EasyChair Preprint

№ 656

A survey of feature space reduction methods for context aware processing in IoBT networks

Andre Harrison, Darius Jefferson, Adrienne Raglin, Brian Jalaian and Michael Lee

November 29, 2018

Symposium: 23nd International Command and Control Research and Technology Symposium

Topic 7: Human Information Interaction

Title: A survey of feature space reduction methods for context aware processing in IoBT networks

Authors:

Andre V. Harrison
U.S. Army Research Lab, RDRL-CII-B
2800 Powder Mill Rd, Adelphi, MD 20783
andre.v.harrison2.civ@mail.mil

Darius Jefferson II
U.S. Army Research Lab, RDRL-CII-B
2800 Powder Mill Rd, Adelphi, MD 20783
darius.e.jefferson.ctr@mail.mil

Adrienne J. Raglin
U.S. Army Research Lab, RDRL-CII-B
2800 Powder Mill Rd, Adelphi, MD 20783
adrienne.j.raglin.civ@mail.mil

Michael Lee
U.S. Army Research Lab, RDRL-CII-B
2800 Powder Mill Rd, Adelphi, MD 20783
michael.h.lee.civ@mail.mil

Brian Jalaian
U.S. Army Research Lab, RDRL-CII-B
2800 Powder Mill Rd, Adelphi, MD 20783
brian.a.jalaian.civ@mail.mil

A survey of feature space reduction methods for context aware processing in IoBT networks

Abstract:

The military use of the Internet of Things within a battlefield environment aims to combine the information collected from a system of heterogeneous sensors and actuators in order to create a cohesive model of the relevant battlefield so that intelligent agents can provide risk-aware decisions or take proper actions, and to collectively give warfighters an edge. To do inference and reasoning under uncertainty efficiently, the most important and relevant features regardless of modality must be identified for each given context, classification task, and classification approach. This can minimize the computational costs required to build a specific model, increase the accuracy of the model and may also allow the model to be generalized. However, the dynamic and adversarial nature of the battlefield may mean that the availability and reliability of sensors will vary over time. Adding a certain amount of redundancy in the set of features used to train an ensemble of classifiers may improve model robustness and minimize uncertainty. One approach to achieve this is by modeling the feature space so that the likely importance of a given set of features can be estimated when context, classification task, or approach is varied. To efficiently understand the shape of a given feature space and to locate clusters of features in a locally distributed fashion, we surveyed methods to select important features and to describe or explore a given feature space.

Introduction:

The increase in the number of things that can sense and react to events continues to increase as Internet of Things (IoT) devices are used in more locations and for more tasks. As these systems become more co-located, as a network of wearables systems, or in the form of intelligent environments, the capabilities and intelligence of these systems can be improved through the fusion of their information. This is true in both the commercial sector as well as within the military domain of battlefield environments. The combined utility of these systems and their coordinated sensing and activity will substantially alter future military capabilities. Already we have shown that the combined sensing of a set of heterogeneous sensors on a common object can improve the ability of machine learning algorithms when attempting to predict the state of the measured entity (Dennison et al. 2016; Vlachostergiou et al. 2018). The use of heterogeneous IoT devices in a military setting or for military objectives is referred to as the Internet of Battlefield Things (IoBT) (Kott, Swami, and West 2016), but leveraging the capabilities of IoT devices under battlefield constraints adds a unique challenge to research around IoT systems.

Typically, within a given context there is a preferred sensing modality when trying to detect the state of an object, but the state of that object may also cause detectable effects within other domains. Sensing the effects caused by the state of an object in every relevant modality and combining that information may not only improve the accuracy of predicting the state of that object within that known context, but the prediction of the state may be made more robust through the simultaneous detection of the context that the state occurs in. This approach may also provide a richer description of the state of the target under observation. The

use of multiple heterogeneous sensors may also enable the simultaneous detection and prediction of multiple target states.

However, an increase in the number of sensors used will also increase the number of dimensions that machine learning algorithms will have to search during training. If nothing is done to reduce the dimensionality of the search space then every learning task would have to search all features concerning the target from every sensor type near the target in order to build a model. However, techniques to efficiently search through an available set of features have already been researched and implemented over many years in the form of dimensionality reduction methods (Hira and Gillies 2015). This has become a fairly standard problem within data science, but many of the current dimensionality reduction techniques are not ideal within a battlefield environment. A popular method to reduce the search space for classification methods is to apply feature extraction to a given set of input features in order to project these input features into a lower dimensional space, however, the lower dimensional features are typically a combination of most if not all of the input features. In a battlefield environment the reliability of a network cannot be guaranteed; sensors may become lost, compromised, or corrupted. In these situations, the network is expected to be dynamic adapting to sensors leaving or joining a given network at unpredictable times. The corruption or compromise of sensor feeds may happen in potentially non-obvious or adversarial, ways. Thus, feature extraction methods that project combinations of all of the original input features into a lower dimensional space would have to be recalculated every time a sensor node is added or removed from the relevant network. Also, the network limitations of the battlefield may not support the transmission of all of the features from every sensor in proximity to the target.

Feature selection methods are another approach to dimensionality reduction that tries to reduce the dimensionality of the search space by only using a subset of the full set of features as an input to train and evaluate a model. This is in some sense more favorable to the constraints of an IoBT network, as feature selection methods only have to be rerun when one of the selected features is lost (or removed) or when an entirely new feature, or set of features, is added to the network. However, feature selection methods that can identify the optimal set of features can be very computationally intensive as these methods often must incorporate the training and evaluation of a specific learning model in order to identify the best set of features. Also, identifying the best set of features has its own inherent computational complexity. Without a well-defined relationship between features and learning methods, selecting the globally optimal set of features to accurately predict an objects state is an NP hard task. In a battlefield environment the unpredictable network and limited bandwidth will also limit available computational resources. Most computation will have to be done at or near the edge, increasing the computation time of feature selection methods or supporting only the simplest of feature selection methods.

The dynamic and potentially uncertain nature regarding the availability of sensors in an IoBT network means that any machine learning algorithms running within this environment will have to quickly and frequently adapt themselves to the changing network dynamics of the battlespace, the potential compromise of sensor and network feeds, and the drift of concepts regarding the classification of state based on mission needs. In some cases, the adaptation of machine learning algorithms can occur slowly and gradually in response to gradually changing patterns or goals. This is true when classification label boundaries drift, the statistical patterns of a given feature or set of features changes, or when new sensor nodes are added. Other adaptations like, when a set of nodes becomes unusable, either due to network connectivity problems or loss of trust, then the trained model needs to be adapted very quickly. Because while the model is being retrained the prior model remains invalid leaving that system down for potentially an unacceptable amount of time. The unpredictability of when a model may need to be quickly retrained is unacceptable in a military setting due to optempo requirements. One method to minimize the impact that the loss a set of features

may have on a given machine learning model is to train an ensemble of different machine learning models on different sets of sparse features where each set is independently able to achieve quality predictions. Thus, when a set of features becomes lost, unreliable, or overly noisy only the subset of machine learning models that they serve as an input to may be compromised. In the simplest scenario the compromised set of models can quickly be dropped from the ensemble of machine learning models while new sets of features can be later identified and added to the set of useful features for a given state. To minimize the overhead that this approach incurs in terms of extra bandwidth and computational power for the redundant features we need efficient methods to identify multiple sets of sparse features. These sets of features may or may not share features between sets, but no feature should occur in every set.

In this paper we conduct a survey of dimensionality reduction methods that may provide guidance or a best place to start towards identifying multiple sets of sparse features for classification and regression methods. The challenge of this goal is that the identification of the best set of features that satisfies this requirement in any given situation is an NP hard problem as there is no pre-defined relationship between a given set of features, the class labels (values), and the learning model, potentially demanding that every possible combination must be evaluated to find a global optimum. Dimensionality reduction methods have taken different approaches to find solutions to this problem either by making certain assumptions about the statistical properties of the underlying features or through heuristic methods, which we discuss further in the following section. However, the primary goal of most of these methods has only been to find a single, approximately optimal solution, not multiple solutions with near equivalent accuracy. We survey the set of approaches in order to determine what methods may best locate these multiple sets of features.

Feature extraction

Feature extraction is one of the two major categories of dimensionality reduction methods (Joshi and Machchhar 2014). It combines all of the input features and transforms them into a new set of features within a smaller dimensional space (Joshi and Machchhar 2014; Khalid, Khalil, and Nasreen 2014). Feature extraction methods aim to prioritize the relevant features obtained from the IoBT network. Feature extraction methods are also useful due to the constrained bandwidth of IoBTs (Kott, Swami, and West 2016). The reduced size of the transformed features means that less bandwidth has to be used to transmit the features if necessary. By combining features and reducing the feature space, the significant elements of the original feature space can be propagated across the network with most of their information still intact. In order to further investigate the advantages and disadvantages of feature extraction, it is necessary to examine its various methods, both linear and nonlinear.

Linear methods

One of the most popular linear methods, as well as being one of the most commonly-used feature extraction methods is Principal Component Analysis (PCA) (Joshi and Machchhar 2014; Khalid, Khalil, and Nasreen 2014; Sharma and Saroha 2015). PCA is an unsupervised feature extraction method that extracts important features from a dataset and transforms them through linear combinations. These linear combinations are known as principal components, or PCs and they are all orthogonal and uncorrelated to each other. PCA maximizes the information obtained from the original features, as measured by the variance of the data. It then ranks each of the PCs by their variance value. The ones with the highest variances are kept for the new representation. It also eliminates the PCs with the least amount of variance, as they are typically deemed as not having significant information. By eliminating these PCs, the dimensional space is reduced while keeping most of the original data intact. Some of its main applications include data compression, facial and image analysis, and pattern recognition (Khalid, Khalil, and Nasreen 2014). One of the major advantages of using PCA is that it useful for extracting important information from noisy or redundant data (Khalid,

Khalil, and Nasreen 2014). It has a better accuracy than some nonlinear methods, as well (Sharma and Saroha 2015). A disadvantage of PCA is that, as a linear method, it will automatically assume that the relationships between features are linear (Joshi and Machchhar 2014; Khalid, Khalil, and Nasreen 2014). Another disadvantage is that the PCs generated by the algorithm are often very difficult to interpret (Sharma and Saroha 2015). PCA is also sensitive to the scaling of the input features (Khalid, Khalil, and Nasreen 2014; Sharma and Saroha 2015).

Another linear feature extraction method is Linear Discriminant Analysis (LDA) (FISHER 1936). Its main goal is to find a line, upon which data points can be projected, that best separates the different classes of input features (Pali, Goswami, and Bhaiya 2014). As a result, unlike PCA, LDA is a supervised method, meaning that it will use a training set of data with labels. If the number of dimensions within the input dimensional space is represented by N and the different classes of features is represented by M, then LDA will project the input features into an M-1 dimensional space (Pali, Goswami, and Bhaiya 2014). By doing so, it can reduce the number of dimensions while still maintaining distinction between classes. Simultaneously, this will maximize the ratio between the determinant of the between-class scatter matrix and the determinant of the within-class scatter matrix (Pali, Goswami, and Bhaiya 2014). The between-class scatter matrix shows how the various features are classified separately from each other, while the within-class scatter matrix captures the distances between features within the same class. LDA is particularly useful in facial recognition (Pali, Goswami, and Bhaiya 2014). Typically, when performing feature extraction of data, PCA is completed first and then LDA is applied (Pali, Goswami, and Bhaiya 2014; Sharma and Saroha 2015). This is done because PCA focuses more on dimension reduction while LDA focuses more on maintaining class distinction.

The final linear feature extraction method we describe in this paper is Independent Component Analysis (ICA). ICA is also an unsupervised feature extraction method and the main objective of ICA is to maximize the statistical independence of the input features of a multivariate signal (Joshi and Machchhar 2014; Khalid, Khalil, and Nasreen 2014). For such a signal, ICA requires that there is an equal number of features and samples (Joshi and Machchhar 2014). ICA assumes that the source signals are linearly independent from each other and are also non-Gaussian (Khalid, Khalil, and Nasreen 2014). It has some advantages over PCA, like having a better probabilistic model that can identify data in an n-dimensional space (Pali, Goswami, and Bhaiya 2014). Another advantage is that it can find a (possibly orthogonal) basis that may reconstruct data more accurately than PCA when dealing with noise (Pali, Goswami, and Bhaiya 2014).

Non-linear methods

A nonlinear extension of the linear PCA feature extraction method is Kernel Principal Component Analysis (KPCA). The main difference between PCA and KPCA is that KPCA performs an extra step before PCA in order to handle nonlinear input data. It will first transform the input data, using a nonlinear function, into a higher-dimensional kernel space (Joshi and Machchhar 2014). It will then perform PCA on the data within that feature space (Joshi and Machchhar 2014). One of the disadvantages of KPCA is that the size of the kernel dimensional space is proportional to the input data squared (Joshi and Machchhar 2014). Thus, the kernel space will increase at a quadratic rate as the amount of input data does, which could affect the processing time of the algorithm. Being that it is a variant of PCA, KPCA is also an unsupervised method.

Another well-known nonlinear method is Locally Linear Embedding (LLE). LLE is also an unsupervised method that assumes the input samples are samples of a manifold and it transforms the manifold's points into a lower dimensional space while preserving information from neighboring points (Arechiga et al. 2017; Ventura 2008). LLE has three major steps that it performs for the set of manifold points (Ventura 2008). First, it finds the nearest k neighbors for every data point. Next, weights are calculated in order to find a

weighted approximation of the original points using a linear combination of those neighbors. Lastly, a matrix containing those weights (W) is used to find a new set of data points within a lower dimensional space while minimizing the reconstruction error. Graphically, a plot of the output coordinates should resemble the topology of the original nonlinear manifold (Arechiga et al. 2017). Regardless of translations, scaling, or rotations, the local relationships between the points during the reduction are comparable to that of the original configuration (Arechiga et al. 2017). The main advantage of this algorithm is that it outputs very sparse weight matrices (since most of the weights are 0) which save computational time and space (Saul and Roweis 2000; Ventura 2008). LLE is another algorithm that could be well-suited to work with other methods (Saul and Roweis 2000). However, one of the disadvantages related to LLE is that the algorithm assumes that each of the original data points has a "preservable" geometric relationship (Arechiga et al. 2017). That is to say, each point and its neighbors are expected to be located on a linear plane so that the reconstruction can be done (He et al. 2008). If the nearest neighbors are not close enough to approximate a linear plane the assumptions of the approach are violated.

Isometric Feature Mapping (Isomap) is an additional, unsupervised nonlinear method (Zheng, Qian, and An 2010). It is very similar to LLE in that it is also nonlinear and assumes the input data sits on a manifold and it also tries to preserve the manifold's intrinsic geometry (Zheng, Qian, and An 2010). The main difference between them is that isomaps estimate the geodesic distances between all points rather than the Euclidean (Arechiga et al. 2017; Zheng, Qian, and An 2010). By doing so, it still preserves the geometry of the manifold but does so on a global scale instead of locally like LLE. Like LLE, it has three steps. The first step is to determine the nearest neighbors, which can be done using k-nearest neighbors, and builds a graph (Zheng, Qian, and An 2010). The second step is to estimate the geodesic distances between all of the points using a shortest-path algorithm such as Dijkstra's algorithm (Arechiga et al. 2017; Zheng, Qian, and An 2010). Lastly, the Multidimensional Scaling (MDS) algorithm is applied to the matrix formed from the shortest-path algorithm (Zheng, Qian, and An 2010). The MDS algorithm will find a lower-dimensional embedding that preserves the distances between those vectors within the matrix (Gupta and Bowden 2011). Since isomaps are an extension of the MDS algorithm, it contains many of the same advantages as MDS but also adds computational efficiency (Zheng, Qian, and An 2010).

Feature selection

Another approach to dimensionality reduction is through feature selection. Feature selection unlike feature extraction does not try to create new features by combining prior features, but instead tries to identify the best subset of a given set of pre-existing features. Because it cannot create new features using feature selection methods depends on the given set of features to be already optimal for a given task. This can mean that the features have been engineered to be relevant to the given task or these features have a human interpretable meaning such that when those features are selected or used in a machine learning algorithm their relevance and behavior is more interpretable.

The final assessment on the utility of a set of features is ultimately based on the performance of a classification (regression) model that uses only those features during the training and testing steps. But including the classification model in the feature selection process is not always efficient and as such feature selection methods typically are divided into one of three categorizes (filter methods, wrapper methods, and hybrid methods). Where the point of discrimination between these categories is based on how a subset of features is identified as the best. For filter-based feature selection methods the feature selection process is wholly independent from the classification method. Instead filter methods estimate the expected utility of a subset of features through different measures (e.g. statistics, information theory, sparse learning, etc.). Wrapper methods actually use a specific classification method to assess the quality of a subset of features based on how well that classifier can be trained using those features. Now, while the classification approach

used is completely interchangeable, the classification method used may affect the final subset of features identified as most relevant. Thus, wrapper methods focus tend to focus on heuristic search methods to arrive at some optimum quickly, though that optimum is typically a local one. For hybrid approaches the feature selection process is part of the classification (regression) algorithm, but within this paper we will primarily focus on the feature selection elements of hybrid approaches. Due to the strong relationship with the learning method must feature selection approaches that we will discuss are supervised.

Filter methods

Filter methods use some form of proxy measure, in place of a classification method, to assess the utility of a feature or a set of features. Because they don't require training to attach a quality measure they can be run much faster than other classes of feature selection methods. However, since they are not tied to a specific classification approach the set of features identified may not be optimal with respect to a given classification method, but they can also be more generic for those same reasons, potentially providing a better starting point for further refinement for a specific classification approach. Filter methods typically use a variety of mathematical functions as a proxy for a predictive model. These functions in lieu of a classification approach aim to measure the similarity (e.g. T-score, F-score, Chi-squared Score (Y. Yang and Pederson 1997)) or relevance (e.g. fisher score, mutual information (Guyon and Elisseeff 2003; Lewis 1992; Y. Yang and Pederson 1997), Pearson correlation coefficient) of a feature to a class label. Typically, these approaches treat each feature independently and rank the importance of a given feature solely based on their relationship with the class label. However, features are often not independent so after the first feature is selected the relevance of any subsequent may be diminished due to redundancies between it and previously selected features.

More advanced filter methods select sets of features in order to maximize the relevance of each feature to a class label, while also minimizing redundancies between selected features. A popular approach that considers both feature relevance and feature redundancy is minimum redundancy maximal Relevancy (mRMR) (Peng, Long, and Ding 2005). In this method each feature is selected sequentially, where each feature receives a score determined by the relevance of that feature with the target label and penalized by the average redundancy of that feature with all previously selected features. The feature with the highest score is selected and then each feature must be rescored again in order to pick the next feature. Measurements of redundancy or relevance can use the same equations, in fact the original formulation for mRMR used mutual information for both relevance and redundancy, however, in (Peng, Long, and Ding 2005), the authors found that using Pearson's correlation measure for redundancy worked better when mutual information was used to measure relevance. Several other methods have been developed to estimate the relationship between features and have taken different approaches to minimize the redundancy between features thereby maximizing the gain when adding a new feature (Brown et al. 2012; Lewis 1992; Novovičová et al. 2007; H. H. Yang and Moody 1999),

Beyond just trying to minimize the redundancy between a feature and the set of features already selected measures like the joint mutual information (Brown et al. 2012; H. H. Yang and Moody 1999) try to maximize the relevance between the selected feature and the target class given a set of already selected features. By considering the conditional mutual information between features these class of measures are able to select new features based on binary feature interactions, where a pair of features have a much stronger relevance to the target class than each feature would have on their own. The limitation with joint mutual information and related approaches is that one feature has to been selected first due to its own relevance to the target class or its interaction with an already selected feature. Pairs of features that individually have a very weak relevance to the target class and do not have a strong interaction with other features may never be selected. This is not the only limitation however; all of the filter selection approaches

that have been discussed only look at binary measure of relevance, redundancy, and conditional mutual information measures; they do not consider higher and more complex interactions.

A more recent approach to filter based feature selection is known as Infinite Feature selection (Eskandari and Akbas 2017; Roffo, Melzi, and Cristani 2015). Infinite feature selection uses an mRMR framework to represent the relationships between all features. Information is represented in an adjacency graph and the importance of each feature is calculated for all possible sets of features including infinitely long ones (with repeats). Features that are selected the most often over more paths are ranked higher. This method was originally developed as an unsupervised approach (Roffo, Melzi, and Cristani 2015), where rather than maximizing the relevance of each feature to the target class each feature was ranked by the magnitude of the variance of each feature. As features with small variance are likely to have poor class discrimination.

Filter based feature selection methods can operate very quickly as they use straightforward concepts of similiarity, correlation, relevance, mutual information, and redundancy. But often these algorithms either assume features are independent, which is often not the case. However, algorithms that try to account for feature redundancy or feature interaction only do so in a limited fashion. Methods that look for feature redundancy and feature interactions (through conditional mutual information) only evaluate binary relationships and then only pairing up features that haven't been selected with features that already have. Redundancies or feature interactions for larger sets of features are often not considered presumably due to the additional computational complexity.

Wrapper Methods

As stated earlier since wrapper feature selection methods don't use a proxy measure they must evaluate a set of features based on how well a specific machine learning algorithm can be trained and evaluated using those features as an input. Hence the set of features considered as optimal is likely to vary with the choice of learning algorithm and the performance metric used (accuracy, F score, precision, recall, etc.). Despite their classification method-based variability, wrapper methods can potentially lead to the development of the best classification models as the input features are directly tied to the classification method. The primary drawback of these methods is that the evaluation time to assess the utility of a set of features can be very long.

Because of the long evaluation times wrapper methods focus on trying to search the available space of possible feature combinations in order to converge to an optimal selection as quickly as possible. However, the identification of the optimal set of features is fundamentally an NP hard problem as there is, in general, no predefined relationship between any of the features and target labels. The most straightforward way to identify the optimal set of features is to use an exhaustive search algorithm that tries every possible combination of features to find the optimal one. This method obviously is not useful for datasets with even a moderately sized number of features as the number of possible feature combinations grows exponentially. Typically, heuristics methods are used that make certain assumptions about the relationships between the data and target classes, but there is no guarantee that assumptions match the data and there often aren't tests to determine when those heuristic assumptions are violated. A well-known wrapper method to search for the optimal set of features is to use the branch and bound method (Narendra and Fukunaga 1977).

The branch and bound (B&B) search method attempts to search the space of all possible feature combinations more efficiently by making assumptions about the maximum accuracy of subsets of features. The B&B method assumes monotonicity of a learning model $J(\cdot)$ where if a set of features $F_i \subset F_j$ then $J(F_i) \leq J(F_j)$. In other words, the accuracy of $J(F_i)$ cannot be greater than $J(F_j)$ if all of the features in F_i are in F_j. The B&B algorithm as described by (Narenda and Fukunaga 1977) forms a tree with all possible combinations of features with the top node consisting of the set of all features. It then performs a depth

first search pruning one feature at each level. When the minimum number of features, m, has been reached the performance of that set of features is set as the minimum performance, J_{min}. The rest of the tree is explored comparing the performance of every node in the tree with J_{min}. If a node on the tree has a performance at or below the current minimum all subsequent child nodes are eliminated from consideration in the search. In this fashion the rest of the tree is searched where the J_{min} is updated whenever a set of m features is found with a performance greater than the current minimum. This method is guaranteed to find the optimal set of features m in a more efficient manner than the exhaustive search, though the cost of the B&B method is still exponential, meaning that a moderate number of initial features may be too large to find an optimal solution in a reasonable amount of time.

Another popular wrapper method is sequential feature selection (SFS) method. This method finds a solution in a much shorter amount of time than exhaustive search with minimal assumptions about the features or their relationship, but the solution found is not guaranteed to be globally optimal. The method consists of a forward and a backward approach where features are sequentially selected or removed, respectively. The 'backward' approach of this method operates in a similar fashion to the B&B method in that the search occurs in a tree-like structure with the top node consisting of the set of all features. However, the 'backward' approach is a breadth-first search method, where the performance of a node is compared with the performance of all of its child nodes. If a node has k features then there are k child nodes and each child node has k-1 features such that the set of child nodes is the set of all possible combinations of k-1 features. The child node with the maximum performance that is also greater than or equal to the performance of the parent node becomes the next parent node and is the only one searched. When no child node has a performance greater than or equal to its parent the search terminates. The 'forward' approach works in a similar fashion, but instead of pruning features at each level a feature is added at each level. The forward approach is typically faster than the backward approach, especially if there are a large number of potential features in the beginning. Training and evaluating a classification model, with a few features, is faster than training one with many features, however because neither approach is guaranteed to find a globally optimal result applying both approaches could identify two separate sets of features, even with the same learning approach and performance metric. Another drawback of this method is that like B&B it assumes a certain monotonicity of the classification function. At each step of SFS a single feature with the largest contribution to the classification accuracy is added or removed, depending on the approach. However, maximizing the addition of a set of k features instead of 1 feature at a time may find a different subset, A subset closer to the final optimal feature subset. Though as k is increased the number of evaluations at each step of the SFS process is increased exponentially.

Sequential floating selection tries to get around some of the limitations of SFS by alternating between the backward and forward steps of SFS (Pudil, Novovičová, and Kittler 1994). In sequential floating forward selection (SFFS), sequential feature selection is done in the forward direction until the set of features selection consists of three features or more. After that point after each new feature is added to the selected set of features SFS is done going backward on those selected set of features. If the set of features is reduced to 2 features or the performance stops or doesn't improve with the removal of any further features SFS in the forward direction is resumed. This is repeated until performance does not improve in the forward or backward direction. In sequential floating backward selection (SFBS) the process is reversed. SFS is done in the backwards direction starting with the full set of features and then alternates to sequential feature selection in the forward direction once two or more features have been removed from the full set. For SFFS and SFBS it is likely that some combination of features will need to be evaluated more than once so typically all performance results for every already evaluated combination of features is saved. SFFS and SFBS are likely to find sets of features closer to the global optimal and explore the space of all possible combinations of features in a manner more thoroughly than sequential feature selection. But they are still not guaranteed

to find a global optimum or even find the same solution. Because of the more exploratory search path they are also likely to take even longer to find a solution. Also, remembering all prior classification results can become memory intensive for very large feature sets. However, the more exploratory pattern of SFFS and SBFS means that more combinations of features will be evaluated and combinations of features that have close to the final optimal set of features may be found and identified.

Hybrid Methods

Typically feature selection and training a learning model are two separate steps. However, certain classes of learning methods contain within them steps to identify the most relevant features, while disregarding out the features that have little to no influence on the final result, often in the form of zeroing out the weight of those features. These methods can be classified as hybrid feature selection methods. Perhaps the most well-known hybrid method is the least absolute shrinkage and selection operator (LASSO) regression methods. LASSO is a regression method that tries to minimize the least squared error, while also minimizing the l1-norm of feature weights to be below a given threshold (Tibshirani 1996). The l1-norm term forces many feature weights to be zeros, effectively removing them from consideration. There are several methods that use an l1-norm regularizer to perform feature selection as part of a regression problem, such as BoLASSO, L1-SVM, and Elastic Net regularization. These form a class of hybrid algorithms based on sparse learning (Bach 2008; Efron et al. 2004; Zare et al. 2013; Zou and Hastie 2005).

Elastic Net regularization aims to extend the LASSO approach by calculating the l2-norm along with the l1-norm effectively combining LASSO and Ridge regularization (Zou and Hastie 2005). This method was developed to handle cases when the number of samples (N_s) is less than the number of features ($N_{f,}$). In the standard LASSO approach, only N_s features can be selected.

In the bootstrapped LASSO algorithm (BoLASSO) LASSO is applied multiple times to data (X,Y) that has been randomly selected with replacement from the pool of training samples (Bach 2008). By repeatedly apply LASSO to randomly selected sets of training data BoLASSO aims to improve the stability of LASSO by identifying features that are consistently selected in the bootstrapped sets of training data. In many feature selection algorithms when new data is added to a given training set the set of features selected often changes making feature selection algorithms less consistent. But with BoLASSO there is an assumption that if the original data that the features are measuring was sparse then there will be a set of features consistently selected in all or most of the feature selection results.

In the LASSO approach the l1-norm regularization term constrains the solution to ensure a level of sparsity, but the amount of sparsity can be scaled up or down through a scaler multiplier to the l1-norm (λ). LASSO solutions with a few features selected will have a larger lambda than solutions with a higher number of selected features. Also, LASSO solutions with more features selected will always contain the features selected when lambda was bigger and fewer features were selected (for the same piece of data). But there is no a priori way to specify lambda in order to ensure a pre-specified number of features are selected in the LASSO solution. However, through a small modification of the least angle regression (LARS) algorithm all LASSO solutions can be efficiently solved for every lambda value (Efron et al. 2004).

Using the LARS algorithm all LASSO solutions can be found for a given set of data, but there is no indication about how consistent those features are if the training set is varied. FeaLect tries to provide some guidance in this front by generating N random subsets from the total set of sample data, without replacement, and then applying LARS to each subset (Zare et al. 2013). Each feature is then scored for each subset based on if they were selected when only k features were selected, for all values of k. This then provides an expectation for each feature overall of the subsets about how likely it is that it will be one of the features selected when only k features are selected in a LASSO solution. Because there is no

thresholding in FeaLect every feature just receives a score and the number of features to be used in a solution can be selected at will. Providing a sparse and more stable set of features, where like BoLASSO it is expected that there is a consistently selected set of features overall iterations of FeaLect.

Decision Trees and Random Forests methods are popular classes of classification and regression algorithms that use a set of weak learners to construct a strong learner (Breiman 2001; Tin Kam Ho 1995). The selection and weighting of weak learners can also be thought of as a feature selection method as only a subset of the weak learners may be selected by the algorithm. A decision tree consists of a set of nodes, branches, and leaves where branches connect nodes to other nodes or terminate at a leaf node. At each node based on the value of a given feature (weak learner) a decision is made to travel down one of two of branches in order to make another decision at the next node. Each leaf node is associated with a class label (classification) or continuous values (regression), so when the path down a tree ends at a leaf the label or value is the answer. Random Forests are related to decision trees in much the same way that BoLASSO is related to LASSO. Random Forests are ensembles of decision trees created through bagging (bootstrapped aggregation). Multiple training sets are created from the full set of sample data through random sampling with replacement to create a set of decision trees where the final result is the mode (classification) or mean (regression) of the ensemble of decision trees. These ensembles of decisions are typically referred to as bagged trees. Random forests have an extra step that distinguishes them from bagged decision trees, because when the best feature is selected for a given decision node it is actually selected from a subset of the set of features rather than the full set.

Decision trees are able to select only the most important features to help make better decisions by pruning features that don't help to make the best decisions, but they are not constrained to prioritize sparse solutions. Specifically, many decisions trees and random forest methods will use a new feature for a decision node when a feature that is already being used at another decision point in the tree could suffice. Regularized trees and regularized random forests are two methods that aim to minimize the use of redundant features by prioritizing features already used in the tree over adding a new feature to the tree (Deng and Runger 2012).

From the standpoint of identifying multiple clusters of sparse features with minimal single points of failure (regularized) random forests present the best already existing approach to identify multiple sets of features with potentially equivalent accuracy. The bootstrapping of the data and the selection of features based on random subsets of features encourages the creation of decision trees that use mostly different features. However, each decision node selects a feature from a different random subset of features depending on the sampling behavior strongly correlated features may always eventually be added somewhere on every decision tree. A simple method around this is to create each decision tree within a random forest from a random subset of the features such that the random sampling of features at each decision node is already from a subsample of the full set of features.

Another issue with decision tress is that decision trees are really modeled as a collection of weak learners so a tree won't really be doing feature selection unless each weak learner only takes one feature as an input and that is done for all types of features. Alternatively, if the tree is trained over a large set of weak learners where the set of learners takes in every possible subset of features of size k over the full set of features, this would also be considered feature selection. However, the first approach limits the types of feature interactions that the tree can make decisions over and the second approach will quickly have a prohibitively large set of weak learners that need to be pruned.

Requirements of future battlefield capabilities

For the task for identifying sparse clusters of features within a volatile network of heterogeneous IoBT systems feature selection methods are probably the most useful type dimensionality reduction to identify subsets of sparse features from a larger set of features. However, feature selection assumes that some of the features from the larger set of features are already useful regarding the learning task. It is unable to resolve any issues or limitations within the features themselves. Feature extraction methods, however can create new features that are potentially more useful than the original features, assuming the underlying assumptions behind the specific feature extraction algorithm used is applicable to the situation. However, those underlying features are still there. Feature extraction is simply a recombination of those original features so within an IoBT network feature extraction methods have to applied in a way that considers the volatility of the network. Generally, feature extraction methods should be applied to features that are linked to each other, where if a feature in that group is lost or becomes unavailable in some way those other features in the group are also likely to have been lost at the same time. It also should be applied to a small number of features for similar reasons. Though, once a set of features have been selected feature extraction can be applied to the selected set of features to create a new set of features with less redundancy and potentially a smaller number of features or vice versa.

Within our survey of feature selection methods, we have shown that several approaches already can identify multiple sets of features for a given problem. BoLASSO, FeaLect, and random forests use random subset sampling, bootstrapping, and bagging, respectively, to create multiple training sets to identify and select different subsets of features. As such these methods are one way to identify multiple sets of features, however, there is little to no control over the variance in performance of these features when applied to the complete dataset or out of sample data. Also, BoLASSO, and FeaLect assume that certain features will always be selected within every permutation of the resampled data creating single points of failure for the ensemble of feature sets. If one of those consistently selected features becomes compromised or is no longer available all solutions become less accurate or may fail entirely. Also, FeaLect and BoLASSO are hybrid methods that assume a linear relationship between the target values and sample data, limiting the problems that they can used for. It should be noted that bootstrapping, random subset sampling, or cross-validation can be applied to any feature selection method in order to identify multiple subsets of features for a given dataset, though there remains no guarantee that one or more features will not be found in all of the feature selection solutions.

Sequential floating selection presents another method to identify multiple subsets of features though as a wrapper method it is computational expensive. By alternatively applying SFS in the forward and backward directions every time a feature is added or removed many combinations of features are likely to be identified along with their associated performance measures. Once a final feature subset is identified previously evaluated sets of features with lower, but still within a given tolerance of the final performance result, can be treated as alternative feature selection solutions, though in this case the number of shared features between all solutions is expected to be relatively large. Though, by using bootstrapping or random subset sampling multiple feature selection solutions may be identified.

None of the methods described within this paper are inherently optimal for the identification of multiple sparse clusters of features given the challenging constraints of IoBT networks. But feature selection methods like BoLASSO, FeaLect, random forests, and sequential floating selection are a first step towards identifying multiple subsets of features, but these methods as they exist do not guarantee that one or more features will not be present in all of the subsets. Random Forests and especially regularized random forests are the best current feature selection method to identify multiple sets of features and only require minor tweaking to ensure no feature is used in every decision tree. By restricting the creation of each decision

tree within a random forest to only a random subset of features a random forest, if large enough, can be built with no feature that spans all solutions.

Another simple method to find multiple subsets of features and ensure there are no shared features between selections, is to iteratively apply a feature selection method such that after a subset of features is selected the feature selection method is applied again to the remaining set of features that weren't selected. However, there is no guarantee that these alternative features will have a comparable predictive performance and they are likely to use a larger number of features. But these two or more sets of features are ensured to use mutually exclusive sets of features. mRMR and other redundancy feature selection methods provide a slightly more controllable method to select an alternative set of features with a controllable overlap of features across feature sets. Unselected features that have a redundant relationship with features that were selected are a good starting point to create an alternate set of features, while features that were selected, and have little redundancy to other features may need to be included in alternative feature sets in order to achieve a certain level of predictive quality.

Collecting, processing, and analyzing all of the relevant data from the available network of IoBT systems is important for the effective command and control of the battlespace. However, the hostile activity between coalition and adversarial forces makes the collection, processing, and interpretation of this data into actionable information challenging as each force uses the assets available to them in order to achieve their own mission while denying or attempting to deny the aims of their adversaries. Thus, it is expected that the future IoBT battlespace will have degraded communication, limited network bandwidth, and data sources whose trustworthiness, reliability, and value will vary over time. In spite of this Warfighters, analysts, and their commanders need trustworthy, reliable, and accurate information about the state of the battlespace and the state of the entities within it regardless of the constraints of the IoBT network. Doing this will increasingly utilize machine learning algorithms that can operate and quickly adapt to changing resources and mission goals, while operating on systems with limited computational power, memory, and bandwidth. But beyond just the need for capable machine learning algorithms is the need for dimensionality reduction methods that can select and identify relevant features for those machine learning algorithms to train on given the constraints of the IoBT environment. Due to the many dynamic constraints of IoBT networks, we have proposed the need for the identification and selection of multiple sparse clusters of features within a given IoBT network. These clusters need not consist of mutually exclusive sets of features, but no single feature should be found within all solutions (if possible). This is a unique requirement that dimensionality reduction methods have not historically been tasked with solving. In fact, several feature selection methods assume that some features exist in all if not most solutions (Bach 2008; Zare et al. 2013). In this paper, we have described several feature extraction and feature selection methods in order to review the operation of these approaches. The aim of this is to highlight how well these algorithms could serve as a first order solution to identify multiple sparse clusters of features and the ways in which they fall short. However, this document serves only as a broad sampling of the wide array of dimensionality reduction methods that have been developed (El Aboudi and Benhlima 2016; Li et al. 2017; Tang, Alelyani, and Liu 2014). The benefit of one method over another will vary depending on the processing, networking, and data constraints of a given situation, though there are some general broad tradeoffs to consider between the approaches that were highlighted above.

Reference:

El Aboudi, Naoual, and Laila Benhlima. 2016. "Review on Wrapper Feature Selection Approaches." *Proceedings - 2016 International Conference on Engineering and MIS, ICEMIS 2016*.

Arechiga, A., E. Barocio, J. J. Ayon, and H. A. Garcia-Baleon. 2017. “Comparison of Dimensionality Reduction Techniques for Clustering and Visualization of Load Profiles.” *2016 IEEE PES Transmission and Distribution Conference and Exposition-Latin America, PES T and D-LA 2016* (2): 1–6.

Bach, Francis R. 2008. “Bolasso.” In *Proceedings of the 25th International Conference on Machine Learning - ICML ’08*, New York, New York, USA: ACM Press, 33–40. http://arxiv.org/abs/0804.1302.

Breiman, Leo. 2001. “Random Forests.” *Machine Learning* 45(1): 5–32.

Brown, Gavin, Adam Pocock, Ming-Jie Zhao, and Mikel Lujan. 2012. “Conditional Likelihood Maximisation: A Unifying Framework for Mutual Information Feature Selection.” *Journal of Machine Learning Research* 13: 27–66. http://jmlr.csail.mit.edu/papers/v13/brown12a.html%5Cnhttp://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Conditional+Likelihood+Maximisation+:+A+Unifying+Framework+for+Information+Theoretic+Feature+Selection#0.

Deng, Houtao, and George Runger. 2012. “Feature Selection via Regularized Trees.” *Proceedings of the International Joint Conference on Neural Networks*.

Dennison, Mark et al. 2016. “Using Cardiovascular Features to Classify State Changes during Cooperation in a Simulated Bomb Defusal Task.” In *Physiologically Aware Virtual Agents Workshop at IVA*, Los Angeles, CA.

Efron, Bradley, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. 2004. “Least Angle Regression.” *The Annals of Statistics* 32(2): 407–99. http://projecteuclid.org/euclid.aos/1083178935.

Eskandari, Sadegh, and Emre Akbas. 2017. “Supervised Infinite Feature Selection.” : 1–10. http://arxiv.org/abs/1704.02665.

FISHER, R. A. 1936. “The Use of Multiple Measurements in Taxonomic Problems.” *Annals of Eugenics* 7(2): 179–88. http://doi.wiley.com/10.1111/j.1469-1809.1936.tb02137.x.

Gupta, Ashish, and Richard Bowden. 2011. “Evaluating Dimensionality Reduction Techniques for Visual Category Recognition Using Renyi Entropy.” (Eusipco): 913–17.

Guyon, Isabelle, and André Elisseeff. 2003. “An Introduction to Variable and Feature Selection.” *Journal of Machine Learning Research (JMLR)* 3(3): 1157–82.

He, Chuan, Zhe Dong, Ruifan LI, and Yixin Zhong. 2008. “Dimensionality Reduction for Text Using LLE.” In *International Conference on Natural Language Processing and Knowledge Engineering, *, 1–7.

Hira, Zena M., and Duncan F. Gillies. 2015. “A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data.” *Advances in Bioinformatics* 2015(1).

Joshi, S K, and S Machchhar. 2014. “An Evolution and Evaluation of Dimensionality Reduction Techniques-A Comparative Study.” *Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference on*: 1–5.

Khalid, Samina, Tehmina Khalil, and Shamila Nasreen. 2014. “A Survey of Feature Selection and Feature Extraction Techniques in Machine Learning.” *2014 Science and Information Conference*: 372–78. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6918213.

Kott, Alexander, Ananthram Swami, and Bruce J. West. 2016. “The Internet of Battle Things.” *IEEE Computer* 49.12: 70–75.

Lewis, David Dolan. 1992. “Feature Selection and Feature Extraction for Text Categorization.” In *Proceedings of the Workshop on Speech and Natural Language - HLT ’91*, Morristown, NJ, USA: Association for Computational Linguistics, 212. http://portal.acm.org/citation.cfm?doid=1075527.1075574.

Li, Jundong et al. 2017. “Feature Selection: A Data Perspective.” *ACM Computing Surveys* 50(6): 1–45. http://dl.acm.org/citation.cfm?doid=3161158.3136625.

Narendra, Patrenahalli M., and K. Fukunaga. 1977. “A Branch and Bound Algorithm for Feature Subset Selection.” *IEEE Transactions on Computers* C-26(9): 917–22. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1674939.

Novovičová, Jana, Petr Somol, Michal Haindl, and Pavel Pudil. 2007. “Conditional Mutual Information Based Feature Selection for Classification Task.” *Proceedings of the Congress on pattern recognition 12th Iberoamerican conference on Progress in pattern recognition, image analysis and applications*: 417–426. http://portal.acm.org/citation.cfm?id=1782964.

Pali, Vivek, Suchita Goswami, and Lalit P. Bhaiya. 2014. “An Extensive Survey on Feature Extraction Techniques for Facial Image Processing.” *2014 International Conference on Computational Intelligence and Communication Networks*: 142–48. http://ieeexplore.ieee.org/document/7065462/.

Peng, Huanchuan, Fuhui Long, and C. Ding. 2005. “Feature Selection Based on Mutual Information Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(8): 1226–38. http://ieeexplore.ieee.org/document/1453511/.

Pudil, P., J. Novovičová, and J. Kittler. 1994. “Floating Search Methods in Feature Selection.” *Pattern Recognition Letters* 15(11): 1119–25. http://linkinghub.elsevier.com/retrieve/pii/0167865594901279.

Roffo, Giorgio, Simone Melzi, and Marco Cristani. 2015. “Infinite Feature Selection.” *Proceedings of the IEEE International Conference on Computer Vision* 2015 Inter: 4202–10.

Saul, Lawrence K., and Sam T. Roweis. 2000. *Introduction to Locally Linear Embedding*.

Sharma, Nitika, and Kriti Saroha. 2015. “Study of Dimension Reduction Methodologies in Data Mining.” *International Conference on Computing, Communication and Automation, ICCCA 2015*: 133–37.

Tang, Jiliang, Salem Alelyani, and Huan Liu. 2014. “Feature Selection for Classification: A Review.” *Data Classification: Algorithms and Applications*: 37–64.

Tibshirani, Robert. 1996. “Regression Selection and Shrinkage via the Lasso.” *Journal of the Royal Statistical Society B* 58(1): 267–88. http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.7574.

Tin Kam Ho. 1995. “Random Decision Forests.” In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, IEEE Comput. Soc. Press, 278–82. https://ieeexplore.ieee.org/abstract/document/598994/.

Ventura, Dan. 2008. “Manifold Learning Examples – PCA , LLE and ISOMAP.” *October* (x): 1–9.

Vlachostergiou, Aggeliki et al. 2018. “Unfolding the External Behavior and Inner Affective State of Teammates through Ensemble Learning : Experimental Evidence from a Dyadic Team Corpus.” In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, Miyazaki, Japan, 192–97.

Yang, Howard Hua, and John Moody. 1999. “Feature Selection Based on Joint Mutual Information.” *Proceedings of International ICSC Symposium on Advances in Intelligent Data Analysis*: 22–25. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.41.4424&rep=rep1&type=pdf%5Cnpapers2://publication/uuid/E0508640-D7DF-41A9-BFA0-794AE77B3E0F.

Yang, Yiming, and Jan O. Pederson. 1997. “A Comparative Study of Feature Selection in Text Categorization.” In *ICML ’97, Proceedings of the Fourthteenth International Conference on Machine Learning*, Nashville, TN, 412–20.

Zare, Habil, Gholamreza Haffari, Arvind Gupta, and Ryan R. Brinkman. 2013. “Scoring Relevancy of Features Based on Combinatorial Analysis of Lasso with Application to Lymphoma Diagnosis.” *BMC Genomics* 14(Suppl 1): S14. http://www.biomedcentral.com/1471-2164/14/S1/S14.

Zheng, Kai-mei, Xu Qian, and Na An. 2010. “Supervised Non-Linear Dimensionality Reduction Techniques for Classification in Intrusion Detection.” *2010 International Conference on Artificial Intelligence and Computational Intelligence*: 438–42. http://ieeexplore.ieee.org/document/5655625/.

Zou, Hui, and Trevor Hastie. 2005. “Regularization and Variable Selection via the Elastic-Net.” *Journal of the Royal Statistical Society* 67(2): 301–20.