Applications of Machine Learning for Electronic Warfare Emitter Identification and Resource Management

Kyle A. Casterline, Nicholas J. Watkins, Jon R. Ward, William Li, and Matthew J. Thommana

ABSTRACT

Electronic warfare (EW) operators face a multitude of challenges when performing single- and distributed-platform sensing and jamming tasks in increasingly dense and agile threat environments. During an engagement timeline, actions often must be taken quickly and based on the partial information available. Recently, the world has observed a boom in artificial intelligence, a suite of data-driven lateral technologies that has already disrupted multiple fields where autonomy and big data are key elements. Although it is not the solution to all EW tasks, artificial intelligence shows promise in offering potential solutions to improve EW efficiency and effectiveness through informed decision-making beyond the capability of a human operator. The Johns Hopkins University Applied Physics Laboratory (APL) Precision Strike Mission Area has invested in research and development in the specific EW tasks of emitter identification and autonomous resource allocation. This article presents promising results from these projects and describes recommended future work in these areas, as well as additional EW applications that may benefit from research in artificial intelligence.

INTRODUCTION

The electronic warfare (EW) operational threat space is one of noncooperative interactions between multiple radio frequency (RF) sensing and transmitting platforms operating in contested spectral environments. This operational space poses many challenges to operators performing tasks such as sensing the electromagnetic (EM) spectrum, effectively managing spectral resources, and sharing critical information across multiple EW platforms, all while jamming threat emitters. These challenges have driven the need for increasingly capable RF systems to process and act on large volumes of information at machine speeds, often with little to no human intervention. The next generation of software-defined RF threat emitters, using increasingly complex agile waveforms, has driven shifts in how future electronic support (ES) and electronic attack (EA) activities are conducted. The jamming platform's overall effectiveness is constrained by its ability to efficiently detect, characterize, jam, and communicate threat waveforms while intelligently managing the RF resources available within the battlespace.

The field of artificial intelligence has received a great deal of attention over the past several years, with multiple major breakthroughs in areas such as object recognition,

natural language processing, and automatic speech recognition. The successful application of machine learning (ML) techniques to other problem domains has generated interest from EW sponsors, operators, and researchers seeking to determine how ML approaches can address EW gaps. This article focuses on several promising results achieved through APL's Precision Strike Mission Area (PSMA) independent research and development (IRAD) projects specific to the EW tasks of emitter identification and autonomous resource allocation. It also includes additional recommended research topics for maturing automated EW approaches that could transition to future military platforms. Furthermore, this article highlights several domain-specific challenges and suggests future research topics within this focus where the use of ML techniques may show promise.

Background

Figure 1 illustrates a scenario that seeks to achieve information dominance and the delivery of overwhelming EW effects against adversaries through the use of collaborative EW. (See the article by Ward et al., in this issue, for details on this vision.) The development of ML applications to improve EW efficiency and effectiveness in the single-platform context is an essential building block for achieving collaborative, autonomous, and adaptive EW capabilities. In this article, we primarily describe the results and knowledge gained from two IRAD projects that applied ML to sensing and emitter identification to investigate automatic modulation recognition (AMR) and autonomous resource allocation. This foundational work demonstrates the merit of these approaches and establishes a growth path to multiplatform, collaborative EW capabilities. The approaches and results presented in this article are platform agnostic, and although the primary platforms considered to date have been airborne, there are probably compelling ground- or sea-based applications as well.

ML APPLICABILITY TO EW TECHNICAL AND OPERATIONAL CHALLENGES

Figure 1 broadly illustrates two challenging topic areas within tactical EW operations for which ML offers promising solutions. This section briefly describes the challenges associated with distributed sensing and distributed resource management.

Engagement of Agile Threat Emitters

EW systems are challenged by the agility of adversary sensors and communications systems that rapidly adapt and operate throughout the EM spectrum. Traditional EW systems must first identify a threat radar to determine the appropriate preprogrammed EA technique. The effectiveness of this approach degrades as radars evolve from fixed analog systems to programmable digital variants with unknown behaviors and agile waveforms.[1] Future radars will probably present an even greater challenge as they will be capable of sensing the environment while adapting transmissions and signal processing to maximize performance and mitigate interference effects. Similarly, communications systems are able to adapt frequency, modulation and coding, and protocol to operate in the presence of various degraded channel conditions with the objectives of maximizing

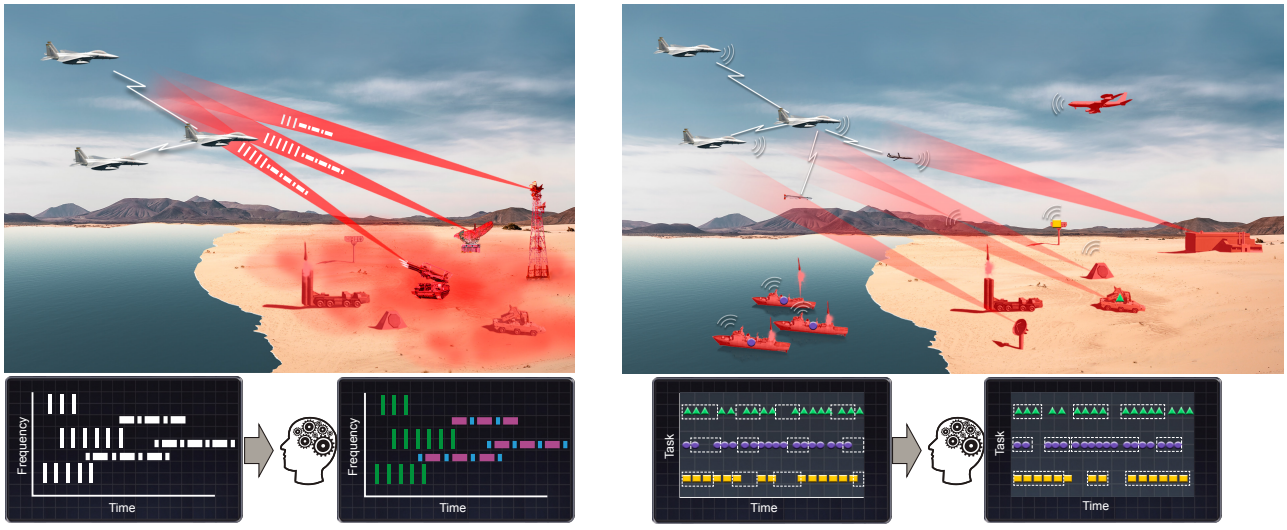

Figure 1. Two topics of interest for collaborative EW within APL's PSMA. The United States will establish spectrum supremacy and deliver overwhelming EW effects against adversaries via the employment of collaborative EW. To realize this objective, two challenges must be overcome: distributed sensing, where ML AMR will identify specific signals of interest (left); and distributed resource management, where sensing and jamming resources are automatically managed in the battlespace (right).

data throughput while minimizing frame error and bit error rates. Furthermore, modern sensor systems' ability to work in a more agile and less deterministic manner have been substantially enhanced by advancements in hardware, software, and adaptive signal processing. Countering these potential threats requires agile EW engagement options that adapt EA techniques based on the current observed operating parameters and modes of the threat at a given snapshot. Often, this kind of engagement will require adapting countermeasures at machine speed on the threat emitter's timescale—in other words, acting much faster than human operator speeds (on the order of milliseconds or microseconds, not seconds). Two Defense Advanced Research Projects Agency (DARPA) programs focus on this problem space: the Behavioral Learning for Adaptive Electronic Warfare (BLADE) program has successfully applied ML techniques to agile communications signals, and the Adaptive Radar Countermeasures (ARC) program has successfully applied ML to threat radar signals.[1,2] Given the body of foundational work established in this area, the application of ML to the engagement of threat emitters is not discussed further in this article.

Wideband Sensing

One of the key challenges for EW systems is providing precise situational awareness of the EM spectrum in real time to characterize the behavior of observed signals and determine what is friendly, threatening, and neutral. Adversary sensing and communication systems are broadening their use of the EM spectrum, requiring ES sensors to simultaneously observe multiple gigahertz of spectrum. Traditional ES systems have limited ability to simultaneously monitor large swaths of the EM spectrum and often resort to scanning a set of narrowband channels. After signal collection, large amounts of signal capture data must be processed on a tactical timescale to characterize emitters and inform EA responses before those measurements become stale. These challenges are further exacerbated when accounting for the operational realities of encountering multiple high-density RF emitters operating at different RF power levels, observing partial signals, and sensing in the presence of high levels of onboard and offboard RF interference.

In addition to being valuable in sensor-based applications (e.g., analog processing), ML methods may also potentially have a role in alleviating ES data-processing bottlenecks. The benefits of wideband sensing can be realized only when such sensing is coupled with a signal processor capable of coping with correspondingly large information rates. Even when computationally expensive operations are decoupled from the full input stream, dense signal environments can still overload downstream resources if detections are not filtered efficiently. ML techniques may have a role in discarding low-priority detections earlier in the processing chain, reducing overall system load and saving computational resources for processing detections that are mission critical. In other words, ML may serve a role by catching critical patterns earlier in the processing chain and in fewer steps. This advantage applies to one particular next-step application for the AMR work presented in this article. The AMR work focuses on only modulation recognition, leaving opportunities for potentially fruitful ML investigations into parameters such as signal characterization via bandwidth, center frequency, pulse repetition interval, angle of arrival, or a combination of these.

Resource Management

The use of an EW platform's sensing and jamming resources must be balanced on the basis of the specific mission and threat environment to effectively engage RF communications and radar targets. An EW mission may include engaging a range of threats, from well-known, less-agile threats that require limited sensing support to complex and adaptable threats that require precise sensing and engagement schedules to track and defeat. Traditional EW sensing and jamming resources from a single platform are managed and scheduled on a timescale that can be allocated before the mission or adapted during the mission by an operator. However, the number of adaptable adversary targets that must be sensed and engaged is growing beyond what is feasible with traditional human-in-the-loop methods. Maintaining effectiveness on threat emitter timescales requires autonomous optimization methods that balance and allocate EW resources at machine speed. Future distributed, collaborative EW missions that include multiple platforms working together and adapting to achieve specific EW mission objectives will necessitate distributed resource management.

This seemingly intractable problem can be simplified if we take a Bayesian perspective. At each point in time, the operator has to consider multiple competing hypotheses of what the adversary is doing. The operator then has to consider what the best course of action is from a set of possible hypotheses. These hypotheses can be enumerated and assigned real values representing the strength of each hypothesis based on the accumulated evidence gathered. Bayesian probability theory allows us to represent units of evidence as real numbers that can be used to strengthen or discard competing hypotheses. This approach allows us to automatically weigh the plausibility of different hypotheses and make decisions that are based on the hypotheses most supported by the evidence. We believe ML has the potential to provide significant capability advancement in resource management for EW premission planning and near-real-time decision-making during EW operations.

EMITTER IDENTIFICATION INFORMED BY AMR

The emitter identification process allows ES and surveillance receivers to distinguish emissions from threats from those that are friendly or neutral. Consequently, ES systems must handle a broad set of received waveforms, from ones commonly used in commercial bands to military-specific radar and communications signals. ES systems have relied on pulse processors that use fixed descriptors to detect, filter, and extract emitter information from received RF modulated pulses. Modulation format is one descriptor that ES systems use when characterizing active emitters while surveying the spectrum.

One of the many challenging tasks an EW system must accomplish is efficiently determining the modulation format of a detected signal. This topic is more broadly known as AMR. Traditional pulse processors treat modulation format as a single feature that can be coupled with other waveform characteristics used to identify active emitters. Many pulse processors will match clusters of pulse descriptors against onboard libraries to identify which emitter is being observed. Successful application of this traditional matching approach presupposes that the signal has been previously observed and its characteristics are known.

A second challenge for AMR is recognizing and interpreting newly observed modulation types or emitter patterns when they are encountered. Novel emissions have proved to be challenging for systems that rely on predefined libraries of known emitter characteristics. Before software-defined threats became a reality, the process of capturing and characterizing novel emissions had been a historically tractable problem. Modern adaptive threats, however, have driven the need for sensing techniques that can rapidly recognize and characterize novel detections at machine speeds. Figure 2 illustrates the application of AMR to a notional tactical EW scenario.

Motivation behind Feature Learning for AMR

Although several deep learning approaches have already been applied to AMR,[3–5] in fiscal years 2018 and 2019, the Feature-based Electronic Attack Trained Hypersurface Responses (FEATHR) IRAD project explored deep feature representation models in the context of AMR. The effort highlighted several distinct advantages of these models over other types of deep neural network (DNN) models directly trained for classification.

Much of the existing work applying deep learning to AMR has focused on using neural network models to learn categorical modulation assignments from a fixed set of labeled examples. A variety of studies have shown this approach to be a viable way of performing AMR. However, training a model solely for the purpose of classification inherently bounds the model's predictions to the set of labeled classes in the training data. This restriction results in the trained model's inability to generalize beyond the set of labels presented during training. Extending the class set for a neural network classifier requires a lengthy process of collecting new examples, labeling the examples, and retraining the model with the extended set.

When training a DNN with a classification objective, a model will typically use a fully connected output layer coupled with a normalized exponential to yield a confidence score across the set of classes being learned. Thus, any features learned are never directly observed because they are internal to the network's architecture. However, this is not necessarily the case if the learning objective

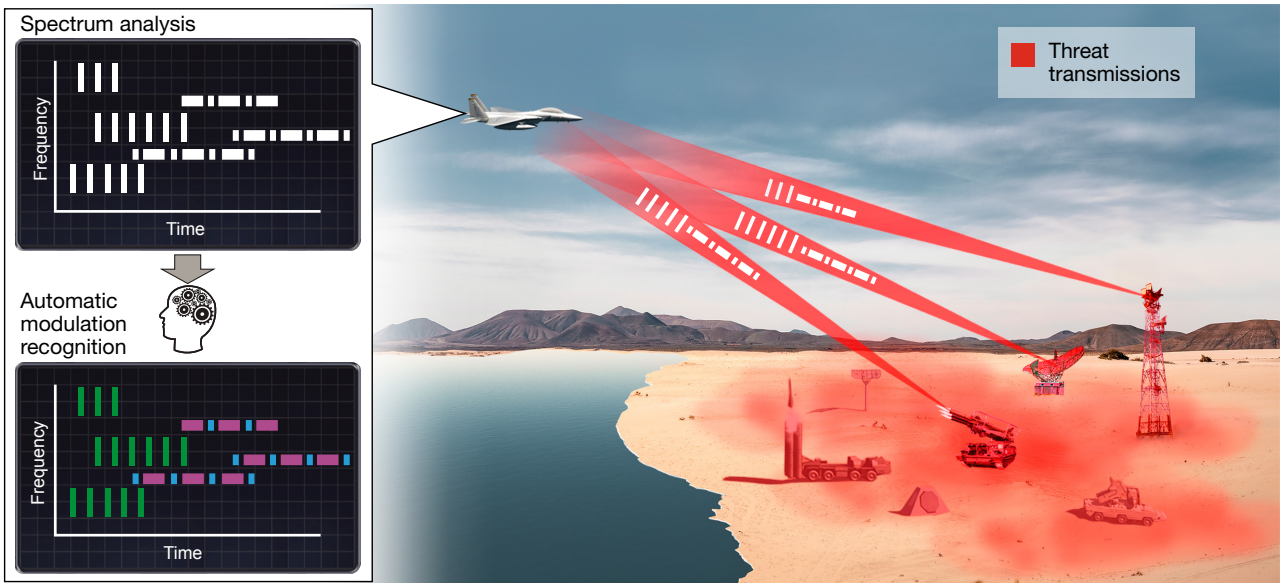

Figure 2. Application of AMR to a notional tactical EW scenario. Shown here is an example of how online AMR may be used to characterize various threats. The upper-left diagram represents a time-frequency distribution of raw detections. The bottom left diagram represents identified radar pulse combinations where each distinct color is used to represent a pulse with specific characteristics.

is targeted at learning features directly. In this context of this problem, features can be thought of as distinctive patterns in the data that are characteristic of a particular signal modulation. This brings us to a set of techniques appropriately called feature learning. While similar to feature extraction, feature learning does not assume predefined rules or transforms for obtaining features directly; rather, desired features are learned through training with an objective task.

To address the need to recognize and categorize open sets of modulations beyond a set of predefined labels, we examine a feature learning approach to performing AMR where features that distinguish differences in modulation types, rather than the modulation types themselves, are learned first. The model's objective is to learn a transformation that maps examples to a position in a multidimensional feature space.

The Triplet Loss

In 2015, researchers at Google Brain published the first paper on the triplet loss showing how to classify individuals.[6] Because the same individual can look different depending on a variety of factors (e.g., lighting, clothing, perspective), researchers realized that there was a need for a method to classify many individuals under nonlinear conditions. By defining the axes of a high-dimensional output space using a neural network, they found that individuals could be clustered together in this space. The term *embedding* is often used to describe the *N*-dimensional vector that corresponds to a position within this output space.

Models trained using the triplet loss, a feature learning approach, yield a transformation that allows data to be mapped into a learned feature space. Examples are aggregated in this space on the basis of learned relationships between modulation classes presented during the training process. This allows us to subsequently analyze these features to both classify known modulations and categorize newly observed examples that are not in our set of existing labels. We implement a two-step approach to first learn a feature representation transform using a residual DNN model[7] trained with the triplet loss. Once the model is trained, we explore two methods for characterizing this feature representation with the learning objectives of modulation classification and anomalous modulation recognition.

The triplet loss is a supervised training objective designed for use with Siamese networks. A Siamese network can be thought of as multiple mirrored instances of a single artificial neural network model, where each instance is initialized and jointly updated the same way throughout the training process. Each output is treated as a mapping of an input example into a position within a common *N*-dimensional Euclidean space (also called an embedding space). For training configurations using the triplet loss, a Siamese network model with three instances is created, as shown in Figure 3.

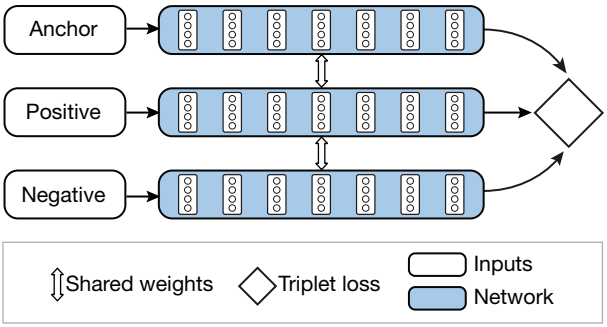

Figure 3. Example of a triplet loss configuration using a Siamese network training configuration. A Siamese network can be thought of as multiple mirrored instances of a single artificial neural network model, where each instance is initialized and jointly updated the same way throughout the training process. Each output is treated as a mapping of an input example into a position within a common *N*-dimensional Euclidean space (also called an embedding space).

The loss, once calculated at the output, is backpropagated through each network instance. There are three types of samples within each batch of training data referenced by the triplet loss: anchors, positives, and negatives. The anchor and positive are two examples that share a common class label, whereas the negative example must belong to a different class. The triplet loss adjusts the relative distances of the anchors, positives, and negatives using Eq. 1:[6]

$$||f(a) - f(p) + \alpha|| \leq ||f(a) - f(n)||. \quad (1)$$

Eq. 1 states that the distance between the anchor and positive must be less than the distance between the anchor and negative by a margin distance of α. Larger margins spread samples farther apart in the *N*-dimensional space being learned. The f in this case is our neural network appropriator that generates an encoding on the basis of an input. Satisfying the triplet loss equation requires that the positive sample be moved closer to the anchor and/or the negative one be pushed farther away. The training process adjusts the axes of the *N*-dimensional space, which simultaneously adjusts the positions of positives and negatives relative to their anchors. Training consists of multiple iterations selecting triplets with new anchors, positives, and negatives. The net effect of this process causes same-labeled examples to gravitate toward one another, forming clusters. Sets of valid triplets can be either formed before training (offline triplet mining) or computed dynamically during training (online triplet mining).[8] Eq. 1 can be rewritten in the following form to perform triplet training:

$$max(||f(a) - f(p)|| - ||f(a) - f(n)|| + \alpha, 0). \quad (2)$$

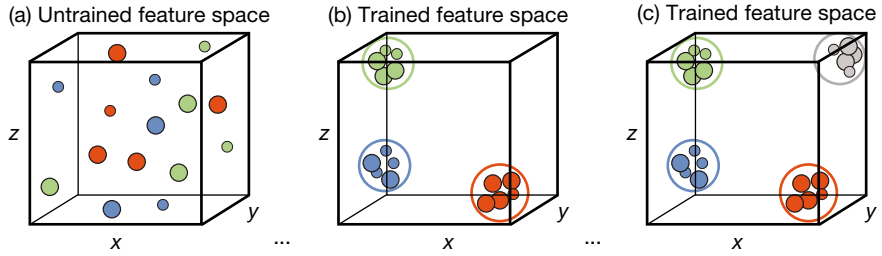

Figure 4. Feature space representations before and after training. (a) Examples are randomly distributed when mapped into the untrained feature space. (b) Once trained, similar examples are aggregated to common locations in the feature space. (c) Placements of new example types depend on the learned features of the existing classes.

Eq. 2 forces the loss to go to 0 when the anchor, positive, and negative are already in the proper orientation. By modifying the loss function, weights and biases of our network are adjusted only when the anchor, positive, and negative are improperly oriented. This version of the equation stabilizes training by preventing weights and biases from being adjusted when the orientation of a given triplet is already valid. The trained *N*-dimensional space is used as our feature representation. When new examples are mapped into the new *N*-dimensional space, the data will be clustered appropriately as shown in Figure 4. In this new feature space, the Euclidean distances between the data points represent learned differences in features.

Using the triplet loss, we create a feature representation where Euclidean distances between samples correspond to learned differences in the data. Once mapped into this feature representation, new waveform examples can be classified through association or identified using unsupervised techniques.

Figure 4a shows a notional feature space representation produced by an untrained neural network model with randomized weights and biases. Examples are randomly distributed throughout the feature space when classes are encoded with an untrained network. Untrained feature spaces do not yet contain meaningful learned features, and there is no way to distinguish classes from one another.

Figure 4b shows a notional feature space where the neural network has been trained with classes A, B, and C using the triplet loss. In the trained feature representation, the weights and biases of the neural network have been adjusted using the triplet loss to properly orient the positions of each class relative to one another. Axes of the learned feature space correlate to features discovered through the training process. In the next section, we explore how a common feature space can allow proper placement of trained classes and also be used to infer the location of new classes on the basis of similarities in learned attributes.

Figure 4c shows an example of identifying a notional, previously unobserved class mapped into an existing feature space. Existing learned features can be exploited to characterize new classes not part of the original training data distribution. Once a common feature representation has been established using a DNN model with the triplet loss, a parametric approach is then used to assign labels to groups of points for classifying new examples. Specifically, Gaussian clusters are used to characterize regions of the feature space that coincide with a specific class. An unsupervised clustering approach is then used to account for regions with unlabeled examples. Our approach uses a clustering technique called OPTICS (ordering points to identify the clustering structure) to assign new clusters to unknown classes.[9] The OPTICS clustering method has several properties that make it a good fit for this application. First, it is a density-based clustering approach and therefore has the ability to reject noisy points located in regions of low density. Second OPTICS uses an additional hierarchical term to account for multiple clusters, each with different densities.

Anomaly Detection Using Learned Features

Once a set of modulation features have been learned, they may be used in performing tasks such as classification or anomaly detection. Figure 5 shows a process that fits a multivariate Gaussian model to the learned features for each modulation type. Anomalies are determined by fixing a log-likelihood threshold value for each

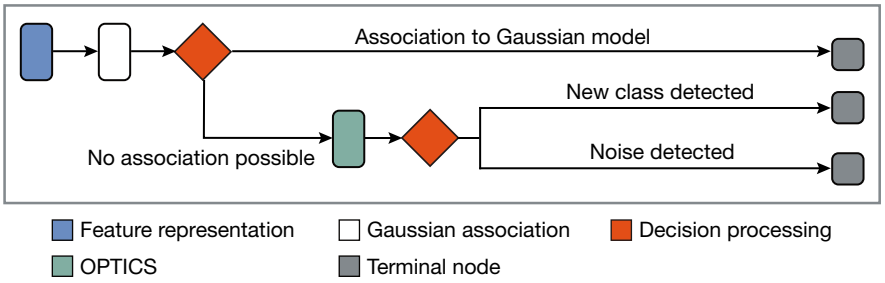

Figure 5. Flow diagram of an anomalous class identification process. This process fits a multivariate Gaussian model to the learned features for each modulation type. Anomalies are determined by fixing a log-likelihood threshold value for each of the Gaussian models. This threshold serves as a decision boundary for associating new examples. Examples that do not associate are also captured and classified as anomalous, indicated by the bottom branch.

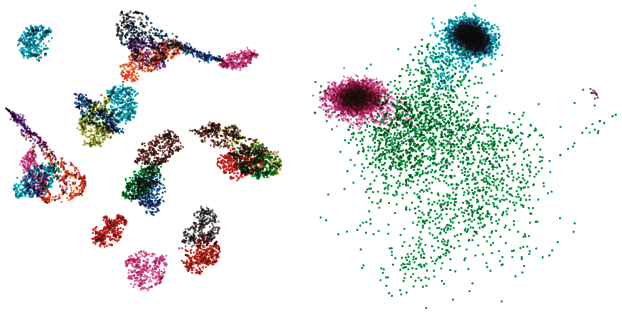

Figure 6. 3-D projections of waveforms mapped within the trained feature space. Left, A projection of waveforms mapped to the trained feature space, where each color represents a different modulation. Positions in this space correspond to different combinations of learned features. Waveforms with similar/matching sets of learned characteristics are placed near each other, forming a cluster. Right, A projection containing only points not associated with any of the modulation clusters produced during the training stage. Upon running OPTICS, green points were identified as noise, while blue and red points were given a new "unidentified modulation" label.

of the Gaussian models. This threshold serves as a decision boundary for associating new examples. Examples that do not associate are also captured and classified as anomalous. This is indicated by the bottom branch in Figure 5. Figure 6 illustrates the 3-D projections of waveforms mapped within the trained feature space.

If we consider more than one type of anomaly detection, an unsupervised clustering step is needed to identify the various possible anomaly types among detections classified as anomalous.

The RF Modulation Data Set

In the experiments described below, we used the open-source RadioML2018 data set released by DeepSig.[10] The data set consists of roughly 2.5 million examples of 24 synthetically generated communications modulation types. Each example is represented as a 1,024-length in-phase and quadrature (IQ) vector sampled in time using floating points. The data include multiple environmental distortions that are commonly observed in collected data. Examples are labeled with a modulation tag as well as a discrete signal-to-noise ratio spanning a range of –20 to 30 dB. The set of modulations in these data are used in two ways: as examples and categories for supervised training and as holdouts for unsupervised anomaly detection. Supervised training uses IQ vectors paired with their appropriate modulation labels, whereas holdout modulations are not given a modulation label. We first learn a feature representation using a subset of 22 modulation types and demonstrate an unsupervised approach for identifying new classes using two modulation categories—frequency modulation (FM) and 16 quadrature amplitude modulation (16QAM)—as holdout classes.

Supervised Training with Holdout Classes

We partitioned our data set into two categories: (1) a supervised category of 22 known classes subdivided into an 80% training partition and a 20% evaluation partition and (2) a category with two holdout classes (FM and 16QAM were withheld during training). Our DNN model was trained on the 22 known modulations (i.e., excluding the two holdouts). Following the procedure described above, we fit multivariate Gaussian distributions to each modulation feature embedding in the training partition. The joint distribution across all known modulations is used to define a threshold distance. This threshold is used as a multidimensional Gaussian boundary for associating new samples to one of the existing classes or for flagging unidentified samples. Samples that fall within this defined multidimensional boundary (indicated by θ in Eq. 3) are associated with one of the existing modulation categories. Otherwise, the samples are given the unidentified label. Note that this step does not assume any prior information about unidentified points; rather, it only excludes outlying or anomalous examples from classification into a known modulation category:

$$(\vec{x} - \vec{\mu})^T S^{-1} (\vec{x} - \vec{\mu}) \leq \theta. \quad (3)$$

Our goal is now to identify our holdout modulations as new clusters among our anomalous examples. To do so, we run OPTICS[9] against the set of anomalous examples identified during the previous step to create labels for examples that fall in regions with sufficient density.

The confusion matrix in Figure 7 shows the accuracy of this process for examples at a signal-to-noise ratio of 10 dB. The true holdout classes (FM and 16QAM) are shown as grayed-out columns. Here we show new cluster 1 and new cluster 2 as identified categories showing a strong correlation with the set of examples in the holdout classes. The unidentified category contains examples that fell outside our distribution threshold and were categorized as noise samples by OPTICS. New cluster 3 is a false detection produced by a dense region of misclassified unidentified samples from other modulation classes.

Next Steps

We view feature learning as a generalizable approach for building rich feature representations of RF modulations, enabling the identification and classification of newly observed RF signals. In many edge applications in the RF domain, it is often impractical to retrain neural network models once they are deployed. Using a consistent learned feature representation coupled with a simpler classification model enables dynamic recognition of RF modulations without the high processing demands of retraining a single model. Postprocessing a feature representation with a simple classification model

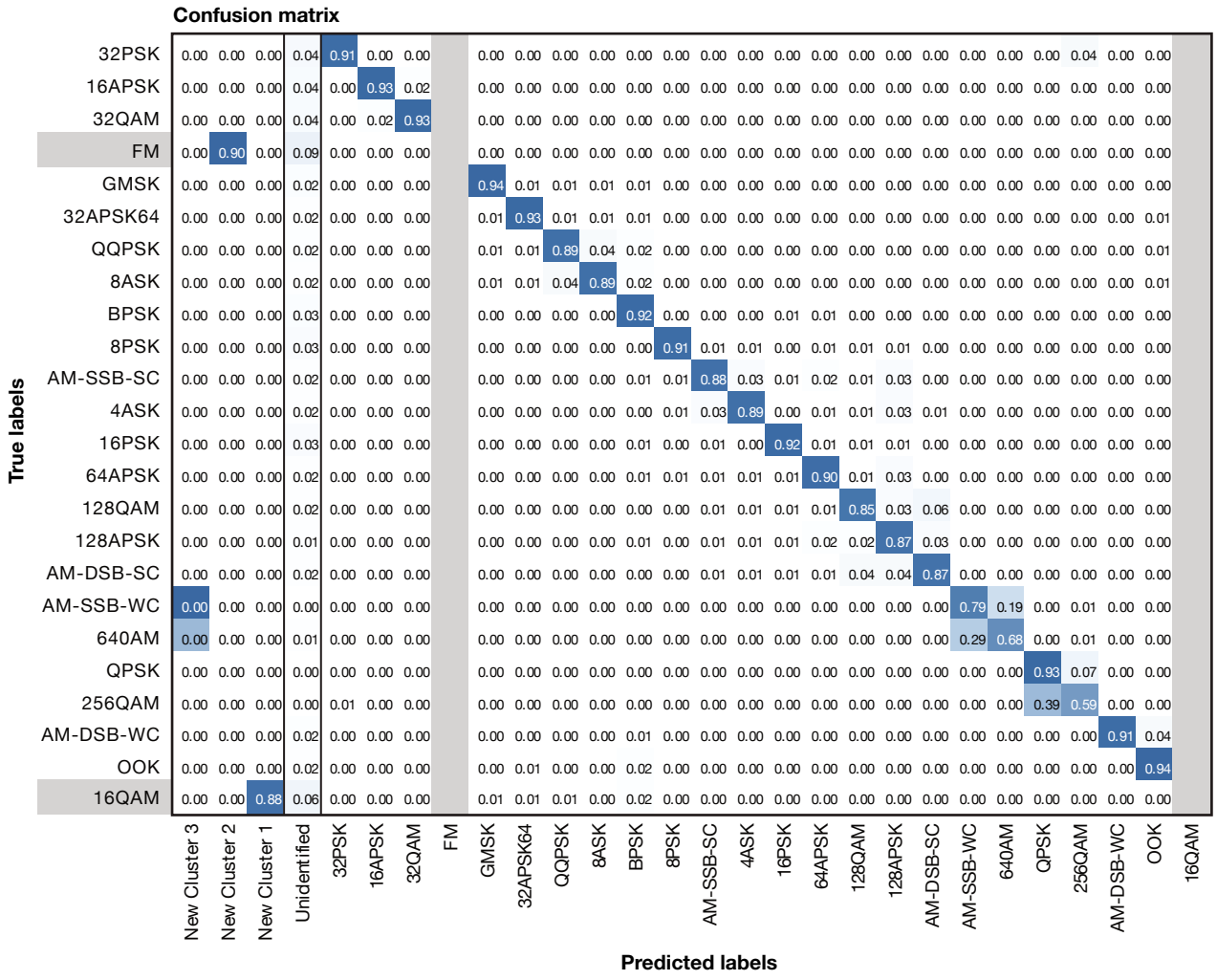

Figure 7. Confusion matrix illustrating the accuracy of the AMR process for examples at a signal-to-noise ratio of 10 dB. The true holdout classes (FM and 16QAM) are shown in the grayed-out columns. New cluster 1 and new cluster 2 are identified categories showing a strong correlation with the set of examples in the holdout classes. The unidentified category contains examples outside of the distribution threshold and categorized as noise samples by OPTICS. New cluster 3 is a false detection produced by a dense region of misclassified unidentified samples from other modulation classes.

requires far less computing power than that needed for retraining. Furthermore, unsupervised techniques can be employed to search for new waveforms using existing learned feature representations that have shown significance in distinguishing other modulations. We are interested in expanding on several elements of this work.

First, we anticipate that several factors may improve this technique's performance. The original implementation of the triplet loss was trained using a far greater class count than what we used in this work. Higher class counts allow the network to learn a richer feature representation capturing finer differences between samples. In our experiments, we used a data set consisting of 24 unique modulation types. However, we anticipate that training with additional unique modulations will improve performance by increasing the variety of the triplets used during training.

Second, when considering a breadth of signal types, note that different types of features are important in distinguishing signal types from one another. For example, the feature set distinguishing 16QAM modulation from quadrature phase shift keying will be different from the feature set distinguishing a linear frequency-modulated pulse from a nonlinear frequency-modulated pulse. Because of this, it makes sense to consider hierarchical patterns in how signals relate to one another. Other active APL efforts have demonstrated success in using hierarchy in performing AMR with DNNs. A potential follow-on research effort may be to explore how feature learning can be combined with signal hierarchies to improve classification performance on a broad set of signal types. One potential solution may involve constructing a nonbinary classification tree using a separate learned feature set at each decision point. This approach

could allow smaller/simpler models to be used because each feature set would no longer have to capture the full set of signal types.

Third, this approach assumes examples are captured as detections represented as an IQ vector isolated in time and frequency. There is probably utility in exploring various other waveform representations, such as complex time-frequency data or sparse detection samples. Furthermore, combining learned modulation-specific features with other contextual data such as direction, wave polarization, or geographic location will be necessary to form specific emitter identities with high confidence. Finally, the source and characteristics of the false detection produced in Figure 7 are an area of further investigation.

AUTONOMOUS RESOURCE ALLOCATION

An automated solution to emitter characterization, as previously described, provides a useful means of mapping observed RF data into specific observed adversary threat emitters. However, this alone is not sufficient for providing a robust EW response. The AMR results presented in the previous section generally assume isolated detections and complete signal captures are available to inform the AMR decision process. The real-world situation an EW platform typically encounters is much more complex. Usually, an EW platform must jam more possible threat emitters across more frequencies than it can simultaneously cover and at duty cycles that do not allow sufficient receiver sampling of the threat environment. Therefore, the limited resources available to an EW system must be appropriately tasked such that accurate snapshots of threat emitters and their corresponding behaviors are balanced with jamming responses. We refer to the problem of how best to allocate our available EW assets as the resource-allocation problem and illustrate it with operational context in Figure 8.

Although in practice we may wish to allocate many assets to optimize system performance, here we consider resource-allocation problems with two distinct action types: sensing and jamming. Sensing refers to the action of detecting and identifying the type of waveforms present in the environment; jamming refers to the action of interfering with an adversary's waveforms. In general, the objective is to maximize jammer on-time and minimize useful sensing time (i.e., jammer off-time). The two problems are interconnected because sensing of the environment informs tailoring and focusing of jammer techniques to improve overall jamming effectiveness, but at the expense of jammer off-time.

In fiscal years 2020 and 2021, PSMA funded the Intelligent Learning Electronic Attack Maestro (IL'EA Maestro) IRAD project to develop a method of addressing the resource-allocation problem. The long-term vision of IL'EA Maestro is to enable future deployment of scalable, distributed, multiplatform approaches to enable autonomous resource allocation. Although one may use myriad potential methods to design such a system, the IL'EA Maestro team has identified the use of model-based stochastic optimization coupled with approximate Bayesian inference as a particularly promising approach. In contrast to standard off-the-shelf reinforcement learning algorithms, this approach allows system designers to incorporate significant domain

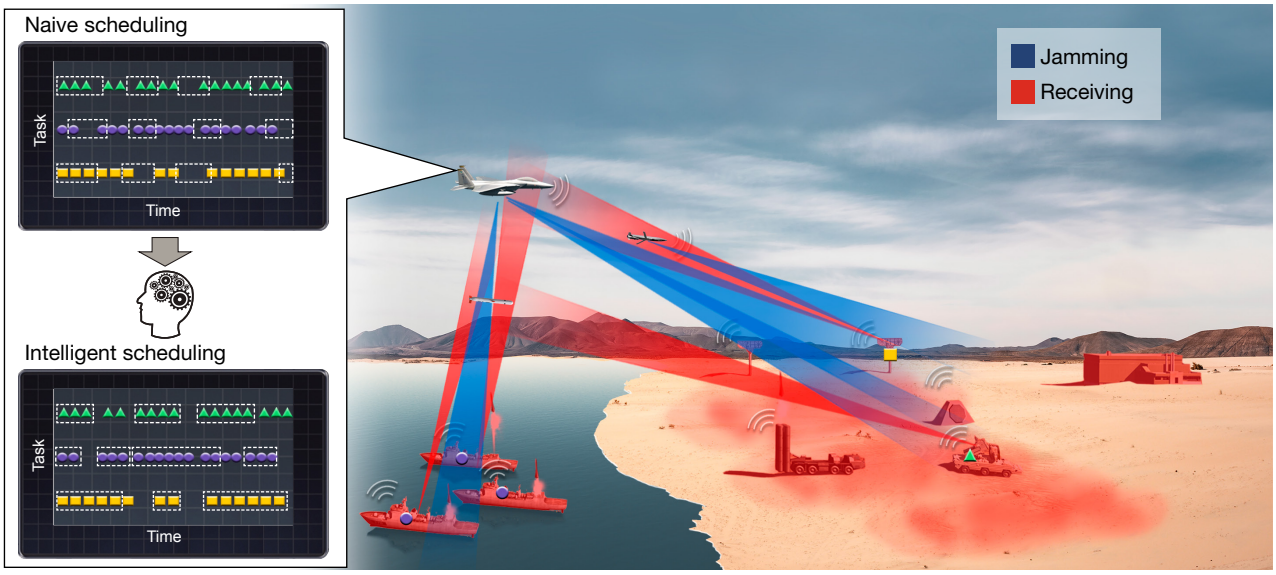

Figure 8. Illustration of autonomous resource allocation in a tactical EW setting. For illustration only, the ES tasks are shown and shapes represent emissions from Red threat emitters. If ES tasks are scheduled without intelligence, sensing intervals do not optimize sensing of pulses (top left diagram), shown as dashed boxes that miss many of the pulses. If tasks are scheduled intelligently, they may be performed more efficiently (bottom left diagram), where ES intervals are dynamically scheduled to capture more pulses.

knowledge (known limitations of adversary capabilities, observed behaviors based on prior sensing of adversary threats, etc.) into the agent's design. Whereas with a standard reinforcement learning algorithm, we would hope the agent could determine adversary vulnerabilities on its own if given enough training time on a simulator, building such knowledge into the system decreases the learning burden. The net effect is that the system performance increases because less data are required to learn a useful model of the adversary, thus requiring less overall sensing time. In the remainder of this section, we describe the resource-allocation problem and explore early results of the IL'EA Maestro project.

The Resource-Allocation Problem

One way to address autonomous resource allocation is through Bayesian probability theory (see, e.g., Koller and Friedman[11]). We start by initializing a broad set of possible adversarial behaviors given a priori information—for example, expected adversary frequency ranges and timing patterns. We then allocate resources to collecting signals from different sections of the frequency spectrum at different moments in time to build a set of evidence that can support the belief that a particular strategy is followed, or not. After each attempted signal acquisition (i.e., scan), we update our beliefs about which behaviors are possible to reflect the new evidence added to our knowledge base.

To better frame the developments of this section, let us now formally describe a general instance of the resource-allocation problem considered. We consider a partition of the subset of the RF spectrum of interest. We consider each element of the partition to be a particular frequency channel, of which there are a total of c channels. We partition the time axis into discrete elements, each of which is of duration Δt. If we denote by C the set of channels and T the set of times under consideration, we have that the signals of interest under consideration evolve as functions of time on the $C \times T$ product space. Figure 9 shows one potential emitter represented in a discrete time and frequency space, where $C = \{1, 2,..., 9\}$, $T = \{1, 2,\ldots, 20\}$, and colorized grid spaces represent individually received bursts from within a pulsed waveform. Note that the red signal is frequency agile—it jumps from channel 4 to channel 7 between the fourth and fifth bursts.

We denote by S the set of signals of interest in the environment. We assume the ability to identify each received pulse as a particular signal of interest as each pulse is encountered. Importantly, we do not assume full prior information regarding the signal's pattern behavior (e.g., pulse length, frequency hop sequence, duty cycle). We assume that (1) the set of channels each signal can appear on is contiguous (i.e., that it appears between a set minimum and a set maximum frequency); (2) the signals of interest are periodic; and (3) signals are not mutually

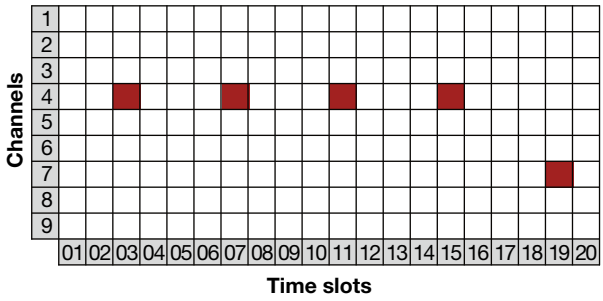

Figure 9. A potential emitter represented in a discrete time and frequency space. Colorized grid spaces represent individually received bursts from within a pulsed waveform. The signal in red jumps from channel 4 to channel 7.

interfering (i.e., two signals cannot occupy the same frequency channel at the same time). Strictly speaking, these assumptions simplify the analysis required of our current prototype. Relaxing these assumptions to accommodate a larger set of possible signals is possible, although doing so comes at a cost of requiring more data to learn useful signal characteristics. Which set of assumptions is best in practice will be determined by context.

We consider a case in which the platforms under consideration have both sensing and jamming capabilities. We assume each transceiver capable of performing a sensing or jamming action can do so only over a contiguous span of channels and that if multiple transceivers are available, they can be allocated independently with the understanding that they can cause self-interference if not coordinated appropriately.

Bayesian Agency

Bayesian agency is a fundamental approach for developing intelligent learning systems that must interact with the world in a meaningful way. It can be considered as a nontraditional, abstract way of designing reinforcement learning agents that have strong assumptions built in about the nature of the world and how to reason about it. Such assumptions can and should be designed in conjunction with human subject-matter experts. These assumptions include (1) the set of hypotheses the agent considers as possibilities for the world; (2) a distribution of prior beliefs regarding the relative likelihood of those possibilities; (3) a method of incorporating data obtained from environmental observations to update the beliefs regarding the agent's underlying hypothesis; (4) a method for assessing the relative value of a potential sequence of future actions; and (5) a method for choosing a particular action to apply to the environment. We may decompose items 1–5 into a set of behavior blocks as illustrated in Figure 10.

We may regard items 1–3 as constituting the agent's learner (object A in the figure), item 4 as constituting the agent's evaluator (object B in Figure 10), and item 5 as constituting the agent's actuator (object C in

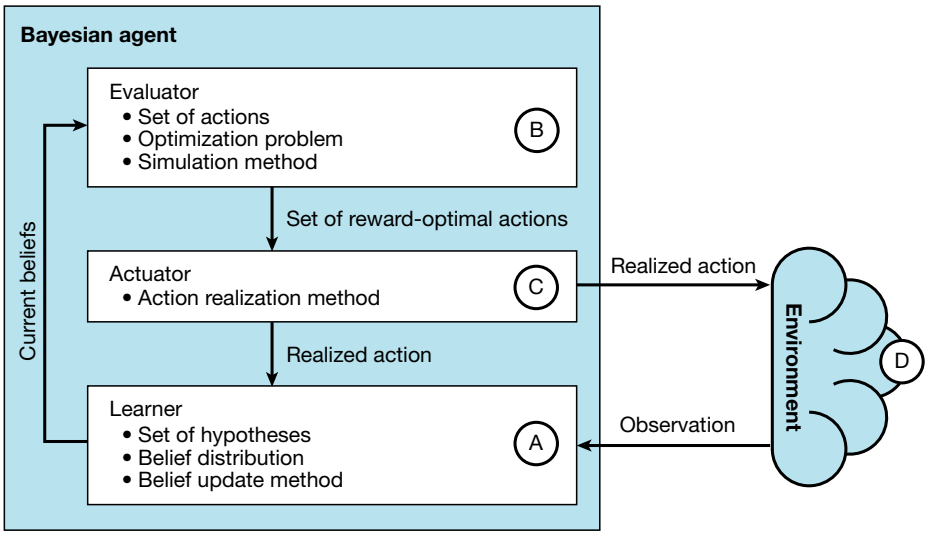

Figure 10. Architecture of a Bayesian agent. Items are decomposed into a set of behavior blocks. The agent interfaces with the real through the actuator, which applies an action to the environment, and the learner, which takes in an observation from the environment.

Figure 10). We see that the agent interfaces with the real world (the environment; object D in the figure) through the actuator, which applies an action to the environment, and the learner, which takes in an observation from the environment.

A strength of this approach is that appropriate selections for the learner, evaluator, and actuator can significantly improve the rate at which the agent learns to effectively interact with its environment. It is worth noting that particular choices of assumptions can restrict the agent from being general in the sense that the set of feasible control policies admitted by the assumptions may be strictly smaller than the set of all feasible control policies. If we so choose, we can design the learner, evaluator, and actuator to mimic standard general-purpose reinforcement learning algorithms such as Q-learning, deep Q-learning, policy gradient, and so forth.[12] However, by constructing them with a specific purpose in mind, we can attain good performance with less training time than we can using an off-the-shelf approach. Therefore, the agent's designers are themselves in control of how much generality is lost.

Before discussing any particular technical details involved in developing a Bayesian agent for the resource-allocation task considered here, let us first discuss some abstract principles required to design a Bayesian agent that facilitates a high level of operation: approximate Bayesian belief propagation, belief-driven action evaluation, and action selection.

Approximate Bayesian Belief Propagation

In an ideal setting, Bayes' theorem (Eq. 4) is an optimal method for updating belief distributions (i.e., the set of probability distributions over the set of possible hypotheses; see, e.g., Pearl[13]). However, when the sets of hypotheses and/or possible observations grow too complex, exact belief propagation becomes intractable. Concretely, without some context-dependent simplifying assumptions, the computation of the marginal probabilities required by the Bayesian belief update expression requires marginalizing over all possibilities. This operation typically grows exponentially with the size of the observation set because all combinations of observations must be explicitly considered. There is a wide body of literature studying approximate Bayesian learning/belief propagation.[14]

$$P(A|B) = \frac{P(A,B)}{P(B)}. \tag{4}$$

The key takeaway is that exact belief propagation, in the sense that the designed agent follows Bayes' theorem exactly as prescribed, is usually too complex to be reliably incorporated into an agent's design. This does not imply that nothing reasonable can be done. However, it *does* imply that a certain amount of subject-matter expertise is required in choosing an appropriate method of belief propagation for a particular application. For details regarding how this can be done in our context, refer to the example in the appendix.

Belief-Driven Action Evaluation

We now move to the topic of belief-driven action evaluation. Under ideal circumstances, we would have access to a perfect model of the world, be it a set of analytical mathematical relations or a computational oracle. However, this is often not the case, as with resource allocation for EW. Nonetheless, some means of evaluating the future consequences of present and past actions and observations is necessary for an agent to make intelligent decisions. Instead of attempting to learn or design a perfect model of the agent's environment, we use the distribution of the agent's current beliefs for this subtask.

From the above, we see how to design a mechanism for updating a distribution over the agent's set of possible hypotheses. Using such a distribution, it is easy to understand that it is possible to compute statistics on the return awarded to an agent for a particular sequence of actions. If the set of hypotheses and their associated

belief distributions are sufficiently simple, this action may be done analytically, and closed-form evaluations can be used to simulate the objective. However, the general case requires that some form of sample-based approximation algorithm be implemented.

It is at this point worth noting that many methods can be used to evaluate the consequences of actions. As is the case in standard reinforcement learning, we could simply evaluate the expected return of each action and select the best alternative according to our current beliefs. However, we could equally well design instead an optimization problem, the solution—or approximate solution—of which informs the agent's actions. Perhaps the most central distinction to be made on this point is that an optimization problem can be specified with known explicit constraints, where the design of a reward function hopes to sufficiently encode constraints through associating negative rewards (i.e., penalties) with operationally infeasible state/action pairs. A reader fluent in modern control theory will recognize this distinction: one that separates receding horizon optimal control and model predictive control. In the former, the common practice is to use a penalty function to enforce constraints on control signals. In the latter, explicit constraints are usually built into the optimization problem used to implicitly define the control law (see, e.g., Borrelli, Bemporad, and Morari[15]).

Action Selection

The last subtask to consider in the design of a Bayesian agent is action selection. This may seem trivial. It is tempting to assert that one can always simply select the best action as specified by the optimization method developed in the previous subsection. However, such actions are not always unique. In fact, empirically, resource-allocation problems often include many actions with the same expected value. This may occur, for example, if several signals to be jammed are given the same priority weight and are equally likely to appear at a particular moment in time. In this case, one must carefully consider how to select a particular optimal action.

Of course, one can select an action randomly (even arbitrarily) over the set of all optimal actions without explicitly affecting the predicted quality of the system's performance. However, it has been observed to be beneficial to select actions according to a secondary purpose. For example, if one can easily generate statistics of the action's rewards, one can choose the expected reward as a primary objective and minimizing variance among actions with the optimal expected reward as a secondary purpose. To do so, we can order cost-optimal actions according to their estimated variance and select the action exhibiting the least variance. Doing so can significantly improve the agent's reliability by reducing noise in the effect of the agent's decisions.

Evaluation of Current Status

Now we can detail the current status of our solution, some of the underlying design decisions made, and its current performance. The choice for a set of hypotheses is straightforward enough: given prior assumptions about how many signals exist in the environment and their corresponding periods, a finite set of possible signals can be generated. This is our set of hypotheses.

As data are collected from the environment, they are used to update the agent's beliefs regarding the hypotheses. As evidence is collected from the environment using scanning resources, the probability associated with certain hypotheses will increase, indicating they are more likely to be true. In the case of our solution, we make certain axiomatic assumptions about the world, which we try to ground in realism. This is done in an effort to limit the extent of the hypothesis set to only cases that are operationally plausible.

The method for updating the distribution of beliefs is somewhat more abstract. If the agent has performed a scan at a particular time t and has observed a particular signal q at time t, the agent then rules out any hypotheses that state that signal q does not exist at time t. The process accumulates more information until eventually only the particular hypotheses that are true for the environment remain. Likewise, we update the agent's beliefs whenever a signal is observed on a particular channel. Such an observation can be used to support any hypotheses about which signals appear on the particular channel. At a minimum, this causes the agent to believe that the signal is more likely to remain in the particular channel in the immediate future. However, some observations can change the agent's beliefs more dramatically—for example, by informing the agent that a particular signal can exist on a channel on which it had hitherto been unobserved. The method for evaluating the value of the agent's action is likewise involved.

Abstractly, the goal of the resource-allocation agent is to learn what is happening in the environment and how best to interfere with the adversary's actions. To this end, a stochastic optimization problem is designed on the basis of the agent's current belief state. If the agent is more certain about how the environment is going to act at a particular moment, the objective prioritizes jamming adversarial signals highly. If the agent does not have a strong belief about what is going to happen presently or in the near future, information gathering in the form of scanning is prioritized. When well specified, such an optimization problem induces a behavior that quickly discovers, characterizes, and jams highly valued adversarial signals. Concurrently, it gradually learns of the behavior of lower-value signals and how best to block them without sacrificing performance with respect to high-value signals.

To investigate this further, let us look at an example of a resource-allocation problem and our current

method's performance with respect to it. We consider an environment with 24 frequency channels and eight distinct adversary emitters. The period lengths, pulse lengths, and interpulse lengths for each of the signals were generated randomly. The signals were dispersed randomly in the frequency-time space, where the signals were allowed to jump to different frequencies so long as (1) the emitter could operate on the desired frequency and (2) the frequency was available (i.e., not in use by other adversarial emitters). The signals were assigned utility values taking positive integer values ranging from 1 to 4 (chosen at random). Utility was accrued for the agent whenever a signal was jammed, with the amount awarded being the signal's value.

Figure 11 summarizes the statistical performance of three resource-allocation algorithms: (1) a strategy that jams adversarial emitters to accrue optimal expected rewards after collecting sufficient emitter pattern data (in blue), (2) uniformly random jamming across channels (in orange), and (3) jamming performance assuming perfect knowledge of the emitter patterns (in gray). Note that each algorithm was scored on its ability to jam appropriate signals, and the signals present had myriad scoring weights. Signals were assigned weights roughly in proportion to their perceived importance so that higher scores correlate with jamming important adversary capabilities (i.e., those associated with higher utility values) more frequently.

One thousand sample runs were performed. Histograms of each strategy's scores are given in Figure 11, where the dashed line of the corresponding color gives the simulation's sampled mean performance. Uniformly random jamming performs worst among the three, with the intelligent resource-allocation method performing second best and the perfect jamming strategy performing best. Note that the perfect strategy is not possible to implement in practice because it is noncausal, requiring perfect information about the future, which an implemented agent would not have access to at run time.

We may be interested in more than just the raw jamming capabilities of the agent, however. Therefore, we have compared the learning performance of the intelligent resource agent against a uniform searching strategy in Figure 12. Here, the blue histogram (and its associated mean) depicts the performance of an intelligent agent taking actions to reduce its uncertainty regarding the operating environment as best as possible at each epoch (i.e., time discretization interval), and the orange histogram (and its associated mean) depicts the performance of an agent that scans uniformly at random at each epoch. The intelligent agent learns the adversarial signal's frequency ranges and timing patterns at a faster rate than the agent that processes information acquired by scanning uniformly at random at all times.

Next Steps

At this stage, it is worth noting that there are clear gaps between what has been developed with regard to solving the resource-allocation problem and what would be required of an operational system. It is easiest to note that at least some, if not all, of the simplifying assumptions made in the current problem statement will need to be altered to adhere to real-world systems. Although such work is ongoing and is of technical interest, it is beyond the scope of this article.

The current agent considers only the control of a transceiver. In practice, each platform is likely to be equipped with several distinct sensors and emitters. Although the particular implementation details of the agent will unavoidably change when this generalization

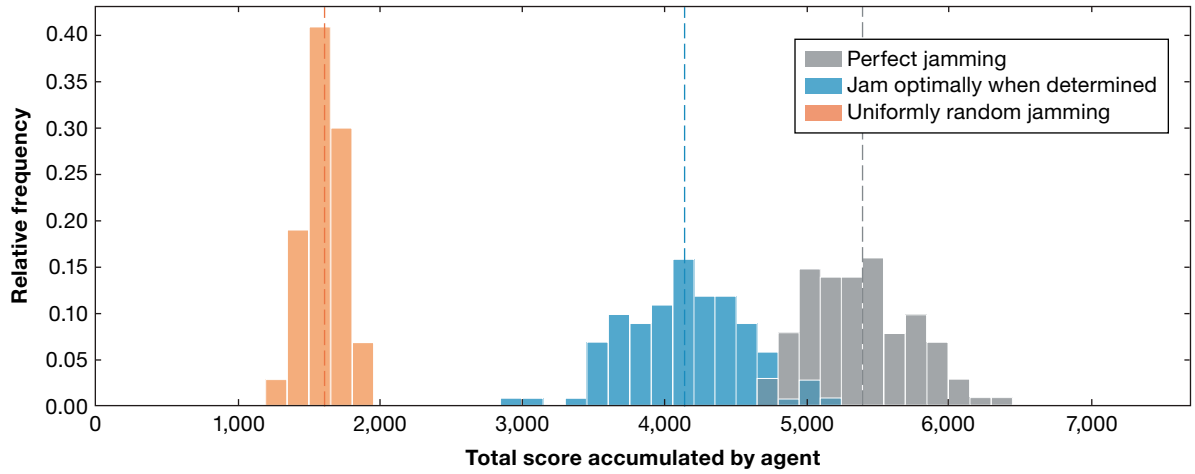

Figure 11. Comparison of jamming performance between resource-management strategies. A strategy that jams adversarial emitters to accrue optimal expected rewards after collecting sufficient emitter pattern data is shown in blue, a uniformly random jamming across channels is shown in orange, and jamming performance assuming perfect knowledge of the emitter patterns is shown in gray. The dashed line of the corresponding color gives the simulation's sampled mean performance.

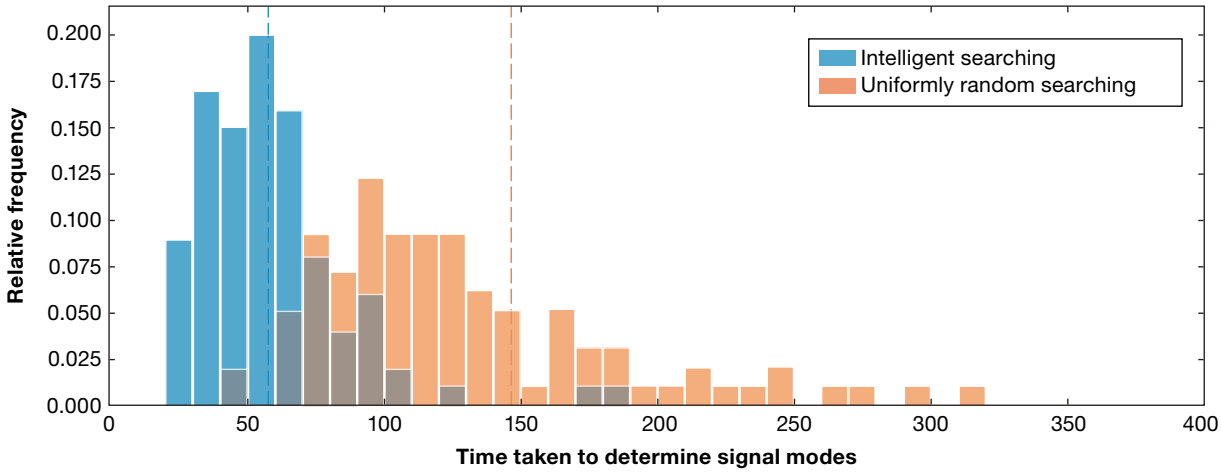

Figure 12. Comparison of learning performance between resource-management strategies. The blue histogram (and its associated mean) depicts the performance of an intelligent agent taking actions to reduce its uncertainty regarding the operating environment as best as possible at each epoch, and the orange histogram (and its associated mean) depicts the performance of an agent that scans uniformly at random at each epoch.

is considered, the basic architecture will remain unchanged. Changes will be limited to the evaluator.

The current solution we have developed for the resource-management problem is an agent designed to take actions based on Bayesian assumptions regarding its adversary's waveforms. As the agent accumulates knowledge of its environment, it can update its belief distributions, leading to better possible jamming actions. The agent is different from traditional reinforcement learning agents because it strongly incorporates subject-matter expert knowledge and makes use of model-based stochastic optimization methods for taking actions. The model-based stochastic optimization methods allow designers to place specific constraints and boundaries on the agent to prevent it from taking actions that are unfruitful from a designer's perspective. Our results indicate that this method shows promise for use in single-agent environments.

We believe that with additional research, this method can be extended to multiagent environments composed of collaborative EW platforms. Of primary concern is the communication and processing delay of distributed platforms. Because waveforms travel at the speed of light, optimizing this method across a battle group of EW platforms remains a challenge. At present, we explicitly consider only the control of a single transceiver, but we hope that the underlying methodology can extend to the multiple transceiver case without requiring a severe algorithmic redesign.

CHALLENGES OF APPLYING ML TO EW

Although the potential benefits of applying ML are compelling, there are many barriers to achieving an envisioned future of ML-enabled autonomous behavior within tactical EW platforms. ML techniques can serve as a powerful tool for developing models capable of making robust, data-driven predictions. However, when developing a model that makes data-driven inferences, it is critical that there first be a source of data representative of the problem at hand. A corpus of training data that sufficiently captures the RF environment of interest must be available for training ML algorithms. Let us consider two types of RF data sources and their roles in constructing RF data sets suitable for enabling ML applications in EW.

RF Collection

An obvious approach to producing an RF data set is to directly record signals of interest from a relevant environment. This can be a challenge depending on the context of the problem being addressed. Software-defined radio has made capturing large volumes of signals more accessible. However, it can still be a challenge to capture the breadth of signal types needed to form a robust data set from a collection. Also, a signal collection will always contain a bias toward the specific environment and the receiver equipment with which it was captured. Real-world data impurities, such as channel fading, multipath, and interference, can prove problematic if a clean signal set is desired. Under other conditions, collected data can yield an advantage in validating models against environmental factors to which they may not be robust. An additional challenge in working with existing captures is that most do not contain sufficient annotations needed for supervised learning efforts. This often means that additional data munging/manual labeling is required before the collection can be used.

RF Simulation

An alternative approach to generating RF data is simulation. A distinct advantage of this approach is that any characterized signal can be produced with parameters varied beyond those that could be feasibly collected. However, generated RF data will be only as realistic as the effects and impairments programmed into the simulation framework. Furthermore, little additional effort is required to capture any desired metadata while simulating. This means that labeled data for training supervised ML algorithms can typically be more easily added to simulated data sets. Conversely, accounting for the proper RF effects can make synthesizing adequately realistic signal data a challenge. The fidelity at which observational effects must be modeled largely depends on the problem to which the data will be applied. A rigorous approach to validation, such as cross validation with labeled RF collections, is also needed to ensure that the synthetic data used are appropriately representative of the true RF signals they are modeled after.

Other Considerations

Once the aforementioned challenges associated with ML training data are overcome, additional challenges should be considered before applying ML to an EW problem. First, many possible tactical host platforms for future integration of ML algorithms may be limited by size, weight, and power. Additional size, weight, and power typically reduces mission time or functions. This means that applying typical ML algorithm-training approaches within an enterprise server facility with an abundance of graphics processing unit resources is not feasible on a tactical platform. Similarly, high-bandwidth data links to connect platforms to these facilities may not be available to support ML applications. Furthermore, approaches for reliable offline training with limited retraining and/or online learning are needed to bring ML to tactical EW platforms. The EW community has a long history of relying on hardware-in-the-loop and range testing and evaluation to characterize the effectiveness of EW techniques. Improved testing and evaluation infrastructure is required to characterize the nondeterministic behavior of ML algorithms and build trust and confidence within the development, sponsor, and operator communities.

CONCLUSION

PSMA IRAD investments into emitter identification and resource management have laid the groundwork for the future of intelligent and autonomous EW platforms. The results of research into AMR and autonomous resource allocation have yielded promising results that showcase the ability for platforms to use data-driven techniques and address agile threats quickly and effectively. Although these foundational efforts demonstrate the feasibility of ML solutions in addressing EW gaps, we recommend numerous follow-on research efforts to mature these solutions.

One such future research vector involves extending the problem context from specific modulations or signal types to specific emitter IDs. This poses several challenges that must first be addressed. The problem of determining an emitter-specific ID will probably drive the need for other contextual information to be processed effectively. Other relevant non-RF factors, such as geographic location, time of day/month/year, or other mission-specific priors, may influence how captured RF emissions are interpreted and thus prioritized.

Although test/development frameworks are capable of mirroring real-world scenarios (such as the environment model used for DARPA's Spectrum Collaboration Challenge[16]), integration with these high-fidelity spectrum environment models will be needed to develop and test the next iteration of ML-enabled RF systems. Several other PSMA efforts on this front seek to accelerate the development and demonstration of future collaborative EW systems. The ongoing Small-scale Broadband, Low-latency Environment (SaBLE) effort focuses on the development of a hardware-in-the-loop RF environment emulation infrastructure. The Collaborative and Adaptive Systems EW Simulation effort proposes an event-driven simulation framework designed with modular platform interaction models to simulate complex engagements. By varying the level of fidelity dynamically, the simulations address temporal resolution challenges encountered when modeling collaborative EW engagements at the signal level.

Ultimately, any fielded, automated solution to EW gaps will act in concert with human operators in a manner that is understandable to the operators and is explainable to other members of the command chain. Engineering an interface that allows for both manual tuning of different signals for prioritization and manual tuning of the balance of effort devoted to observation and jamming is straightforward at a technical level. It can be accomplished by allowing the user to manually input different objective functions into the agent's optimization online. However, it is challenging to create such an interface that an operator can cognitively manage. There are many paths to address this task; however, all will require significant design effort and consultation with relevant experts (e.g., human factors engineers, candidate operators).

Finally, incorporation of automated solutions into EW missions requires some level of mission planning. The Bayesian agency approach to autonomous resource allocation may also extend to mission planning to enable optimal placement and employment of EW platforms and EW techniques. Application of the work described in the Autonomous Resource Allocation section to

mission planning would need to take into account objectives, the environment, and an accurate model of the adversary. Often, the scenarios of interest are based on competing adversarial objectives in contested environments that yield an advantage for the defender. A mission planning application should incorporate the prior information of the operator in the formulation of the hypotheses. If this is done correctly, the hypotheses under consideration by the algorithm will be much more relevant to the mission objective. However, the operator also needs to incorporate uncertainty into the priors, which will result in a nonzero probability across a set of hypotheses that are possible but considered less plausible on the basis of the operator's knowledge.

In this article, we described two active areas of research at APL that are applying ML techniques to EW domain-specific challenges. We further outlined EW gaps and recommended research topics where we anticipate ML will serve a role in future EM operational environments. The successful adoption of these technologies will probably drive changes in how future EW systems are developed, tested, and maintained.

REFERENCES

[1]Defense Advanced Research Projects Agency. "Changing how we win: DARPA technologies that are making a difference today." Mar. 2017. https://www.darpa.mil/attachments/DARPA_Changing-HowWeWin.pdf.

[2]Defense Advanced Research Projects Agency. "Adaptive Radar Countermeasures (ARC)." https://www.darpa.mil/program/adaptive-radar-countermeasures (accessed Aug. 2, 2021).

[3]A. K. Nandi and E. E. Azzouz, "Algorithms for automatic modulation recognition of communication signals," *IEEE Trans. Commun.*, vol. 46, no. 4, pp. 431–436, Apr. 1998, https://doi.org/10.1109/26.664294.

[4]A. Fehske, J. Gaeddert, and J. H. Reed, "A new approach to signal classification using spectral correlation and neural networks," in *1st IEEE Int. Symp. New Frontiers Dyn. Spectr. Access Netw.*, Baltimore, MD, Nov. 2005, pp. 144–150, https://doi.org/10.1109/DYSPAN.2005.1542629.

[5]T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," arXiv, submitted Feb. 12, 2016; last revised Jun. 10, 2016, https://arxiv.org/abs/1602.04105.

[6]F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," arXiv, submitted Mar. 12, 2015; last revised Jun. 17, 2015, https://arxiv.org/abs/1503.03832.

[7]K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv, Dec. 10, 2015, https://arxiv.org/abs/1512.03385.

[8]O. Moindrot. "Triplet loss and online triplet mining in TensorFlow." Oliver Moindrot blog, Mar. 2018.

[9]M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," *ACM SIGMOD Rec.*, vol. 28, no. 2, pp. 49–60, Jun. 1999, https://doi.org/10.1145/304181.304187.

[10]T. J. O'Shea and N. West, "Radio machine learning dataset generation with GNU radio," in *Proc. 6th GNU Radio Conf.*, vol. 1, no. 1, 2016, https://pubs.gnuradio.org/index.php/grcon/article/view/11.

[11]D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA: MIT Press, 2009.

[12]R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT press, 2018.

[13]J. Pearl, *Causality: Models, Reasoning and Inference*. Cambridge, UK: Cambridge University Press, 2009.

[14]E. T. Jaynes, *Probability Theory: The Logic of Science*, G. L. Bretthorst, Ed. Cambridge, UK: Cambridge University Press, 2013.

[15]F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge, UK: Cambridge University Press, 2017.

[16]D. M. Coleman, Ed. "The DARPA SC2 Colosseum Test Bed," *Johns Hopkins APL Tech. Dig.*, vol. 35, no. 1, 2019, pp. 1–78, https://www.jhuapl.edu/TechDigest/Detail?Journal=J&VolumeID=35&IssueID=1.

APPENDIX. AN EXAMPLE BELIEF-DRIVEN EMITTER MODEL—LEARNING AND PREDICTING

Following are details on the process of learning a belief-driven model of an adversary emitter's transmission schedule in frequency and time. These details supplement the Autonomous Resource Allocation section. The example that follows is notional—the calculations are not an exact reflection of those in the discussed IL'EA Maestro implementation, nor is the set of emitters. Rather, this example is pedagogical, intended to give the reader a concise sense of the complexity of implementing an appropriate Bayesian model for the resource-allocation problem.

Problem Specification

Because our focus is on pedagogy, we consider a minimum working example instead of a realistic one. The assumptions stated in the article continue to apply here as well. Consider an environment with two emitters. Initially, we know neither the emitters' timing patterns (i.e., periodicity, pulse length, interpulse length, or rising edge time) nor the emitters' range of possible operating frequencies. Each will be learned for each signal by way of observing emissions, storing the data obtained from the emissions, and processing the data appropriately. The remainder of this discussion describes a Bayesian approach to achieving the following goals: timing pattern prediction and emitter operating frequency prediction.

Updating Beliefs of Timing Patterns

In this section, we discuss the data storage and processing techniques required to identify the emitters' timing patterns. Although the data processing step is ultimately just an application of Bayes' theorem, $P(A|B) = P(B|A)(P(A)/P(B))$, the complexity of the application demands careful consideration. Notably, there is a need to propagate observed data forward in time perpetually in order to correctly identify each signal's characteristics. That is, the belief update process is not Markovian. What we have seen in the past (even the arbitrarily distant past) has an effect on our beliefs about an emitter's signal going forward. Because we may well need to run this process for an arbitrarily long time to perform a mission and we can neither store nor process online an arbitrarily large amount of data, we must develop a means of storing the relevant pieces of information in the infinite stream of observer data in a useful manner.

In the implementation referenced in this article, we develop this means by using a priori knowledge of a value (T) to provide the upper bound on the signal's periodicity and then constructing an exhaustive set of all feasible timing patterns for a given emitter. The size of this table grows cubically, that is, $O(T^3)$, with the magnitude of the given upper bound on the signal's periodicity. Although this process is of polynomial complexity and so can be thought of as tractable colloquially, note that this level of complexity may not be appropriate for a computation that needs to be completed in real time for large T practically. For example, even in the modest case where $T = 10$, there are 330 possible hypotheses. For the case in which $T = 100$, that number grows to 333,300. For $T = 1{,}000$, the number is (approximately) 333 million. This fact limits the utility of the implementation discussed here to handling signals with a periodicity of 1,000 quantization intervals or less. A quantization time of 1 ms implies that the pulses could be at most 1 s. Although outside the scope of our discussion here, promising research addressing this complexity involves storing data in a novel data structure and processing it in a more sophisticated way.

For the present discussion, let us consider the case in which $T = 3$, which induces a set of eight possible timing hypotheses. For each possible period length p, we may write the collection of possible timing patterns as a set of finite fields of length p, where each finite field is written as a list of binary values such that the entry is 1 if the signal is active at times t modulo p and 0 otherwise. Written thusly, we have for $p = 1$ the set {(1)}; for $p = 2$ the set {(01) (10)}; for $p = 3$ the set {(001), (010), (100), (011), (110), (101)}; and for $p = 4$ the set {(0001), (0010), (0100), (1000), (0011), (0110), (1100), (1001), (0111), (1110), (1101), (1011)}. Note that each set is the collection of p-digit binary numbers, with all 1s occurring in a contiguous block, when considering the fact that the topology of a finite field loops ends around to "connect" to each other. That is, all activity must either be a rising or falling edge or be contained between two such edges. Note also that constant signals may be mapped without loss of generality to a $p = 1$ signal and do-nothing signals need not be considered. Thus, for $p > 1$, we do not explicitly consider fields that are either all 1s or all 0s.

If we allow each signal to have $T = 3$, each signal generates its own set of eight possible timing patterns. It is our task to use data to eliminate timing patterns that are not supported by observed data in a computationally efficient manner. To enable this, we must make some notes regarding how observations inform the timing decisions. There are a few important cases to highlight.

1. If we take an observation and see a particular signal, that signal must be active at that particular time, modulo some periodicity.
2. If we take an observation and do not see any signal, that does not mean that no signal is active, as the signal may be active in a different frequency channel.
3. If we take an observation and see a signal in a time bin such that the signal appears between two previously seen emissions for a fixed periodicity p, then the signal must have been active for all times between the previous time modulo periodicity observation and the current time modulo periodicity, provided that the signal is actually a period p signal.

The first two observations are straightforward. If a signal is seen, it is active. If it is not seen, we do not obtain any new information regarding the signal's activity (absent any further side information that may be present—i.e., known time correlations between distinct signals). The third observation is somewhat obtuse but improves the data efficiency of the model-building process drastically when implemented.

To develop an understanding of what is happening in this case, suppose for a signal with $T = 3$ that the first observation is made at time $t = 5$ (Figure A-1). The time modulo period value for $p = 1$ is 1 (as it always must be), and so (1) remains a viable hypothesis. The time modulo period value for $p = 2$ is 1 (count: 1, 2 | 1, 2 | 1), and so if the signal is of periodicity 2, it must have the timing pattern (10). This is because we have seen the period active at time modulo period 1, and if it was also active at time modulo period 2, then it would always be active, and so by default must be a periodicity 1 signal. The time modulo period value for $p = 3$ is 2 (count: 1, 2, 3 | 1, 2), and so all $p = 3$ hypotheses of the form (x1x) are still possible. The time modulo period value for $p = 4$ is 1 (count: 1, 2, 3, 4 | 1), thus allowing all hypothesis of the form (1xxx) to be possible.

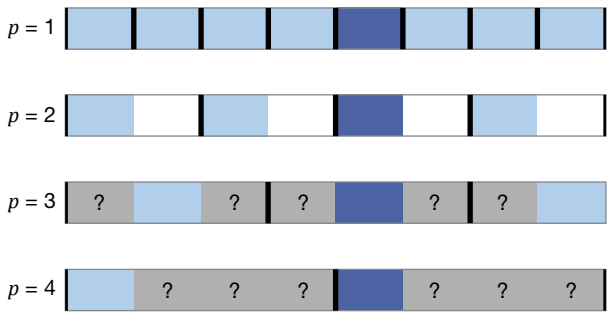

Figure A-1. Example illustration of determining signal periodicity. This example is based on observation at time interval 5.

Now, suppose we make an observation at time $t = 7$ (Figure A-2). The time modulo period value is 1 for $p = 2$. If the signal has periodicity 2, nothing changes: the timing pattern must be (10). The time modulo period value is 1 for $p = 3$. This then implies that the timing pattern must be (110) if the signal is of periodicity 3. The time modulo period value is 3 for $p = 4$. Using only the first two observations from above, this would then imply that all hypotheses of the form (1x1x) are still viable. However, using the third fact implies that the timing pattern must indeed be (1110) if the signal is of periodicity 4. Incorporating this subtle detail helped us learn faster, to the point where we now know what the timing pattern must be for each allowable periodicity, were the assumed periodicity value is actually true. Further observations would eventually allow us to eliminate certain periodicity values because we know that if a signal is active when it should not be, that periodicity is not possible. Thus, under the assumptions outlined in this article, we can fully determine the signal's timing characteristics from partial, intermittently observed information.

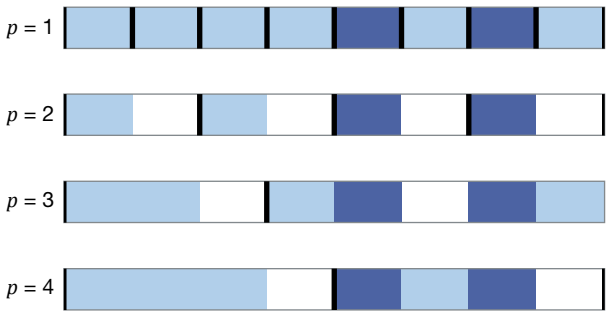

Figure A-2. Example illustration of determining signal periodicity. This example is based on observation at time interval 7.

Because this discussion is intended to clarify core concepts and not provide exhaustive detail, we will not explain here how the presence of multiple signals affects the learning of timing patterns. Suffice it to say that known (or, rather, assumed) correlations between the signals' timing behavior can help us learn timing information even faster.

Updating Beliefs of Channel Occupancy

In this section, we discuss the learning of the channel occupancy models used internally in the IL'EA Maestro prototype detailed in this article. As we did in the section discussing learning timing patterns, we will focus here on pedagogy instead of realism. Because using known (or, rather, assumed) frequency spectrum correlations between emitted signals is an integral piece of the model, we discuss an example with two distinct signals. It is our task to determine where in the frequency spectrum each will be in the near future given our knowledge of what we have observed in the past and our assumptions about co-occurrence.

With each signal, we start an assumption that all detectable emissions will occur between a priori known lower and upper frequency bounds. We quantize the spectrum into discrete frequency bins that we will describe as channels. For the purposes of demonstrating how knowledge of frequency spectrum correlation could potentially be used in learning an appropriate emitter model, we assume that the two signals cannot occur on the same channel at the same time (for example, because of interference).

Before any observations of either signal are made, we initialize each signal's distribution of possible operating frequencies as a uniform random variable. As observations are made, we keep a record of a few quantities for each tracked signal so as to design updated beliefs of channel occupancy (which are more informative than the uniform prior). Namely, we track the lowest and highest operating channel observed for a given signal and the time since the last observation of the signal on each channel observed.

Intuitively, we would like the distribution to be concentrated primarily between the observed lower and upper operational bounds. Additionally, we would like channels where the signal has appeared before to have more "weight" in the model than those that have not been confirmed to be in use, and we would like this discrepancy of weight to be itself weighted by how old the observation is at the time the belief is to be evaluated. If there were no correlations between signals, we could then assign probabilities for each signal without consideration of the other, with the assumed density taking on only the information observed for each signal individually. Let us walk through this case for a signal with 10 plausible frequency bands.

If we have seen the signal as low as channel 3 and as high as channel 9, we would want to assign most of the probability mass to channels in the set {3, 4, 5, 6, 7, 8, 9} and some small probability (possibly zero) to channels in the set {1, 2, 10}. Thus, we may introduce some scale factor w between 0 and 1, such that $w >> 0$ and the probability assigned to $P\{[3, 9]\} = w$ and $P\{1, 2, 10\} = (1 - w)$. This form takes care of the first intuition from the preceding paragraph. To take care of the time-value intuition, we must introduce some other term such that a new observation almost certainly determines the future location of

the signal but older information is not forgotten. There are many ways to accomplish this, but for simplicity's sake, consider the following form.

Assign each frequency channel a buffer to accumulate a weight value. Whenever a certain amount of time has passed (e.g., the estimated period of the signal under consideration), multiply the current channel buffer weight by some scalar less than 1 and add to each buffer a large value if the signal has been seen on the channel since the last weight update and zero otherwise. We can consider each buffer value to be a relative likelihood of occurrence, albeit one that is not appropriately scaled to be a probability. However, when we update our channel occupancy beliefs, we can rescale appropriately. Using the intuitions from above, we can sum together the buffer weights for the channels in {[3, 9]} and use this sum as a scale factor to obtain a conditional probability: if the signal is in {[3, 9]}, then its probability of being in a particular channel c in {[3, 9]} is assumed to be its buffer weight value divided by the summed scale factor. We can do likewise if the signal is in {1, 2, 10}. If we then consider the scale factor w from above to be an assumed conditional probability of occurrence, the net belief we obtain is $P(\text{channel}) = w(Bc/B1)$ for channels in {[3,9]} and $P(\text{channel}) = (1 - w)(Bc/B2)$ otherwise, where Bc is the buffer weight for channel c, $B1$ is the scale factor for the first case (where the signal is assumed to be between the minimum and maximum previously seen values), and $B2$ is the scale factor for the second case (where the signal is assumed to be outside the interval of prior observations).

Now we consider incorporating the assumption that signals cannot co-occupy a frequency channel. Note that we can accommodate finer-grained temporal correlation models, which, for example, include information about how a signal moves through different frequency channels over time. Considering the process described above for the single signal case, we can see that the primary means for accommodating time correlation between signals is in manipulating the buffer weights appropriately. Intuitively, observations of a signal other than the one being predicted on a particular channel should decrease the likelihood that the considered signal is indeed there, as we are assuming a negative correlation between the occurrences of two distinct signals on a single channel.

To accomplish this effect in the prediction model outlined, we multiply each buffer weight by a scaling factor that takes the value 1 if the signal under consideration is the most recent signal seen on a particular channel and the value $(1 - e^{-\alpha\tau_c})$ otherwise, where α is some positive scaling constant and τ_c is the time since the most recent signal has been observed. As τ_c grows larger, the influence observing some signal on the channel decreases, eventually to zero. That is, the multiplication factor converges to one in the limit of large τ_c and thus has no effect after much time has passed. However, if a different signal has been observed recently, it can significantly reduce the model's belief that the signal under consideration is present in the current channel.

Kyle A. Casterline, Force Projection Sector, Johns Hopkins University Applied Physics Laboratory, Laurel, MD

Kyle A. Casterline is a researcher and section supervisor in the Intelligent Combat Platforms Group of APL's Force Projection Sector. He has a BS in electrical engineering from Pennsylvania State University and an MS in computer science from Johns Hopkins University. Kyle has worked on a variety of research projects within APL's electronic warfare program area throughout his career. He has experience leading artificial intelligence development teams, with a focus on radio frequency and signal processing. His research interests are in machine learning and artificial intelligence applied to the domains of autonomous sensing and digital signal analysis. His email address is kyle.casterline@jhuapl.edu.

Nicholas J. Watkins, Force Projection Sector, Johns Hopkins University Applied Physics Laboratory, Laurel, MD

Nicholas J. Watkins is a senior professional staff member in the Intelligent Combat Platforms Group of APL's Force Projection Sector. He has a bachelor's degree in electrical engineering from Wilkes University, where he graduated *summa cum laude*, and he completed his doctorate in George J. Pappas's group at the University of Pennsylvania. His research interests lie at the intersection of stochastic processes, control, and optimization. His work at APL focuses on the development and analysis of autonomous systems, with applications in electronic warfare, vehicle guidance, and cybersecurity. His email address is nick.watkins@jhuapl.edu.

Jon R. Ward, Force Projection Sector, Johns Hopkins University Applied Physics Laboratory, Laurel, MD

Jon R. Ward is a project manager in APL's Force Projection Sector. He has a BS in electrical engineering from North Carolina State University, an MS in electrical engineering from North Carolina State University, and a PhD in electrical engineering from the University of Maryland, Baltimore County. Jon is the scientific advisor for APL's electronic warfare program area, advising on electronic warfare techniques, tactical communications, and general science and technology needs in support of sponsor engagement and internal investment. He has experience leading teams in communication system development and analysis, modeling and simulation, and lab and field test and evaluation of wireless communication systems. Jon is an active member of IEEE and current president of the Chesapeake Bay Chapter of the Association of Old Crows (AOC). His email address is jon.ward@jhuapl.edu.

William Li, Force Projection Sector, Johns Hopkins University Applied Physics Laboratory, Laurel, MD

William Li is a former researcher and data scientist in the Intelligent Combat Platforms Group of APL's Force Projection Sector. He earned a BS and an MS in electrical and computer engineering from the University of Louisville's Speed School of Engineering. William has extensive experience with deep learning and reinforcement learning techniques. He was previously named the winner of the APL's Reconnaissance Chess Competition. Additionally, William has made significant contributions to the study and development of reinforcement learning–based agents for air-to-air combat and electronic warfare.

Matthew J. Thommana, Force Projection Sector, Johns Hopkins University Applied Physics Laboratory, Laurel, MD

Matthew J. Thommana is a researcher in APL's Force Projection Sector. He has a BS in electrical engineering and mathematics from the University of Iowa and is pursuing an MS in computer science from Georgia Tech. He has strong foundations in computer science and communication systems. After a stint in the commercial world, Matthew has worked on SDB II testing and integration, cockpit displays and penetration testing, and radio frequency automation and testing. He currently works on machine learning tasks applied to radars. He has a patent on distributed spectrum harvesting. His research interests lie at the intersection of distributed systems and decision-making and machine learning. His email address is matthew.thommana@jhuapl.edu.