K = 15$ showing (A) MLP SNR = 40 dB and (B) MLP SNR = 20 dB

TABLE 5 Set of multiple threat and non-threat objects: Precision, sensitivity, and specificity measures (to 2d.p.) for each of the classes C_k when SNR = 20 dB and $P^{(k)} = P/K = 5000$ for the MLP classifier

C_k	Precision	Sensitivity	Specificity
Guns (C_1)	1.00	1.00	1.00
Hammers (C_2)	1.00	1.00	1.00
Knives (C_3)	1.00	1.00	1.00
Knuckle dusters (C_4)	1.00	1.00	1.00
Screwdrivers (C_5)	0.95	0.96	1.00
Scissors (C_6)	1.00	1.00	1.00
Bracelets (C_7)	0.83	0.85	0.99
Belt buckles (C_8)	0.99	0.98	1.00
Coins (C_9)	0.70	0.72	0.98
Earrings (C_{10})	0.75	0.75	0.98
Keys (C_{11})	0.90	0.95	0.99
Pendants (C_{12})	0.67	0.53	0.98
Rings (C_{13})	0.71	0.77	0.98
Shoe shanks (C_{14})	0.99	1.00	1.00
Watches (C_{15})	0.99	0.99	1.00

`n_estimators = 50` and `max_depth = 2` have been shown to lead to accurate results. However, this problem is more challenging, as it involves attempting to classify data from the samples in $(\mathbf{x}, \mathbf{t}) \in D_8^{(\text{test},(2))}$ that are only constructed from samples of the unseen $G^{(2)}$th geometry, and, therefore, the previous hyperparameters are no longer optimal. This is illustrated in Figure 21 for the case where SNR = 40 dB and, here, the average κ score obtained from considering the situations when instances of the chef, cutlet, meat cleaver, Santoku, and Wusthof knife geometries as being unseen is presented. This suggests the optimal performance will be for a very limited region where `n_estimators` ≈ 30 and `max_depth = 1` and, away from this, the performance of the classifier will be poor.

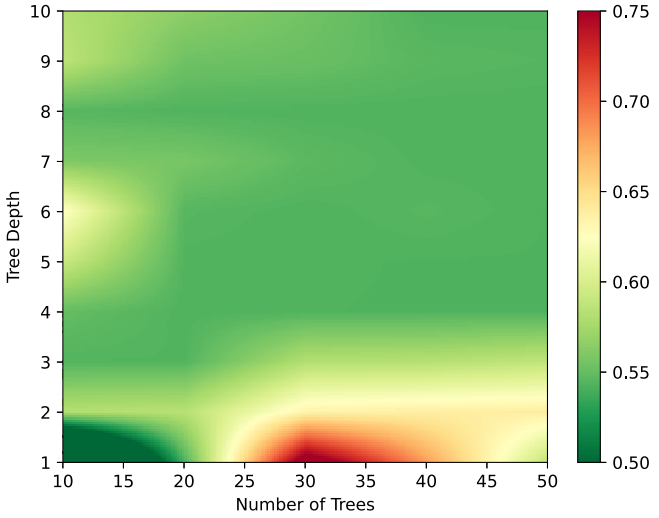

FIGURE 21 Set of multiple threat and non-threat objects: Overall performance of the gradient boost classifier for different values in the hyperparameter space, for $P^{(k)} = 2000$ when $K = 8$ and SNR = 40 dB, showing average κ score for different values of numbers of trees (`n_estimators`) and tree depth (`max_depth`)

TABLE 6 Set of multiple threat and non-threat objects: Comparison of volumes for different knife models

Knife	Volume (m^3)
Chef	1.46×10^{-5}
Cutlet	3.28×10^{-6}
Meat cleaver	6.50×10^{-5}
Santoku	2.51×10^{-5}
Wusthof	3.48×10^{-5}

The poor performance of the gradient boost classifier for this problem for a large range of hyperparameters is due to its inability to correctly classify the cutlet knife geometry, with the classifier instead predicting this as a tool rather than a weapon in the majority of cases (indicating bias against this geometry) and, additionally, for other geometries, the relatively high degree of uncertainty that is associated with $\gamma_2(\mathbf{x})$ despite $\gamma_{2,50}$ being high (indicating a high variance). This can further be explained by the comparison of the knife volumes using a fixed $\alpha = 0.001$ m shown in Table 6, where it can be seen that the cutlet knife geometry has a volume that is an order of magnitude smaller than that of the knives. The MPT spectral signature depends on the object's volume as well as its materials and geometry and, as each cutlet knife tends to be associated smaller volumes to those considered in $D_8^{(\text{train})}$ this has contributed to the classifier not being able to recognize it.

The situation can be improved by increasing the standard deviations s_α and s_{σ_*}, so that $D_8^{(\text{train},(2))}$ includes MPT spectral signatures that are closer to that of the omitted $G^{(2)}$th geometry. In Table 7, we consider three alternatives A, B, and C to the previous control choice. Then, in Figure 22, we repeat the investigation shown in Figure 21 for cases A, B, and C. In this figure, we observe that the classifier has less variability and performs increasingly well over a wide range of hyperparameters, as s_α and s_{σ_*} are increased. In the limiting case of C, the overall performance of the classifier is uniform with $\kappa = 0.75$ over the complete space of hyperparameters considered. Furthermore, in Figure 23, we show, for case C, approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$, to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, obtained for the case where the training samples are taken as $(\mathbf{x}, \mathbf{t}) \in D_8^{(\text{train},(2))}$, with either the chef, cutlet, meat cleaver, Santoku or Wusthof geometry being treated as unseen, in turn. These results were obtained with `n_estimators = 50` and `max_depth = 2` with SNR = 40 dB. From this figure, we observe that $\gamma_{2,50} \approx 1$ for the unseen chef, meat cleaver, Santoku, or Wusthof knives suggesting the most likely class is C_2 (a weapon) with small interpercentile and interquartile ranges, which indicates a high degree of certainty associated with the prediction and also a low variability. However, when the unseen object is a cutlet knife, $\gamma_{1,50} \approx 1$ with small interpercentile and interquartile ranges, which indicates that the classifier is still

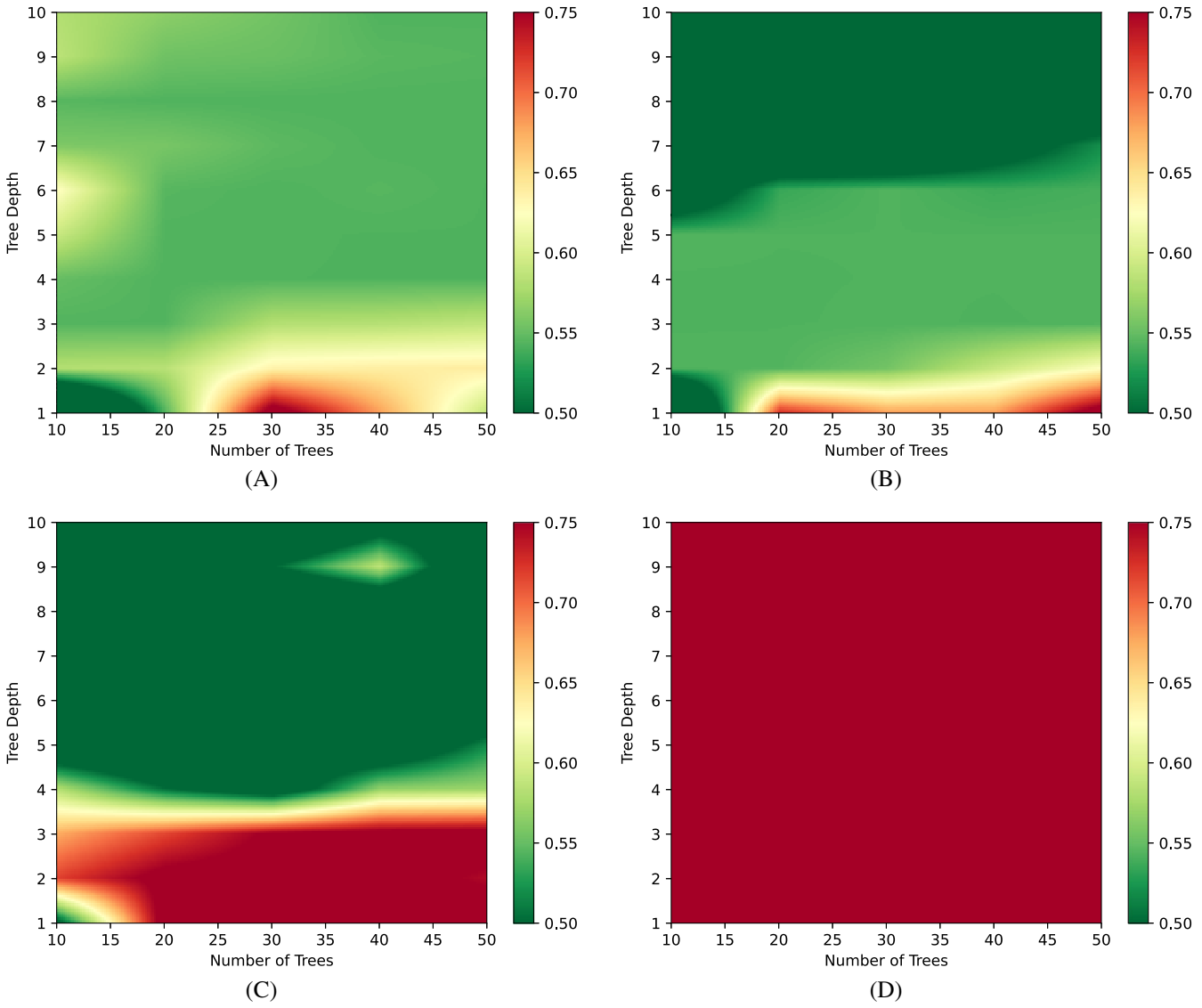

FIGURE 22 Set of multiple threat and non-threat objects: Overall performance of the gradient boost classifier for different values in the hyperparameter space, for $P^{(k)} = 2000$ with $K = 8$ and SNR = 40 dB, showing average κ score for different values `n_estimators` and `max_depth`, for different scaling regimes (A) Control, (B) A, (C) B, and (D) C

consistently misclassifying this object as a tool, instead of a weapon, despite the classifier being trained over a wider range of object sizes and conductivities. Hence, the classifier remains biased against this geometry. We conjecture this is due to the significant difference in the shape of the MPT spectral signature for the cutlet knife geometry shown in Figure 33 of Reference 1, compared to the other knives and gun geometry on which the classifier is trained.

6 | CONCLUSION

This article has presented a novel approach to training ML classifiers using our simulated MPT-Library, which has been enhanced by simple scaling results to create large dictionaries of object characterizations at a low computational cost. We have employed tensor invariants of MPT spectral signatures as novel object features for training ML classifiers and considered a range of both probabilistic and non-probabilistic classifiers. We have presented a well-reasoned approach for justifying the performance of different ML classifiers for practical classifications problems using both uncertainty quantification using statistical analysis and ML metrics. Furthermore, we have explored the ability of our classification approaches to classify unseen threat objects.

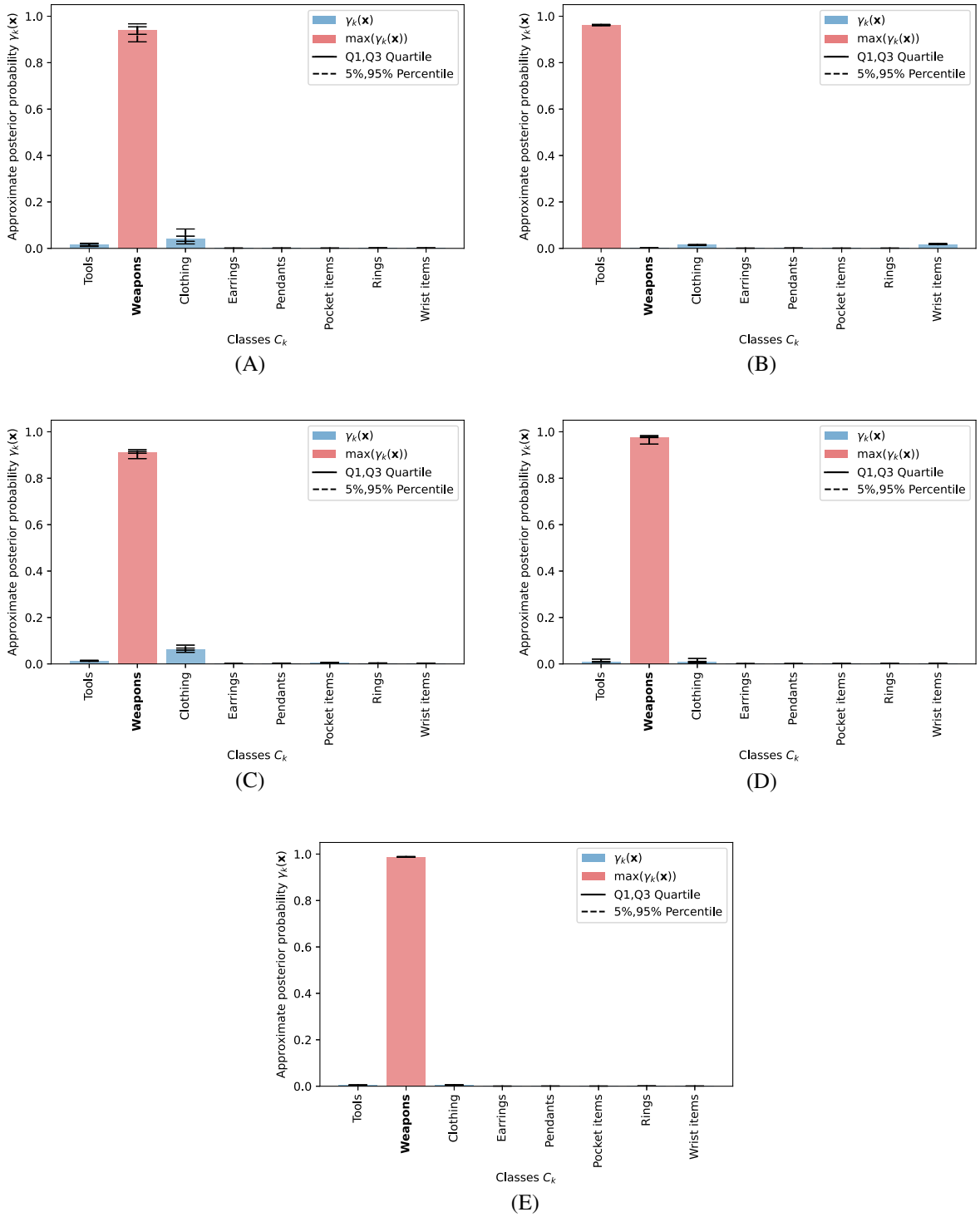

FIGURE 23 Set of multiple threat and non-threat objects: Approximations $p(C_k|\mathbf{x}) \approx \gamma_k(\mathbf{x})$ to posterior probabilities $p(C_k|\mathbf{x})$, $k = 1, \ldots, K$, using the gradient boost classifier for $P^{(k)} = 2000$ when $K = 8$ and SNR = 40 dB showing the case when $D^{(\text{train})}$ is constructed using the scaling regime C and cases where $D^{(\text{test})}$ is constructed of instances (A) chef, (B) cutlet, (C) meat cleaver, (D) Santoku, and (E) Wusthof

TABLE 7 Set of multiple threat and non-threat objects: List of the parameters for the sampling distributions considered

Scaling regime	s_α	s_{σ_*}
Control	$0.0084m_\alpha$	$0.0236333m_{\sigma_*}$
A	$0.02m_\alpha$	$0.05m_{\sigma_*}$
B	$0.05m_\alpha$	$0.1m_{\sigma_*}$
C	$0.1m_\alpha$	$0.2m_{\sigma_*}$

For the $K = 8$ class coin classification problem, we have found that the logistic regression classifier performs well to discriminate between different denominations of British coins. Other classifiers could also be used, but would be unnecessarily computationally intensive for this classification problem. Our choice of logistic regression was motived by a statistical analysis of the dictionary for this problem, which showed it had the desired properties needed for this classifier to perform well. We have seen significant benefits from increasing the number of frequencies considered in the MPT spectral signature as well as increasing the size of the training data set, which all led to an improvement of the accuracy of the classifier by reductions in its variability. A classifier of this type could help with the automated sorting of coins and in fraud detection. It also has potential applications in coin counting. Our results are useful for coin designers as it could help them to design new coins that have greater differences in their spectral signature to make them easier to classify.

For the multi-class problem, involving the discrimination between threat and non-threat objects, we have found improvements in the accuracy of the classification by using MPT spectral signatures over a larger range of frequencies compared to a narrow range, and, once again, also by increasing the size of the training data set. The hyperparamters for each of the classifiers were optimized for our multi-class dictionaries leading to excellent performance by decision trees, random forest, gradient boost and SVM (SNR = 40 dB) and good performance by SVM (SNR = 20 dB) and MLP classifiers. The best performing probabilistic classifier for both the $K = 8$ and $K = 15$ class classification problems and both SNR = 40 dB and SNR = 20 dB noise levels being the gradient boost algorithm. The gradient boost algorithm is also seen to perform well on the classification of unseen objects, provided the training set contains sufficiently similar MPT spectral signatures from other objects. This classifier could help in security screening applications such as transport hubs as well as in parcels transportation.

ACKNOWLEDGMENTS

Ben A. Wilson gratefully acknowledges the financial support received from EPSRC in the form of a DTP studentship with project reference number 2129099. Paul D. Ledger gratefully acknowledges the financial support received from EPSRC in the form of grants EP/R002134/2, EP/V049453/1, and EP/V009028/1. William R. B. Lionheart gratefully acknowledges the financial support received from EPSRC in the form of grants EP/R002177/1, EP/V049496/1, and EP/V009109/1 and would like to thank the Royal Society for the financial support received from a Royal Society Wolfson Research Merit Award.

ENDNOTES

*Note that the entries of $\mathbf{t}_p$ are all 0 except for $(\mathbf{t}_p)_{C_k} = 1$ corresponding to the kth class.

†We will return their properties shortly.

‡Note that we follow the convention used by `scikit-learn` for $\mathbf{C}$, other references use a different convention where the rows and columns are swapped.

§The sample mean and sample standard deviation are used as an approximation to m_X and s_X, respectively.

¶Although we chose $\alpha \sim N(m_\alpha, s_\alpha)$ and $\sigma_* \sim N(m_{\sigma_*}, s_{\sigma_*})$ we should not expect the parent distributions $p(\mathrm{x}_{1+(m-1)M}|C_1)$ and $p(\mathrm{x}_{1+(m+2)M}|C_1)$, for $(\tilde{\mathcal{R}}[\alpha B, \omega, \sigma_*, \mu_r])_{ij}$ or $(\mathcal{I}[\alpha B, \omega, \sigma_*, \mu_r])_{ij}$, respectively, to be normally distributed, due to the powering operation involved in the scaling in Lemma 3 of Reference 26, which, for large s_α, can have significant effect. However, the invariant I_1 only involves summation and will not change the distribution further for independent variables. The invariants I_2 and I_3 do involve products and so are likely to further change the parent distribution, but, compared to the root finding in eigenvalues, these are much smoother operations and so their effects are expected to be smaller.

ORCID

Paul D. Ledger https://orcid.org/0000-0002-2587-7023
William R. B. Lionheart https://orcid.org/0000-0003-0971-4678

REFERENCES

1. Ledger PD, Wilson BA, Amad AAS, Lionheart WRB. Identification of metallic objects using spectral magnetic polarizability tensor signatures: object characterisation and invariants. *Int J Numer Methods Eng*. 2021;15:3941-3984.
2. Wilson BA, Ledger PD. Bawilson94/mpt-library: version 1.0.0 mpt-library; May 2021. 10.5281/zenodo.4876371.
3. Ledger PD, Lionheart WRB. Characterising the shape and material properties of hidden targets from magnetic induction data. *IMA J Appl Math*. 2015;80(6):1776-1798.
4. Ledger PD, Lionheart WRB. An explicit formula for the magnetic polarizability tensor for object characterization. *IEEE Trans Geosci Remote Sens*. 2018;56(6):3520-3533.
5. Ledger PD, Lionheart WRB. Understanding the magnetic polarizability tensor. *IEEE Trans Magn*. 2016;52(5):6201216.
6. Ledger PD, Lionheart WRB, Amad AAS. Characterisation of multiple conducting permeable objects in metal detection by polarizability tensors. *Math Methods Appl Sci*. 2019;42(3):830-860.
7. Ledger PD, Lionheart WRB. The spectral properties of the magnetic polarizability tensor for metallic object characterisation. *Math Methods Appl Sci*. 2020;43:78-113.
8. Wilson BA, Ledger PD. MPT-calculator; 2021. Accessed October 15, 2021. https://github.com/BAWilson94/MPT-Calculator
9. NGSolve; 2021. Accessed October 12, 2021. https://ngsolve.org
10. Schöberl J. C++11 implementation of finite elements in NGSolve. Technical report, ASC Report 30/2014, Institute for Analysis and Scientific Computing, Vienna University of Technology; 2014.
11. Schöberl J, Zaglmayr S. High order Nédélec elements with local complete sequence properties. *COMPEL-Int J Comput Math Electr Electron Eng*. 2005;24(2):374-384.
12. Schöberl J. NETGEN — an advancing front 2D/3D-mesh generator based on abstract rules. *Comput Vis Sci*. 1997;1(1):41-52.
13. Özdeğer T, Davidson JL, Van Verre W, Marsh LA, Lionheart WR, Peyton AJ. Measuring the magnetic polarizability tensor using an axial multi-coil geometry. *IEEE Sens J*. 2021;21:19322-19333.
14. Özdeğer T, Ledger PD, Lionheart WRB, Peyton AJ. Measurement of GMPT coefficients for improved object characterisation in metal detection. *IEEE Sens J*. 2022;22:2430-2446.
15. Makkonen J, Marsh LA, Vihonen J, et al. KNN classification of metallic targets using the magnetic polarizability tensor. *Meas Sci Technol*. 2014;25:055105.
16. Van Verre W Özdeğer T, Gupta A, Podd FJW, Peyton AJ. Threat identification in humanitarian demining using machine learning and spectroscopic metal detection. Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning (IDEAL); 2019:542-549; Springer.
17. Marsh LA, Ktistis C, Järvi A, Armitage DW, Peyton AJ. Three dimensional object location for multiple objects using a walk-through metal detector. *Meas Sci Technol*. 2014;25:055107.
18. Karimian N, O'Toole MD, Peyton AJ. Electromagnetic tensor spectroscopy for sorting of shredded metallic scrap. Proceedings of the IEEE SENSORS 2017 - Conference; 2017; IEEE.
19. Rehim OAA, Davidson JL, Marsh LA, O'Toole MD, Peyton AJ. Magnetic polarizability spectroscopy for low metal anti-personnel mine surrogates. *IEEE Sens J*. 2016;16:3775-3783.
20. Zhao Y, Yin W, Ktistis C, Butterworth D, Peyton AJ. Determining the electromagnetic polarizability tensors of metal objects during in-line scanning. *IEEE Trans Instrum Meas*. 2016;65:1172-1181.
21. Zhao Y, Yin W, Ktistis C, Butterworth D, Peyton AJ. On the low-frequency electromagnetic responses of in-line metal detectors to metal contaminants. *IEEE Trans Instrum Meas*. 2014;63:3181-3189.
22. Rehim OAA, Davidson JL, Marsh LA, O'Toole MD, Armitage D, Peyton AJ. Measurement system for determining the magnetic polarizability tensor of small metallic targets. Proceedings of the IEEE Sensor Application Symposium; 2015.
23. Makkonen J, Marsh LA, Vihonen J, et al. Improving the reliability for classification of metallic targets using a WTMD portal. *Meas Sci Technol*. 2015;26:105103.
24. Davidson JL, Abdel-Rehim OA, Hu P, Marsh LA, O'Toole MD, Peyton AJ. On the magnetic polarizability tensor of US coinage. *Meas Sci Technol*. 2018;29:035501.
25. Ammari H, Chen J, Chen Z, Volkov D, Wang H. Detection and classification from electromagnetic induction data. *J Comput Phys*. 2015;301:201-217.
26. Wilson BA, Ledger PD. Efficient computation of the magnetic polarizability tensor spectral signature using POD. *Int J Numer Methods Eng*. 2021;122:1940-1963.
27. Bishop CM. *Pattern Recognition and Machine Learning*. Springer; 2006.
28. Golub GH, Loan CFV. *Matrix Computations*. Johns Hopkins University Press; 1996.
29. Smith OK. Eigenvalues of a symmetric 3×3 matrix. *Commun ACM*. 1961;4:168.
30. Schmidt K, Sterz O, Hiptmair R. Estimating the eddy-current modeling error. *IEEE Trans Magn*. 2008;44(6):686-689.
31. Kuhn M, Johnson K. *Applied Predictive Modeling*. Springer; 2013.
32. Manning CD, Raghavan P, Schütze H. *An Introduction to Information Retrieval*. Cambridge University Press; 2009.
33. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning*. Springer Series in Statistics; 2009.
34. Géron A. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media; 2019.
35. Cristianini N, Shawe-Taylor JA. *Introduction to Support Vector Machines and other Kernel-Based Learning Methods*. Cambridge University Press; 2000.

36. Richard MD, Lippmann RP. Neural network classifiers estimate Bayesian a-posteriori probabilities. *Neural Comput*. 1991;3:461-483.
37. Friedman JH. Greedy function approximation: a gradient boost machine. *Ann Stat*. 2001;29(5):1189-1232.
38. Elith J, Leathwick JR, Hastie T. A working guide to boosted regression trees. *J Anim Ecol*. 2008;77(4):802-813.
39. Powers DMW. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Int J Mach Learn Technol*. 2011;2:37-63.
40. Shmueli B. Multi-class metrics made simple, part II; 2020. Accessed January 31, 2020. https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1
41. Shmueli, B. Multi-class metrics made simple, part III; 2020. Accessed January 31, 2020. https://towardsdatascience.com/multi-class-metrics-made-simple-the-kappa-score-aka-cohens-kappa-coefficient-bdea137af09c
42. Cohen J. A coefficient of agreement for nominal scales. *Educ Psychol Meas*. 1960;20(1):37-46.
43. Sammut C, Webb GI. *Encyclopaedia of Machine Learning*. Springer Science & Business Media; 2011.
44. Powers DMW. What the F-measure doesn't measure: features, flaws, fallacies and fixes; 2015. arXiv preprint arXiv:1503.06410.
45. Wilson BA. Characterisation and Classification of Hidden Conducting Security Threat Objects using Magnetic Polarizability Tensors. Ph.D. thesis, Swansea University; 2022. Submitted.
46. Jones H. Copper-nickel 90/10 and 70/30 alloys technical data. Technical report CDA Publication TN31, Copper Development Association and others; 1982.
47. Ho CY, Ackerman MW, Wu KY, et al. Electrical resistivity of ten selected binary alloy systems. *J Phys Chem Ref Data*. 1983;12(2):183-322.
48. Snoek J, Larochelle H, Adams RP. Practical Bayesian optimization of machine learning algorithms. *Adv Neural Inf Process Syst*. 2012;25:2960-2968.
49. Falkner S, Klein A, Hutter F. BOHB: robust and efficient hyperparameter optimization at scale. ICML'2018; 2018.
50. Yoo Y. Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches. *Knowl-Based Syst*. 2019;178:74-83.

How to cite this article: Wilson BA, Ledger PD, Lionheart WRB. Identification of metallic objects using spectral magnetic polarizability tensor signatures: Object classification. *Int J Numer Methods Eng*. 2022;1-36. doi: 10.1002/nme.6927