Optimizing Feature Selection for Semi-Supervised Machine Learning Classifiers

Anastacia MacAllister, Jordan Belknap, Danielle Clement, Megan McConnell, Stephen Summers

Lockheed Martin Corporation

Fort Worth, TX

anastacia.m.macallister@lmco.com, jordan.l.belknap@lmco.com, danielle.m.clement@lmco.com, megan.a.mcconnell@lmco.com, stephen.t.summers@lmco.com

ABSTRACT

In 2018 the United States Department of Defense (DoD) released their Artificial Intelligence (AI) strategy summary. The report highlights several key AI technologies the nation needs to maintain a competitive edge. The DoD asserts that one such area, AI based predictive maintenance, is integral to ensuring equipment like aircraft and armored vehicles stay mission ready. Unfortunately, the DoD maintains varied fleets of equipment, often at lower quantities than commercial industry. These small quantities of equipment at varying operating conditions makes collecting representative data sets, often required for AI, challenging. Another factor complicating the creation of AI for military applications, is the lack of insight into which variables best capture complex processes. This can make it challenging to determine which variables are important factors to include in an AI model. While subject matter experts (SMEs) often provide insight into data, they may not identify the optimal combination of features. There could be bias in the SMEs recommendation, or for security reasons they may not know the true nature of the variables. As a result, another method of selecting optimal features for AI models is needed. While existing literature contains ample work on feature selection, only limited work exists dealing with small data sets. This paper describes work using Binary Particle Swarm Optimization (B-PSO) to optimize the accuracy of a Self-Organizing Maps (SOMs) based AI model for predictive maintenance trained using a small real-world data set. Testing results show that using B-PSO to select training features produces a classifier with up to 95% accuracy, 98% precision, and 72% recall. This new method increased some AI model accuracy metrics by 15% over the original baseline.

ABOUT THE AUTHORS

Anastacia MacAllister, Ph.D., is a Machine Learning Researcher at Lockheed Martin's Skunk Works®. Her research focuses on developing machine learning algorithms using sparse or imbalanced data sets. Dr. MacAllister is currently working on developing machine learning methods for prognostic health management. She is also developing machine learning tools to help reduce pilot's cognitive load. Dr. MacAllister received her Ph.D. from Iowa State University of Science and Technology in Mechanical Engineering and Human-Computer Interaction.

Jordan Belknap is an Artificial Intelligence Researcher at Lockheed Martin's Skunk Works. She is currently the Principal Investigator of AI Based Prognostic Health Management. She specializes in developing AI solutions for highly complex systems with noisy, partially labeled data. She received her Masters' from Georgia Institute of Technology in Computer Science with a focus in Interactive Intelligence.

Danielle Clement, Ph.D., is a Technical Fellow at Lockheed Martin Skunk Works. She is currently researching verification and validation approaches for autonomous systems. Dr. Clement received her Ph.D. from the University of Texas at Arlington in Computer Science Engineering

Megan McConnell is responsible for the technical strategy and roadmaps for the Development Innovation Team. With over 15 years of aerospace experience, she has held positions of increasing responsibility throughout the corporation. Ms. McConnell has served as a Country Chief Engineer, Landing Gear IPT Lead, worked on the Hydraulics and Actuation IPT, Affordability, and functioned in the Simulation and Systems Integration Labs. She specializes in closing

 ©2020 Lockheed Martin Corporation. All Rights Reserved.

open ended projects and integrating management with engineering. Ms. McConnell holds a Bachelor of Science in Mechanical Engineering from Oklahoma State University and a Master's in Business Administration from Texas Christian University.

Stephen Summers is a Systems/Software Engineer at Lockheed Martin Aeronautics Company on the Power and Thermal Management System (PTMS). He works with production and field support for system hardware failures looking for trends and undetected failures. Stephen received a B.S. degree in Computer Engineering from Texas A&M University and a M.S. degree in Electrical Engineering from University of Texas at Arlington.

©2020 Lockheed Martin Corporation. All Rights Reserved.

Optimizing Feature Selection for Semi-Supervised Machine Learning Classifiers

Anastacia MacAllister, Jordan Belknap, Danielle Clement, Megan McConnell, Stephen Summers

Lockheed Martin Corporation

Fort Worth, TX

anastacia.m.macallister@lmco.com, jordan.l.belknap@lmco.com, danielle.m.clement@lmco.com, megan.a.mcconnell@lmco.com, stephen.t.summers@lmco.com

INTRODUCTION

New technologies and capabilities are being rapidly integrated into battlefield technology, providing warfighters with the tools necessary to address 21st century challenges and threats. While this increase in capability can provide immense benefit to the warfighter, it also adds complexity (Drezner, 2009; Grammich, Arena, Younossi, Brancato, & Blickstein, 2008). This complexity can introduce numerous engineering and human factors challenges that need to be addressed to ensure equipment performs as intended (Iriarte, 2018; Hawley & Swehla, 2018). Specifically in the engineering field, new technology often requires increased maintenance monitoring and highly skilled maintainers. Previously, traditional one-size-fits-all maintenance monitoring and scheduling was driven by engineering models of degradation and skilled operator observations. This conservative strategy was moderately effective when little information was recorded about aircraft operations. However, this strategy occasionally led to unnecessary replacement of equipment and to extensive downtime when an unexpected failure occurred. To combat these issues, manufacturers are beginning to test and implement increasingly affordable commodity sensors to add health reporting capabilities to new equipment. The sensors capture performance and operation data that can then be leveraged to predict when a piece of equipment will perform sub-optimally or predict when a part failure may occur. Ultimately, the goal is to be prognostic rather than reactive when conducting maintenance. However, the amount of sensor data being generated is outpacing a human's ability to analyze it. In addition, even with smaller amounts of data, many times humans are not able to identify higher order interactions between recorded variables. As a result, events might go unnoticed and propagate into larger issues, meaning the sensors are not being fully utilized.

To combat this flood of data, organizations are turning to machine learning (ML) and artificial intelligence (AI) (Bean, 2018; Anshuk Gandh, Carmen Magar, 2013; How real-time data is transforming five industries, 2019; Libert & Beck, 2018). Even the United States Department of Defense (DoD) is beginning to recognize the impact ML and AI can make. In 2018 the DoD released their Artificial Intelligence (AI) strategy summary (United States Department of Defense, 2018). The report highlights several key AI technologies the nation needs to maintain a competitive edge. The DoD asserts that one such area, AI based predictive maintenance, is integral to ensuring equipment like aircraft and armored vehicles stay mission ready. As a result, the DoD plans to spend two billion dollars over the next five years to develop and deploy ML and AI methods (Fryer-Biggs, 2018).

While technology companies like Google, Amazon, and Microsoft have had great success applying machine learning to their domain (Columbus, 2019; Kobielus, 2019; Spencer, 2019; Wiggers, 2019), the military domain provides unique challenges when developing machine learning systems for prognostics. Specifically, tech companies often deal with consumer data that is large scale and simple to collect. They can collect data sets on consumer behavior that have millions or billions of data points, all of which are labeled with the behavior or event being recorded. This large-scale dataset with corresponding labels is the gold standard for machine learning. Unfortunately, the DoD maintains varied fleets of equipment, often at lower quantities than commercial industry. These small quantities of equipment at varying operating conditions makes collecting representative data sets, often required for AI, challenging. Another factor, complicating the creation of AI for military applications, is the lack of insight into which variables best capture complex processes. This can make it challenging to determine which variables are important factors to include in an AI model.

While challenging, developing ML for these types of applications is not impossible. Often, however, these types of data sets require careful selection of ML algorithms and variables used to create these models to ensure accurate results.

 ©2020 Lockheed Martin Corporation. All Rights Reserved.

This paper presents a novel method that uses optimization to select a combination of variables to create ML classifiers from small data sets with many parameters. The method uses Binary Particle Swarm Optimization (B-PSO) to select a combination of variables for training Self-Organizing Map (SOM) based semi-supervised classifiers. This classifier is then used to detect anomalies in an aircraft subsystem. The method presented has a wide variety of applications for the creation and optimization of ML classifiers where little might be known about the system. The paper below first presents a brief background on SOMs and feature selection problems. It then describes the novel algorithm and presents method testing results.

BACKGROUND

When creating a machine learning algorithm, properties of the data itself can drive which algorithm is best suited to the task. Machine learning approaches can normally be categorized as supervised, unsupervised or semi-supervised. Supervised approaches use input and output pairs to generate a model which maps inputs to outputs, while unsupervised techniques group items by similarity and do not require knowledge of outputs. To generate a model with supervised learning techniques, a large and relatively balanced data set is often required. For example, a supervised learning technique looking to discriminate between normal and abnormal samples would require many labelled examples of both normal and abnormal readings. Since unsupervised techniques do not require these labels, they are frequently employed to gain better understanding of unfamiliar and unbalanced data sets (Russell & Norvig, 2016). Semi-supervised learning balances the two approaches. In data sets best suited for semi-supervised learning techniques, some of the input data does have supporting labels, or output mappings. Semi-supervised techniques use this label information to either (a) apply the given labels to the unlabeled input data as in supervised approaches or (b) serve as limits or constraints on an unsupervised learning process (Chapelle, Schölkopf, & Zien, 2006). The dataset discussed in this work is comprised of sensor readings that are predominantly assumed to be non-anomalous data, with limited examples of labeled anomalies. This limited amount of labeled data and highly unbalanced nature of the data set suggests a semi-supervised approach such as Self-Organizing Maps, discussed in the next section.

Self-Organizing Maps (SOMs)

Understanding key properties of the dataset can help drive machine learning model selection. Due to the properties of the Power and Thermal Management System (PTMS) data set, a semi-supervised classification approach was derived from unsupervised SOMs. Self-organizing maps, introduced by Kohonen in the early 1980s (Kohonen, 2012), are an unsupervised learning technique. SOMs cluster similar data together such that data items that are close together in the input space are also close together in the output space. They are used extensively in anomaly detection because of their strengths in clustering, feature identification, data exploration, and visualization. When SOMS are used to cluster data, the resulting clusters can be labeled with known information, supporting semi-supervised techniques. SOMs approximate a multi-dimensional data set into a two-dimensional grid. SOMs are comprised of a set of nodes, also called neurons, that are defined by a typically two-dimensional index and a weight vector, which has the same number of elements as an input data item.

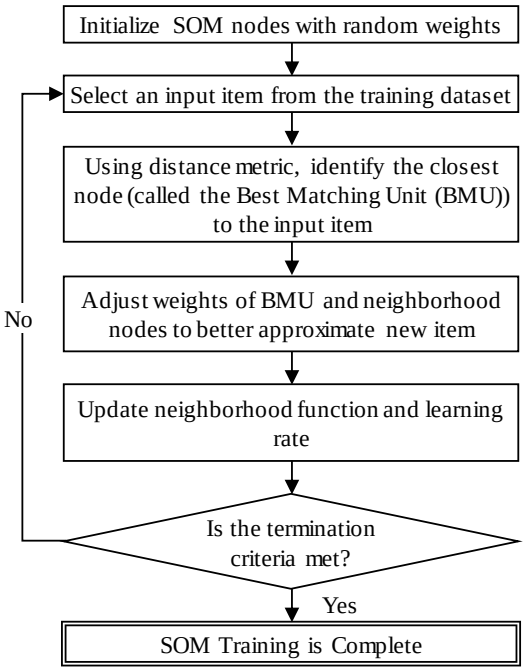

Figure 1. SOM Training Process

Figure 1 illustrates the training process for SOMs. SOMs are typically initialized with random weights. As the SOM is trained, the weights of each node are adjusted to approximate the provided input information. For each input item, the SOM node whose weight is the closest to the input item is identified. This node is called the Best Matching Unit (BMU). Once the BMU is found, the weights of the BMU and the nodes in the neighborhood of the BMU are updated to better approximate the new data. After the weights of the nodes are updated, the neighborhood function and the learning rate, which are

© 2020 Lockheed Martin Corporation. All Rights Reserved.

parameters that guide the learning process, are updated. This training process is repeated over the set of inputs until some termination criteria is met (typically number of inputs seen, minimization of error, or convergence properties).

More formally, SOMs can be defined as a mapping from an input vector $\vec{x} = [x_1, x_2, \dots, x_m]$ to a node i with a weight $\overrightarrow{w_i} = [w_1, w_2, \dots, w_m]$. During training, weights are updated per equation (1), where $\overrightarrow{w_i}(t+1)$ is the updated weight for node i after processing input $\vec{x}$ in training timestep t. We define $d(\vec{x}, \vec{y})$ to be the distance between vector $\vec{x}$ and vector $\vec{y}$. Euclidean distance is frequently employed as the distance function.

$$\overrightarrow{\boldsymbol{w_i}}(\boldsymbol{t}+\mathbf{1}) = \overrightarrow{\boldsymbol{w_i}}(\boldsymbol{t}) + \boldsymbol{\Theta}(\boldsymbol{t}) * \boldsymbol{\sigma}(\boldsymbol{t}) * \boldsymbol{d}(\vec{\boldsymbol{x}}, \overrightarrow{\boldsymbol{w_i}}) \qquad (1)$$

As reflected in equation (1), the update is based on a gaussian neighborhood decay rate $\Theta(t)$, defined in equation (2), as well as a learning rate $\sigma(t)$, defined in equation (3). In early stages of training, the neighborhood around the BMU might be very large, but as the map grows more mature, the neighborhood shrinks. This means that early updates to the map have a more global impact, while later updates act on a more local level. The learning rate defines the impact of the changes. In this way, the neighborhood defines the scope of the changes based on a given input and the learning rate defines the scale of those changes. Often, the optimal values and bounds of these parameters, such as neighborhood decay rate and learning rate, are not known beforehand. These parameters, frequently referred to as hyperparameters, are usually computed through experimentation and testing.

$$\boldsymbol{\Theta}(\boldsymbol{t}) = \boldsymbol{e}^{-\frac{\boldsymbol{d}(\vec{\boldsymbol{x}}, \overrightarrow{\boldsymbol{w_i}})^2}{2\boldsymbol{\sigma}(\boldsymbol{t})^2}} \qquad (2)$$

$$\boldsymbol{\sigma}(\boldsymbol{t}) = \boldsymbol{\sigma_0} \boldsymbol{e}^{-\frac{\boldsymbol{t}}{\boldsymbol{\lambda}}} \qquad (3)$$

Feature Selection

When applying machine learning techniques, selecting which aspects of your data are the best to use to train your algorithm is critical. Each data point typically contains multiple variables, or features (for example, a position data point will have latitude, longitude and altitude features). While it may seem that using all the available features for machine learning would be the best choice, that is typically not the case. Features can be redundant or related, which leads to their influence on the outcome being over-emphasized. In addition, the more features that are used to train a model, the more overall data points are needed for training, and the less likely the model is to generalize well to new cases. While relying on human expertise can be an excellent early approach to feature selection, automating the feature selection process removes human biases and allows for the process to be applied to environments where subject matter experts may not be readily available.

Two types of dimensionality reduction approaches are typically applied to address data complexity issues: feature extraction or feature selection (Li, et al., 2017). Feature selection chooses a subset of features to consider from the larger set of features, where feature extraction uses transformation techniques such as Principal Component Analysis (PCA) to map the existing high-dimensional data sets into lower dimensions (Alelyani, Tang, & Liu). Feature selection can be preferred when it is desirable for features to relate to real world data items. There is also a cost benefit to feature selection, as reducing the number of data points required results in a reduction of the storage space needed to maintain datasets. Unsupervised learning techniques, such as the clustering approach used in SOMs, can develop vastly different clusters based on the selected features, so identifying valuable features is also critical for model performance.

Feature selection is typically categorized as the mechanism by which the features are selected. Categories of selection algorithms include filter methods, wrapper methods, and embedded methods (Chandrashekar & Sahin, 2014). Filter methods identify promising features based on statistical or information theoretic properties of the data. Wrapper methods use the algorithm itself as a mechanism to select promising features – the best features are the ones that result in the highest performance of the algorithm. Embedded methods are those where the algorithm itself includes a feature selection component.

Dimensionality reduction is valuable as a preprocessing step for many machine learning modelling applications, so feature selection appears in much of the machine learning literature. As examples, Prudhvu, Sai, et al. applied genetic algorithms as a wrapper technique to select relevant features for Radial Basis Functions across multiple data sets (Raj

 © 2020 Lockheed Martin Corporation. All Rights Reserved.

& Kumar), and Ijaz and Choi applied PCA as a feature extraction method before using SOMs to cluster electromyographic signals (Ijaz & Choi, 2018). While our previous work relied on human subject matter expertise (SME) to identify the most relevant features when using SOMs for PTMS anomaly detection, extending the approach to incorporate algorithmic selection of features has promise to remove human biases and extend the approach to systems where SME input is unavailable, while maintaining or improving model performance.

METHODS

There are multiple challenges which must be addressed when constructing and deploying ML algorithms. The first is data. An ML classifier's accuracy is highly dependent on the quality of the training data. Therefore, clean quality data must exist to generate a robust ML model. Once the data is collected, it then must be formatted properly to be ingested by the algorithm for training. This ingestion algorithm must also be thoughtfully selected to match the characteristics of the available data. Only after the completion of these collection, pre-processing, and model selection steps can an ML model be trained and tested. This section describes these steps in detail then discusses the training and testing methodology employed.

Data Collection and Pre-Processing

To build a quality data set for ML model creation, flight test data was pulled off the aircraft post flight. From here the subsystem subject matter expert (SME) then converted data to readable format, filtered out irrelevant features, and time synced the data. Features of the final data set included temperatures, pressures, derivatives, and set points determined by deterministic models. This preprocessing was an important step, since it ensured that the ML model could have access to clean data that accurately represented the subsystem behavior the SME was trying to classify.

The data set consisted of 120,000 flight hours, shared between 342 tail numbers. Amongst those 342 tail numbers, 2 aircraft each exhibited a unique identified anomaly. Both anomalies specifically dealt with air pressure leakage during two different subsystem configurations. The remaining data set was unlabeled and contained several types of data including: aircraft performing at baseline, performing with minor deviations from baseline, and potentially unidentified anomalies. These characteristics meant the data set had a significant amount of noise. There were 3,594 files, 543 of which contained known anomalous behavior. Each data file had a frequency of 1 Hz, meaning there was a data entry for every second of the flight run, and for every data entry there were 197 variables. With a frequency of 1 Hz, 120,000 hours of flight time, and 197 variables, this resulted in 1.4 billion data points. This amount of data is impossible for a human to examine manually and training an ML algorithm on a fraction of this data set far exceeds the processing power of any standard machine.

Feature Vector Creation

Due to the large number of variables in the data set, a down selection was required. Down selecting ensures that an ML classifier is only being trained on the most relevant variables in the data set, reducing the chances of the model picking up on unwanted trends. The variables used to build an ML classifier are often referred to as the feature vector. The feature vector is a list of features, or variables, whose values are considered by the ML algorithm for classification. As shown in Figure 2, the feature vector is fed into the ML algorithm, and the algorithm generates a model based on the information provided by the feature vector. This means that the robustness and accuracy of the model depends on the selection of features.

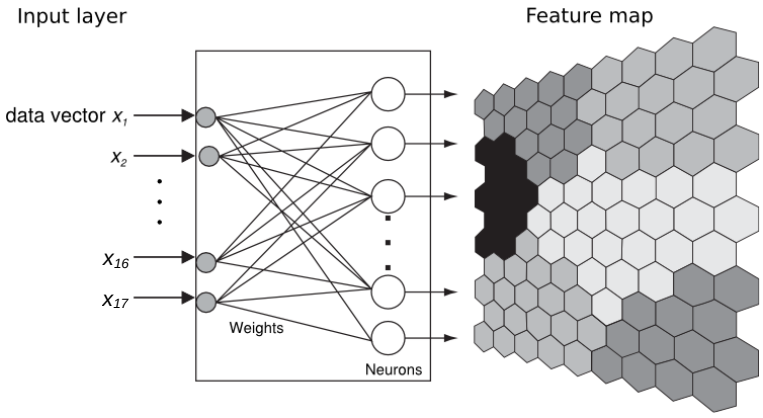

Figure 2: SOM Feature Vector

Reprinted from the AI for 5G: Research Directions and Paradigms, by the Science China Information Sciences, 2018, Retrieved from https://arxiv.org/pdf/1807.08671.pdf

© 2020 Lockheed Martin Corporation. All Rights Reserved.

Multiple methods for downsizing a feature vector exist. At a fundamental level, these methods are performing dimensionality reduction- a technique used to identify features of high value and discard features of low value. Two of the most common dimensionality reduction algorithms are Principal Component Analysis (PCA) and Independent Component Analysis (ICA). However, while both algorithms can produce viable results, the resulting features are not human readable (Lever, 2017; Hyvarinen, 1999). For this application it was important to the SME that they be able to interpret features used by the ML model. To address the human readability problem, the authors initially chose to have the dimensionality reduction driven by engineering design and subsystem expertise (Macallister, Belknap, Clement, Summers, & Hellstern, 2019). However, this methodology relies heavily upon the availability of a subsystem expert and removes the possibility of identifying valuable features the expert may not have considered. As an alternative, the authors chose a method called binary particle swarm optimization, or B-PSO. B-PSO is a technique used to find an optimal combination of variables for capturing specific behavior. No derivative features are produced, and is more efficient than grid or random search (Taijia Xiao, Dong Ren, Shuanghui Lei, 2015). It is straight forward to implement and is tolerant of open-ended problems. One downside is that a global optimum is not guaranteed, and can get stuck in a local optimum, especially in high dimensional problems (Jamian, Abdullah, Mokhlis, Mustafa, & Bakar, 2014). B-PSO is interconnected with model selection, and the training process will be discussed in the SOM Training section. However, using this methodology, 17 variables were identified as the optimal features for identifying this specific anomaly.

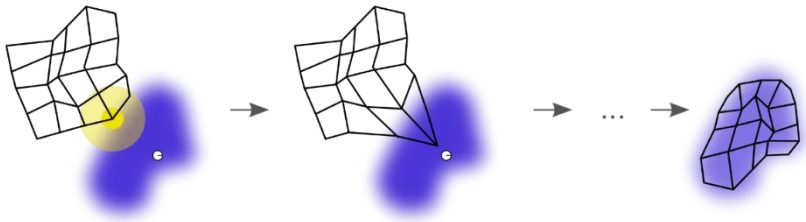

Figure 3: SOM Training

Reprinted from Wikimedia Commons, by Mcld, 2010, Retrieved from https://commons.wikimedia.org/wiki/File:Somtraining.svg

Machine Learning Model Selection

After data collection and feature vector creation, the next step was to choose an ML algorithm. Machine learning algorithms fall into three separate categories: supervised, semi-supervised, and unsupervised. Each category of algorithm performs well with specific types of data, which means that algorithm selection is driven by data characteristics. Supervised learning requires fully labeled, evenly distributed data. Semi-supervised requires partially labeled data and can deal with unbalanced data. Unsupervised requires no labels and can also deal with unbalanced data. Due to the characteristics of the PTMS data -partially labeled and unbalanced - supervised algorithms were not a viable option for anomaly detection. Semi-supervised and unsupervised learning algorithms would both perform well with the data. However, to leverage the limited information about known anomalies, a semi-supervised classifier was the ideal option and was selected for the work.

Within the realm of semi-supervised learning, one-class support vector machines (SVMs), kmeans, and self-organizing maps (SOMs) are frequently used methods of anomaly detection. SVMs are traditionally a supervised learning method but can be adapted to make use of limited known information. However, they exhibit extreme sensitivity to hyperparameter selection, and adequate hyperparameter selection has proven to be challenging (Hsu & Chang, 2003). For these reasons, SVMs were not chosen for initial method exploration. Kmeans is also a commonly used clustering technique and is quick to implement. It is a proven method for anomaly detection and allows for use of limited known information. The weaknesses of kmeans are that the algorithm is very sensitive to initial hyperparameter values, namely the number of clusters and their initialization locations (Chen, Qin, Liu, Liu, & Li, 2010). Kmeans assigns data points to randomly initialized clusters based on which cluster is closest in n-dimensional space. However, the updates to each cluster have no effect on neighbors; each cluster independently shifts and grows. This means that reaching a global optimum is challenging (Raykov, Boukouvalas, Baig, & Little, 2016). Self-organizing maps (SOMs) perform similarly to kmeans with some notable differences. It is also a clustering algorithm, and randomly initializes locations of clusters. However, the method is less sensitive to initial hyperparameters. As the algorithm trains on a growing data set, the location of each cluster is updated based upon the data points assigned to it. Early in the training cycle, newly assigned data points have a significant effect on the location of the cluster. As training continues, the effect of a newly assigned data point lessens, resulting in fine tuning of cluster shape and location, as shown in Figure 3. This molding of the clusters to match the data means that SOMs outperform kmeans

© 2020 Lockheed Martin Corporation. All Rights Reserved.

when finding the global optimum. Additionally, SOMs are more suitable for visualization of the data-providing clear insight into cluster behavior, which allows users to identify what makes an anomaly an anomaly (Chen, Qin, Liu, Liu, & Li, 2010). This information has the potential to inform engineering experts about behaviors and anomalies previously unidentified. Resulting in a decrease in data noise.

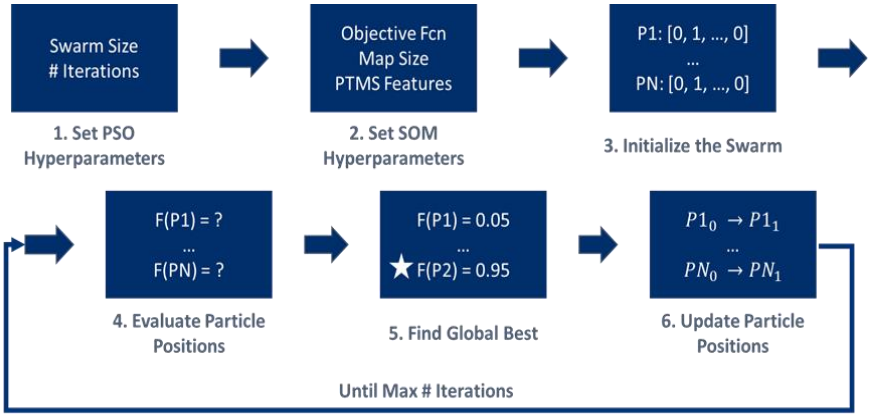

Figure 4: B-PSO Training Process

Performing this review indicated that SOMs were the best fit for the problem space and data. Robustness to ambiguities of error identification and steady state performance, handling unbalanced data, and the possibility of gleaning new information about the data set made it a promising method.

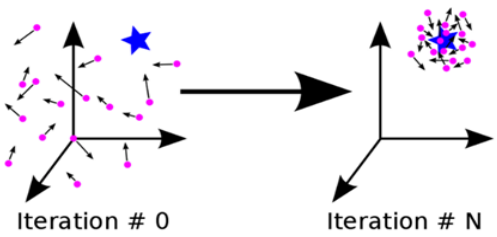

Figure 5: B-PSO Convergence

SOM Training

Once the model and the methodology for identifying the feature vector were selected, the next step was the B-PSO training as shown in Figure 4. As mentioned earlier, B-PSO is used in combination with the SOM methodology. The B-PSO algorithm is responsible for selecting the feature vector for the SOMs to train on, uses the performance metrics of those SOMs to update the selection of the feature vector, and the SOMs eventually converge on a feature vector that results in optimal SOM performance as shown in Figure 5.

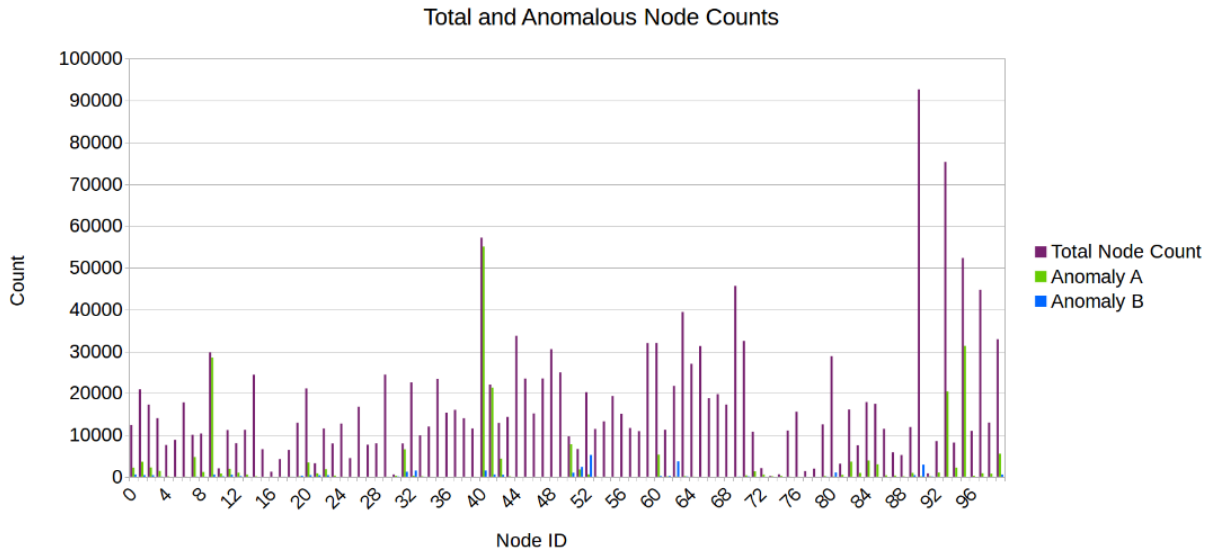

Figure 6: Anomaly Proportion by Node

© 2020 Lockheed Martin Corporation. All Rights Reserved.

To run B-PSO, several training hyperparameters needed to be set. PSO hyperparameters included swarm size, and number of iterations. SOM hyperparameters included the size of the map, learning rate, neighborhood decay rate, and number of training iterations. The optimal hyperparameters often depend on the data set characteristics and are often determined through experimentation and analysis of results. Once the hyperparameters were decided upon, the B-PSO runs began. For this work, a subset of approximately 3,500 flight run files were used as the training data set for SOMs. This training data was pulled at random, with the remaining portion set aside to use for testing and validation of the trained classifier. A training data point consisted of a time series data point measuring selected pressure and temperature sensor data from the PTMS system flight run files. The training data set serves to establish the optimal feature vector for this problem space. The parameter combination was then tested using three independently trained SOMs whose results were then averaged.

Once the optimal feature vector was identified, and the associated unsupervised SOM was trained, a contextual phase was applied. The contextual phase utilized semi-supervised machine learning methodology to overlay known anomalies back onto the SOM. Sample output from the contextual phase is shown in Figure 6. This figure shows the node ID, the number of data points in each node, and the proportion of anomalous points in each node. The hypothesis was that the anomalies would cluster in like areas of the SOM, demonstrating that anomalous flight runs contained distinguishably different behavior. Comparing the total vs anomalous counts in each node, Figure 6 shows that nodes 10, 41, and 42 contain a high proportion of data from anomalous flight runs. Preliminary analysis of the results, like those in Figure 6, suggested that the SOM could in fact differentiate between anomalous and non-anomalous flight data. The ability of the SOM to differentiate between the anomalous and non-anomalous flight run data meant that the method could be used to build an automatic classifier. This automatic detection could help aid the maintainers by identifying potential issues in the aircraft subsystem before they become critical. In addition, an automatic method of detection increases the likelihood of an anomaly being identified, since the current amount of data is too large for the maintainer team to sort through and is too complex for standard deterministic methods.

SOM Classifier Creation

Once preliminary training results demonstrated the SOM's ability to separate anomalous and non-anomalous data, the next step was to build a classifier. This classifier automatically determines if a flight run was anomalous or not. Building this classifier required a way to use the time series data from within a flight run file, to classify an entire flight run as anomalous or not. To accomplish this, the authors used the information from Figure 6. Inspecting the graphs for each individually trained SOM, often showed that much of the data from the anomalous flight runs fell into a handful of nodes. These nodes were designated as anomaly indicators. A flight run file was classified as anomalous if a certain number of its data points fell into these anomalous nodes. This certain number of data points was referred to as the threshold. Initially, the threshold determining the optimal number of data points was unclear, so the threshold value for making the anomaly decision was turned into a hyperparameter and varied. Designating the threshold as a hyperparameter allowed the authors to experimentally determine the threshold corresponding to the maximum classification accuracy. This threshold method was necessary in part because of the uncertainty associated with the

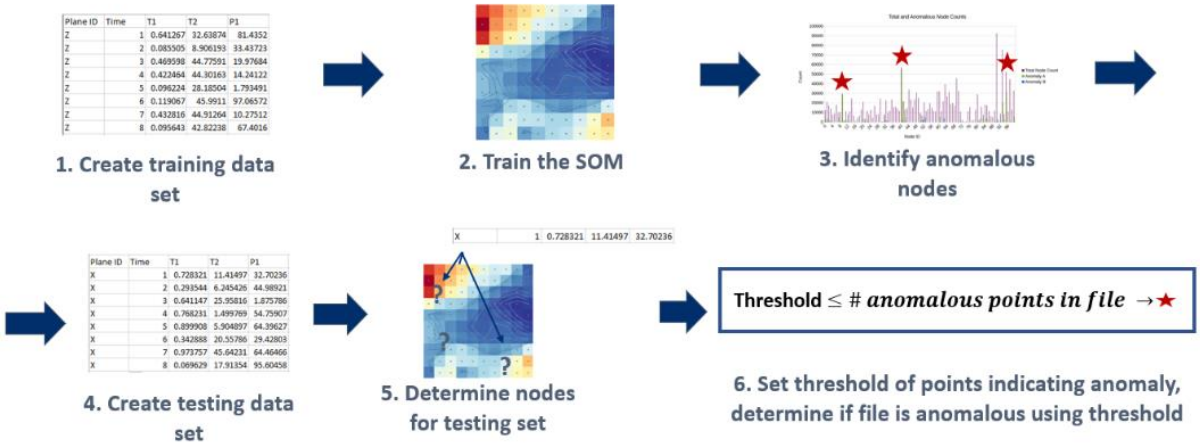

Figure 7: SOM Training and Classification Process

© 2020 Lockheed Martin Corporation. All Rights Reserved.

labeled data. Recall that a flight run was designated anomalous or not based on SME analysis. The SME generally labeled a flight run anomalous only after a known issue was found. While the flight run file was known to have anomalous behavior, one could not be certain that all the measurements within a file were truly anomalous. Some of the data could in fact be partially normal. As a result of this coarse labeling, uncertainty was injected into the system. As such, the system needed to be tolerant to the existence of some non-anomalous data contained in a flight run labeled anomalous. The threshold method allowed for this to be considered.

The entire training and testing process for the classic cluster classifier is shown in Figure 7. The first training step involves creating a data set that contains examples of both anomalous and non-anomalous data. The existence of anomalous data in the data set is important since it allows the researchers to determine where the known anomalies cluster. The second step trains the SOM using the training method outlined in the Background section. After the SOM is trained, the next step involves determining where the limited number of data points from known anomalous files fall. From here the last step is to test how accurate the trained classifier is at predicting anomalies in a testing data set using a predetermined threshold. The next section provides the results from this process and discusses their implications.

RESULTS AND DISCUSSION

The goal of the PSO-SOM classifier was to select a set of features that maximizes the F1 score measure of performance. F1 score was used as a measure of performance rather than overall accuracy because of the unbalanced nature of the dataset. While the PSO optimization routine removed the need to explicitly select a set of features, a number of other parameters did require specification. Often ML requires tuning what are called hyperparameters. These hyperparameters vary depending on the machine learning method selected. For the SOM classifier the hyperparameters consisted of map size, and threshold for anomalous file designation. For the results presented below these hyperparameters along with the PSO driven feature selections were varied in an attempt to find the optimal combination, resulting in the most accurate classifier. Accuracy results presented below include overall accuracy, recall, and precision. The overall accuracy is a measure of how many flight run files the classifier correctly labeled as anomalous or non-anomalous. However, this metric does not provide a complete picture of classifier performance, especially with highly unbalanced data sets, so recall and precision are also used to augment the results presentation. Recall can be thought of as a measure of how many true positives a classifier misses. If a classifier has a very low recall, then it is missing many of the anomalous flight runs. Precision can be thought of as a measure of how accurate the classifier is when it predicts a flight run contains anomalous behavior. If a classifier has a low precision it is not very accurate when it flags a file.

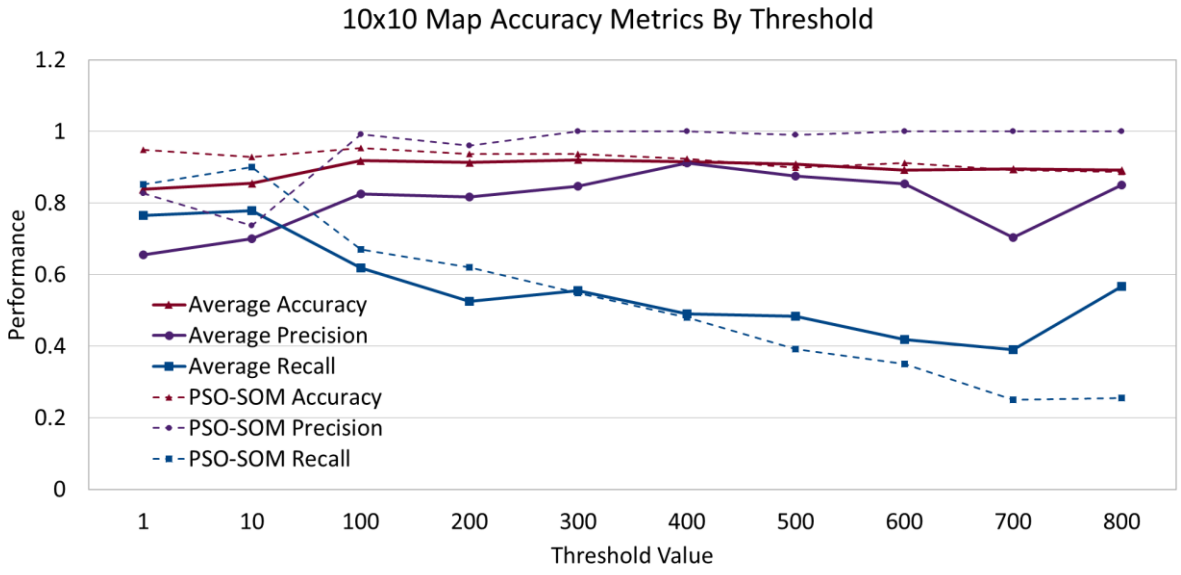

Figure 8: 10x10 SOM vs PSO-SOM Accuracy Metrics

© 2020 Lockheed Martin Corporation. All Rights Reserved.

Use cases influence the level of precision and recall required in ML classifiers. For example, say letting a defective part off an assembly line is very expensive. In that case an ML system would require high recall. It would be conservative and flag parts as defective even if there was only a slight suspicion. In another application, taking a machine out of service for maintenance may be very expensive. In that situation, an ML classifier with high precision would be needed since high confidence would be required when taking a machine off-line. The testing results discussed below present the implications of these tradeoffs as they correspond to the tabulated results. Note that classifier accuracies presented are an average based on three independent SOM classifiers for a given hyperparameter combination. This average measure is intended to damp out any variation associated with randomly selecting testing and training data.

The first PSO-SOM based flight run classifier used a ten by ten SOM map. This map size was fixed and the threshold for classifying a file as anomalous was varied. In this classifier, each parameter combination was tested using three independently trained SOMs whose results were then averaged. For each run, seven-hundred flight files were used for training, five hundred for testing, and five hundred for validation. The number of files used for training was dictated by the memory constraints of the computer. Figure 8 shows the overall accuracy, precision, and recall values for each threshold. The solid lines are the baseline results using a SME selected feature vector. The dotted lines represent the accuracy metrics for SOMs trained using PSO selected features. On the graph, a score of one indicates perfect performance or one-hundred percent correct answers. In practice, a classifier rarely reaches one-hundred percent accuracy. In order to deploy a classifier, generally accuracy is desired to be above the high eighties. Looking at the results, the graph suggests that for a ten by ten map size a threshold of one-hundred produces the highest overall accuracy for both the SME and PSO selected feature vectors, while also providing a high recall and precision. The graph also shows that a threshold increase results in a decrease in recall and an asymptotic increase in precision for both feature selection methods. This means that as the threshold goes up, it requires that more points fall into the anomalous nodes before a flight is flagged as anomalous. This results in higher prediction precision. Remember, however, in some cases a higher recall might be more desirable. If a user desired a higher recall, given the classifier results in Figure 8 they would most likely select a threshold of ten. However, they would have to accept a decrease in precision. Ultimately, it is up to the user to make the tradeoff depending on the application. For the SME's application a higher precision is required since repairing the subsystem can be time consuming and expensive. As a result, when a system is flagged they desire high confidence that something is not performing as expected. This desire for high confidence drove the selection of a higher precision at the expense of recall for the ten by ten map.

These accuracy metric and threshold related trends hold true for both the SME selected features and the PSO selected feature SOM classifiers. However, PSO-SOMs demonstrate higher accuracy metric scores. On average the PSO-SOM

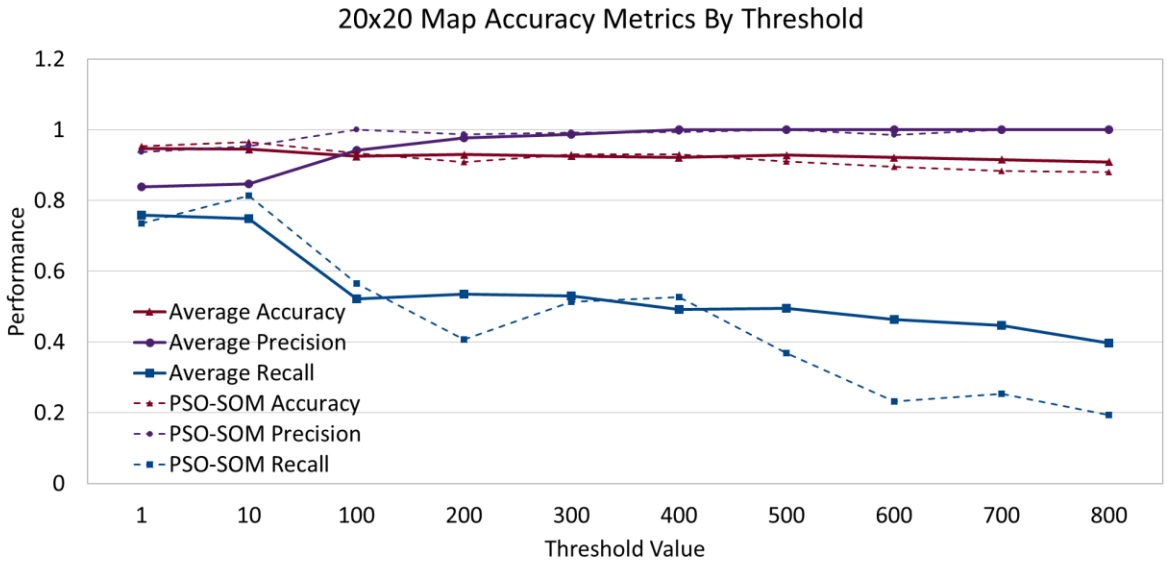

Figure 9: 20x20 SOM vs PSO-SOM Accuracy Metrics

© 2020 Lockheed Martin Corporation. All Rights Reserved.

method increased accuracy, precision, and recall by about fifteen percent. This demonstrates the ability of the PSO feature selection method to help increase classifier accuracy. Results suggest optimization-based feature selection method can aid in situations where the optimal combination of features is unknown or if a SME is unavailable. This increase in accuracy also means that the classifier is more trustworthy, increasing its usefulness and deployability. Ultimately, these increases are especially encouraging due to the dataset limitations that drove the selection of the semi-supervised classifier.

The second SOM based flight run classifier tested used a twenty by twenty SOM map. Again, the map size was fixed and the threshold for classifying a file as anomalous was varied. Each parameter combination was tested using three independently trained SOMs whose results were then averaged. For each run, seven-hundred flight files were used for training, five-hundred for testing, and five hundred for validation. Figure 9 shows the overall accuracy, precision, and recall values for each threshold. Again, the solid lines are the SME selected features and the dashed are the PSO-SOM selected features. Looking at the results, the graph suggests that for a twenty by twenty map size a threshold of ten produces the highest overall accuracy while also providing a high recall and precision. As with the ten by ten map, testing results generally show that increasing the threshold decreases recall and increases precision to a point. However, the twenty by twenty classifier seems to be less sensitive to threshold changes than the ten by ten map. While both classifiers ultimately produce similar overall accuracies, the twenty by twenty classifier seems to provide less of a tradeoff between precision and recall. However, the twenty by twenty map results do not respond as well to PSO-SOM optimization. At a threshold of ten, the twenty by twenty map produces accuracy results that are not as high as the ten by ten map. This could suggest that the twenty by twenty map is suffering from overfitting to the training data set, since accuracy results for the validation data set are lower. If overfitting is occurring, this could suggest that the ten by ten map would generalize better when deployed to detect new anomalous flight runs than the twenty by twenty map. In order to determine if over fitting is occurring, more aircraft data, not included in the initial data set would be required. Seeing how well each method detects anomalies in new aircraft would help determine the model with the ability to best generalize.

Overall, the results demonstrate that a fairly accurate ML classifier can be created using challenging data. In addition, using PSO to optimize classifier feature selection can help increase detection accuracy metrics. This ability to auto-select features can help machine learning practitioners create more accurate classifiers to help maintainers identify anomalous flights even when data SME's are unavailable or little is known about a system. Ultimately, this new PSO-SOM method will allow machine learning anomaly detection to be applied to a wider range of problems more accurately.

CONCLUSION AND FUTURE WORK

With increases in equipment complexity, organizations, such as the DoD, are turning to data driven analysis via machine learning to help pinpoint issues. This monitoring is made possible by increasingly economical data collection. While the technology industry has demonstrated success analyzing data for trends using traditional ML and artificial intelligence, there exist unique challenges outside of this domain. Challenges such as data collection can make using traditional machine learning methods infeasible. Another factor complicating the creation of AI for military applications, is the lack of insight into which variables best capture complex processes. This can make it challenging to determine which variables are important factors to include in an AI model.

The work presented in this paper discusses the development of a semi-supervised ML method to identify anomalous performance in an aircraft subsystem. Occasional anomalous performance and data labeling difficulties made it challenging to construct a machine learning model. To combat this the authors developed a Self-Organizing Map (SOM) based classifier and paired it with an automated feature selection mechanism using Binary Particle Swarm Optimization (B-PSO). Classifier testing results demonstrated that it was able to detect anomalous PTMS performance around 95% of the time using both a ten by ten and a twenty by twenty map. Also, results demonstrated the ability to use the B-PSO feature selection method to increase accuracy metrics over a SME specified feature vector. However, results also showed that classifier hyperparameters play an important role determining more nuanced measures of performance and preventing overfitting. As a result, these hyperparameters need to be carefully tuned based on the application of the classifier. Ultimately, the tool helps to decrease maintainer analysis workload when looking for issues in the subsystem output data. Moving forward, future work on the classifier will focus on pinpointing specific causes of anomalous system behavior, further aiding the maintainers and hopefully reducing equipment downtime.

 © 2020 Lockheed Martin Corporation. All Rights Reserved.

REFERENCES

Alelyani, S., Tang, J., & Liu, H. (n.d.). *Feature Selection for Clustering: A Review.*

Anshuk Gandh, Carmen Magar, R. (2013). *How technology can drive the next wave of mass customization.* McKinsey on Business Technology.

Bean, R. (2018). *The State of Machine Learning in Business Today*. Retrieved from Forbes: https://www.forbes.com/sites/ciocentral/2018/09/17/the-state-of-machine-learning-in-business-today/#5dbf6f93b1de

Benabdeslem, K., & Lebbah, M. (n.d.). *Feature Selection for Self-Organizing Map.*

Chandrashekar, G., & Sahin, F. (2014, 1). A survey on feature selection methods. *Computers and Electrical Engineering, 40*(1), 16-28.

Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning.* Cambridge, MA.: MIT Press.

Chen, Y., Qin, B., Liu, T., Liu, Y., & Li, S. (2010). *The Comparison of SOM and K-means for Text Clustering.* 268-274: Canadian Center for Sciences and Education.

Columbus, L. (2019). *The Most Innovative Companies of 2019 According to BCG*. Retrieved from Forbes: https://www.forbes.com/sites/louiscolumbus/2019/03/24/the-most-innovative-companies-of-2019-according-to-bcg/#50b7ee82486d

Drezner, J. (2009). *COMPETITION AND INNOVATION UNDER COMPLEXITY.* RAND.

Fryer-Biggs, Z. (2018). *The Pentagon plans to spend $2 billion to put more artificial intelligence into its weaponry - The Verge*. Retrieved from The Verge: https://www.theverge.com/2018/9/8/17833160/pentagon-darpa-artificial-intelligence-ai-investment

Giannopoulou, E., & Mitrou, N. (2018). Extensive experimental evaluation of self-organizing maps for automatic classification of a multi-class multi-label corpus. *IEEE Access, 6*, 67385-67403.

Goay, C., Abd Aziz, A., Ahmad, N., & Goh, P. (2019, 12 1). Eye diagram contour modeling using multilayer perceptron neural networks with adaptive sampling and feature selection. *IEEE Transactions on Components, Packaging and Manufacturing Technology, 9*(12), 2427-2441.

Grammich, C., Arena, M., Younossi, O., Brancato, K., & Blickstein, I. (2008). *A Macroscopic Examination of the Trends in U.S. Military Aircraft Costs over the Past Several Decades.*

Hawley, J., & Swehla, M. (2018). *The New Equipment is Here, Now Comes the Hard Part: Cognitive and Sociotechnical Challenges in Network-Enabled Mission Command.* ARL.

How real-time data is transforming five industries. (2019). Retrieved from Innovation Enterprise Channels: https://channels.theinnovationenterprise.com/articles/how-real-time-data-is-transforming-five-industries

Hsu, C.-W., & Chang, C.-C. &.-J. (2003). *A Practical Guide to Support Vector Classification.* Department of Computer Science and Information Engineering, National Taiwan University.

Hyvarinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 626-634.

Ijaz, A., & Choi, J. (2018, 4 1). Anomaly Detection of Electromyographic Signals. *IEEE Transactions on Neural Systems and Rehabilitation Engineering, 26*(4), 770-779.

Iriarte, M. (2018). *Test and measurement sector grapples with standardization, complex systems*. Retrieved from Military Embeded Systems: http://mil-embedded.com/articles/test-measurement-grapples-standardization-complex-systems/

Jamian, J., Abdullah, M., Mokhlis, H., Mustafa, M., & Bakar, A. (2014). Global particle swarm optimization for high dimension numerical functions analysis. *Journal of Applied Mathematics, 2014*.

Kobielus, J. (2019). *Looking ahead to Next '19, Google puts AI at the center of cloud hyperscaling*. Retrieved from Silicon Angle: https://siliconangle.com/2019/03/25/looking-ahead-next-19-google-puts-ai-center-cloud-hyperscaling/

Kohonen, T. (2012). *Self-organizing maps* (Vol. 30 ed.). Springer Science & Business Media.

Lever, J. K. (2017). Principal Components Analysis. *Nature*, 641-642.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R., Tang, J., & Liu, H. (2017). Feature Selection: A Data Perspective. *ACM Computing Surveys (CSUR), 50*(6), 1-45.

Libert, B., & Beck, M. (2018). *Machine Learning Is A Moneyball Moment For Companies*. Retrieved from Forbes: https://www.forbes.com/sites/barrylibert/2018/08/31/machine-learning-is-a-moneyball-moment-for-companies/#4bac378e44ec

Macallister, A., Belknap, J., Clement, D., Summers, S., & Hellstern, G. (2019). Prognostic Health Management Using Semi-Supervised Machine Learning. *I/ITSEC.*

 © 2020 Lockheed Martin Corporation. All Rights Reserved.

Raj, P., & Kumar, S. (n.d.). *Feature Selection using Binary PSOGSA and Radial Basis Network with a Novel Fitness Function.*

Raykov, Y., Boukouvalas, A., Baig, F., & Little, M. (2016, 9 1). What to do when K-means clustering fails: A simple yet principled alternative algorithm. *PLoS ONE, 11*(9).

Russell, S., & Norvig, P. (2016). *Artificial intelligence: a modern approach.* Malaysia: Pearson Education Limited.

Spencer, M. (2019). *Artificial Intelligence Hype Is Real*. Retrieved from Forbes: https://www.forbes.com/sites/cognitiveworld/2019/02/25/artificial-intelligence-hype-is-real/#33aa935d25fa

Taijia Xiao, Dong Ren, Shuanghui Lei, J. (2015). Based on grid-search and PSO parameter optimization for Support Vector Machine. *Proceeding of the 11th World Congress on Intelligent Control and Automation*, 1529-1533.

United States Department of Defense. (2018). *Summary of the 2018 Department of Defense Artificial Intelligence Strategy.*

Wiggers, K. (2019). *Google researchers improve reinforcement learning by having their AI play Pong | VentureBeat*. Retrieved from Venture Beat: https://venturebeat.com/2019/03/25/google-researchers-improve-reinforcement-learning-by-having-their-ai-play-pong/

 © 2020 Lockheed Martin Corporation. All Rights Reserved.