ARL-TR-8345 ● APR 2018

US Army Research Laboratory

Current and Future Applications of Machine Learning for the US Army

by Michael Lee, Ramakrishna Valisetty, Alexander Breuer, Kelly Kirk, Brian Panneton, and Scott Brown

ARL-TR-8345 ● APR 2018

US Army Research Laboratory

Current and Future Applications of Machine Learning for the US Army

by Michael Lee, Ramakrishna Valisetty, Alexander Breuer, Kelly Kirk, and Brian Panneton
Computational and Information Sciences Directorate, ARL

Scott Brown
Parsons, Baltimore, Maryland

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED (From - To)**
April 2018	Technical Report	January 2017–October 2017

4. TITLE AND SUBTITLE Current and Future Applications of Machine Learning for the US Army	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Michael Lee, Ramakrishna Valisetty, Alexander Breuer, Kelly Kirk, Brian Panneton, and Scott Brown	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) US Army Research Laboratory ATTN: RDRL-CIH-C Aberdeen Proving Ground, MD 21005	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TR-8345
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

Primary author’s email: <michael.s.lee131.civ@mail.mil>.

14. ABSTRACT

Recent major advances in the technology known as deep learning have reawakened global interest in machine learning and its potential to transform many technologies. In this report we review the different types of machine learning, outlining some of the places where it is currently used in Army research. Then we prognosticate where we see it applied to future Army research and operations. In doing so, we also identify gaps in current machine learning research that need to be filled for it to reach its full potential.

15. SUBJECT TERMS

machine learning, neural networks, supervised learning, unsupervised learning, artificial intelligence, automation

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Michael Lee
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER (Include area code)**
Unclassified	Unclassified	Unclassified	UU	64	410-278-5888

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

Acknowledgments

Special thanks to Dr E Chin, Dr B Henz, and Dr R Namburu for helpful discussions and Dr J Ramsey for careful reading of the manuscript. Funding was provided by the 731 Project for High-Performance Computing.

1. Introduction

Machine learning (ML), broadly defined, is a class of computer algorithms that automatically optimize parameters to process a given input and yield a desired output. A classic example of ML is linear regression whereby a line is found that optimally fits (passes through) a set of points. A more recent example of ML is a classification task such as labeling a million-pixel image with a single word like "cat".

For many applications, ML accomplishes the same tasks that a human could do just as well. However, ML shines in 2 cases: 1) when the number of tasks is unwieldy, say, in the millions, and/or 2) the dimensionality of the problem is beyond the understanding of the human mind. A simple example of a task that a human could do, but would be too difficult, is to simultaneously monitor thousands of security cameras in real time looking for suspicious behaviors. Perhaps an ML approach could spot anomalous events and share only those video clips with human watchers. Better yet, the anomalous images could be tentatively labeled with words such as "masked intruder at Entrance #1" to aid the security guard in only focusing on pertinent information.

In addition to reducing the burden for humans, ML can piece together complex interconnections that a human might not recognize. For example, an ML algorithm could detect that out of a million bank accounts, 5 of them seem to have transactions in sync with each other even though they are not sending or receiving money to each other or to a common third party.

Given ever-increasing computational resources for both handheld and stationary devices, it behooves us to imagine where ML can transform how wars are fought. Certainly ML is already having an impact on scientific research within the US Army, but one can also easily imagine operational applications such as autonomous vehicles and improved surveillance.

The primary goal of this document is to inspire personnel within the Army and Department of Defense to think about what could be possible with ML and what research investments may be fruitful to achieve those possibilities.

2. A Quick Tour of Machine Learning Algorithms

For the purposes of our discussion, ML methods can be roughly divided into 4 categories: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

2.1 Supervised Learning

In supervised learning, training data has labels that are considered true statements (a.k.a. ground truth). An example of labeled data would be a series of pictures of dogs and cats where each picture has a corresponding notation as "dog" or "cat". A machine learning algorithm, once trained, would attempt to determine the correct label based on just looking at the picture (i.e., pixel values). Many of the rapid advances in recent years for machine learning have resided in the realm of supervised learning.

One specific advance is deep neural networks (a.k.a. deep learning). Essentially, complex mathematical functions (i.e., artificial neural networks or ANNs, for short) are optimized (trained) to convert high-dimensional data (e.g., an image) into something as simple as a label. This would be an example of a classification task.

2.1.1 Decision Trees

Decision trees (DTs)[1] are a supervised learning method used for classification and regression. Earlier uses of DTs were in operations research and as analytical decision support tools. A DT appears more like an inverted tree. A decision process starts at the ground level and can reach, via different branches, any leaf that represents the final decision. However, in practice a DT works like a flow chart with many decision nodes and paths comprising the decision processes for arriving at the final decisions. Decision trees are thus multiclass classifiers. They can handle both real and binary data and are simple to interpret and visualize. DTs are used to give statistical interpretation by combining the probabilities along the decision paths and in the process are used to discover critical events. Other scenarios can be added easily and DTs can be combined.

Among drawbacks, DTs can become unstable even with small variations in the data where completely different DTs can emerge. They can become large and complex and are prone to overfitting. Furthermore, if some events dominate, the DTs can also become biased. The cost of using a DT is exponential in the number of decision points. ID3 is a popular algorithm and the decision events are chosen on the basis of maximum possible information gain.[2] Greedy algorithms are used with emphasis on local knowledge at the internal nodes to reduce cost.[3] Unfortunately, globally optimal DTs elude such a process. Still, multiple suboptimal DTs can be postulated and combined as classifiers in an ensemble learning.

Although a DT can be thought of as a navigator through a maze of decision events, DTs are put to use in designing complex industrial plant operation systems, aircraft navigation systems, self-driving cars, and so on. An example, cited in Russell and Norvig,[4] illustrates how an automated flight controller for a Cessna was designed and why it performed better than humans. Faced with the choice of designing from the unwieldy application of the first principles of flight controls, aerodynamics, blade propulsion, and so on, the designers turned to test pilots who put the plane through a set of maneuvers and mapped the results back to learn the science of flying. The flight control DT was extracted using the C4.5 system.[5]

2.1.2 Bayesian Learning

In Bayesian Learning (BL), the most probable hypothesis, h, is sought given data, D, and some domain knowledge.[6] The familiar Bayesian theorem,

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}, \tag{1}$$

gives the maximum a posteriori hypothesis and is difficult to use. Most often the maximum likelihood hypothesis, $P\ (D|h)$, is used and sought assuming a uniform prior, $P\ (h)$. If the error in the known output space is Gaussian, then the likelihood hypothesis assures an error minimization in the sense of the sum of the squared error (similar to linear regression). Therefore, conventional back propagation, gradient descent methods, and even regularization efforts to control the variance in neural nets are seen as a particular application of BL.

Because BL deals in probabilities, computing confidence levels is easy for both the regression and classification outputs. In fact, BL is applied to develop the neural networks (NNs) and study their results in the model space (M) addressing the following issues in a rigorous manner:

- the distribution of weights given the data (D) is analyzed over a set of models (M), $P\ (w|D, M)$,
- the distribution of outputs given the data (D) is analyzed over a set of models (M), $P\ (y|D, M)$,
- and the distribution of models given the data, $P\ (M|D)$.

The maximum a posteriori hypothesis is also interpreted using information theory as a sum of 2 lengths:

- a length representing the miscalculation error in the model, and
- the length representing complexity in the NN model (or size of the hypothesis).

There are many proposed algorithms called "Minimum Description Length" to handle the tradeoff in the model prediction error and model complexity.[7]

BL methods are applied in medical research and molecular biology where data is sparse. This may be because of the experimental protocols. Although useful for assessing many aspects of the ML algorithms, BL methods are difficult to apply directly and are combined with simulation-based Monte Carlo–like techniques. In general problems, noise is not always Gaussian. Other probability distributions are complex to arrive at and numerical integration over a large number of input variable space or parameter space becomes difficult. Researchers are using Monte Carlo simulations to overcome this difficulty.

2.1.3 Bayesian Inference and Belief Networks

Bayesian inference was used first[8] in Bayesian networks (BNs) to arrive at the probability for a final decision. The architecture for a BN looks a lot like that of any NN, but it does not work quite in the same way as in an NN. BNs are edges in a Bayesian probability-informed DT. BNs are pruned by invoking the inference from the conditional probability independence, drastically removing many decision events that feed into the decision path. As one traverses along the path of the decision events, a probability dependence is implied on the prior events but not a physical dependence. Because of the pruning done, there is always a finite but low probability for a negative decision. In practice, many decision events are not binary.

Probability inference can jump over decision events, like C depends on B and B depends on A but C also depends on A (directly), that is, not sequentially. Also, new conditional events can feed into the decision events partway on the BN decision paths. So BN graphs are topological and acyclical. BNs are therefore called graphical models or belief networks.[9] While a strong prior knowledge is a must for the construction of BNs, it also alleviates the difficulty of overcoming sparsity of knowledge on some decision events. BNs are used to dynamically update the probability distribution as new evidence comes in.

BNs are implemented via hidden Markov models (HMMs) in speech recognition and text processing. Central to the development of HMMs is the assumption that the transition probability for the next transient state in a Markov chain depends only upon the current state and not the previous ones. Discarding prior dependencies makes predicting the future state difficult, but the training is continued under the assumption that the statistical nature of the state remains time invariant. A description of HMMs can be found in Jurafsky and Martin.[10] BNs are now used in many other fields as well, such as engineering, physical sciences, medicine, sports, and law.

2.1.4 Naïve Bayes

While BNs give the probability for a final event using the conditional probability of the many prior events, a reverse application of the Bayesian principle is used with great success in classification problems via Naïve Bayes. Assuming all attributes in the final event are independent of each other, then the reverse application allows for an updating of the classification (maximum a posteriori value) for the prior event given a new sample of the attributes' values.

It appears very simple and widely used in consumer enterprises and social network enterprises; however, because probabilities for some attributes can become zero, updating can become tenuous at times. The individual attribute probabilities must be smoothed out to avoid making a null classification. Naïve Bayes started originally in text retrieval.[11] It is now also used in automated medical diagnosis systems.

2.1.5 Regression

While linear regression leads to quantitative outcomes (e.g., fitting a line through a set of points), logistic regression is used for binary classification. The former method can be made to work for the multiclass classification by channeling the quantitative outcome into selected ranges.

There are 2 issues in applying linear regression in ML. The first one is overfitting, which occurs when the number of features drop down to 10 or less per weight from the hypothesis. The second one is the computation, which when the number of features runs into millions can become challenging. To overcome these issues, regularization techniques have been developed which, in addition to reducing the model prediction error, also seek to reduce the numerical values of the computed coefficients. Added to the cost function in a Ridge regression is a penalty equal to the sum of the square magnitude of the coefficients (L2-regularization), and, in a Lasso regression, the sum of the absolute values of the coefficients (L1-regularization.)

Coefficient shrinkage prevents overfitting in Ridge regression. Because there is a possibility for some coefficients to drop down to zero, Lasso regression also allows for a sparse hypotheses by enabling some feature dropping. These regressions are preferred in ML communities over stepwise regression techniques to identify the feature space in the hypothesis proposals because the otherwise combinatorial choice of relevant parameters is automatically obtained from optimization.

2.1.6 Similarity Learning

Similarity learning (a.k.a. distance metric learning) is a straightforward ML approach that classifies input by its closeness to previously classified objects. Simplicity is the method's strong point; however, if there are too many database objects the method can become slow. Ideas to speed up the method include dimensionality reduction (described in Section 2.5.2), sparsification, and hashing.[12]

2.2 Information Theory

Although information theory started in the 1940s from the seminal work of Shannon as a way to minimize noise in communications,[13] today the theory finds wide applications in machine learning, genetics, neurobiology, particle physics, statistics, and so on. Even though it is widely used for achieving lossless JPEG-like compression, the theory gives fundamentally correct abstractions not only for comparing communication or data streams, but even the belief systems themselves.

Traditionally a machine learning algorithm is said to be properly trained if overfitting is eliminated (by reaching a balance between the bias and variance in the predictions.) But this is unsatisfactory in many fields; for example, in the medical field the false negatives are sought to be minimized and the true positives are sought to be maximized. So the basic question is what benefit is it to have a 90% correct prediction versus an 85% one? This brings to the fore the concepts of information theory into machine learning and the issues such as 1) Is the machine learning algorithm actually working? and 2) Can it be improved further?

These issues are examined not in relationship to the algorithmic details, but rather in relation to how the predictions are used further down the line. As quoted in Hu, ". . . learning is an entropy-decreasing process and pattern recognition is 'a quest for minimum entropy'. The principle behind entropy criteria is to transform disordered data into ordered one (or pattern). . ."[14]

Furthermore, using the entropy concept from the information theory[13] and joint and marginal distributions of the predicted and target results, Hu proposed the following learning measures: 1) joint information, 2) mutual information, 3) conditional entropy, 4) cross entropy, and 5) Kullback-Leibler divergence to probe the issues of similarity and symmetry instead of the traditional empirical learning criteria (such as an error rate, an error bound, a cost measure, a classification margin, etc.). Some of the information theory informed machine learning algorithms[15] are proposed as follows:

1) Information theoretic clustering

    a) Mutual information criterion

b) Information bottleneck method

c) Information theoretic co-clustering

2) Information theoretic semi-supervised learning

a) Entropy-based approaches

b) Information rate-based approach

3) Information theoretic feature selection

a) Mutual information-based feature selection

b) Maximum-relevance minimum-redundancy

c) Joint mutual information

d) Information fragments

e) Conditional mutual information maximization

f) Other information theoretic measures

4) Information theoretic metric learning

a) Minimizing an information theoretic distance measure

b) Coding length-based approach

c) Application in information retrieval

d) Factor graph

2.3 Graph-Based Machine Learning

Graph-based ML is a semi-supervised learning used for understanding how groups form in various domains such as social networks, biological clusters, and brain networks. A class of data that maps into a graph of clusters showing dense connections within the clusters and sparse connections between the clusters is best served by graph-based ML.

Deploying unsupervised clustering on "big data" problems is made difficult by the fact that the number of optimal clusters is unknown, clusters can dynamically form and unform, there is uncertain variance in the data samples, and the challenge in coming up with a cost function to describe the situation.

Most data such as video, image, text, and social are often unlabeled or multilabeled. For example, many semi-supervised learning tasks deal with data points that can naturally belong to multiple labels (e.g., an image with a mountain can be labeled under “adventure” and “west”). Therefore, correlations exist among the multiple labels that the algorithms have to deal with.

Data are often mapped into a graph of nodes and edges, such that the nodes correspond to labeled and unlabeled data points, and the edges reflect the similarities between data points. Formal solutions to these graphs are intractable because graph properties and spatial relations are not available to begin with.

Graphs come in all sizes and shapes, and can be combined from multiple sources and from multiple types of data representations (e.g., image pixels, object categories, and chat response messages). Graph-based semi-supervised learning is deployed to seek a function to describe the graph with the following properties: 1) it should be close to the given labels on the labeled examples, 2) it should be smooth on the whole graph, and 3) it should be consistent with the label correlations.[16] Algorithms such as agglomerative clustering require knowledge of first-degree neighbors and incremental merging of nodes. Factors like a cluster population, how some nodes are widely connected within a cluster, and how some nodes have external connections to other clusters, help to incrementally optimize the cluster graph.

2.4 Nonparametric Machine Learning Algorithms

Parameters are the weights in a machine learning algorithm, and a machine learning algorithm is called a parametric algorithm if the number of weights used in it is fixed up front. Therefore, a linear curve used to fit a dataset may be termed a 2-parameter machine learning algorithm. Examples of parametric algorithms in machine learning are NNs, Naïve Bayes, logistic regression, linear discriminant analysis, and so on. Although these models have the benefit of being simpler in scope and speedier in results delivery, they suffer from oversimplifying assumptions and poor fit as more data becomes available over time. Algorithms such as k-nearest neighbors, decision trees such as CART and C4.5, and radial basis function (RBF) kernel support vector machines (SVM) do not make any functional mapping assumptions to label the data and belong to the class of machine learning algorithms called “nonparametric”.[17,18]

2.4.1 Kernel Function Methods and Support Vector Machines

Support vector machines use a linear kernel and thus belong to a class of kernel function methods that are used to separate a 2-way labeled N-dimensional dataset into 2, separated from each other by the largest margin possible using an (N −1)-dimensional hyper plane.[19,20] For nonlinear classification, a kernel trick is used that maps a given dataset into an even higher dimensional space to achieve clearer classification. SVMs do not directly provide probability estimates; these are calculated using 5-fold cross-validations, which are expensive. SVMs lead to overlapping target classes when the dataset has more noise. Let n equal the number of features and m, the number of samples, then, the following recommendations are made:

- for $n > m$, use logistic regression or SVM without a kernel (linear kernel),
- for $n \approx m$, use SVM with Gaussian kernel,
- for $n < m$, introduce more features and use logistic regression or SVM without a kernel, and
- for $n >> m$, SVMs lead to poor predictions.

SVMs are used as classifiers but may also be used for regression and anomaly detection. SVMs are used in many fields as follows:

- Display advertising, image-based gender detection, content-based image retrieval, large-scale image classification, image segmentation systems, facial expression classification,
- Handwritten characters recognition, text and hypertext categorization, texture classification,
- Protein-fold and remote homology detection, protein classification, human splice site recognition, identification of alternative exons and chemotherapy effect on survival rate,
- Generalized predictive control (SVM-based) method to the problem of controlling chaotic dynamics in plants with small parameter perturbations, dynamic reconstruction of chaotic systems from interspike intervals using least squares SVMs,
- Inverse geosounding problem, seismic liquefaction potential, underground cable temperature, and land cover classifier,
- Data classification using SVM,
- SVM and decision-tree modeling,
- Personal recommendation system for news websites,
- Intrusion detection and detecting steganography in digital images,
- Particle and quark-flavor identification in high-energy physics, and

- Object detection and 3-D object recognition.

2.4.2 Ensemble Bagging, Boosting, and Stacking

Many classifiers such as naïve Bayes, logistic regression, and shallow decision trees are weak learners. They are low variance type and do not overfit or high bias type, therefore, cannot easily learn hard learning problems. However, because they do well on parts of the input feature map, taken together, a bunch of weak classifiers can do a better job overall than a single classifier. The challenge here is to select classifiers suitable for different parts of the input features space and then tally the votes of the different classifiers. While this addresses the divide and conquer approach to an input map, other issues remain such as data fusion, confidence estimation for the outputs, if all the statistical information from the input map is thoroughly wrung out or not, and the ever-present issue of reduction of computational cost. These issues are discussed at length by Dietterich.[21] Ensemble methods seek improved outcomes, often using a number of ML models. These models are often proposed with slight architecture variations to the same ML algorithm that is at task, but this is done in a manner that ensures a model-related statistical interpretation for the outcomes. Ensemble methods are used for spam filtering. Boosting and bagging ensemble methods include AdaBoost, gradient tree boosting, and XGBoost.

2.4.3 Boosting

Boosting is implemented in 2 stages. In the first stage, subsets of the original data are created that were known to contain features prone to misclassification. In the second stage, a series of weak classifiers are deployed and their results are combined using a weighted majority vote-based cost function. The classifiers are sequentially deployed with each classifier receiving improved outcomes from the previous classifiers in an iterative manner. The implementation resembles a logistic regression. The loss functions are replaced by the product of the hypotheses of the underlying classifiers and the confidence levels in their classifications. Gradient descent method is used to obtain a better overall classification by incrementally improving upon the votes in the subclassifiers.

2.4.4 Bagging

Bagging, which stands for bootstrap aggregating, seeks to decrease the variance in the prediction.[22] Some samples in the training dataset are linearly combined and added back to the training dataset. This approach allows one to tweak the already expected classification and improve the stability and accuracy of machine learning algorithms used in statistical classification and regression.

2.4.5 Stacking

In stacking, several models are applied to the bootstrapped samples of the training data to identify the specific portions of the input data for which different models have difficulties predicting the desired outcome.[23] The outputs of these models are then used to train a Tier-2-type classifier to correct the misses in the first set of outcomes. A logistic regression is employed on the subclassifiers to arrive at the final classification.

2.4.6 Instance-Based Learning

Instance-based learning (IBL) (also called memory-based learning, lazy learning, and case-based learning)[24,25] covers a family of algorithms that do not strive to do ML on each and every new data sample. The algorithms instead rely on memory. Some earlier instances of data samples and outputs are stored in memory, and on the new instance of a data sample, they rely on a comparison of the new sample with the stored samples to come up with a prediction. Algorithms like IBL are used to predict on a new data sample by computing the distances or similarities between this instance and the stored instances and by averaging some selected k-nearest neighbors. Locally weighted linear regression is another algorithm and RBF network is another implementation. Naturally, the computational complexity of classifying a sample becomes O(N) where N is the number of samples stored in the memory. Thus, the advantages for the IBL depends upon the data domain, data size, and noise in the data.

These networks also evolve and adapt by having some old samples replaced by the new ones if new results are deemed better, but at the risk of introducing some drift over the time in the model. Examples of IBL are the k-nearest neighbor algorithm, kernel machines, and RBF networks. IBL is used as a second-opinion diagnostic tool in the medical field for knowledge discovery. Most IBL methods work only for real inputs and, unlike DTs and Bayes classifiers, do not need a training phase. IBL is nonparametric, that is, it has no prior model assumptions.

2.4.7 Computational Learning Theory

Machine learning is a form of inductive learning. The learning depends upon the previously learned outcomes to label the new data samples. While any learning algorithm seeks to learn as fast as possible and with as few misses as possible, there remains the issues of uniqueness of the algorithm, time of learning, and feasibility of learning. These are the issues studied under computational learning theory (CLT).

For computational theorists, an algorithm is feasible if the learning by it is done in a polynomial time of computation, that is, $O(N^k)$, where N is the problem size and k is some polynomial power. These algorithms are said to belong to the polynomial time class or simply "P". The other kind are said to belong to the class of "NP" or nondeterministic polynomial time class. Essentially, if an algorithm belongs to a "P" class, its learning can be verified in polynomial time, but if it belongs to the "NP" class, then its learning will be hard to verify.

Many approaches are proposed using different data agglomeration techniques to augment the limited available datasets and inference principles such as variations of probability theory (frequency based, Bayesian, etc.). Specific CLT approaches include exact learning, probably approximately correct (PAC) learning, Vapnik-Chervonenkis (VC) theory, Bayesian inference, algorithmic learning theory, and online machine learning. CLT led to many practical algorithms, for example, PAC theory inspired boosting, VC theory led to SVMs, and Bayesian inference led to belief networks.

2.4.8 Artificial Neural Networks: A Versatile Strategy Born of Simplified Neuroscience

An artificial neural network (ANN) is a brain-inspired model to process information. The first ANN was created by Warren McCulloch and Walter Pitts in 1943. It was a very simplistic model resulting in logic functions such as "a or b" and "a and b".[26] In this section of the report, we discuss a variety of ANNs and what their potential use cases are. These ANNs include deep learning networks, convolutional neural networks, recurrent neural networks, and autoencoders.

Feed Forward Neural Networks The algorithms for the simple regression and classification problems are easily implemented with a feed forward neural network (FFNN) architecture in which stacked layers of neurons (or compute nodes) are assumed. This architecture is also called a multilayer perceptron.[27]

A data sample is input to the neurons in the left-most layer and the results of the FFNN are extracted from the neurons in the right-most layer. Connections are not assumed among neurons belonging to any one layer, but neurons belonging to adjacent layers are fully connected. Information coming into a neuron from neurons on the left side via these connections is amplified via weights, and the set of weights for the entire FFNN is called the parameter space. These parameters are initially set to a set of random values and are updated as the FFNN is updated in the learning process. The amplified input is presented to the activation functions at the neurons and outputs generated are fed to the neurons in the layer to the right. Depending upon the overall problem, known logistic or regression functions are selected as activation functions for the neurons. All the functions in the entire network are said to belong to what is called a feature map. As the output travels to the neurons in the right-most layer, the results are compared with the expected values in a supervised learning and an error or a cost function is computed. This function is minimized by either propagating the error to the neurons in the layers to the left via the back propagation algorithm[28] or via stochastic gradient descent methods.[29] After this step, the randomly initialized parameters get updated and the learning continues by presenting another data sample to the network.

Even though the computational cost in an FFNN depends on the number of layers and the number of neurons in it, the learning objective repeatedly leads to the selection of the architecture. More than the architecture, the learning process selected for the FFNNs is frequently more important. Since FFNNs often try to extract results from datasets that are statistical in nature, the learning process should guard against introducing unnecessary bias and variance in the results. The FFNNs are trained, guarding against this issue by carefully selected training and testing protocols, and by trying to minimize a second error called the cross-validation error. Unlike in numerical physical simulations in which the convergence is sought by a monotonic decrease in some error, learning in FFNNs are evaluated against the bias error and variance error to eliminate an overfit of the data.

As topics in ML increased from text recognition, speech processing, video processing, sequence modeling, threat detection, threat posture, anomaly detection, and so on, input data is processed to present only the salient features to the FFNNs. The FFNNs themselves are given new architectures by making/unmaking new neural connections. Information and feedback is held in memories at the level of the neurons. The memory is used to make new decisions in the learning process. The NNs described in the remainder of this section highlight these features.

Hopfield Networks Hopfield networks (HNs) emulate the associative memory function of a human brain.[30] HNs are trained to learn one or more patterns. Given a new data sample approximating an already learned pattern, the HN is able to recollect the correct pattern. The network is able to do this even when the new sample is corrupted with noise or even if some connections in it are broken. As noted in Russell and Norvig 1975[4] (p. 571), if an HN is trained on a set of photographs, then afterwards, the HN will recognize every photo even if a piece of one of the photographs is presented. The network is able to do this, not by storing the original set of photographs in its memory, but by having the weights trained on the original set alone. HN can be used to recognize or classify features from text, voice, and images that are already trained into its memory. HNs are also used to solve combinatorial optimization problems such as the "traveling salesperson problem".

HN is an ergodic network because any node in an HN can be reached from any other node directly. All nodes are initially input. The nodes exist in 2 states: "fired" or "not fired". All nodes are fully interconnected to each other and output from a node is routed to all other nodes as input, so firing of one node can set off different patterns. The sign function is used as an activation function with the activation levels set to be either +1 or –1. If there are N nodes, there will be N^2 weights and the training can get computationally expensive. Unlike a human brain, which is able to recollect a whole image from only a few stored features, an HN seeks to remember every pixel input to it. To reduce the dimensions and cost of training, the input may be transformed into principle component analysis (PCA)-based feature extractors. An HN can store up to 0.15 N images, where N is the number of neurons in the network.

Boltzmann Machines Boltzmann machines (BMs) are similar to HNs in that they have an extra layer or a group of hidden nodes that are never shown any of the initial input.[31] Neurons are in a binary state and output from one is fed to all others as input. Learning starts with random weights, but unlike in an HN where learning evolves deterministically, learning in a BM continues stochastically using probability-based contrastive divergence using Markov chains. BMs are inspired by the Boltzmann distribution often found in real physical systems. BMs undergo state transitions that resemble a simulated annealing search for the configuration that best approximates the training set. Because firing of neurons occurs in a nondeterministic manner, the network will not settle in one stable state and a probability distribution of activation patterns can be ascertained. BMs are trained to recognize gray images or probability distributions. It is reported in Russel and Norvig[4] (p. 596) that BMs are a special case of belief networks.[32]

Convolutional Neural Networks Convolutions are familiar in physical sciences. For example, a time series is convoluted with a kernel in Fourier transform (FT) to obtain its frequency content. Similarly, in the realm of ML, convolutions are employed to reduce the input datasets to obtain feature maps that are fed into ML algorithms. Thus, convolutional neural networks (CNNs), or deep CNNs, are used for image processing but can also be used for text and speech processing. CNNs have a long history starting with the observation that the visual cortex in the brain responds to small overlapping regions in the visual field before a final image is recognized. After many attempts, the modern implementation of CNN started emerging from the seminal work of LeCun et al.[33]

Before an image's pixel values are input to the ML classifiers, the input is reduced through a series of convolutions and pooling operations. The convolution step tries to extract features such as lines or edges from the input image, that is, the spatial relationships in the input images. For this purpose, small chunks of an input pixel matrix are convoluted (i.e., dot product) with a selected convolution filter, which is a matrix much smaller in size than the input matrix. The pixel dataset is usually large, for example, there are 10,000 pixels in a 100 × 100 pixel image. A convolution filter with a very small size, say 5 × 5 pixels, is used to convolute with equal-sized chunks of the input matrix. As the convolution operations are performed, sliding from left to right and top to bottom with a stride on the input matrix, what emerges are a matrix of values, called a feature map, which is slightly smaller in size than the input matrix. This feature map is repeatedly subjected to convolutions with different filters to arrive at the final feature map that is much smaller in size than the original matrix.

In between the convolution operations, the feature maps are downsampled with yet another step called pooling. Similar to the convolution operations, a small chunk of the matrix of a feature map is selected and the most dominant information from that chunk is determined using pooling types such as maximum, average, summation, and softmax. This operation is completed again by sliding across the feature map with a stride. Nonlinearity is often introduced into the feature map via an operation called ReLU (rectified linear unit), which selects only the positive values in the matrices.

This final feature map is input to one or more fully connected layers leading to classifiers. The neural network classifiers for CNNs are designed to exploit the local nature of the features in the image, meaning that a feature such as an edge somewhere in an image is not necessarily the same as another edge somewhere else in the image. This spatial locality is assured by not allowing connections from a neuron in one layer to all the neurons in the next layer but only to a few.

Deep Learning CNNs' ability in learning feature representations from large datasets have been generalized as priors and used to obtain body part classifiers and pose regressors in sequence modeling, object detection, and pose and intent recognition. Intent recognition requires, in addition, establishing a deeper understanding of the interplay between the identified smaller objects in an image like, for example, among an elbow, a baseball mitt, and the baseball. Deep architectures are proposed with hidden layers, containing smaller NNs, in parallel for these problems. Researchers have trained a cascade of regression-based CNNs for human pose estimation and combined those using weak spatial NNs in deep learning architectures. Correspondence becomes an issue for these NNs, which is established in some NNs via the "intra-class alignment", that is, alignment of parts identified within a class, and "key point identification", an issue which is learning the intent of parts within a class are important to overall learning.

Deep learning affords the ability to create complex manifolds with hierarchical structure.[34] The issues in designing a CNN are types of convolution filters, when to introduce nonlinearity, pooling types, and the neural net local connections. A great many deep learning CNNs have been developed since the 1990s. However, beginning in 2012, the following nets have perked up interest in the ML community: AlexNet,[35] ZFNet,[36] VGGNet,[37] GoogLeNet,[38] ResNets,[39] and DenseNet.[40]

Generative Adversarial Networks Generative adversarial networks (GANs) or deconvolution GANs are a class of networks developed in the last few years.[41] These networks learn from an input dataset of images representing an object class. Later, the network is able to reproduce or reconstruct images typical of that object class without ever needing any of the images from the dataset as input. In essence, the network has learned the object class and is able to reconstruct images that can easily pass for those in the class. Instead of a complete image, a brush stroke like a sketch is input to generate real-looking objects.

Learning in these networks uses a game between 2 adversaries: a generator network that tries to generate realistic objects, and a discriminator network that attempts to identify if it came from an input from the object class or from the generative model. As the game concludes, the generator reproduces the feature distribution of the object class so exactly that the discriminator network is unable to differentiate the generated object from the real one. Both parts of the network are usually trained using stochastic gradient descent with exact gradients computed by maximum likelihood.

The tug between the generator network and the discriminator network makes it, although not a GAN, a reinforced learner too. GANs are extended beyond images to video streams and robot behaviors, but their true calling is in reconstructing super-resolution high-definition images. Super-resolution GANs are proposed to recover realistic textures and fine-grained details from images that have been heavily downsampled.

Long Short-Term Memory Networks, Gated Recurrent Units Long short-term memory (LSTM) networks are a special type of recurrent neural networks (RNNs) that solve the so-called "vanishing gradient problem". When the input dataset is large, the tendency is to propose a network with many hidden layers and neurons. In training such a network, the gradient can become vanishing/exploding at some neurons. The LSTM algorithm was developed[42] mainly for solving the vanishing gradient problem. Each neuron has a memory cell and 3 gates: input, output, and forget. The input gate determines how much of the information from the previous layer gets stored in the current neuron. The output gate determines how much of the next layer gets in on the current neuron. The forget gate altogether enables dropping of the current neuron to overcome bottlenecks in the learning process. This long-term memory capturing feature of the neurons is what enables the LSTMs their sequence modeling capability. They are used successfully in many fields, especially when data are sequential, for example, language processing, speech recognition, machine translation, image captioning, video classification, and even bioinformatics. By combining with CNN-enabled priors, LSTMs are used for generating captions for the images.

Recently, gated recurrent units (GRUs) were developed and are similar to LSTM.[43] The difference is that the GRUs have one less gate and are wired separately. For each neuron, they have an update gate that determines how much information to keep from the last state, and how much information to let in from the previous layer, and a reset gate that is wired differently. They always send out their full state, and they do not have an output gate. Another difference and simplification is that GRUs do not have a memory (cell) state. The memory is associated with the state from previous steps. GRUs are easier to train and less expensive than LSTMs.

Bidirectional LSTMS and GRUs Memory in neurons in the LSTMs[44] and GRUs is stored from the past states. Unlike this situation, neurons in the bidirectional LSTMs and bidirectional GRUs use information from the future, like in autofilling a text, and updates the neurons on the backward pass. So instead of advancing on features such as on an edge, these bidirectional LSTMS and GRUs do things like filling in a hole.

Using Wavelets to Preprocess Input Just as an FT is used to extract the frequency content in a time series, wavelets are constructed as kernel functions for convolution with not only time signals, but also images. Unlike the kernel in the FT that extends from negative to positive infinity, kernels are selected for wavelets to be active within specific temporal or spatial windows to extract local features. Thus, wavelets readily provide both time and frequency information[45] and start appearing in signal processing and signal compression applications.

Once convolutions started appearing in CNNs, wavelets became an attractive choice for building the feature libraries for the images as well. For natural images, Olhausen and Field[46] showed that the most common image features allow for sparse linear representations by a redundant dictionary of basis functions that resemble Gabor wavelets. By redundant, it is meant that the number of basis functions available exceeds the pixel count in the images. This allows for more stable representations for the common image features, which can be represented by a few nonzero coefficients irrespective of the locality of the features in the image, and are also invariant with respect to translation, magnification, and rotation. The dictionaries are learned by minimizing the feature rebuild error with a sparsity-inducing penalty. Learning algorithms are developed that go by the name "Sparse Frame" for implementing this approach[47] under Defense Advanced Research Projects Agency (DARPA) funding. The dictionaries are useful for tasks such as image recovery, image classification, image compression, image reconstruction via super-resolution, and biomedical imaging (MRI and tomography).

The time localization extraction ability of the wavelets is also exploited for denoising human brain signals and accurately identifying the spikes to input to brain activity classifiers.[48]

2.5 Unsupervised Learning

In unsupervised learning, the input data is unlabeled (i.e., there are no ground truths to train against). Imagine that in the example of image classification, the "dog" and "cat" labels are missing, and all that is available is a randomly assorted series of dog and cat pictures. What can a computer do with this information? For one thing, it can assume that it is being given a bunch of images of N classes of objects and it needs to sort those images into one of those N classes. Even the variable N (denoting the number of classes) could be an unknown quantity. So, the computer simply sees a series of images and tries to bin them on their similarities and differences of each picture to each other. The method described here is loosely called "clustering" and is one of the primary classes of methods in unsupervised learning. Another major class of unsupervised learning algorithms involves converting high-dimensional data (e.g., the pixel values of an image) into a lower-dimensional manifold (e.g., a small set of classifiers).

2.5.1 Clustering

Clustering is an unsupervised learning approach to finding similar subsets of data. There are 2 traditional types of clustering algorithms: k-means and hierarchical.

K-means The k-means clustering method aims to find k number of clusters for a given set of multidimensional points, where the variable, k, is given by the user. It is best used when the data has compact groups of data, rather than long-stretched-out groups.

Hierarchical Hierarchical clustering sequentially aggregates groups of points together until there are essentially a few groups comprising all of the data. With further heuristic measures, the number of clusters can be ascertained from the data rather than being specified by the user.

2.5.2 Manifold Learning

Manifold learning is also an important form unsupervised learning, whereby high-dimensional data (e.g., a million pixels of an image) is converted to low-dimensional data (e.g., a small vector of numbers describing the mathematic traits of the image). Traditionally, input data is considered as a set of N-dimensional vectors and output is a set of M-dimension vectors, where M can be as small as 2 to 3.

Dimensionality Reduction Dimensionality reduction (DR) is a transformation of high-dimensional data into a lower dimensional space. Manifold learning is the automated process of achieving dimensionality reduction, of which there are many methods.

PCA A common DR algorithm in this regard is principal component analysis (PCA), whereby the most important dimensions are extracted. Mathematically, this corresponds to obtaining the largest eigenvalues and eigenvectors of the covariance matrix between the input vectors. In layman's terms, PCA asks, "What are the most distinguishing characteristics of a group of objects?" and then plotting the objects along those dimensions only. The distinguishing (i.e., first) component could be a mixture of qualities. For example, in describing human body shapes the most differentiating component could be the sum of the height and weight of the person.

kPCA To make PCA more generalizable to a wider class of problems, it might make sense to transform one or more of an input vector's elements prior to computing the covariance matrix. In this case, we are modifying the kernel and hence the term, kernel-PCA, or kPCA.[49]

Isomap While it is outside the scope of this document to consider every manifold learning algorithm out there, it is worth mentioning a few that potentially offer superior solutions for certain datasets. Isomap[50] uses neighborhood clustering to build graphs and measure connective distances. This permits nonlinear manifolds to emerge as principal components.

Sparse Dictionary Learning Sparse dictionary learning is an unsupervised method of feature extraction. Imagine we divided up a series of images into 8 × 8 blocks of pixels. The most common 8 × 8 blocks would fill out a dictionary of features common among the collection of images. This dictionary could be used, for example, to come up with a compression scheme, whereby the images could be written as a combination of common dictionary blocks. Popular methods include orthogonal matching pursuit.

2.5.3 Autoencoders

Advances in supervised deep learning can be transferred to the field of unsupervised learning using a special ANN called an autoencoder (AE).[51,52] Essentially, an autoencoder optimizes a function of input data that maps its output to the input data (a.k.a. an identity function). What distinguishes AEs from the identity function is that the network layers are bottlenecked so that the data is forced to be compressed in some way. Types of bottlenecks include 1) using a smaller number of nodes in the middle, 2) imposing sparsity in the middle nodes, and 3) imposing some kind of constraint on the middle node's weights (e.g., L2-norm). This ensures that given enough training data, there is no way that the bottleneck layer of the AE NN will end up being simply the identity operator. Instead, the AE becomes a lower dimensional representation of a higher dimensional manifold (namely, the input data).

AEs are primarily used to encode, that is, to compress, a dataset and recreate it back.[51,53] AEs are also used to extract many features in a dataset and use such extracted features as priors in a convolutional NN. AEs are often symmetric with respect to the layered nature of the network. The hidden layers are designed with progressively reduced nodes to the middle layer of the network. AEs are trained to predict the input. Sparse AEs, variational AEs, and denoising AEs are some variations of the AEs. HNs and BMs are simple classifiers and so are the AEs, but their purpose is to identify specific objects in large datasets, like a cat in a photo.

Once a good feature representation is given, a supervised learning algorithm can do well. But what happens if there are too many features/objects in an input, and if their meaning changes out of sequence? In domains such as computer vision and speech and natural language processing, these issues are apparent. Because priors/ features help, and there are an abundance of such priors/features in the natural world, the question becomes, "Are there algorithms that can automatically learn feature representations and improve upon them in subsequent iterations?" Sparse AEs[54] do surprisingly well in this regard. Unlike in AEs, the number of neurons increase hidden layer by hidden layer as one moves into the center of the network. The network is still symmetric between the input layer and the output layer. The input is encoded in more neurons at the center. On the back pass from the output side, instead of passing the input, it is spiked with some noise that forces some neurons to drop out, thus the term sparse, and, thus the ability to code more features comes into play.

2.5.4 Variational AEs

Variational AEs employ the same network structure as regular AEs, but they learn an approximated probability distribution in the input data.[55] They employ Bayesian inference and independence information to rule out some dependence in the features in the data and drop connections between neurons in the neighboring layers in the learning process.

2.5.5 Denoising AEs

Denoising AEs are AEs for which the input data is fed with some noise but require the AE to still learn the original input and reproduce it without noise.[56] This makes the network learn the broader details in the input sample while the smaller details are drowned out by the noise and become difficult to learn.

2.5.6 General Applications of UL

While unsupervised learning (UL) methods may not be as advanced as supervised learning (SL) approaches, the simple fact is that most data out there, especially as regards Army research, is unlabeled. Thus, the question becomes, "How can unlabeled data be learned?" One common idea is determining to what extent different input features are correlated with each other (a.k.a. regression.) This could affect the experimenter's choice of which data to regularly record and which to discard. Another aspect of identified correlations is that it implies a similar underlying principle for those features. For example, 2 microphones placed on opposite sides of a vehicle picking up the same frequency hum may indicate a similar source point of that signal that might be triangulated based on phase delays and relative amplitudes (similar to the way our hearing localizes sound sources). Another value of UL, as alluded to earlier, is grouping of similar data packets, leading to labeling. Suppose we collected a series of animal images and found that a cluster of them had similar properties. This cluster could then be labeled by a human (or intelligent agent) as "cat". Moreover, UL can be used to deduce connectivity (i.e., graphs/networks/trees). For example, the series of animal pictures could group large and small animals, and under large animals, find 2 subclusters of elephants and lions. Finally, UL can be used to parse source signals from mixed input (e.g., the cocktail party problem, whereby we want to extract a voice from the din via 2 or more microphones).

2.6 Semi-Supervised Learning

We have been introduced to both supervised learning and unsupervised learning. It is natural to ask the question, "Is it possible to improve the performance of a supervised learner if one can provide additional data, even if they are unlabeled?" Semi-supervised learning[57] is an attempt to answer this question in the positive. In short, semi-supervised learning attempts to solve the same kind of problem as supervised learning—predict the labels of unseen data, but attempts to exploit any unlabeled data that may exist in addition to labeled data. Incorporating unlabeled data is important, since it is frequently the case that can easily access unlabeled data with which one can augment one's labeled data.

Including unlabeled data can improve the accuracy of classification or regression, but there are a few key assumptions that must be met for semi-supervised learning to be applicable:

- The label function $f(x)$—that is, the function we are trying to learn—is smooth in regions in which we have a high density of sample points. This results in $|x - y| \leq \epsilon \Leftrightarrow f(x) = f(y)$ for some small for classification problems.

- If one forms clusters with a distance metric $d(x, y)$, then points that belong to the same cluster are likely to have the same class. Equivalently, the separation boundaries between classes must lie in a low density region.
- The data lie in a low-dimensional manifold, even if it is embedded in a high-dimensional space.

These assumptions arise from the typical approaches to semi-supervised learning. Assuming the label function is smooth allows one to infer class labels onto unlabeled points from nearby labeled neighbors; the motivation for the cluster assumption is similar. The manifold assumption arises from the curse of dimensionality: as the dimensionality rises, pairwise distances become more similar—and therefore less useful for discrimination—unless the data lies in a low-dimensional manifold.

2.7 Reinforcement Learning

Reinforcement learning may be described simply as "learning what to do"[58]—that is, a learning agent is placed in an environment with the ability to make observations, perform actions, and measure rewards. The goal of the learner is to maximize its reward for its actions, and it is to learn how to do that through trial-and-error search of its environment. One should notice that the concept of reward is defined loosely; rewards may be immediate or delayed.

Reinforcement learning may be conceptualized as an approach to characterizing learning problems. In this conceptualization, reinforcement learning is somewhat distinct from both SL and UL, though both approaches bear some similarity to reinforcement learning. Perhaps the most important distinction is that SL and UL are concerned with determining the best categories for data objects, but do not consider how that categorization fits into a larger problem of how to *act*. On the other hand, reinforcement learning explicitly treats the problem of choosing actions to maximize rewards. Thus, a reinforcement learning problem may contain subproblems that resemble SL or UL.

A conundrum that arises in learning how to choose actions is the balancing of exploration and exploitation. Maximizing rewards requires exploitation of solutions from experience that have provided good rewards in the past; nevertheless, exploiting past solutions precludes learning new solutions. A learner could easily become trapped in a locally optimal solution if it does not explore the solution space to discover new approaches. This balancing of exploration and exploitation is not typically considered in classical supervised learning approaches.

3. Currently Available Software and Tools for Machine Learning

1) Caffe[59] supports many different types of deep learning architectures (CNN, RNN, LSTM, and fully-connected) and is geared toward image classification and image segmentation. It also supports graphics processing unit (GPU)-based acceleration using the CuDNN library from Nvidia.

2) Deeplearning4j[60] is a deep learning programming library written for Java with wide support for deep learning algorithms. These algorithms all include distributed parallel versions that integrate with Apache Hadoop and Spark.

3) TensorFlow[61] is a library for machine learning across a range of tasks. It was originally developed by Google to meet their needs for systems capable of building and training NNs to detect and decipher patterns and correlations, analogous to the learning and reasoning, which humans use.

4) Theano[62] is a numerical computation library for Python, where computations are expressed using a NumPy syntax and compiled to run efficiently on either CPU or GPU architectures.

5) Keras[63] is a library that contains numerous implementations of commonly used NN building blocks such as layers, objectives, activation functions, optimizers, and a selection of tools to facilitate working with image and text data. It is essentially a front end to Deeplearning4j, Tensorflow, or Theano.

6) Microsoft Cognitive Toolkit[64] is a deep learning framework developed by Microsoft Research. Microsoft Cognitive Toolkit describes NNs as a series of computational steps via a directed graph.

7) MXNet[65] is a scalable deep learning framework used to train and deploy deep NNs. It can be used with multiple languages including C++, Python, Julia, MATLAB, JavaScript, Go, R, Scala, Perl, and Wolfram.

8) Scikit-learn[66] is a Python module with a variety of unsupervised and supervised learning approaches.

9) Torch[67] is a scientific computing framework with support for machine learning algorithms, with primary emphasis on using GPUs. It has a convenient scripting language, LuaJIT, and an underlying C/CUDA implementation. PyTorch extends Torch capabilities to Python.

10) Dlib-ML[68] is a C++ toolkit containing machine learning algorithms.

11) Chainer[69] is a Python-based deep learning framework that includes automatic differentiation application programming interfaces (APIs) based on dynamic computational graphs as well as object-oriented high-level APIs to build and train NNs.

12) Neon[70] is Intel Nervana's reference deep learning framework, similar in ease of use to Keras.

4. Potential ML-Enabled Army-Relevant Applications Encountered in Our Lab during First Year of Study

This report is a product of our project from the fiscal year 2017. The other related deliverable is the analysis of how current machine learning tools can be applied to various US Army Research Laboratory (ARL) and Army-relevant problems. While Sections 4.1 through 4.10 are only a minuscule representative of the potential applications, they hint to the wide reach that ML may be able to impact Army research and operations.

4.1 Assessment of Planetary Gear Health

In collaboration with Dr Adrian Hood (ARL/Vehicle Technology Directorate), we are trying to identify the progression of damage in helicopter transmission gears by observing accelerometer signals. One of the main challenges of condition-based maintenance of vehicles (air and ground) is how to convert sensor signals (e.g., accelerometers and microphones) into information about the current health of each part/system. The current state of the art is usually to apply an FT and then sum up the peaks to create a metric of vibration.[71] The next steps for analysis might be to use convolutional filters to better decompose the raw signal. Furthermore, it is an open question whether deep learning can be used to correlate the hierarchical frequency/temporal nature of the vibrations to known damage states.

4.2 Assessment of Fatigue and Cracking in Vibrating Load Tests (Maneuver Sciences Campaign)

Pitch/catch ultrasound is a nondestructive method that can localize crack formation in materials. While there are already physics-motivated estimation techniques,[72] we were curious if the output signals could be directly fed into a neural network and correlated with emerging crack length. The POC for this research is Dr Robert Haynes (ARL/Vehicle Technology Directorate).

4.3 Automated 3-D Tissue/Organ Segmentation from CT Scans for Soldier Protection

Medical images such as CT (computed tomography) scans and MRIs (magnetic resonance images) can be obtained fairly rapidly, but the subsequent analysis and conversion into useful information is a bottleneck. For example, a group at ARL (POC: Dr Sikhanda Satapathy/WMRD) uses 3-D organ models segmented from CT scans to simulate, via finite elements, the effects of various ballistics and loads. Obtaining the 3-D segmented tissues via CT scans is a laborious process, taking roughly 24 man-hours. We propose that unsupervised and/or supervised learning could accelerate this task without sacrificing the accuracy obtained by an expert modeler.

Automatic segmentation for recreating 3-D representations of biological data from 2-D scans has been pursued in the medical community for some time. Utilizing an unsupervised clustering approach to solve this problem, a goal would be to create a system that can take in an image sequence from something like a CT or MRI dataset and generate a 3-D representation of the data using automatic segmentation. In this case the subject will focus on segmentation of medical images; however, this method could theoretically apply to any scan datasets for use in reproducing an accurate 3-D representation. One of the primary goals of this project is separation of each tissue type. The primary types of tissues include soft tissue such as skin, organs, brain tissue, and bone, including the skull.

The first attempt at achieving this is comparing what is possible using just image thresholding alone to remove any data above a particular threshold value. This works well to pull out just the bone alone. However, thresholding allows any other objects to remain behind that may be in a similar range of values, including part of the machine. With models generated by clustering, it is possible to separate the bone from the other objects in the image. Currently, the clustering model we are using (DBSCAN) includes some noise that needs to be cleaned up. This is a future problem that this project is working to correct. Another problem that is introduced by thresholding is that any material that is located inside of another object is not possible to separate using only thresholding. For example, brain tissue would be lost if done only using thresholding. By utilizing clustering methods, the brain tissue or any soft tissue can be separated from the skull; however, there is still the issue of noise being included in the final model.

The clustering method used here affords an additional benefit of providing each cluster as a separate 3-D model. This allows for each cluster to be viewed to see each different part of the model that was generated. Using something like thresholding will only provide a single solution according to the initial parameters. Clustering and thresholding, together, could be the best “unsupervised” solution. Supervised learning, with sufficient ground truth data, may be the ultimate, most robust solution.

4.4 Machine Learning for Armor Mechanics Problems

Better understanding of armor mechanics allows for lighter and more efficient protection of Army personnel and equipment. Machine learning is applicable both for discovery of armor mechanics at high rates, and for optimization of protection packages. These 2 problems are intertwined. Unsupervised learning can detect interesting behaviors of materials from empirical data; for example, the change in penetration as one transitions from ballistic rates to hypervelocity rates, or nonhomogeneous material properties in rolled homogeneous armor steel of sufficient thickness. Supervised learning could show the independent variables that best predict protection to effect better armor designs, and can even be used to automatically optimize a protection package given a set of constraints.

4.5 Automated Optical, Thermal, and Acoustic Monitoring of the Additive Manufacturing Process

Additive manufacturing (AM) (e.g., 3-D printing) has the potential to improve sustainment by providing replacement parts more quickly than the traditional logistics chain. The current limitation of AM is durability and reliability of the printed part. ML could be used to monitor the layer-by-layer build process and detect problems before it is too late to fix them. Reinforcement learning could then be used to apply the appropriate fix before continuing the programmed deposition.

4.6 Supervised and Unsupervised Learning of Soldier Personnel Databases

The Army Study to Assess Risk and Resilience in Servicemembers (STARRS) program is an ongoing study to understand, in part, the factors related to suicide.[73] As such, the data collected hold a treasure trove of information that may be of interest to Army leaders regarding Readiness and Sustainment. To fully tap into the data will require ML to find the deep connections between various factors.

4.7 Automated First-Pass Analysis of Video Streaming Data

Data analysts can only process so much data in a given time period. As the "flash flood" of data increases exponentially over time, ML-enabled processes will be needed to ferret out significant from nonsignificant data. In the case of video streams, an ML-based tool could be used to select only frames or intervals where certain desired objects are identified. As long as false negatives are low, this should greatly ease the burden of human operators without the risk of overlooking important data.

4.8 Evaluation of Human-Annotated Maintenance Reports Toward Sensor-Based Anomaly Detection in Vehicles

Currently acquired data in the field is likely incomplete in being able to detect when unplanned maintenance events will occur in Army systems. The question is whether (with current data) we can predict when problems will likely occur, before they actually occur, to improve readiness.

4.9 Use of ML to Assess Whether Specific CPU Processes Are Malicious or Friendly

Cybersecurity is an increasing concern for the Army, especially as it relates to the unique environments that it must endure; specifically, the contested electromagnetic spectrum and constant targeted assault from the adversary. Toward this end, we think that the latest tools in ML, such as RNNs and deep reinforcement learning, will help correctly detect and ameliorate intrusive threats that may not be easily detectable by traditional pattern recognition. Furthermore, we foresee reasoning processes developed with deep reinforcement learning will ease the burden of human cyber defense agents.

4.10 Use of ML as a Mechanism for Information Dispersal in a Contested Environment

Traditional network coding uses linear transformation to divide, distribute, and disperse information from sender to receiver. We believe that it may be possible to use nonlinear transformations derived from ML to divide, encrypt, and compress data for reduced bandwidth environments while improving data integrity.

5. ARL Research Using Machine Learning

Machine learning is either currently being used, or could be used, in many research projects at the ARL. Using data collected from the posters presented at the November 2016 ARL Open Campus Open House (see Appendix), we list some of the research projects that either use ML or might be able to benefit from it. Our list of ML-related ARL research efforts is by no means complete.

6. Army Operational Applications

While machine learning has technically been around since the early 19th century with the invention of linear regression by Gauss, we believe that the newest advances in ML will impact the Army in ways we cannot currently imagine. In this section, we outline the many areas of Army operations that we think will be enhanced and what kinds of ML methods might be employed.

6.1 Military Intelligence

Military intelligence encompasses information gathering and analysis as it pertains to what commanders need to make the best decisions. Processing must be automated as ever larger amounts of data are collected. The main problems to consider are the volume, velocity, veracity, and variety of data. Large volume (a.k.a. big data) requires smart distribution of the data over many compute nodes. Velocity requires fast computing and networking connected to the data streams. Veracity is a question of trust in the source of the information and anomaly detection. Variety amounts to the application of different trained models using many different ML algorithms. We outline the different types of data and analysis requirements in this subsection.

6.1.1 Natural Language Processing

There are big benefits to having computers distill out important concepts and sections of text from large databases of text gleaned from various media sources. Another recently reported ML breakthrough is accurate text translation between different languages.[74] A challenge unique to the Army is translating from languages that are not common, and therefore have fewer professional translators. In the realm of artificial general intelligence (AGI), it is professed by some groups that natural language processing will be a foundation of human-like cognition.[75]

6.1.2 Data Mining

Given the proliferation of data generated by humans, sensors, and agents, a big question is what residual value that data contains beyond the immediate use justifying its collection. Data mining can be both a statistical and machine learning effort to find patterns in the data that otherwise would have been missed by human operators.[76]

6.1.3 Anomaly Detection

Traditionally, anomaly detection is performed by first identifying clusters of known data and characterizing the distribution that the data falls under. Then, as new inputs are processed, they are either identified as falling into or outside of the original distributions. If they are outside of the known distributions, they are considered anomalies. Many of the following types of anomaly detection systems could be useful to the Army:

- Cyber intrusion detection: network traffic that is out of the ordinary. McPAD and PAYL[77] are 2 such examples of software currently in use that use anomaly detection.
- Pattern of life anomalies: visuals and biometrics of people acting in ways different from the norm, suggesting that they may be performing some adversarial action.
- Condition-based maintenance: signals that are not typical for the material/ system at its age in current lifecycle.
- Soldier anomalies: reasons to believe soldier biometrics are out of the ordinary.
- Foreign item detection: visuals of objects not recognized in a database of known materiel.

6.2 Autonomy

6.2.1 Automated Target Recognition

Automated target recognition (ATR) is a very mature field that has been using machine learning for decades.[76,78–84] Some relevant questions going forward are as follows:

1) To what extent will current advances in deep learning enhance ATR?
2) Will more sophisticated algorithms require more complex/power-hungry onboard computing?
3) Can ML be robust against various deceptive obfuscations of the target?
4) To what extent could reinforcement learning be used to make real-time trajectory adjustments?

6.2.2 Robotics

The use of machine learning in robotics is also such a vast field as to require a document unto itself. The areas where ML continues to make sense include sensing, navigation, locomotion, and decision-making. Sensing, at present, will benefit from all of the advances in computer vision. Navigation, besides use of standard GPS, could benefit from egomotion,[85] that is, motion estimation based upon its own perceptions. Locomotion could be learned, not programmed, which would lead to not only faster development times, but also the ability to rehabituate under new environments or damaged modalities (e.g., losing 1 of 4 legs). Finally, as the number of robots exceeds the number of human operators, it will be necessary for robots to make decisions on their own on how to carry out their defined missions. It will have to make calls such as, "Do I go back to home base because battery is low?" or "Do I continue onward a little and then self-destruct?"

6.2.3 Self-Healing

Besides robotics, it is ultimately desired to have any system correct itself when damaged or not working at full capacity. This requires intelligence at some level to autonomously diagnose deficiencies and problems and rectify those issues with the resources available to it.

6.2.4 Ethics

To the extent that autonomy is learned through machine learning, the question will be, "How will the autonomous system respond to situation X?" The problem here is with a system that has potentially lethal force, how can we be sure that it will only use its force correctly and lawfully?[84] We surmise that there will have to be extensive testing of a machine-learned algorithm before it will possess the actual ability to use lethal force, even if it is tied with human-in-the-loop decision-making.

6.3 Training Intelligent Agents through Playing Games

A flurry of research in recent years has been looking at using machine learning to autonomously play various video games. In some cases the reported algorithm now exceeds human game playing. In other cases there are still challenges dealing with long-term memory. For the US Air Force, intelligent agents have been successfully trained on combat-centric flight simulators that closely mimic real life.[86] The questions for the Army include the following:

- Can intelligent agents be attached to robotic platforms?
- To what extent can intelligence be general enough to deal with the diverse set of situations encountered in real life versus a video game?
- Can we trust the action of a trained agent when we may not understand its logic?
- To what extent will an agent be able to work with a human?

6.4 Cybersecurity

Machine learning has played an integral role in cybersecurity over the last decade.[13,16,77,87–91] Specifically, ML can be used for anomaly detection, detecting specific patterns indicative of known threats, and discerning network behavior as potentially being produced by malicious agents. As the field continues to intensify, the question will be whether ML will keep security one step ahead of the adversary who may use ML to obfuscate detection.[92]

6.5 Prognostic and Structural Health Monitoring

A long-term vision is that every mechanical system in use by the Army will have some amount of internal sensing regarding the current and projected health of the system. The relevant questions are as follows:

- Can we discern the current health of a system or system component from a limited number of sensors?
- Can onboard ML predict the health of a system or system component after exposure to a specific environmental or ballistic insult?

6.6 Health/Bioinformatics

6.6.1 Sequence Mining

As the number of genome sequences continues to grow exponentially, the computational effort required to compare sequences obtained in the field may become unmanageable. Machine learning can reduce the necessary comparisons by classifying the sequence at various levels of taxonomy.

6.6.2 Medical Diagnosis

Artificial intelligence has long held the promise of transforming medicine.[93] In recent years, machine learning is already making great strides in detecting malignancies in various tissues.[94] It could just as well be used to describe traumatic injury or post-traumatic stress disorder (PTSD)[95] with a plan of treatment.

6.7 Analysis

A significant component of the Army focuses on the analysis of operations, systems, and research and testing. Traditionally, analysts use a large swath of tools, including machine learning, in the form of multidimensional regression, clustering, and dimension reduction. With the emergence of deep learning, a new set of tools should be possible that allow for more efficient processing of larger datasets that require more sophisticated models. For example, it should be possible to extract features and physical properties from video streams taken during a test that might exceed current standard practices.

6.8 Other Uses for Machine Learning

- Adaptive User Interfaces (AUIs) and Affective Computing: ML could be used to determine the mental and/or emotional state of the user and offer up an interface suitable to that state. In addition, variable AUIs could serve variations in users. For example, some users might prefer audio feedback versus visual feedback.

- Recommender Systems: One of the most popular recommendation systems is the one that chooses the next movie that a user wants to watch based on ratings from previously watched movies (e.g., the so-called "Netflix problem"). For Army purposes, recommendations for logistics resupply could be made based on feedback from previous usage and inventory accounting.

- Search Engines/Information Retrieval: Traditionally, search engines return document "hits." The new paradigm is to answer the user's question in a concise form rather than simple pattern matching.
- Sentiment Analysis: Traffic on both social media and various sensors trained on environments could detect not just critical keywords or the presence of specific objects, but also deduce the likelihood of a possible attack.
- Tailored Propaganda: Traditionally done by dispersing leaflets, propaganda these days can be distributed through social media. The ML angle is how to target propaganda to the right demographics with the most convincing message. Also, it is important to quickly detect and subvert propaganda from adversaries targeted to our own personnel/people.

7. Research Gaps in Machine Learning

One of the goals of this study is to identify gaps in current research that could limit the full potential of ML for use both in Army research and operations. This section borrows from the strategic planning work of ARL Campaign Scientists Dr Brian Henz and Dr Tien Pham (unpublished).

7.1 How to Fit Army Data/Questions into Current Methods

Traditionally, half the battle in employing ML to a particular domain is figuring out how to adapt available tools and algorithms. This is more acute for a lot of the problems that the Army faces that might be unique compared to other academic, commercial, or governmental uses. The first problem that any data analyst faces is adapting the data to the statistical or ML model they want to use. Not all data uses continuous variables or is a time-series. Discrete/labeled data can be very tricky to manage since the labels may not easily be converted into something mathematical. An example of this in natural language processing is how words are often converted into high dimensional one-hot vectors. Another example might be how to convert large amounts of maintenance reports into predictions about how a particular vehicle will fare over time.

In addition, Army requirements go beyond the typical commercial sector use in terms of needing to detect not just objects and people, but also their intent and posture. This will require the development of new models. Another big requirement is explainability, as outlined by a recent DARPA program: what were the factors that led an ML algorithm to make a specific decision? In a real-life event, if an ML algorithm were to proclaim the presence of an important target without human verification, could we trust that determination?[96]

7.2 High-Performance Computing

As computationally demanding ML tasks are envisioned, developers are using multithreaded, parallel, and heterogeneous architectures (GPU, many-core) to speed up calculations. Distributed implementations of ML are far less common than GPU versions because of the inherent network bottlenecks associated with internode communication in distributed computing and the substantial advantage of GPUs versus CPUs in terms of single precision floating point performance. Besides a strong current reliance on GPUs, bio-inspired neural computing aims to find non-von Neumann architectures to perform ML more efficiently and potentially faster. An example of this is the IBM neuromorphic chip.[97] Future research should focus on how to distribute ML processing such that network communication is minimized between nodes. Also, to what extent can unsupervised learning algorithms like clustering can be mapped to neural networks?

Other things to consider:

- Current ML software (specific neural networks) performs best with a small cluster of GPUs.
- Most nonneural network-based ML algorithms are not highly parallel or not parallel at all.
- Another Army specific challenge is analyzing largely unlabeled datasets (e.g., with unsupervised learning). Manually labeling clusters would be a form of semi-supervised learning.

7.3 Unique Size, Weight, Power, Time, and Network Constraints

With travel into remote areas or any area far from a friendly base, the Army must limit the size, weight, and power of systems. Furthermore, in the “heat of battle,” time is critical. For example, one cannot wait for an operational simulation to finish while they are being fired at. Finally, network bandwidth can be highly constrained in regions where other commercial transmitters dominate, or in situations where limiting radio communication improves stealth.

In this multiply constrained environment, machine learning will need to be performed efficiently and often in an isolated fashion. The diametric opposite condition would be training a large neural network using a large data repository, which is often the case for state-of-the-art machine learning feats. The commercial sector is developing self-driving vehicles, which will presumably use low-power computational devices (e.g., field-programmable gate arrays, mobile GPU) for autonomous driving, road/obstacle detection, and navigation. However, the Army will have a lot more requirements including autonomous sensors and actuators, situational awareness/understanding, communication/ cooperation with humans, and a wide range of battlefield devices. This will require several factors more computing power and algorithm-specific hardware for optimal miniaturization and low power consumption.[98]

7.4 Training/Evaluating Models with Cluttered or Deceptive Data

Operational environments are expected to have higher than usual density of static and dynamic objects in a chaotic environment. Furthermore, one fully expects active deception to avoid notice. We also want to be able to develop algorithms that are robust enough to at least be aware of deception and dial down their certainty estimates accordingly.

7.5 Training a Model with Small and Sparse Data

Breakthroughs in CNN-based target classification can be attributed, in part, to the availability of thousands of examples of each object class. In Army scenarios there may be limited data for certain people and objects. One ultimately will need one-shot[99] or multishot classifiers where a few representative data entries are sufficient to learn a new class. The best option so far is "knowledge transfer", by which new classes are learned by tweaking a subset of all of the parameters of previously trained models. The idea is that with fewer parameters to optimize, less data would be required to modify those parameters.

7.6 Training Models Specifically for Army-Relevant Targets

Even for object classes that we can generate plenty of imagery (e.g., friendly objects), we need to train our own models to recognize Army-relevant classes from potentially thousands of images per class. The Army also uses other sensing modalities not typically found in commercial vehicles (e.g., thermal and radar). Thus, models need to be trained for these atypical sensing devices. Fundamentally, atypical sensing devices may require novel neural network topologies for optimal accuracy and compactness.

7.7 Incorporating Physics in Reasoning

One interesting area worth pursuing is combining model and simulation with machine learning. There are many ways this could be done. For example, ML can be used to derive the starting parameters for a simulation. In addition, ML can be used to process the output of simulations. An intriguing new area is developing physics-based or physics-like simulation that uses ML-like models/equations. One such application would be predicting "What if?" scenarios. For example, "What if I run over this tree? What will happen next?"

7.8 Soft Artificial Intelligence

Machine learning is traditionally thought of as hard (i.e., mathematical) manifestations of artificial intelligence. It is possible that eventually, all AI tasks will be reduced to mathematics. For now, however, some intelligent tasks appear to be more reasoning- or emotion-based. For tasks in the previously described methods, ML does not adequately address the following soft AI characteristics.

7.8.1 Human-like Reasoning

Humans do not always reason completely logically, but they also have the ability to piece together incomplete information and make "best guess" decisions. Encoding this behavior has been a challenge for a several decades.[100]

7.8.2 Emotions

Emotions appear to be motivation/objective functions that drive humans to certain ends. For example, happiness may lead to inactivity or a pursuit of productive creativity. Fear, on the other hand, may lead to holding back. Do computers need emotions to operate more effectively or are they better off having 100% objectivity? This is both a philosophical question and a future research direction. For now though, there is no question that in the context of human-agent teaming, computers will need to accurately interpret human emotion to achieve the best group outcome.[101]

7.8.3 Social Communication

Interactivity with humans is a foremost concern for Army research going forward. A similar issue is how communication will occur between different computer systems that are not necessarily designed by the same laboratory. One area of research has been using computers to teach social communication in people who have difficulties in this area.[102] Once again, for human-agent teaming, agents will need to be able to participate in social interactions and follow social norms in the company of humans.

7.8.4 Creativity

Creativity is often thought of as a random merging of ideas combined with novel elements whereby a discrimination function decides on the functionality and/or aesthetics of the newly created items. In some ways, creativity is already being demonstrated by certain computer laboratories. For example, computers can be imbued with certain aspects of creativity for the purpose of design.[103]

7.8.5 General Intelligence

The ultimate goal of AI is the merging of many narrow intelligence algorithms into a unified intelligence, much like a human mind.[75] It is likely that even early so-called artificial general intelligences (AGI) will have some superhuman abilities given that many narrow AI tasks are already better than human for certain tasks. One major goal of AGI is to automate certain tasks currently performed by humans.

7.8.6 Artificial Super Intelligence

A machine learning study would not be complete without mentioning the speculation of many philosophers that machine learning will eventually be able to improve its own programming leading to an exponential improvement in capability, perhaps far exceeding human intelligence. These visions are both utopian[104] and dystopian.[105] The hope is that super intelligence will solve many of the world's current problems.

8. Conclusion

In this work we reviewed the different classes of machine learning and described some of the more commonly used methods. We then noted a small subset of examples of how ML is being used at ARL. Finally, we prognosticated where ML could be applied to various Army domains in the future and outlined some of the challenges that need to be addressed to achieve this outcome. We hope that this document will inspire future researchers and decision makers to continue to invest in research and development to fully utilize ML to help advance the US Army.

9. References

1. Howard RA, Matheson JE. Influence diagrams. Decis Anal. 2005;2(3):127–143.
2. Quinlan JR. Induction of decision trees. Mach Learn. 1986;1(1):81–106.
3. Safavian SR, Landgrebe D. A survey of decision tree classifier methodology. IEEE T Syst Man Cyb. 1991;21(3):660–674.
4. Russell S, Norvig P. Artificial intelligence: a modern approach. 2nd ed. Englewood Cliffs (NJ): Prentice Hall; 1995. p. 1–75.
5. Quinlan JR. Combining instance-based and model-based learning. In: Machine Learning. Proceedings of the Tenth International Conference; 1993 June 27–29; University of Massachusetts, Amherst (MA): Morgan Kaufmann; c1993. p. 236– 243.
6. Heckerman D, Geiger D, Chickering DM. Learning Bayesian networks: the combination of knowledge and statistical data. Mach Learn. 1995;20(3):197–243.
7. Barron A, Rissanen J, Yu B. The minimum description length principle in coding and modeling. IEEE T Inform Theory. 1998;44(6):2743–2760.
8. Wright S. Correlation and causation. J Agri Res. 1921;20(7):557–585.
9. Good IJ. A causal calculus (I). Br J Philos Sci. 1961;11(44):305–318.
10. Jurafsky D, Martin JH. Speech and language processing. 2nd ed. Prentice Hall Series in Artificial Intelligence. Upper Saddle River (NJ): Pearson Prentice Hall; 2008.
11. Maron ME, Kuhns JL. On relevance, probabilistic indexing and information retrieval. J ACM. 1960;7(3):216–244.
12. Bellet A, Habrard A, Sebban M. A survey on metric learning for feature vectors and structured data. CoRR. 2013;abs/1306.6709.
13. Shannon CE. A mathematical theory of communication. ACM SIGMOBILE Mob Comput Commun Rev. 2001;5(1):3–55.
14. Hu BG. Information theory and its relation to machine learning. In: Deng Z, Li H, editors. CIAC'15. Proceedings of the 2015 Chinese Intelligent Automation Conference; Fuzhou (China). Lecture notes in electrical engineering, vol 336; Springer Berlin Heidelberg; c2015. p. 1–11. doi:10.1007/978-3-662-46469-4_1.

15. Wei X. Information theory and machine learning. University of Illinois at Chicago. [accessed 2018 Mar 29]. https://pdfs.semanticscholar.org/d6d9/b285738560963810bf15c68210a21b05de23.pdf.

16. Zha ZJ, Mei T, Wang J, Wang Z, Hua XS. Graph-based semi-supervised learning with multiple labels. J Vis Commun Image R. 2009;20(2):97–103.

17. Raschka S. Machine learning FAQ. [accessed 2018 Mar 29]. https:// sebastianraschka.com/faq/docs/parametric_vs_nonparame tric.html.

18. Brownlee J. Parametric and nonparametric machine learning algorithms. [accessed 2018 Mar 29]. https://machinelearningmastery.com/parametric-and-nonparametric-machine-learning-algorithms.

19. Vapnik V, Chervonenkis A. A note on one class of perceptrons. Automat Rem Control. 1964;25(1):103.

20. Boser BE, Guyon IM, Vapnik VN. A training algorithm for optimal margin classifiers. COLT ’92. Proceedings of the Fifth Annual Workshop on Computational Learning Theory; 1992 July 27–29; Pittsburgh, PA. New York, (NY): ACM Press; c1992. p. 144–152. doi.org/10.1145/130385.130401.

21. Dietterich TG. Ensemble methods in machine learning. In: Multiple Classifier Systems: First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings; Berlin, Heidelberg: Springer Berlin Heidelberg; 2000. p. 1–15.

22. Breiman L. Bagging predictors. Mach Learn. 1996;24(2):123–140.

23. Wolpert DH. Stacked generalization. Neural Networks. 1992;5(2):241–259.

24. Simoudis E, Aha DW. Special issue on lazy learning. Artif Intell Rev. 1997;11 (1–5).

25. Aggarwal CC. Data classification: algorithms and applications. 1st ed. Yorktown Heights (NY): Chapman & Hall/CRC Press; 2014.

26. Stergiou C, Siganos D. Neural networks. [accessed 2016 Dec 27]. https:// www.doc.ic.ac.uk/ nd/surprise_96/journal/vol4/cs1 1/report.html.

27. Bishop CM. Neural networks for pattern recognition. New York (NY) USA: Oxford University Press, Inc; 1995.

28. Bryson A, Ho Y. Applied optimal control: optimization, estimation, and control. 1975. Hemisphere (NY): p.177–203.

29. Bottou L. Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010; Springer; 2010; p. 177–186.

30. Hopfield JJ. Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci USA. 1982;79(8):2554–2558.

31. Hinton GE, Sejnowski TJ. Learning and relearning in Boltzmann machines. Parallel Distrilmted Processing. 1986;1.

32. Neal RM. Connectionist learning of belief networks. Artif Intell. 1992;56(1):71– 113.

33. LeCun Y, Bengio Y. Convolutional networks for images, speech, and time-series. In: Arbib MA, editor. The handbook of brain theory and neural networks. MIT Press; 1995;3361(10).

34. Lin HW, Tegmark M. Why does deep and cheap learning work so well? arXiv preprint arXiv:1608.08225. 2016.

35. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. Advances in Neural Information Processing Systems. NIPS'12. Proceedings of the 25th International Conference on Neural Information Processing Systems, vol 1; 2012 Dec 3–6; Lake Tahoe (NV). Red Hook (NY): Curran Associates, Inc; c2012. p. 1097–1105.

36. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. ECCV 2014. 13th European Conference on Computer Vision; 2014 Sep 6–12; Zurich (Switzerland). Lecture Notes in Computer Science, vol 8689. Springer; c2014. p. 818–833.

37. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. CoRR. 2014;abs/1409.1556.

38. Szegedy C, Liu W, Jia Y, Sermanet P, Reed SE, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. CoRR. 2014;abs/1409.4842.

39. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. CoRR. 2015;abs/1512.03385.

40. Huang G, Liu Z, Weinberger KQ. Densely connected convolutional networks. CoRR. 2016;abs/1608.06993.

41. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial networks. ArXiv e-prints. 2014.

42. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–1780.

43. Chung J, Gülçehre Ç, Cho K, Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR. 2014;abs/1412.3555.

44. Wang C, Yang H, Bartz C, Meinel C. Image captioning with deep bidirectional LSTMs. MM ’16. Proceedings of the 2016 ACM on Multimedia Conference; 2016 Oct 15–19; Amsterdam (the Netherlands). New York (NY): ACM; c2016. p. 988– 997.

45. Torrésani B. Introduction to the special issue on wavelet and time-frequency analysis. J Math Phys. 1998;39(8):3949–3953.

46. Olshausen BA, Field DJ. Sparse coding with an overcomplete basis set: a strategy employed by V1. Vision Res. 1997;37(23):3311–3325.

47. Xie J, Hu W, Zhu SC, Wu YN. Learning sparse frame models for natural image patterns. Int J Comput Vision. 2014;114:91–112.

48. Quiroga RQ, Nadasdy Z, Ben-Shaul Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. Neural Comput. 2004;16(8):1661–1687.

49. Schölkopf B, Smola A, Müller KR. Kernel principal component analysis. In: Gerstner W, Germand A, Hasler M, Nicoud JD, editors. ICANN'97. 7th International Conference on Artificial Neural Networks; 1997 Oct 8–10; Lausanne (Switzerland). Lecture Notes in Computer Science, vol 1327. Springer Berlin Heidelberg; c1997. p. 583–588.

50. Tenenbaum JB, De Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. Science. 2000;290(5500):2319–2323.

51. Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science. 2006;313(5786):504–507.

52. Autoencoders. Stanford (CA): Stanford University. [accessed 2017 Jan 5]. http:// ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/.

53. Bourlard H, Kamp Y. Auto-association by multilayer perceptrons and singular value decomposition. Biol Cybern. 1988;59(4):291–294.

54. Ranzato MA, Boureau YL, LeCun Y. Sparse feature learning for deep belief networks. In: Platt JC, Koller D, Singer Y, Roweis ST, editors. NIPS 2007.

21st Annual Conference on Neural Information Processing Systems; 2007 Dec 3-6.

55. Vancouver BC (Canada). Curran Associates Inc; Advances in Neural Information Processing Systems 20: Proceedings of the 2007 Conference. La Jolla (CA): Neural Information Processing Systems Foundation; c2009. p. 1185–1192.

56. Kingma DP, Welling M. Auto-encoding variational bayes. ICLR 2014. Proceedings of the 2nd International Conference on Learning Representations; 2014 Apr 14–16; Banff (Canada). arXiv preprint arXiv:1312.6114. 2013.

57. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol PA. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res. 2010;11:3371–3408.

58. Chapelle O, Schölkopf B, Zien A. Semi-supervised learning. In: Dietterich T, editor. Adaptive computation and machine learning. Cambridge (MA): MIT Press; 2006. p. 524.

59. Sutton R, Barto A. Reinforcement learning: an introduction. 1st ed. In: Dietterich T, editor. Adaptive computation and machine learning. Cambridge (MA): MIT Press; A Bradford Book; 1998.

60. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T. Caffe: convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093. 2014.

61. Gibson A, Nicholson C, Patterson J, Warrick M, Black AD, Kokorin V, Audet S, Eraly S. Deeplearning4j: distributed, open-source deep learning for Java and Scala on Hadoop and Spark. 2016.

62. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, et al. TensorFlow: large-scale machine learning on heterogeneous distributed systems. 2015. arXiv:1603.04467.

63. Theano Development Team. Theano: a Python framework for fast computation of mathematical expressions. arXiv e-prints. 2016;abs/1605.02688.

64. Chollet F, et al. Keras, GitHub. [accessed 2018 Mar 29]. https://github.com/fchollet/keras; 2015.

65. The Microsoft cognitive toolkit. [accessed 2018 Mar 29]. https://www.microsoft.com/en-us/cognitive-toolkit/.

66. Chen T, Li M, Li Y, Lin M, Wang N, Wang M, Xiao T, Xu B, Zhang C, Zhang Z. MXNet: a flexible and efficient machine learning library for heterogeneous distributed systems. CoRR. 2015;abs/1512.01274.

67. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–2830.

68. Collobert R, Kavukcuoglu K, Farabet C. Torch7: A Matlab-like environment for machine learning. In: BigLearn, NIPS Workshop; 2011 Dec 16–17; Sierra Nevada, Spain.

69. King DE. Dlib-ML: a machine learning toolkit. J Mach Learn Res. 2009;10:1755–1758.

70. Tokui S, Oono K, Hido S., Clayton J. Chainer: a next-generation open source framework for deep learning. In: Proceedings of workshop on machine learning systems in the 29th Annual Conference on Neural Information Processing Systems (NIPS); 2015 Dec 7–12; Montreal, Canada. Neon. [accessed 2018 Mar 29]. http://neon.nervanasys.com/docs/latest/.

71. Hood A, LaBerge K, Lewicki D, Pines D. Vibration based sun gear damage detection. Volume 5: 25th International Conference on Design Theory and Methodology; ASME 2013 Power Transmission and Gearing Conference. ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference; 2013 Aug 4–7; Portland (OR).

72. Ihn JB, Chang FK. Pitch-catch active sensing methods in structural health monitoring for aircraft structures. Struct Health Monit. 2008;7(1):5–19.

73. Nock MK, Stein MB, Heeringa SG, Ursano RJ, Colpe LJ, Fullerton CS, Hwang I, Naifeh JA, Sampson NA, Schoenbaum M, et al. Prevalence and correlates of suicidal behavior among soldiers: results from the Army study to assess risk and resilience in servicemembers (Army STARRS). JAMA Psychiat. 2014;71(5):514– 522.

74. Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Krikun M, Cao Y, Gao Q, Macherey K, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. CoRR. 2016;abs/1609.08144.

75. Goertzel B, Pennachin C. Artificial general intelligence. 1st ed. New York (NY): Springer-Verlag Berlin Heidelberg; 2007 (vol. 2). p. 509.

76. Salerno J, Hinman M, Boulware D. Building a framework for situation awareness. AFRL/IFEA, Air Force Research Lab, Rome (NY) 2004.

77. Perdisci R, Ariu D, Fogla P, Giacinto G, Lee W. McPAD: a multiple classifier system for accurate payload-based anomaly detection. Comput Netw. 2009;53(6):864–881.

78. Rogers SK, Colombi JM, Martin CE, Gainey JC, Fielding KH, Burns TJ, Ruck DW, Kabrisky M, Oxley M. Neural networks for automatic target recognition. Neural Networks. 1995;8(7):1153–1184.

79. Nasrabadi NM. Hyperspectral target detection: an overview of current and future challenges. IEEE Signal Proc Mag. 2014;31(1):34–44.

80. Steinberg AN. An approach to threat assessment. IFUSION2005. In: Proceedings of the 8th International Conference on Multisource Information Fusion; 2005 July 25–29; Philadelphia (PA). Washington (DC): IEEE, Vol. 2; p. 8.

81. Sharma S, Tiwari R, Shukla A, Singh V. Identification of people using gait biometrics. Int J Mach Learn Comput. 2011;1(4):409–415.

82. Zahiri SH, Seyedin SA. Swarm intelligence based classifiers. J Frankl Inst. 2007;344(5):362–376.

83. Roff HM. The strategic robot problem: lethal autonomous weapons in war. J Military Ethics. 2014;13(3):211–227.

84. Lin P, Bekey G, Abney K. Autonomous military robotics: risk, ethics, and design. San Luis Obispo (CA): California Polytechnic State University; 2008.

85. Osteen PR, Owens JL, Kessens CC. Online egomotion estimation of RGB-D sensors using spherical harmonics. In: 2012 IEEE International Conference on Robotics and Automation (ICRA); 2012 May 14–18; St Paul (MN). IEEE; c 2012. p. 1679–1684.

86. Ernest N, Carroll D, Schumacher C, Clark M, Cohen K, Lee G. Genetic fuzzy based artificial intelligence for unmanned combat aerial vehicle control in simulated air combat missions. J Def Manag. 2016;6(144):2167–0374.

87. Mitola J. Cognitive radio—an integrated agent architecture for software defined radio. [dissertation]. [Stockholm (Sweden)]: Royal Institute of Technology (KTH); 2000.

88. Singh J, Nene MJ. A survey on machine learning techniques for intrusion detection systems. Int J Adv Res Comput Commun Eng. 2013;2(11):4349–4355.

89. Shon T, Moon J. A hybrid machine learning approach to network anomaly detection. Inform Sciences. 2007;177(18):3799–3821.

90. Bkassiny M, Li Y, Jayaweera SK. A survey on machine-learning techniques in cognitive radios. IEEE Commun Surv Tut. 2013;15(3):1136– 1159.

91. Seewald AK, Gansterer WN. On the detection and identification of botnets. Comput Secur. 2010;29(1):45–58.

92. Arsene L. Machine learning for cybersecurity not cybercrime. 2017 Jan 1. [accessed 2018 Mar 29] http://www.darkreading.com/partner-perspectives/ bitdefender/machine-learn ing-for-cybersecurity-not-cybercrime/a/did/1327904.

93. Szolovits P, Patil RS, Schwartz WB. Artificial intelligence in medical diagnosis. Ann Intern Med. 1988;108(1):80–87.

94. Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, Thrun S. Dermatologist-level classification of skin cancer with deep neural networks. Nature. 2017;542(7639):115–118.

95. Galatzer-Levy IR, Karstoft KI, Statnikov A, Shalev AY. Quantitative forecasting of PTSD from early trauma responses: a machine learning application. J Psychiat Res. 2014;59:68–76.

96. Gunning D. Explainable artificial intelligence (XAI). Arlington (VA): Defense Advanced Research Projects Agency (DARPA) [accessed 2017 Aug 8]. http:// www.darpa.mil/program/explainable-artificial-intelligence.

97. Hsu J. IBM’s new brain [news]. IEEE spectrum. 2014;51(10):17–19.

98. Keller J. SWaP: how size, weight, and power are transforming the military electronics industry. Military & Aerospace Electronics. 2013 Jun 1. [accessed 2018 Mar 29]. http://www.militaryaerospace.com/articles/print/volume24/issue-6/news/trends/swap--how-size--weight--and-power-aretransforming-the-military-.html.

99. Li FF, Fergus R, Perona P. One-shot learning of object categories. IEEE T Pattern Anal Mach Intel. 2006;28(4):594–611.

100. Davis E, Marcus G. Commonsense reasoning and commonsense knowledge in artificial intelligence. Commun ACM. 2015;58(9):92–103.

101. Cowie R, Douglas-Cowie E, Tsapatsoulis N, Votsis G, Kollias S, Fellenz W, Taylor JG. Emotion recognition in human-computer interaction. IEEE Signal Proc Mag. 2001;18(1):32–80

102. Wainer AL, Ingersoll BR. The use of innovative computer technology for teaching social communication to individuals with autism spectrum disorders. Res Autism Spect Dis. 2011;5(1):96–107.

103. Pollack JB, Hornby GS, Lipson H, Funes P. Computer creativity in the automatic Design of Robots. Computer. 2006;36(2).

104. Kurzweil R. The singularity is near: when humans transcend biology. New York (NY): Penguin Books (Nonclassics); 2006.

105. Bostrom N. Superintelligence: paths, dangers, strategies. 1st ed. Oxford (UK): Oxford University Press; 2014.

Appendix. Technical Posters from the 2016 ARL Open Campus Open House that Referenced Machine Learning

This appendix appears in its original form, without editorial change.

A.1 Analysis & Assessment

Grynovicki, J., “Human System Integration Modeling for Improved Performance” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/AA07.pdf)

- Combining modeling and actual data acquired from experiments, we foresee as being a future ML / uncertainty quantification task.

Acosta, J., “Augmenting Threat Analysis Capabilities Using Intelligent Threat Agents”
(https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/AA09.pdf)

- The core of intelligent agents may be deep learning models.

Montoya, J., “Tools for EO/IR Sensing System Performance Analysis” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/AA14.pdf)

- Improved sensing to the point of classification, is inherently a supervised learning task.

A.2 Human Sciences

Marathe, A., “Continuous Multifaceted Soldier Characterization for Adaptive Technologies”
(https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS01.pdf)

- Dynamic learning to predict Soldier state function is potentially a reinforcement learning endeavor.

Vettel, J., “Individual Differences in Human Variability for Translational Neuroscience”
(https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS02.pdf)

- Unsupervised learning can be used to cluster and characterize the space of human variability

Boynton, A., “Field Assessment of Dismounted Soldier Performance” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS05.pdf)

and

DeCostanza, A., “Real-Time Assessment of Group Dynamics” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS08.pdf)

- Sensor data and various metrics could be used to predict performance (regression) and assign appropriate workloads (classification)

Gaston, J., “Real-World Perceptual Augmentation” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS12.pdf)

- Localization algorithms could be used, for example, to convert field of view into descriptive labels and highlighted points of interest.

Diego, M., "Distributed Soldier Representation" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS16.pdf)

- ML can be used to reduce the information from simulations to bite-sized chunks suitable for engaged Soldiers.

Oie, K., "Human System Integration-Cybernetics" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS20.pdf),

Evans, William A., "Human-Robot Interaction" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS22.pdf),

Davis, T., "Manned and Unmanned Collaborative Systems Integration" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS23.pdf),

and

Chen, J., "Human-Robot Interaction & Human-Agent Teaming" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS24.pdf),

- Machine learning is often used to translate input from language/format to another.
- This may be useful for improving human-system interactions, and appropriately reducing the burden on Soldiers to interpret inputs from an ever-increasing array of systems.

Dickerson, K., "Similarity Metrics for Multimodal Cueing" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/HS21.pdf)

- ML may provide unique means for fusing multimodal data optimized for human consumption.

A.3 Information Sciences

Scanlon, M., "Acoustic Sensors & Processing" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS04.pdf)

- Supervised learning may allow classification-type tasks in the realm of acoustic inputs.

Sullivan, A., "Radar Technology for Detection of Concealed Targets" (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS05.pdf)

- Generative deep learning can be used to develop realistic models.

Rao, R. and Shuowen Hu, “Cross-Modal and Extended Range Face Recognition” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS08.pdf)

- This project uses supervised learning on visible and IR facial images.

Rao, R., “Human Detection in the Wild” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS09.pdf)

- This work extends the state-of-the-art in pedestrian detection via various ML algorithms.

Srour, N., “Sensor, Data and Information Processing, and Fusion for Situational Understanding” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS10.pdf)

- Supervised learning on multimodal data

Suri, N., “Intelligent Information Management for the Battlefield” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS11.pdf)

- Perhaps, reinforcement learning could be used to prioritize information management?

Klavans, J., “Social Computing” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS12.pdf)

- The key here is combining the multilingual technologies developed in-house with some of the big data strategies currently being used for machine translation.

Young, S., “Computational Intelligence” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS13.pdf) and

Summers-Stay, D., “Reasoning Under Uncertainty” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS14.pdf)

- While traditionally we think of supervised and reinforcement learning, the next step in autonomy is where agents and vehicles can have higher levels of intelligence (e.g., reasoning)

Kwon, H., “Joint Text & Video Analytics”, (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS15.pdf)

- The poster’s abstract says it best: "Develop methods for enhancing situational awareness through joint Natural Language (NL) Text and Video analytics for: NL summarization of video, visual question-answering, ontology-supported activity recognition, multimodal representation of event semantics"

Raglin, A., “Discovery Mechanisms for Engendering Creative Decision Making” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS17.pdf)

- The ability of an intelligent to serve up the right information at the right time may be enhanced by the use of reinforcement learning (e.g., "was this helpful?")

Moore, T., “Data-Driven Analysis of Collaboration Structure and Evolution” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS19.pdf)

- Different unsupervised learning algorithms may be part of the workflow for studying this area.

Sadler, B., “Mobility & Cognitive Networking in Harsh Environments” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS21.pdf) and

Tobin, R., “Wireless Networking in Resource Constrained Environments” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS22.pdf)

- Autonomous agents, which underpin the cognitive network, may benefit from supervised and reinforcement learning.

Harang, R., “Characterizing Burstiness in Intrusion Detection” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS24.pdf),

Erbacher, R., “Cognitive Foundations of Cyber Analysts” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/IS25.pdf),

and

Cam, H., “Risk Model Roadmap from Events to Parameters” (https://www.arl.arm y.mil/www/apps/ocoh-tech-followup/posters/IS26.pdf)

- Work in this area requires ML algorithms yet to be discovered appropriate for training on limited data.

A.4 Sciences for Maneuver

Berman, “Energy For Maneuver” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/MAS02.pdf) and

Lee, I., “Self-Sustaining Energy for Robotics and Autonomous Systems” (https://w ww.arl.army.mil/www/apps/ocoh-tech-followup/posters/MAS04.pdf)

- Autonomous, intelligent agents will likely be used in a lot of the decision making for future self-sustaining energy systems.

Riggs, M. and Hood, A., “Probabilistic-Diagnostic Informed Innovations for Power Transmission Light weighting″ (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/MAS13.pdf) and

Hall, A., “Virtual Risk-informed Agile Maneuver Sustainment (VRAMS)” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/MAS28.pdf)

- Supervised learning with diagnostic data will lead to deeper situational awareness on the health state of a system.

Fields, M., “Meta-Cognition, Self-Reflection and Proprioception” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/MAS22.pdf)

- This work tackles some of the more far-reaching goals of AI/machine learning necessary to allow agents to truly be peers with their human counterparts.

Owens, J., “Semantic Spatial Understanding” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/MAS23.pdf)

- This work relies on principles of supervised and reinforcement learning.

A.5 Sciences for Lethality & Protection

Satapathy, S., “Modeling Brain Response to Blast and Ballistic Loading” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/SL03.pdf)

- We have been looking at using unsupervised learning algorithm to automatically yield the 3-D segments required for this project’s simulations.

Allik, B., “Vision Based Navigation” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/SL14.pdf)

- Automatic target recognition has always been a major consumer of ML algorithms.
- From the poster: "Technical challenges include low frame rates, blur, latency, gun survivability, dynamic range, resolution, etc."

A.6 Materials Research

Holmes, L., “Additive Manufacturing Research” (https://www.arl.army.mil/www/apps/ocoh-tech-followup/posters/MS14.pdf)

- One long-term goal is to use machine learning to expedite the application of additive manufacturing to the Army.

List of Symbols, Abbreviations, and Acronyms

2-D	2-dimensional
3-D	3-dimensional
AE	autoencoder
AGI	artificial general intelligence
AI	artificial intelligence
AM	additive manufacturing
ANN	artificial neural network
API	application program interface
ARL	US Army Research Laboratory
ATR	automated target recognition
AUI	adaptive user interfaces
BL	Bayesian learning
BM	Boltzmann machines
BN	Bayesian network
C4.5	a decision-tree generation algorithm
CART	classification and regression trees for machine learning
CISD	Computational and Information Sciences Directorate
CLT	computational learning theory
CNN	convolutional neural network
CPU	central processing unit
CT	computed tomography
DARPA	Defense Advanced Research Projects Agency
DBSCAN	density-based spatial clustering of applications with noise
DOD	Department of Defense
DR	dimensionality reduction
DT	decision tree

FFNN	feed forward neural network
FPGA	field-programmable gate array
FT	Fourier transform
GAN	generative adversarial network
GPS	global positioning system
GPU	graphics processing unit
GRU	gated recurrent unit
HMM	hidden Markov model
HN	Hopfield networks
HPC	high-performance computing
HRED	Human Research and Engineering Directorate
IBL	instance-based learning
kPCA	kernel principal component analysis
LSTM	long short-term memory
ML	machine learning
MRI	magnetic resonance imaging
NN	neural network
NP	nondeterministic polynomial
P	probability
P	polynomial
PAC	probably approximately correct
PCA	principal component analysis
POC	point of contact
PTSD	post-traumatic stress disorder
RBF	radial basis function
ReLU	rectified linear unit
RNN	recurrent neural network

SEDD	Sensors and Electron Devices Directorate
SL	supervised learning
SLAD	Survivability and Lethality Directorate
STARRS	Study to Assess Risk & Resilience in Servicemembers
SVM	support vector machine
UL	unsupervised learning
VTD	Vehicle Technology Directorate
VC	Vapnik-Chervonenkis
WMRD	Weapon and Materials Research Directorate

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIR ARL IMAL HRA RECORDS MGMT RDRL DCL TECH LIB
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
1 (PDF)	RDMR AEA D WADE
41 (PDF)	DIR ARL RDRL CI B HENZ R NAMBURU T PHAM Y SELF-MEDLIN L SOLOMON RDRL CIH S M VINDIOLA J MONACO RDRL CII S YOUNG RDRL CII A G WARNELL RDRL CII B S RUSSELL B JALAIAN RDRL CIN D T BRAUN H CAM M DE LUCIA G SHEARER RDRL CIN T A SWAMI RDRL D A KOTT RDRL HR P FRANASZCZUK RDRL HRF C V LAWHERN RDRL SES E H KWON R RAO RDRL SL T STADTERMAN RDRL SLE W A BEVEC M MARKOWSKI RDRL WML B B BARNES RDRL WMM B G GAZONAS RDRL WMM D C GOBERT C KUBE B MCWILLIAMS J SOUTH RDRL WMM G J ANDZELM RDRL WMP S SCHOENFELD RDRL WMP B C HOPPEL S SATAPATHY RDRL WMP D R DONEY RDRL WMP R D HORNBAKER RDRL VT A C KRONINGER P OSTEEN RDRL VT M A HALL R HAYNES RDRL VT P A HOOD