Deep Learning for Situational Understanding

Supriyo Chakraborty*, Alun Preece†, Moustafa Alzantot‡, Tianwei Xing‡, Dave Braines§†, Mani Srivastava‡
*IBM T.J. Watson Research Center, †Crime and Security Research Institute, Cardiff University, ‡UCLA, §IBM UK

Abstract*—Situational understanding (SU) requires a combination of insight — the ability to accurately perceive an existing situation — and foresight — the ability to anticipate how an existing situation may develop in the future. SU involves information fusion as well as model representation and inference. Commonly, heterogenous data sources must be exploited in the fusion process: often including both hard and soft data products. In a coalition context, data and processing resources will also be distributed and subjected to restrictions on information sharing. It will often be necessary for a human to be in the loop in SU processes, to provide key input and guidance, and to interpret outputs in a way that necessitates a degree of transparency in the processing: systems cannot be "black boxes". In this paper, we characterize the Coalition Situational Understanding (CSU) problem in terms of fusion, temporal, distributed, and human requirements. There is currently significant interest in deep learning (DL) approaches for processing both hard and soft data. We analyze the state-of-the-art in DL in relation to these requirements for CSU, and identify areas where there is currently considerable promise, and key gaps.

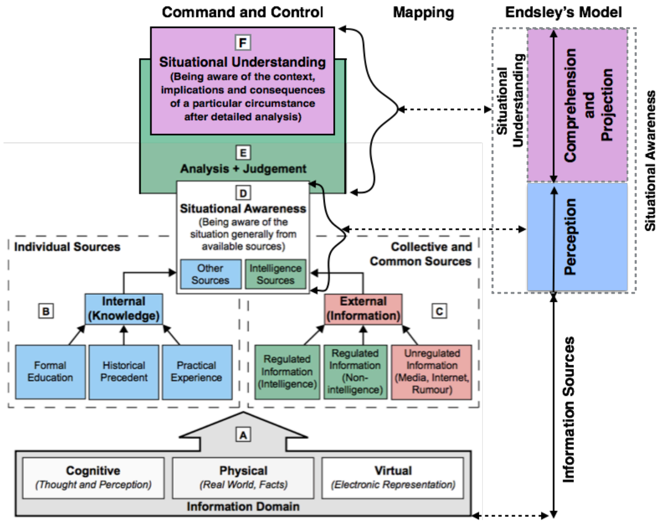
Command and Control
Mapping
Endsley's Model
F
Situational Understanding
(Being aware of the context, implications and consequences of a particular circumstance after detailed analysis)
E
Analysis + Judgement
D
Situational Awareness
(Being aware of the situation generally from available sources)
Other Sources
Intelligence Sources
Individual Sources
Collective and Common Sources
B
Internal (Knowledge)
C
External (Information)
Formal Education
Historical Precedent
Practical Experience
Regulated Information (Intelligence)
Regulated Information (Non-intelligence)
Unregulated Information (Media, Internet, Rumour)
A
Cognitive
(Thought and Perception)
Physical
(Real World, Facts)
Virtual
(Electronic Representation)
Information Domain
Situational Understanding
Comprehension and Projection
Perception
Situational Awareness
Information Sources

Fig. 1. Mapping between different notions of situational understanding.

I. Introduction

Decision making in complex multi-variate domains such as air traffic control, ship navigation, emergency response, military command and control, and so on, often depends heavily on the *situational understanding* of the decision maker. This notion of understanding, in general is defined as the "product of applying analysis and judgment to the unit's situation awareness to determine the relationships of the factors present, and form logical conclusions concerning threats to the mission accomplishment, opportunities for mission accomplishment, and gaps in information" [1]. This is asserted as corresponding to *Level 2 situational awareness* (SA) in Endsley's widely-used model, as shown in Fig. 1. In fact, Endsley's SA model provides us with an operational definition of understanding as – the spatio-temporal *perception* of environmental events (level 1), followed by their *comprehension* within a specific context (level 2), and finally, the ability to *project* or predict potential future events (level 3) due to change in variables such as time, or other events (also see the UK military doctrine [2]).

In this work, we explore the functional blocks required to achieve situational understanding in an ad-hoc military coalition characterized as a federation of agents working collaboratively towards a common mission objective. These are coalition agents and therefore are also part of separate administrative domains, that determine their local data collection and data sharing policies.

Motivated by the widespread usage of deep-learning based techniques in domains such as health care, criminal justice system, finance, as well as military decision making [3], [4], we begin by translating the operational definition of situational understanding (provided by Endsley's model) using components of a distributed learning framework. First, to gain *perception* of the environment, the distributed model should be trained to identify events occurring at different levels of spatio-temporal granularity. These events are collectively recorded in the time-series data collected by the various agents. The coalition context for these agents can further introduce local constraints to regulate the flow of information between the agents, thereby adding additional complexity and complication. Second, to aid *comprehension*, the model must not operate as a black-box. In other words, the model must be interpretable [5], and its output explainable using human understandable semantics. Finally, the model itself should be generative, with ability to accurately *project* into future states.

An illustration of a multi-layer view of a coalition network in shown in Fig. 2. Layer 1 depicts the different coalition agents (blue, green and yellow regions). Each agent collects multi-modal data locally, and cooperates with other agents, within constraints of their own organizational data-sharing policies. Layer 2 shows information pooling from human and machine agents required to achieve situational understanding. Coalition Situational Understanding (CSU): Based on the above description, coalition situational understanding can be broken down into the following learning challenges.

- **Distributed Learning Algorithm:** The very existence of a coalition is contingent on the premise that the *whole is greater than the sum of the parts*, i.e., the shared model of the environment, learned using the information combined

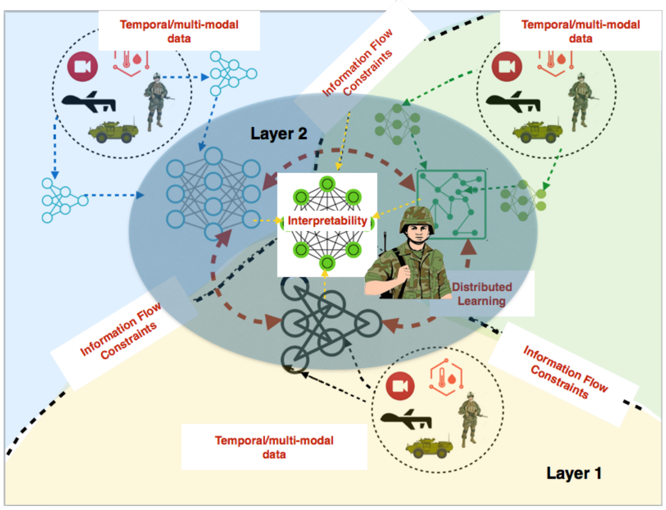
Temporal/multi-modal data
Temporal/multi-modal data
Information Flow Constraints
Layer 2
Interpretability
Distributed Learning
Information Flow Constraints
Information Flow Constraints
Temporal/multi-modal data
Layer 1

Fig. 2. A multi-layer view of an ad-hoc military coalition. Information exchange occurs between the different agents under various domain-specific constraints to achieve distributed learning of an interpretable model.

from all the agents is greater than just the individual pieces of information. However, to train a shared model and realize the above goal, the learning algorithm used should, (1) be able to adjust to the variability in network topology connecting the various agents; (2) be sensitive to the reliability of the training data available from the agents; (3) account for the different granularities at which information is made available (e.g., raw data or model parameters) by the agents; and finally, (4) meet the privacy requirements of the agents.

- **Multiple time-scale learning:** Coalitions are often formed to monitor a particular geographic region for event(s) of interest. However, the periodicity of the monitored events could be different. The shared model, should have the power to use the collective information from the agents to learn events that manifest themselves at different time scales. For example, on a particular road segment, the volume of traffic (or the congestion level) on a weekday, may be solely dependent on the time-of-day. However, the congestion level over a weekend, might depend on the schedule of a nearby sporting event. Thus, congestion is predicted as a result of two different events occurring at different time scales.
- **Model interpretability and tellability:** These attributes refer to the bi-directional flow of information between models and humans. While deep learning based models are motivated by neuro-scientific advancements in the understanding of the working of the human brain, a critical distinction, that has often been made between the two, is attributed to the human ability to “think” [6]. Informally, it is this ability to *think*, that allows humans to not only make a prediction, but also *justify* or *rationalize* it through a series of logically consistent and *understandable* choices leading up to the prediction. This justification, in turn, enables the decision maker to implicitly or explicitly associate a measure of *confidence* to the prediction and use that to determine the next steps of necessary actions. The counterpart to the human thought process in deep learning models is often referred to as *interpretability* [5]. The ability to interpret a prediction enables semantically meaningful information to flow from the models to humans.

  We refer to the information flow from humans to models as *tellability*. The notion of tellability is different from enriching the training data set with samples corresponding to the new hypothesis classes that we want the model to learn. Instead, it implies adding prior information to the model that is not part of the training data. Tellable information is typically based on human experience and not limited to the training data alone.

Our contributions: In this paper, we make two contributions. First, we translate the problem of coalition situational understanding within a learning framework. We do so, by mapping the operational definition from Endsley’s model to different components of the learning framework. For each of these components, we then summarize the state-of-the-art and perform gap-analysis. Second, we provide our preliminary results, using techniques that are aimed at bridging the gap for each component, and list the open problems that need to be solved in order to attain situational understanding.

II. Distributed Model Learning

We broadly categorize the challenges involved in designing an effective distributed learning algorithm into two groups. First, any learning strategy should account for the heterogeneity in network topology of the agents effecting the communication costs of data (or parameter) sharing, and also make suitable adjustments for unreliable training data. Second, in a more adversarial setting, the learning algorithm should be resilient to intentional tampering of training data with the aim to subvert model learning, and also preserve the privacy of the agents sharing their local data.

A. Comparison of Learning Strategies

We did a preliminary analysis of different distributed learning strategies and compared them in terms of their communication efficiency. In particular, we focussed on (1) federated, (2) incremental, and (3) diffusion, based strategies which have the highest communication efficiency, in terms of communication power consumed for model training. Once the learning algorithm has completed the model training phase, all the agents are expected to have the same shared model.

Federated Learning is a centralized learning strategy. Learning is done via a federation of agents, which are coordinated by a central server [7]. In each iteration, the agents pull the global model from the central server, and train the model separately on their local datasets. After training for several epochs, parameter updates from all agents are aggregated by the central server. The agent parameters are averaged and used to update the global model parameters.

Incremental Learning strategy uses a cyclic path connecting the agents to propagate the model update. The distribution network is represented by a graph, where agents correspond to nodes in the graph. An edge is placed between two nodes if

Strategy	Comm #	AvgDistance	TotalComm
Federated	N x 2	1/6	$\Theta(n)$
Incremental	1	$\sqrt{log^2 n/n}$	$\Theta(log^2 n/n)$
Diffusion	N x 2	$\sqrt{log^2 n/n}$	$\Theta(log^2 n)$

TABLE I
COMMUNICATION COST OF DIFFERENT STRATEGIES

they have a direct line of communication, i.e. they are neighbors of each other within a certain distance. While, finding such a cycle – also called a Hamiltonian cycle – in general, is an NP-complete problem, prior work has demonstrated that they can be found with high probability in random geometric graphs that we use in our experiments [8], [9]. During training, each agent transmits model parameters to the next agent in the cycle. Thus, in any iteration, there is only a single active agent.

Diffusion Learning strategy also makes use of the Hamiltonian cycle described above. However, in this strategy, all the agents perform model training using their local datasets in parallel. Once training is complete, the model parameters are then shared with the neighboring nodes. This allows for faster dissemination of the model parameters and is also in accordance with asynchronous learning, where different agents learn with models of different parameters at every iteration.

Experimental Setup We use a setup of n nodes uniformly distributed over the unit square region $[0, 1]^2$ as our simulated network. Each node maintains its own private dataset that can be used for training a shared model. Within this network, the distributed learning strategies are implemented as follows:

In case of federated learning, we exploit the symmetry of the distribution region, and compute the optimal location of the central server to be the center of the square with coordinate $[0.5, 0.5]$. The average communication cost for every agent is thus a constant, equal to $1/6$. The total communication cost is of the order $\Theta(n)$, which is proportional to the number of agents in the network.

In case of incremental learning and diffusion learning strategies, the communication cost depends on the edges that form the Hamiltonian cycle. Since the graph is constructed based on communication constraint, i.e., edges between two nodes exist only when their distance is less than certain radius. Setting this radius to $r_n \propto \Theta(\sqrt{log^2 n/n})$, ensures that the Hamiltonian cycle exist with a high probability. The average communication cost is no greater than $\Theta(log^2 n/n)$. A comparison of the total communication cost for each of the three strategies is shown in Table II-A.

We use a Multilayer Perceptron (MLP) network with a single hidden layer of 128 neurons as the shared model and train it using the MNIST dataset. Data is randomly partitioned among ten agents. In every iteration, the agents train their own model for 20 epochs, using local datasets and a batch size of 100 data samples. The optimization algorithm we are using is the vanilla stochastic gradient descent. After every iteration, we use the testing dataset to evaluate the performance of each agent, and communicate the model with other agents as per the learning strategy.

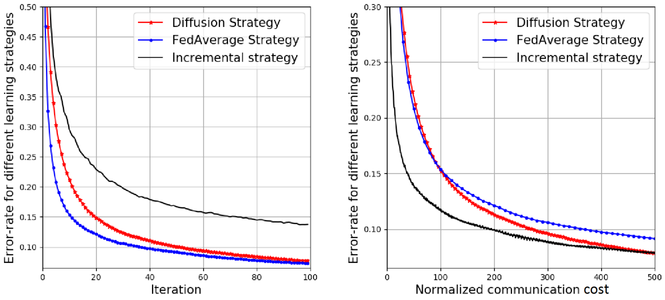

Fig. 3. Comparison of the learning strategies in terms of the error rate (expressed as a percentage) and the approximate communication cost.

Communication Cost Analysis We use two different metrics for analyzing the communication cost. The first one is the *error rate* which we define as:

$$\text{Error Rate} = 1 - \mathbb{P}(\text{accurate classification}).$$

The total communication cost is the sum of all the pairwise communication costs between different clients. We further approximate the pairwise communication cost between nodes i and j, denoted by $E_{i,j}$ as,

$$E_{i,j} \sim d_{i,j}^2$$

where $d_{i,j}$ is the distance between the two nodes.

Fig. 3 shows the change in error-rate for different learning strategies. For incremental strategy, the curve shows the performance of the model propagated in the network. While for federated learning and diffusion learning, the transition curves are drawn as averages of testing error-rate for all the agents in the networks. Fig. 3 (a) shows the change in the error-rate curve for the three different strategies as a function of the number of iterations and Fig. 3 (b) shows the error-rate with regard to the normalized communication cost.

From our preliminary analysis, it is clear that federated learning consumes the most energy. This is because all the agents are assumed to be able to communicate with the central server, which requires them to have sufficient radio power. In addition to this, setting up a central server for storing and updating parameters is also resource intensive. In comparison, for the incremental and diffusion strategies, the average communication distance can be optimized using the Hamiltonian cycle, which also provides an upper bound on the radio range for each client.

B. Prior approaches for privacy-aware distributed learning

We now summarize prior work that has been done in the area of privacy-aware and adversarial learning strategies.

Shokri et. al. [10] considered a setup where data are distributed between various agents, each wanting to keep their own data private. The goal was to learn a model over the combined data. The authors proposed a technique where models were trained locally, and instead of raw data only the model parameters were shared. In fact, to guarantee privacy, the model parameters were perturbed using differentially private noise before sharing. The shared parameters were averaged and passed back to the entities for the next iteration. Model inversion attacks were demonstrated in [11]. The authors showed that shared models leak information and are vulnerable even against a "black-box" adversary (that interacts with the model only via inputs and outputs). More recently, Abadi et. al. [12], demonstrated the training of a large differentially private neural network (with non-convex objectives) and a privacy budget. In fact, they proved that their model is resilient to model inversion attacks against a stronger adversary that also has knowledge of the training algorithm and the model parameters.

Generative adversarial network (GAN) is an unsupervised framework, used to train a generative model in an adversarial setting. Two models are simultaneously trained: a generative model G that captures the data distribution, and a discriminative model D, that estimates the probability that a sample came from the training data rather than G [13]. The goal is to train the generator G such that the error rate of the discriminator is close to 50%, which implies that D is unable to distinguish between a training and a fake sample. Note, while the discriminator has access to the training samples the generator does not. The weights of the generator are adjusted based on the output of the discriminator. In a distributed setup, each agent trains a discriminator model (that has access to the local training data), a shared generator model is trained based on the feedback received from the discriminator models of the different agents. Sometimes, training data are injected with malicious samples to allow attackers to induce vulnerability into the model and mount black box attacks without knowledge of the model's parameters. Adversarial training is a technique that trains a model explicitly on adversarial examples to make it robust to attacks. Scaling of an adversarial training to large data sets has been proposed in [14].

C. *Gap analysis within coalition context*

Distributed learning in the coalition context is confounded by concerns of data privacy of the agents and heterogeneity in terms of both network topology and in the form of the shared data. In addition, for large models, parameter sharing is also extremely expensive in terms of the memory and bandwidth consumed, especially for resource constrained agents [15].

In future, we plan to extend our investigations to model network effects including bandwidth constraints more accurately, allowing us to better study the impact of the network on learning algorithm. In addition, we will also introduce appropriate penalty functions, to handle reliability of the training data and study their impact on the error-rate.

III. Multi Time-Scale Learning

Situational understanding in a coalition setting is a complex task that requires fusion of information from different sources in order to make accurate decisions. One major challenge it to handle the heterogeneity between the data sources with respect to their time scales. This can be due to the different sampling rates at which data are collected or due to the difference in periodicity of the patterns recorded in the data. We explain these challenges in detail below.

- **Multi time-scale data sources**: If different data sources collect data at different frequencies, challenges arise when trying to learn a model from these data sources. For example, if a GPS sensor collects data at the rate of one sample per minute (0.01667 Hz) and a microphone collects data at a significantly higher rate of $16,000$ Hz. Fusing these two sources of data together is clearly challenging as they vary significantly in their data rate.
- **Multi time-scale data patterns**: Multi-time scale pattern learning can be a challenging problem even when using data from a single source. There may exist patterns at different scales within the data which require a scale-invariant learning algorithm. For example, consider the task of modeling the volume of road traffic to predict future congestion levels. Clearly, traffic congestion patterns exist at multiple time scales: at the scale of specific hours within the day, or particular days of the week, and even during specific seasons of the year. Another example is language modeling for text understanding. In this case, patterns exist at the character-level, word-level, and sentence-level. An effective model should be able to learn and exploit all these patterns together instead of modeling only the single level patterns.

A. *Prior approaches for multi-time scale learning*

Recurrent neural networks (RNNs) are theoretically capable of learning long-term temporal dependencies regardless of their scale. However, in practice, this is difficult to achieve due to the long-term memory requirements while RNN suffer from vanishing gradient problem [16]. These practical difficulties, induced by multi-scale patterns in time-series data while doing sequence prediction and classification, led to the invention of the ClockWork RNN (CW-RNN) model architecture [17]. CW-RNN is a powerful modification of the standard RNN architecture in which the hidden layer units are partitioned into separate modules. Each module can process inputs at its own *clock rate*. Units within each module are fully connected to each other, while only units that are in faster modules (with smaller clock period) are connected to units in other slower modules (with larger clock period). The evaluation results of CW-RNN when tested on sequence generation and sequence classification tasks in [17], shows that CW-RNN can achieve significantly better results compared to both standard RNNs and also Long Short-Term Memory (LSTM) networks [18]. In addition, CW-RNNs are also computationally more efficient than an equivalent RNN or LSTM with the same number of hidden neurons, as hidden neurons are updated only at their assigned clock rates. However, CW-RNN require manual setting of the clock rates for each group of hidden neurons while an effective multi-scale learning algorithm should be able to automatically learn these rates from the data.

An approach for learning multi-scale temporal patterns by discovering the latent hierarchical structure of the sequence was recently proposed by the authors in [19]. This model,

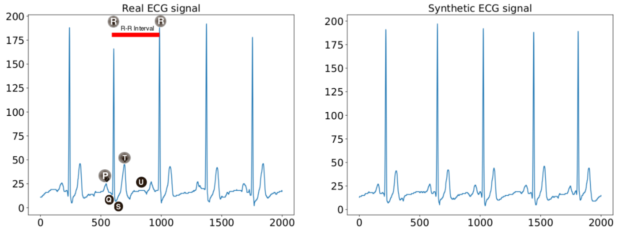

Fig. 4. Real ECG signal vs synthetic output generated by a multi-layer recurrent neural network.

called *hierarchial multiscale recurrent neural network* (HM-RNN), does not require assigning fixed clock rates or explicit boundary information. The model is able to adaptively find the proper relationship between the patterns at different scales in the sequence.

Multi-scale learning is also challenging problem when the scale variation exist in dimensions other than time, such as, in images where objects appear at different scales. While the state-of-the-art techniques in object recognition, namely convolutional neural networks (CNN) [20], use convolution and pooling layers for satisfying the translation and scale invariance requirements to detect patterns in images, scale invariance, remains a challenge as CNN accuracy can drop significantly due to change of scale of objects in test images. This was also demonstrated in [21], where the performance of a CNN on scaled version of CIFAR-10 [22] dataset dropped by 43.22% of its performance on the standard CIFAR-10. To counter these challenges, variations of CNN, such as the Scale-Invariant Convolutional Neural Networks have been proposed that detect objects and are resilient to scale variations.

B. Multi-time scale pattern learning using ECG data

We investigated the multi-scale pattern learning problem by training a deep recurrent neural network for identifying patterns in an electrocardiography (ECG) signal. ECG signals have multiple repeating patterns composed of waves, typically identified by the letters P, Q, R, S, T, and U. To successfully learn the patterns and synthesize an ECG signal, the interval between each of these waves as well their periodicity has to be preserved. In addition, the high sampling rate of the signal, implies that the current value of the signal can be dependent on prior values that were hundreds of time steps ago – long range dependence. Therefore, the model has to exhibit long-term memory capabilities in order to model these long range dependencies.

We used a variation of the hierarchical recurrent neural network architecture, proposed in [23], to perform our experiments. Fig. 4 shows 2000 samples each of the true ECG signal and the signals synthesized by our model. The results demonstrates that our model architecture was able to successful generate an output signal that closely matched the real signal.

C. Gap analysis within coalition context

In a coalition setting, multi-scale learning is an inherent problem due to the different time-scales at which data is collected by the various agents. In addition, in coalition scenarios involving surveillance or monitoring, there is a often a necessity for learning very long-term dependencies (i.e., the prediction or classification result may depend heavily on measurements that happened over a long period of time). While learning very long-term dependencies is a problem for standard RNN and LSTM networks, multi-scale learning algorithms such as CW-RNN and HM-RNN exhibit better results for this goal. In future, we also want to combine these models with episodic delineation techniques for improved results [24].

IV. Model interpretability and tellability

This attribute represents the extent to which the human is in the loop for the CSU problem. As discussed in Section I and illustrated in Figure 2, the human user typically has a number of interaction points with the data models, generally in terms of setting requirements and preferences. These interactions need to be bi-directional – where on one hand, the model predictions are interpretable and on the other hand, humans are able to inject prior knowledge into the model training process. Specifically, where the human provides input to the system in terms that change the representation and reasoning of the corresponding model — for example, by providing key information currently unknown to the CSU system — we denote the model as tellable by humans. We believe that analyzing model interpretability will also help in identifying pathways using which prior knowledge can be injected into the models, making them tellable. Therefore, in the remaining section, we will focus primarily on model interpretability.

One can be misled into believing that interpretability is simply the ability to explain the working of the learned model (Layer 2 in Fig. 2) itself in a human understandable way. In fact, a closer inspection of even the human thought process reveals that we do not actually interpret the working of the model (in this case of our brain) in terms of its low-level parameters. We do make predictions but we do not justify our predictions based on the learning algorithm used by the brain or the way it chooses to represent information (model parameters) in the model. Instead, we choose to provide our justifications, more often than not in a post-hoc setting, based on prior information correlating model response and physical observations. *This implies that one can define interpretability at multiple different levels*: in terms of low-level model parameters and learning algorithms used to train the model, or in terms of the functionality of the model or even a combination of both.

In fact, as observed in [5], the notion of interpretability is not even a monolithic concept but reflects several different dimensions which are summarized below:

- **Model Transparency:** This is defined in terms of three parameters: (i) *simulatability* – whether a human can

use the input data together with the model to reproduce every calculation step necessary to make the prediction. This allows the human to understand the changes in the model parameters caused by the input and the influence of the training data on the model parameters; (ii) *decomposability* – whether there is an intuitive explanation for all the model parameters; and finally (iii) *algorithmic transparency* – which is essentially an ability to explain the working of the learning algorithm. For example, choice of a linear regression model versus a highly non-linear neural network.
- **Model Functionality:** This is defined in terms of (i) *textual description* – explaining the predictions. To do so, one might use a model for prediction and another model to generate an explanation; (ii) *visualization* – another common means of explaining the working of a model is through visualization of the parameters. One popular approach to visualize high-dimensional distributed representations is using the t-SNE mechanism [25]; and finally (iii) *local explanation* – where instead of explaining the entire mapping of a model, local changes introduced by a specific input vector for a given output class is computed. Gradient of the output is used to identify specific weights and the local changes that are influenced by the input vector.

Note that some of the above dimensions, while leading to better interpretability of the model, can however lead to loss in model efficiency. For example, a linear model using simple features, even with a large amount of training data, will not be able to match the predictive capability of a neural network for highly non-linear data.

We now summarize prior work based on the above interpretability dimensions. We then identify challenges that are unique to the CSU setting.

A. Prior approaches for model interpretability

We provide a classification of prior works based on the dimensions discussed above.

Deep learning networks are created by composition of individual units that are differentiable. This allows the gradient-descent based learning algorithms to back propagate and adjust the weights of these units to minimize the error function. Recently, the layer-wise relevance propagation algorithm [26], [27] (LRP), has been proposed that uses this property of the individual units to decompose the output of a deep neural network in terms of its input variables and provide transparency into which features of the data were used for the model output. It is a principled method which has close relation to Taylor decomposition and is applicable to arbitrary deep neural network architectures.

The LRP technique has been used for EEG data analysis in [28]. Relevance score for each input data point is computed towards the final decision and is then visualized as a heat map providing interpretability.

Our familiarity with textual explanations make them a useful method for explaining a model to humans. Recent work in this space has thus focused on learning the textual explanation. The authors in [29], combine two modular components – a generator and encoder – to operate together and learn candidate rationales for a prediction. Rationales are simply subsets of the words from the input text that satisfy two key properties. First, the selected words represent short and coherent pieces of text (e.g., phrases) and, second, the selected words must alone suffice for prediction as a substitute of the original text. For a given input text, the generator specifies a distribution over possible rationales. The encoder then maps the rationale to task specific values. The distribution that minimizes the regularized encoder loss function is then used as the rationale.

Similarly, word vectorization models are used to capture the semantic context of a word, and explain the result of the operation on the vectors [30]. Model visualization for understanding the working of recursive neural networks has been proposed in [31]. The above technique provide insight into the functionality of the neural network models.

B. Gap analysis within coalition context

We consider the problem of model interpretation within a coalition setting. Any decision made using the common model has to be adequately justified for it to be accepted by all the coalition agents. Such a justification can only be generated using an interpretable model. Furthermore, the policy-constraints of an agent can mandate that it cannot share raw data with other agents. In such a scenario, the member can train a local model and share the predictions of the model (instead of raw data). For the other agents to use this prediction, it should not only be associated with a confidence score, but also a justification for the prediction based on an interpretable model.

The coalition setting also presents unique challenges that can influence the design of the interpretable model. The local models at each agent maybe be non-homogeneous in terms of their architecture, making it difficult to use techniques such as layer-wise relevance propagation (LRP). Similarly, the policy-based constraints might make it difficult to learn the correct interpretation because of incomplete information sharing.

V. Approach Overview and Open Problems

In the previous sections, we defined the coalition situational understanding problem in terms of operational components of a distributed deep learning framework. We also identified several gaps in current architectures such as lack of support for (1) distributed model learning in a heterogenous and possibly adversarial setting; (2) multi-scale pattern learning from time series data; and (3) human-in-the-loop activities via interpretable and tellable models. These gaps need to closed to enable such a framework to work within a coalition context. In addition, model learning is also a data and computation intensive process. While we can address the need for computing power by pooling in more resources, obtaining the required amount of training data, might not always be feasible, especially in an ad-hoc coalition setting. Finally, deep learning models are poor at representing uncertainty – a key ingredient in the human-machine interaction loop. The quantification of uncertainty is also related to the interpretability aspect of the models, as it allows humans to associate a degree of trust to the machine output [32].

In comparison, Bayesian reasoning provides a unified framework for model building, inference, prediction and decision making. There is explicit accounting for uncertainty and variability of outcomes. Finally, the framework is also robust to model overfitting and Bayes rule provides an automatic "Occam's Razor" effect, penalizing unnecessarily complex models [33], [34]. However, for reasons of computational tractability of inferences, Bayesian reasoning is restricted primarily to conjugate and linear models.

The above leads us to the observation that there exists elements in the Bayesian reasoning and deep learning frameworks that complement each other. This observation has been exploited in recent work on probabilistic machine learning in general (see [35] and references therein) and Bayesian Deep Learning (BDL) in particular [36]. In short, BDL aims to integrate deep learning and Bayesian models within a uniform probabilistic framework. Motivated by the above advances in BDL, we sketch an initial architecture for addressing the coalition situational understanding problem below.

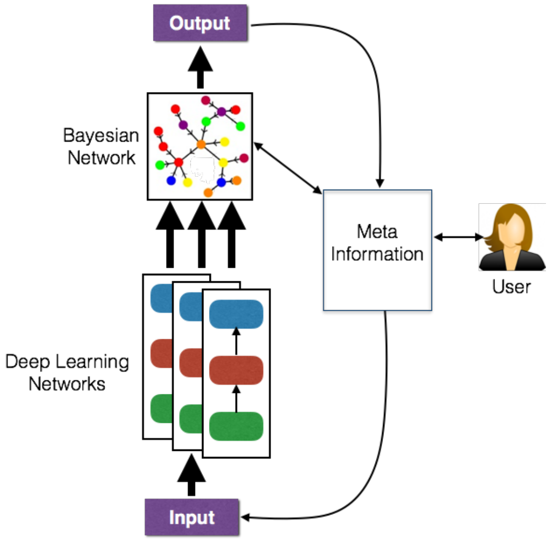

Fig. 5. Human-in-the-loop CSU approach

Fig. 5 illustrates our approach to the CSU problem with an emphasis on the human-in-the-loop dimension and can be considered a 'vertical' view of the 'horizontal' plane previously depicted in Section I, Fig. 2. We view the interpretable deep learning networks, handling time-series data and extracting patterns at multiple time-scales, feeding their output (e.g., next predicted state) into a Bayesian network. We have already established that deep learning models perform exceedingly well when it comes to *perception* tasks such as object recognition, speech recognition, text modeling and so on. The Bayesian network connects the output from the various models into a reasoning network which then can be used to draw inferences leading to CSU. Such a reasoning network is also amenable to user input in the form of priors (i.e., it is tellable), and some initial work has been done in this area, referred to as collaborative DL [37]. This ability to assign priors on the Bayesian network can help reduce the amount of training data required for model training. In addition, the notion of automatic relevance determination (ARD), has also been studied in Bayesian networks in [38], and is very similar to the layer wise relevance propagation techniques applied in deep learning – allowing interpretability. Therefore, the above architecture can not only be trained within a unified probabilistic model, but can also fill in the gaps required for achieving CSU.

We envision, that a user will interact with the system via a meta-information layer that will allow bidirectional exchange of messages. On one hand, a human can use this layer to provide priors – these priors can be on the hidden units of the deep learning models, parameters defining the neural network or the model parameters specifying the causal inferences (represented in the figure as a Bayesian network). As mentioned earlier, these priors help avoid overfitting, especially when sufficient training data is unavailable [36]. On the other hand, the model output will be translated to human-consumable form (e.g., semantically meaningful sentences) by this layer. As the situation develops, the agents (human and machine) can both contribute additional meta information to better describe the problem context, additional data sources and/or services.

Currently, work is underway to realize the architecture scoped in Figures 5 in the context of a number of pilot applications involving a mix of DL techniques from the ones surveyed in earlier sections, together with other signal processing techniques, with an aim to begin addressing the gaps identified above.

VI. Conclusion and Future Work

In this paper we have established that, to be effective in addressing the SU problem in a coalition context, a DL-based approach must be able to incorporate:

1) multi-modal data;
2) time-series data;
3) distributed learning with information flow constraints;
4) classification and model-building (low-level and high-level fusion);
5) human-in-the-loop factors (interpretability and tellability).

All of these characteristics may be present in non-coalition SU settings also, but all are likely to always be present in the coalition context.

Beyond characterizing the coalition SU problem, the main contribution of this paper has been twofold: (1) to analyze the current state-of-the-art in DL against each of the above requirements, and to identify gaps and (2) to propose an outline systems architecture to support ongoing research into addressing the gaps.

Our immediate research agenda consists of grounding the proposed architecture in a set of initial implementations, addressing a small number of challenge problems featuring combinations of the above requirements. Going forward, we are committed to placing our datasets and code in the public domain to promote collaboration in what is, after all, a large problem space.

Acknowledgements

This research was sponsored by the U.S. Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the UK Ministry of Defence or the UK Government. The U.S. and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copy-right notation hereon.

References

[1] B. C. Dostal, "Enhancing situational understanding through the employment of unmanned aerial vehicles," *Army Transformation Taking Shape…Interim Brigade Combat Team Newsletter*, no. 01–18, 2007.

[2] "Understanding: Joint Doctrine Publication 04 (JDP 04)," Ministry of Defence, UK, 2010.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 5 2015.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[5] Z. C. Lipton, "The mythos of model interpretability," *CoRR*, vol. abs/1606.03490, 2016.

[6] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," *CoRR*, vol. abs/1604.00289, 2016.

[7] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[8] J. Petit, "Hamiltonian cycles in faulty random geometric networks," in *2nd International Workshop on Approximation and Randomization in Communication Networks*, 2001.

[9] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, 2005.

[10] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15, 2015.

[11] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15, 2015.

[12] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16, 2016.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *Annual Conference on Neural Information Processing Systems*, 2014, pp. 2672–2680.

[14] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *CoRR*, vol. abs/1611.01236, 2016.

[15] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'14, 2014.

[16] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[17] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork rnn," *arXiv preprint arXiv:1402.3511*, 2014.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] J. Chung, S. Ahn, and Y. Bengio, "Hierarchical multiscale recurrent neural networks," *arXiv preprint arXiv:1609.01704*, 2016.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[21] Y. Xu, T. Xiao, J. Zhang, K. Yang, and Z. Zhang, "Scale-invariant convolutional neural networks," *arXiv preprint arXiv:1411.6369*, 2014.

[22] A. Krizhevsky, V. Nair, and G. Hinton, "The cifar-10 dataset," 2014.

[23] M. Alzantot, S. Chakraborty, and M. Srivastava, "Sensegen: A deep learning architecture for synthetic sensor data generation," in *Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on*. IEEE, 2017, pp. 188–193.

[24] T. D. Kelley and S. McGhee, "Combining metric episodes with semantic event concepts within the symbolic and sub-symbolic robotics intelligence control system (ss-rics)," 2013.

[25] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[26] A. Binder, G. Montavon, S. Bach, K. Müller, and W. Samek, "Layer-wise relevance propagation for neural networks with local renormalization layers," *CoRR*, vol. abs/1604.00825, 2016.

[27] A. Binder, S. Bach, G. Montavon, K.-R. Müller, and W. Samek, *Layer-Wise Relevance Propagation for Deep Neural Network Architectures*, 2016.

[28] I. Sturm, S. Bach, W. Samek, and K. Müller, "Interpretable deep neural networks for single-trial EEG classification," *CoRR*, vol. abs/1604.08201, 2016.

[29] T. Lei, R. Barzilay, and T. S. Jaakkola, "Rationalizing neural predictions," *CoRR*, vol. abs/1606.04155, 2016.

[30] M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. H. Hovy, and N. A. Smith, "Retrofitting word vectors to semantic lexicons," *CoRR*, vol. abs/1411.4166, 2014.

[31] A. Karpathy, J. Johnson, and F. Li, "Visualizing and understanding recurrent networks," *CoRR*, vol. abs/1506.02078, 2015.

[32] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, 2016.

[33] D. J. MacKay, "Bayesian methods for adaptive models," Ph.D. dissertation, California Institute of Technology, 1992.

[34] I. Murray and Z. Ghahramani, "A note on the evidence and bayesian occam's razor," 2005.

[35] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, vol. 521, no. 7553, pp. 452–459, 2015.

[36] H. Wang and D.-Y. Yeung, "Towards bayesian deep learning: A survey," *arXiv preprint arXiv:1604.01662*, 2016.

[37] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," *arXiv preprint arXiv:1409.2944*, 2015.

[38] R. Neal, "Assessing Relevance Determination Methods Using DELVE," in *Neural Networks and Machine Learning*, 1998.