Argumentation-Based Online Incremental Learning

H. Ayoobi*[1], Student Member, IEEE, M. Cao[2], Senior Member, IEEE, R. Verbrugge[1] and B. Verheij[1]

***Abstract*—The environment around general-purpose service robots has a dynamic nature. Accordingly, even the robot's programmer cannot predict all the possible external failures which the robot may confront. This research proposes an online incremental learning method that can be further used to autonomously handle external failures originating from a change in the environment. Existing research typically offers special-purpose solutions. Furthermore, the current incremental online learning algorithms can not generalize well with just a few observations. In contrast, our method extracts a set of hypotheses, which can then be used for finding the best recovery behavior at each failure state. The proposed argumentation-based online incremental learning approach uses an abstract and bipolar argumentation framework to extract the most relevant hypotheses and model the defeasibility relation between them. This leads to a novel online incremental learning approach that overcomes the addressed problems and can be used in different domains including robotic applications. We have compared our proposed approach with state-of-the-art online incremental learning approaches, an approximation-based reinforcement learning method, and several online contextual bandit algorithms. The experimental results show that our approach learns more quickly with a lower number of observations and also has higher final precision than the other methods.**

***Note to Practitioners*—This work proposes an online incremental learning method that learns faster by using a lower number of failure states than other state-of-the-art approaches. The resulting technique also has higher final learning precision than other methods. Argumentation-based online incremental learning generates an explainable set of rules which can be further used for human-robot interaction. Moreover, testing the proposed method using a publicly available dataset suggests wider applicability of the proposed incremental learning method outside the robotics field wherever an online incremental learner is required. The limitation of the proposed method is that it aims for handling discrete feature values.**

***Index Terms*—Argumentation-Based Learning, Online Incremental Learning, Argumentation Theory, General Purpose Service Robots**

I. Introduction

THE development and application of domestic service robots are growing rapidly. Whereas basic household robots are already common practice [1], the study of General Purpose Domestic Service Robots (GPSR) able to do complex tasks is increasing [2], [3]. Due to the dynamic environment around GPSRs, they need to efficiently handle noise and uncertainty [4].

On the hardware level of GPSRs, any kind of system failure should be avoided. On a practical level, which involves persistent changes in the environment, it becomes much more difficult to account for all possible external failures at design time. Therefore, it is important to note that confronting unforeseen failures is mostly the default state for GPSRs, rather than an exceptional state as often described in the literature. There are some solutions for external failure recovery in the literature, which involve using simulations for the prediction of external failures [5] and logic-based reasoning to account for external failures [6], [7]. However, in most of these cases, the solutions are proposed for specific applications. In the following, we use the word "Failure" instead of the word "External Failure" for conciseness. This means that the focus of our research is not on system/hardware failures. In this paper, we propose an argumentation-based incremental online learning method for recovering from unforeseen failures.

A. Argumentation

Argumentation is a reasoning model based on interaction between arguments [8]. Argumentation has been used in various applications such as non-monotonic reasoning [9], inconsistency handling in knowledge bases [10], and decision making [11]. In [12], Dung has defined an Abstract Argumentation Framework (AF) as a pair of the arguments (whose inner structures are unknown) and a binary relation representing the attack relation among the arguments. Extending Dung's idea, some arguments can support a conclusion and others might be against (attacking) that conclusion in the bipolar argumentation framework [13]. Both the Bipolar Argumentation Framework (BAF) and the Abstract Argumentation Framework (AF) are used in the proposed argumentation-based learning approach.

B. Argumentation in Machine Learning

According to a recent survey by Cocarascu et al. [14], the works using argumentation in supervised learning are listed as follows. Argumentation-Based Machine Learning (ABML) [15] uses the CN2 classification approach [16]. This method uses experts' arguments to improve the classification results. The paper by Amgoud et al. [17] explicitly uses argumentation. There are other approaches for improving classification using argumentation in the literature [18].

Machine learning techniques have also been used for argumentation mining [19], [20], [21]. Bishop et al. combined argumentation with machine learning to prevent failure in deep neural network based break-the-glass access control systems [22].

In contrast with the aforementioned methods, we do not use argumentation for improving the current machine learning approaches or resolving conflicting decisions between current classification methods; instead, we focus on the development of an online incremental learning method. Moreover, the

[1] Department of Artificial Intelligence, Bernoulli Institute, Faculty of Science and Engineering, University of Groningen, The Netherlands.

[2] Institute of Engineering and Technology (ENTEG), Faculty of Science and Engineering, University of Groningen, The Netherlands.

*The corresponding author.

proposed method only uses class labels for the testing phase and not for the training. Therefore, it can be utilized in open-ended (class-incremental) scenarios as well [23].

C. The Expansions

This research is an expansion of the conference paper [24]. The specific expansions are listed as follows.

- The comparison of our proposed Argumentation-Based Learning (ABL) approach with multiple associative reinforcement learning approaches or contextual bandit algorithms has been added to the paper. Contextual bandit algorithms are the most relevant approach to study types of scenarios similar to those presented in this paper.
- Formalizing the proposed method. This includes formalizing the updating procedure of the hypotheses generation unit and hypotheses argumentation unit, formalizing the process of generating hypotheses from the BAF and formalization of the first and second guess generation. In this way, the specification of the method is fully precise and non-ambiguous.
- Extending the proposed method to handle multiple successful recovery behaviors rather than only one successful recovery behavior at each state. Real-world robotic scenarios sometimes have multiple successful recovery behaviors for a failure state.
- Specifying the algorithms in the proposed method by adding pseudocodes to explain argumentation-based learning in more detail. In this way, the computational details of our implemented algorithms are fully explained.
- Validating the argumentation-based learning method outside the robotics scenarios, using a publicly available machine learning dataset (from the UCI repository). This emphasizes the applicability of the proposed method as a general technique for online incremental learning and it shows that this method is not limited to robotics applications.

The rest of this paper is organized as follows. The required background is presented in Section II. Section III introduces the scenarios used in this research. In Section IV, the proposed method has been explained in more detail. Section V presents the experiments and the results obtained from this research. The discussion is presented in Section VI. The conclusion is given in Section VII.

II. Background

The Abstract Argumentation Framework (AF) and Bipolar Argumentation Framework (BAF) are the building blocks of the online incremental learning approach proposed in this paper. AF, BAF and online incremental machine learning algorithms are formally defined in this section.

A. Formal Definition of Abstract Argumentation Framework

An argumentation framework defined by Dung [12] is a pair *AF = (AR, R_{att})* where *AR* is a set of arguments, and R_{att} is a binary relation on *AR*, i.e. $R_{att} \subseteq AR \times AR$. The meaning of $A\ R_{att}\ B$ is that *A attacks B* where *A* and *B* are two arguments. In order to define the *grounded extension* semantics in *AF*, which is used in the proposed learning method, some semantics should be defined first.

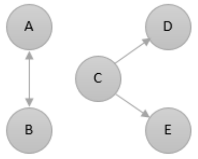

Figure 1: An abstract argumentation framework (AF)

(**Conflict-Free**) Let $S \subseteq AR$. S is conflict-free iff there is no $B, C \in S$ such that B attacks C.

(**Acceptability**) An argument $A \in AR$ is *acceptable* with respect to a set *S* of arguments iff for each argument $B \in AR$: if *B* attacks *A* then *B* is attacked by at least one element of *S*.

(**Admissibility**) A conflict-free set of arguments *S* is *admissible* iff each argument in *S* is acceptable with respect to *S*.

(**Characteristic Function**) The *characteristic function* F_{AF} in an argumentation framework *AF = (AR, R_{att})* is defined as follows:

$F_{AF} : 2^{AR} \rightarrow 2^{AR}$ and

$F_{AF}(S) = \{A | A \text{ is acceptable with respect to } S\}$.

(**Grounded Extension**) The *grounded extension* of an argumentation framework *AF*, denoted by GE_{AF}, is the least fixed point of F_{AF} with respect to set-inclusion [12]. Since F_{AF} is a monotonic function with respect to set inclusion [12], the existence of the fixed point for this function follows from the Knaster-Tarski theorem [25].

Example: Consider the argument set $AR = \{A, B, C, D, E\}$ and the attack relations given by $R_{att} = \{(A, B), (B, A), (C, D), (C, E)\}$ as demonstrated in Fig. 1. Then the conflict-free sets of arguments would be {}, {A}, {B}, {C}, {D}, {E},{A, C}, {A, D}, {A, E}, {B, C}, {B, D}, {B, E}, {D, E}, {A, D, E}, {B, D, E}. Among these, only the sets of {}, {A}, {B}, {C}, {A, C}, {B, C} are admissible. The grounded extension is {C}, which is the least fixed point of F_{AF}.

It can be proved that the grounded extension of the abstract argumentation framework utilized in the proposed argumentation-based learning method is the singleton admissible sets which do not have both incoming and outgoing edges.

B. Formal Definition of an Abstract Bipolar Argumentation Framework

An Abstract Bipolar Argumentation Framework (*BAF*) [13] is an extension of Abstract Argumentation Framework by adding a support relationship. A *BAF* is a triple of the form $< AR, R_{att}, R_{sup} >$ where *AR* is the finite set of arguments, $R_{att} \subseteq AR \times AR$ is the *attack* set and $R_{sup} \subseteq AR \times AR$ is the *support* set. Considering A_i and $A_j \in AR$, then $A_i\ R_{att}\ A_j$ means that A_i attacks A_j and $A_i\ R_{sup}\ A_j$ means that A_i supports the argument A_j.

The semantics of BAF are as follows:

(**Conflict-Free**) Let $S \subseteq AR$. S is conflict-free iff there is no $B, C \in S$ such that B attacks C.

(**Admissible set**) Let $S \subseteq AR$. S is admissible iff S is conflict-free, closed for R_{sup} (if $B \in S$ and $B\ R_{sup}\ C \Rightarrow C \in S$) and

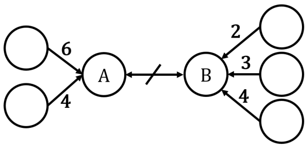

Figure 2: The supporting weights in the Bipolar Argumentation Framework (BAF).

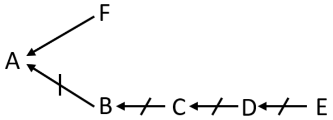

Figure 3: A Bipolar Argumentation Framework (BAF).

S defends all its elements. For instance in Fig. 3, $\{A, C, E, F\}$ is an admissible set since E defends C (i.e. E attacks D which itself is attacking C) and C defends A and no argument attacks F. Therefore, {A, C, E, F} defend all its elements.

(**Preferred extension**) The set $E \subseteq AR$ is a preferred extension iff E is inclusion-maximal among the admissible sets. An inclusion-maximal set among a collection of sets is a set that is not a subset of any other set in that collection.

(**Supporting Weights**) Like [26] the support relations in our model also have an assigned weight. Therefore, a node with higher sum of supporting weights can attack nodes with lower sum of supporting weights. For instance, Fig. 2 shows that the aggregated supporting weight of the argument A is $6+4=10$ and the corresponding supporting weight for the argument B is $2+3+4=9$. Therefore, argument A can attack and defeat B. The $\nrightarrow$ arrows show attack relations and the $\rightarrow$ arrows demonstrate support relations in Fig. 2 and Fig. 3. The formal definition of the supporting weights function is defined in Eq. 8 in Section IV-D.

Figure 3 shows a bipolar argumentation framework. The admissible sets are {}, {E}, {A, C, E}, {A, C, E, F}. The preferred extension in this BAF is {A, C, E, F}.

C. *Formal Definition of Online Incremental Machine Learning Algorithms*

We define an incremental learning approach that uses a sequence of data instances $d_1, d_2, ..., d_t$ for generating the corresponding models $M_1, M_2, ..., M_t$. In case of incremental online learning, each data instance d_i incrementally updates the model and $M_i : \mathbb{R}^n \rightarrow \{1, ..., C\}$, where C is the number of class labels, is representing the model which depends on M_{i-1}. The online learning is then defined as an incremental learning which is also able to continuously learn. Incremental learning approaches have the following properties:

- The model should adapt gradually, i.e. M_i is updated using M_{i-1}.
- The previously learned knowledge should be preserved.

A recent study on the comparison of the state-of-the-art methods for incremental online machine learning [27] shows that Incremental Support Vector Machines (*ISVM*) [28], [29] together with LASVM [30], which is an online approximate SVM solver, and Online Random Forest (*ORF*) [31] outperform the other methods. The comparison methods used in our paper have been chosen based on the aforementioned survey [27].

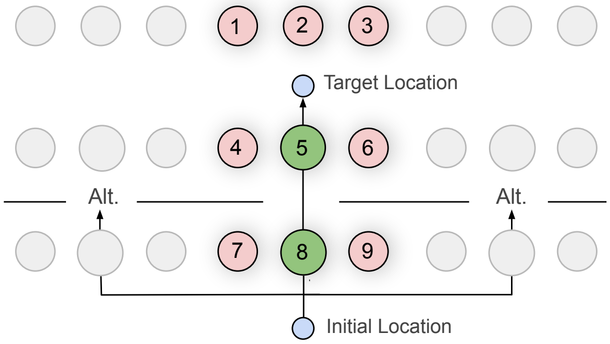

Figure 4: Schematic overview of the possible failure state scenarios. Only the green location is relevant for finding the best recovery behavior. Alt. stands for the Alternative Route recovery behavior.

The proposed argumentation-based incremental learning approach uses the bipolar argumentation framework to model the visited data instances and generate relevant hypotheses. Subsequently, the abstract argumentation framework is used to model the defeasibility relations (i.e. the attack relations) between the current set of generated hypotheses and predict the best action (recovery behavior) for an unforeseen incoming data instance. Furthermore, the model incrementally gets updated as new data instances enter the model.

III. Scenarios

The performance of the different methods is tested using three test scenarios. The aim of the first two test scenarios is to model a situation where a programmer has provided an initial solution (e.g., a top level behavior such as entering the room), while (s)he has not accounted for all possible failures (e.g., objects and persons blocking the entrance), but allows the robot to find new solutions whenever a (previously unseen) failure occurs.

The basic setup of the first two test scenarios is illustrated in Fig. 4. The high-level behavior of the robot aims to proceed from the initial location to the target location using three entrances. Different obstacles might be on its way to the target location. In these scenarios, an agent observes all the obstacle locations at once and chooses a single recovery behavior (action) for recovering from that failure state. The agent can reach the goal if it chooses the best recovery behavior; otherwise, it fails to reach the goal.

In order to make the explanation of the method simpler, we first concentrate on finding only the best recovery behavior for each failure state. In the method Section IV, we will also explain how to generalize the method to scenarios where multiple recovery behaviors might be successful in a failure state.

A. *Recovery Behaviors*

Whenever the robot is confronted with a failure state, it may use any of the following recovery behaviors to resolve the issue. The run-time of each recovery behavior in seconds is presented in parentheses in front of each recovery behavior:

- Continue (2s): This solution is only useful if the failure has resolved itself (e.g., the obstacle moved away just after the failure).
- Push (5s): The robot can try pushing any obstacle.
- Ask (4s): The robot can try to ask any type to move.
- Alternative Route (Alt) (10s): The robot can move to another entrance to reach the target location.

It is important to note that choosing Alternative Route as the best recovery behavior may not always lead to success, because the robot may again be confronted with new obstacles (Fig. 4). Moreover, the best recovery behavior not only depends on the run-time of each recovery behavior, but also on the type, the color and the location of the obstacles.

B. Test Scenario 1

In this scenario, three types of obstacles (ball, box or person) with four colors (red, blue, green or yellow) can be presented in one of the locations 1 to 6 (Fig. 4). Locations 7 to 9 play no role in this scenario. There can be either zero or one combination of color-type in each location. Only location number 5, marked in green (Fig. 4), is relevant for choosing the best recovery behavior. It is important to notice that the robot does not know this fact and it should infer that the only effective location is location number 5 by observing different failure states in the environment.The agent observes all the obstacle locations at once and chooses a single recovery behavior (action) at each state. A new state is generated randomly at each time step. The number of possible combinations of the color-type in each location is 13 (3 types × 4 colors + "no obstacle" = 13). Since there are 6 locations in this scenario, the number of all possible states in this scenario is $13^6 = 4,826,809$.

Notice that colors can have meaningful interpretations for each type of obstacle. For instance, the red object might be heavy and cannot be pushed, while green ones are light. On the other hand, red people can be more cooperative and move out of the robot's way when being asked. Therefore, the colors can represent any realistic feature for the people and the objects. Using the colors instead of these realistic features simplifies the scenarios with fewer features.

C. Test Scenario 2

This scenario is more complex than the first scenario since each color-type combination can be presented in any of the nine possible locations. Here, only the green locations 5 and 8 are required for determining the best recovery behavior. Once again, the robot does not know this fact and it should infer that the only effective locations are the location number 5 and 8 by observing different failure states in the environment. The agent predicts a single recovery behavior (action) while it can observe all the obstacle locations at once at each state. The number of all possible states in this scenario is $13^9 = 10,604,499,373$.

D. Test Scenario 3

The third scenario has a different purpose and context. It shows the applicability of the proposed method outside the robotics field. The recent study on online incremental machine learning techniques [27] used the publicly available datasets from the UCI machine learning repository [32]. We also used the SPECT heart dataset from the UCI machine learning repository. This dataset represents the diagnosis of cardiac Single Proton Emission Computed Tomography (SPECT) images. Each of the images (patients) is even classified as normal or abnormal. The database of 267 SPECT image sets has been processed for extracting features that summarize the original SPECT images. We randomly selected 40 out of 267 data instances and fed them incrementally to the incremental learning approaches in order to compare the results.

The SPECT heart dataset has recently been used in various researches [33], [34], [35], [36].

Notice that we do not use class labels in the training and that the label for each class is determined autonomously based on a trial and error procedure in our proposed method. Class labels are only used for testing the performance of the model for prediction on an unforeseen data instance.

IV. Method

In this section, we will discuss the proposed argumentation-based learning method for recovering from an unforeseen failure state.

A. Argumentation-Based Learning (ABL)

In order to explain *ABL*, we first use a simplified version of the previous test scenarios where there is only one location ahead of the robot (instead of 6 or 9). When there is no obstacle ahead of the robot, the best recovery behavior is "Continue".

Assume that the robot confronts a blue-ball blocking the entrance. Since there is no pre-trained model yet, the robot tests different recovery behaviors in order of their run-time to find the best one. Supposing that pushing the ball was successful in this case, the robot should learn from this experience.

However, unlike the traditional tabular reinforcement learning techniques, only learning the best recovery behaviors (actions) for exactly the same experiences (states) is not enough. We need a learning approach capable of inferring the correlated feature values (each feature value is the color or type of the obstacle at each location or an empty location with no color and type) for choosing the best recovery behavior. This is known as *generalization* in the machine learning literature. For instance, confronting a red ball and a green ball with the same recovery behavior of pushing, the robot should make a new hypothesis *push a ball*. Therefore, the next time the robot confronts the yellow ball, it can easily infer that *Push* is the best recovery behavior.

Confronting a yellow ball with *Alternative Route* as the best recovery behavior contradicts the previous hypothesis. Therefore, a new hypothesis is made: *Push a ball unless it's yellow*. From an argumentation perspective, we can see each hypothesis as an argument. Therefore, the second generated hypothesis can attack and defeat the first argument. This is inspired by human agents who make new hypotheses from their perceptions and reason about the best course of action at each state.

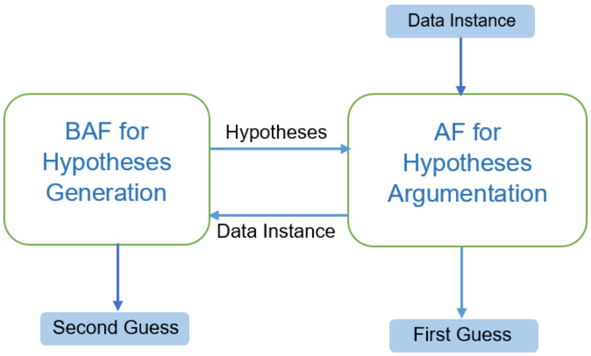
Data Instance
BAF for Hypotheses Generation
Hypotheses
AF for Hypotheses Argumentation
Data Instance
Second Guess
First Guess

Figure 5: Architecture of the proposed Argumentation-based learning method.

The architecture of the proposed argumentation-based learning method is shown in Fig. 5. A bipolar argumentation framework is used as hypotheses generator unit and an abstract argumentation framework models the defeasibility relation between these generated hypotheses.

Algorithm 1 presents the pseudocode of argumentation-based learning. When a new data instance enters the model, all the combinations of its feature-values and the set of nodes in the grounded extension of the AF will be extracted. Each node (argument) in the AF unit is of the form *precondition* → *post-condition: weight*. According to the similarity between the *preconditions* of the arguments in the grounded extension and the feature values combinations, there will be three possible cases. Either there will be a unique similarity, multiple similarities or no similarity. In case of unique similarity, the post-condition of the argument (which is a recovery behavior) will be used as the first guess and will be applied to the environment to see the result. On the occasion that there exist multiple similarities, the recovery behavior with the highest weight among the arguments will be chosen and its post-condition will be applied to the environment. A successful recovery from the failure state will update the BAF unit using Algorithm 2. On the other hand, failure from recovery will lead to generating the second guess (Algorithm 3), updating the BAF unit (Algorithm 2), generating hypotheses from BAF unit (Algorithm 4) and updating the AF unit (Algorithm 5), respectively.

We now use an illustrative example to explain the proposed method in more detail.

B. Example

Table I shows the best recovery behavior when the robot confronts an obstacle with different colors and types. Notice that this table is only used for this example and a randomly generated table is utilized for each of the 1000 independent runs for the experiments. Figures 6 to 8 show the updating procedure of the model step by step. In the hypotheses generation unit (*BAF*), an arrow → shows a support relation between arguments and $\nrightarrow$ shows an attack relation between them. However, in *AF*, → shows an attack relationship between the arguments.

Referring to Table I, at the beginning of the learning procedure, the robot confronts a Red-Ball (R-Ba). It tests all the recovery behaviors in order of their run-times and finds the *Push* recovery behavior as a success (Table I). Subsequently, the Bipolar Argumentation Framework is getting updated as in Fig. 6. In order to update the *BAF*, first, the best recovery node is added which is *Push* in this case. Then all the possible combinations of the feature-values of the current state are added as supporting nodes. The supporting nodes for *Push* are *R*, *Ba* and *R-Ba*. If there previously exists the same supporting node, its supporting weight will be increased. For instance in Fig. 7, where *8:B-Bo* enters the *BAF*, since *B* and *B-Bo* are new supporting nodes for the *Alt (Alternative Route)* recovery behavior, they are added to the model with a supporting weight equal to 1. On the other hand, *Bo* already exists in the set of supporting nodes for *Alt* and its weight is increased. After updating the supporting weights, a set of hypotheses is generated based on the number of occurrences of each supporting node. For instance, after observing 1:Red-Ball (R-Ba), *R* → *Push* and *Ba* → *Push* are added to the AF unit.

Confronting *2:R-Bo* and using the previously generated

Algorithm 1: Argumentation-Based Learning pseudocode

```
input: Current BAF Graph, Current AF Graph, Data Instance DI
  entering the argumentation-based learning model
output: List of best recovery behaviors called BRB-list
- Extract all feature-value combinations DI and add them to a list
  called Combs.
- Find the set of nodes in grounded extension (GE of AF)
for (all gx in GE) do
    for (all comb in Combs) do
        if (gx.precondition==comb) then
            BRB-list.Add(gx.post-condition)
if (BRB-list is not empty) then
    if (BRB-list.Length==1) then
        - Apply BRB-list[0] to environment and observe the result.
    else
        - Select a recovery behavior in BRB-list with highest weight.
if (result==failure) OR (BRB-list is empty) then
    - Use BAF unit for second guess generation and add these guesses
      to BRB-list by using Algorithm 3
    - Update the BAF unit using Algorithm 2.
    - Generate Hypothesis from the updated BAF using Algorithm 4.
    - Update the AF unit using the generated hypotheses using
      Algorithm 5.
if (result==success) then
    - Update the BAF unit using Algorithm 2.
return BRB-list
```

Order	Color	Type	Best Recovery Behavior
1	Red	Ball	Push
2	Red	Box	Alternative Route
3	Red	Person	Ask
4	Green	Ball	Push
5	Green	Box	Alternative Route
6	Green	Person	Ask
7	Blue	Ball	Push
8	Blue	Box	Alternative Route
9	Blue	Person	Alternative Route
10	Yellow	Ball	Push
11	Yellow	Box	Alternative Route
12	Yellow	Person	Ask
13	None	None	Continue

Table I: Possible combinations of color-type with the best recovery behaviors.

Algorithm 2: Updating Hypotheses Generation Unit

```
input: Current BAF Graph, New Data Instance (DI) and the Best
  Recovery Behavior BRB
output: BAF Graph

- Extract all feature-value combinations of DI and add them to a list
  called Combs.
if (BRB is not in BAF) then
    - Add BRB to the BAF graph;
    - Add bidirectional attack edges between BRB node and all other
      Recovery Behavior Nodes (RBN) in BAF;
for (any item in Combs) do
    - Boolean isNewCombination = true
    for (any sup in BRB.supporting-nodes) do
        if (item == sup) then
            sup.weight += 1;
            isNewCombination = false;
    if (isNewCombination) then
        BRB.supporting-nodes.Add (item);
```

hypotheses (specifically $R \rightarrow Push$), the robot would infer that the best possible recovery behavior is *Push*, which is a wrong choice in this case (Table I). Therefore, the robot tries other recovery behaviors and finds *Alt* as success and updates the model accordingly. Moreover, a bidirectional attack will be added among all the recovery nodes in the *BAF* (in this case, *Alt* and *Push*). Subsequently, the new set of hypotheses is generated to update the hypotheses argumentation unit. Finally, an abstract argumentation framework is updated to model the attack relations between the set of generated hypotheses (arguments). This *BAF-AF* update cycle goes on and on during the learning procedure.

In this small example, seven out of thirteen predictions of the model are correct, and only two are wrongly classified using the proposed argumentation-based learning. In other cases, our system can provide multiple probable guesses. For instance, when *12:Y-P* enters the system in Fig. 8, the *AF* cannot provide any suggestion but the *BAF* will suggest both *Ask* and *Alt* as the candidate recovery behaviors. However, the mapping of the states to the best recovery behavior is randomly generated in all the experiments.

C. Hypotheses Generation Unit (BAF Unit)

This unit has two roles. Firstly, it generates a new set of hypotheses whenever the *AF* unit could not classify the new data instance correctly (1). The second role of this unit is to produce a second guess for the best recovery behavior (2):

1) In order to generate a new set of hypotheses from the constructed *BAF*, only one recovery behavior is considered which is highlighted with a red box in Fig. 6 to 8. The pseudocode shown in Algorithm 4 shows the procedure of hypotheses generation.

The pseudocode of updating the current hypotheses generation graph (*BAF* unit) using a new data instance is shown in Algorithm 2. The only nodes which are getting updated during this process are the best recovery behavior for the current data instance and its supporting nodes. Autonomously identifying the best recovery behavior through trial and error, the update procedure for hypotheses generation takes place. The updating procedure searches for a node in the *BAF* graph with the best recovery behavior and appends all the possible combinations of the feature-values of the current state to the support nodes of the best recovery behavior node. In case that a supporting node already exists in the best recovery behavior node, its supporting weight is incremented.

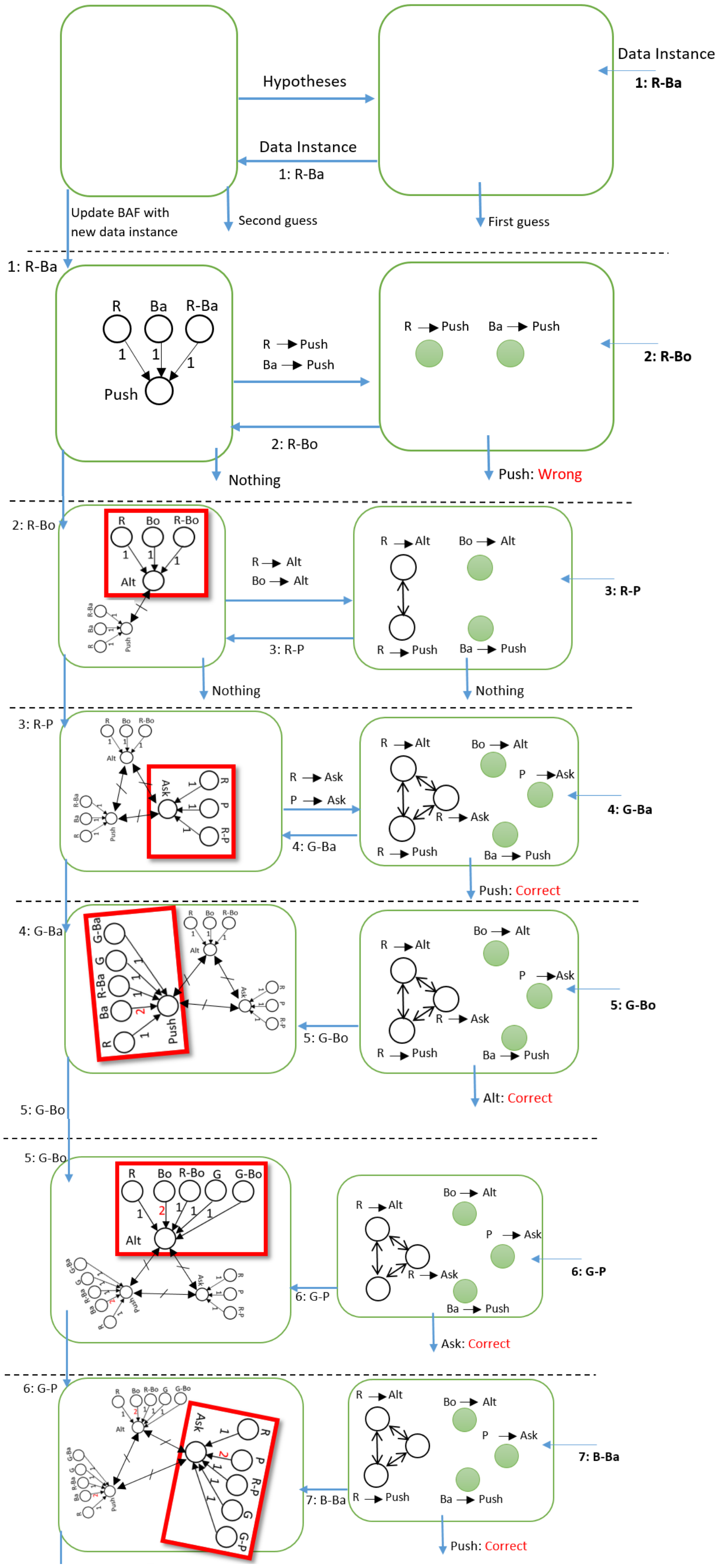

Figure 6: Example of Argumentation-Based Learning for the illustrative example. First part

2) In order to generate a second guess, a new *BAF* should

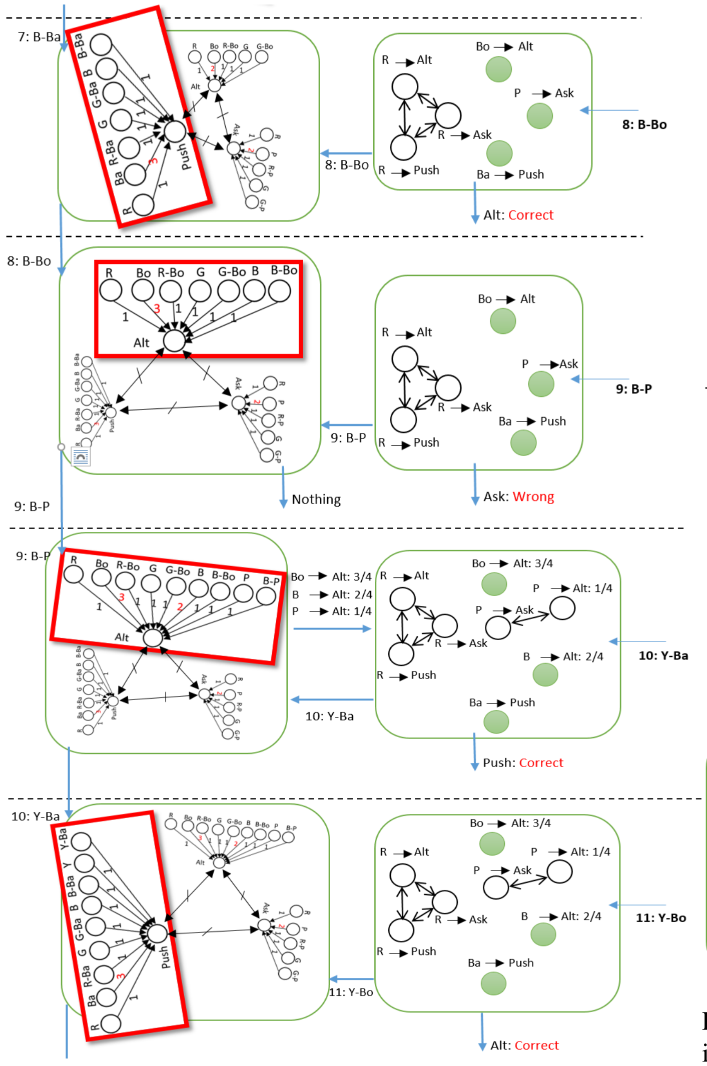

Figure 7: Example of Argumentation-Based Learning for the illustrative example. Second Part

be constructed. For an unforeseen failure state, the set of all possible combinations of feature-values is compared with the supporting nodes of each recovery behavior node. According to the sum of the matching supporting weights, the attack relations are adapted among the recovery behaviors. Therefore, only recovery behaviors with a higher sum of the matching supporting weights can attack the other recovery behavior. For instance, in the example, when *12: Y-P* enters the model for prediction, the *AF* is not be able to guess the best recovery behavior. Constructing a new *BAF* for a second guess, shown in Fig. 9, the calculated weighted sum for the *Alternative Route (Alt)* node is the same as *Ask* and higher than *Push*. Accordingly, the attack relations get updated. Using preferred extension semantics and its intersection with recovery behavior nodes, both *Alternative Route (Alt)* and *Ask* are chosen as the second guesses.

D. Formal Representation of Updating Procedure of the Hypothesis Generation Unit (Algorithm 2)

In Section II-C, the formal definition of online incremental learning is represented. The sequence of labeled data instances $d_1, ..., d_t$ is entering the model and the *BAF* unit gets updated.

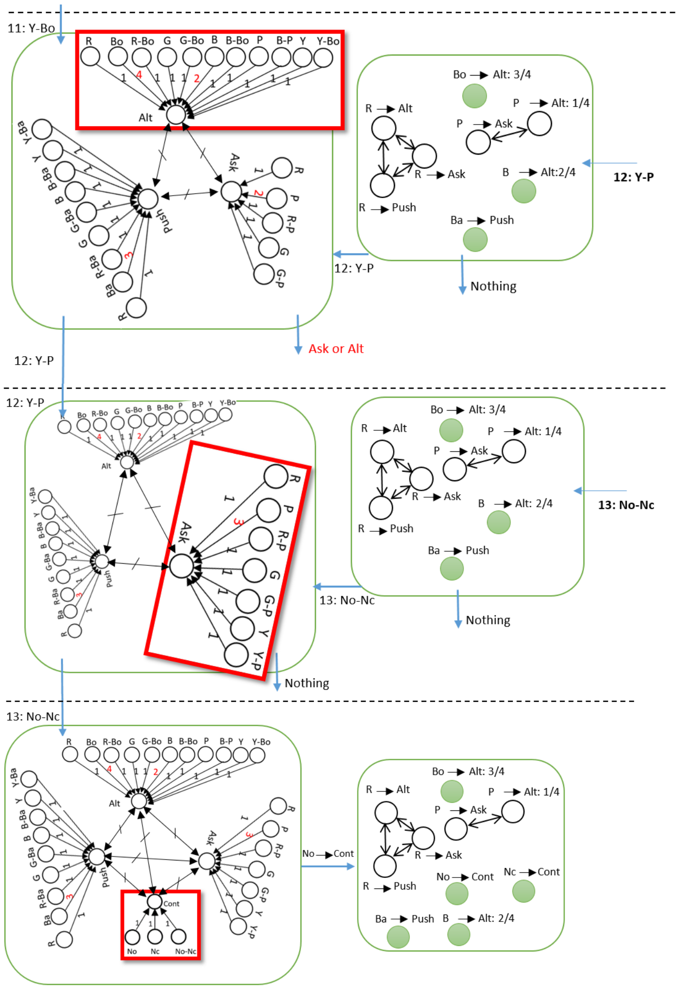

Figure 8: Example of Argumentation-Based Learning for the illustrative example. Third Part

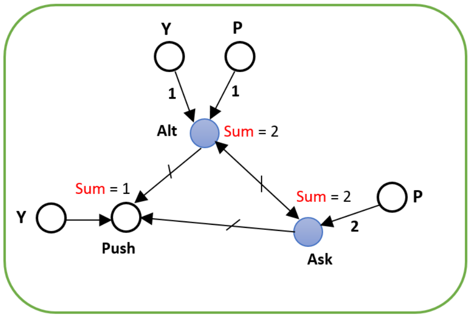

Figure 9: The generated *BAF* when Yellow-Person (*12:Y-P*) enters the model. Blue nodes show the intersection of preferred extensions and recovery behavior nodes.

The hypotheses generation unit is represented by BAF_{t+1} when a data instance d_t is entering this unit.

$$BAF_0 =< AR_0, R_{att_0}, R_{sup_0} >=< \emptyset, \emptyset, \emptyset >$$
$$BAF_{t+1} = update(BAF_t, d_t) \qquad \forall t \geq 0 \qquad (1)$$

In the following lines, the update procedure for the *BAF* model is described. The *BAF* model at time *t+1* is in the following form:

$$BAF_{t+1} =< AR_{t+1}, R_{att_{t+1}}, R_{sup_{t+1}} > \qquad (2)$$

Algorithm 3: Second Guess Generation Pseudocode

```
input: Current BAF Graph, Set of Recovery Behavior Nodes (RBN)
output: Set of recovery behaviors

- Generate a new graph G from BAF with the same set of nodes.
- G.sup = BAF.sup
for Node a in RBN do
    for Node b in RBN do
        if a != b then
            if (a.weight > b.weight) then
                G.attacks.add(attack(a,b));
            else if (a.weight == b.weight) then
                G.attacks.add(attack(a,b));
- return the set of nodes in the preferred extension of G.
```

Using the best recovery behavior at time *t* called BRB_t (This is determined by trial and error in the environment) and the set of all the subsets of feature values in the n-dimensional data instance $d_t = (f_{1_t}, f_{2_t}, ..., f_{n_t})$ (f_{i_t} shows the ith feature value of the n-dimensional d_t vector), called $Combs(d_t)$, the arguments set of the *BAF* gets updated in the following form:

$$AR_{t+1} = AR_t \cup BRB_t \cup Combs_t \tag{3}$$

where

$$Combs_t = \{P \subseteq \{f_{1_t}, ..., f_{n_t}\}\} \tag{4}$$

In addition to the set of all the arguments *AR*, we need to keep track of the set of the Recovery Behavior Nodes (*RBN*) among the arguments in the following way:

$$RBN_t = \{BRB_0, ..., BRB_{t-1}\} \tag{5}$$

The attack relation $R_{att_{t+1}}$ is getting updated using the current set of the Recovery Behavior Nodes RBN_t and the best recovery behavior BRB_t.

$$\begin{aligned} R_{att_{t+1}} = R_{att_t} &\cup \{att(BRB_t, b) | b \in RBN_t\} \\ &\cup \{att(b, BRB_t) | b \in RBN_t\} \end{aligned} \tag{6}$$

The support relations between the arguments are getting updated as follows.

$$R_{sup_{t+1}} = R_{sup_t} \cup \{sup(c, BRB_t) | c \in Combs_t\} \tag{7}$$

For instance in the example, when the *1:R-Ba* enters the *BAF* unit (Fig. 6), all the combinations of this data instance {*R*, *Ba*, *R-Ba*} are added as support nodes to the current best recovery behavior node, which is *Push*.

There is also a weight function $W_t : R_{sup} \rightarrow \mathbb{N}^+$ which specifies the weights of the support relations in R_{sup} at time *t*. Whenever R_{sup} gets updated, the corresponding weights for the support relations update in the following way:

$$\begin{gathered} \forall c \in Combs_t : W_{sup(c,BRB_t)_{t+1}} = \\ \begin{cases} \mathrm{W}_{sup(c,BRB_t)_t} + 1 & \text{if sup(c,BRB}_t) \in R_{sup_t} \\ \\ 1 & \text{otherwise} \end{cases} \end{gathered} \tag{8}$$

Here, $W_{sup(c,BRB_t)_t}$ is the weight of the support relation $sup(c, BRB_t)$ at time *t*. Eq. 8 means that if the supporting node *c* has been already existed in the *BAF* unit, then its weight is incremented. Otherwise, its supporting weight is set to 1.

Algorithm 4: Hypotheses Generation Pseudocode.

```
input: Current $BAF$ Graph, Threshold, the best recovery behavior and
  the latest hypothesis with wrong recovery behavior called WrongRule
output: The set of generated hypotheses

- Choose the Best Recovery Behavior node called BRB.
- Normalize the supporting weights of BRB to [0, 1].
- Sort BRB.supporting-nodes according to their weight values from
  high to low.
- Sum = 0;
- Hypotheses-List = Empty;
for (any sup in BRB.supporting-nodes) do
    if (sup.weight > Threshold) then
        Add sup $\rightarrow$ BRB to the Hypotheses-List;
for any (A $\rightarrow$ BRB) in Hypotheses-List do
    for any B $\rightarrow$ BRB in Hypotheses-List do
        if (A $\supset$ B) then
            Remove (A $\rightarrow$ BRB) from Hypotheses-List;
Add WrongRule.$Precondition$ $\rightarrow$ BRB to Hypotheses-List;
return Hypotheses-List;
```

E. Formal Representation of Generating the Second Guesses using BAF (Algorithm 3)

For generating the second guess using the incoming data instance d_t, another *BAF* should be constructed. Fig. 9 shows the new extracted *BAF* when the *12:Y-P* enters model. It is almost the same as the main hypotheses generation unit. However, only the attack relations R_{att} should be adapted as follows.

$$\begin{gathered} R_{att_t} = \{att(a, b) \ | \ a, b \in RBN_t, \ x, y \in Combs_t, \\ (\sum_{(x,a) \in R_{sup_t}} W_{sup(x,a)_t} \geq \sum_{(y,b) \in R_{sup_t}} W_{sup(y,b)_t})\} \end{gathered} \tag{9}$$

Only the recovery behavior node with the higher aggregated supporting weights can attack the other recovery behavior node in the generated *BAF*. For generating a second guess, the *preferred extensions* semantics is used to choose the best recovery behavior nodes as the second guess. Therefore, the elements in the intersection of the *preferred extensions* set and the set of recovery behavior nodes RBN_t are selected.

F. Formal Representation of Hypotheses Generation (Algorithm 4)

Using the updated Bipolar Argumentation Framework (*BAF*) from the previous subsection, the set of hypotheses can be generated. Therefore, we can inductively define the hypothesis set as follows:

$$\begin{gathered} HS_0 = \emptyset; \\ HS_{t+1} = GenerateHypothesis(BAF_{t+1}, BRB_{t+1}, NC_t); \end{gathered} \tag{10}$$

Here, NC_t is the hypothesis used in the AF unit and was Not Correctly (*NC*) determined the action (recovery behavior). We also count the number of times the recovery behavior BRB_{t+1} was the best recovery behavior until now and call it $CBRB_{t+1}$. Each hypothesis in the hypotheses has the form of *precondition (pre)* → *post-condition (post): weight* where weight is the hypothesis weight. Whenever a hypothesis is shown in the form *pre* → *post* instead of the previous form, it means that the hypothesis weight is equal to 1. The formalization of generating the hypotheses set is as follows:

$$\begin{aligned} HS_{t+1} = \big\{(A \to BRB_{t+1}) : weight \mid A \in AR \setminus RBN_{t+1} \\ , weight = \frac{W_{sup(A,BRB_{t+1})_{t+1}}}{CBRB_{t+1}},\ sup(A,BRB_{t+1}) \in R_{sup_{t+1}} \\ , Normalized(W_{sup(A,BRB_{t+1})_{t+1}}) \geq threshold, \\ \forall a \in A\ \nexists b \in A : a \subset b\big\} \cup \{(NC_t.pre \to BRB_{t+1})\} \end{aligned} \tag{11}$$

Here, the *threshold* ∈ [0,1] and *Normalize* is the linear normalization function for $W_{sup(A,BRB_{t+1})}$. This equation means that when the best recovery behavior is determined, it is used as the post-condition of the hypothesis and its supporting nodes with a weight higher than a specific threshold are chosen as the pre-condition. The hypothesis weight is also computed based on the supporting weight of the supporting node in the pre-condition and the number of times the current recovery behavior was the best recovery behavior so far. Choosing a low *threshold* value means generating more hypotheses. After an extensive set of experiments, we found out that *threshold = 0.4* was a good value in all the experiments.

G. Hypotheses Argumentation Unit using AF

As stated in the previous sections, this unit tries to justify what has been learned so far by updating the attack relations between the arguments (hypotheses). The arguments in this framework can only bidirectionally attack each other when they have the same preconditions but different post-conditions.

When a new data instance enters the model, there are three possible cases for the set of hypotheses in the grounded extension of the *AF*. When the grounded extension of the *AF* is the empty set, the second guess is generated by the *BAF* unit. If one argument with the same post-condition exits in the grounded extension of the *AF*, then this post-condition will be the *AF*'s first guess. If more than one argument with different recovery behaviors in their post-condition was chosen, the weights of arguments determine which argument has more power to be selected. For instance in the example, if blue-ball enters the model after it has been trained using the complete set of data in Table-I, both *B* → *Alt: 2/4* and *Ba* → *Push:1* can be used for prediction. Since the *Ba* → *Push:1* has higher weight, the *Push* recovery behavior will be chosen, which is the correct choice for this failure state. Notice that in the proposed argumentation-based learning method, it can be proved that the grounded extension is a set of the singletons in the *AF*.

Algorithm 5 shows the updating process of the hypotheses generation unit.

H. Formal Representation of Updating Procedure of Hypotheses Argumentation Unit (Algorithm 5)

The hypotheses generation unit is represented by AF_t when data instance d_{t-1} is entering this unit for updating.

$$AF_0 =< AR, R_{att} >=< \emptyset, \emptyset >$$

$$AF_t = update(AF_{t-1}, HS_t) \quad \forall t \geq 1 \tag{12}$$

Algorithm 5: Updating Hypotheses Argumentation Unit

```
input: Current AF Graph, the new set of generated hypotheses HS
   from BAF unit
output: AF Graph

for (all item in HS) do
    - Add item to set of AF.arguments
    - Update the attack relations between arguments as follows
    for (all arg in AF.arguments) do
        if (arg.pre == item.pre) & (arg.post != item.post) then
            AF.attacks.Add(attack(item, arg))
            AF.attacks.Add(attack(arg, item))
return AF
```

In time t the Abstract Argumentation Framework (AF) is:

$$AF_t =< AR_t, R_{att_t} > \tag{13}$$

Here the argument set AR_t is updated at time *t* using all elements in the recently generated hypotheses set HS_t and the previous arguments set AR_{t-1} as follows:

$$AR_t = AR_{t-1} \cup HS_t \tag{14}$$

The attack relationship R_{att_t} is also get updated whenever two arguments have the same preconditions but different post-conditions:

$$\begin{aligned} R_{att_t} = R_{att_{t-1}} \cup \big\{att(x,y) | x \in AR_t, y \in AR_t, \\ x.pre = y.pre \wedge x.post \neq y.post\big\} \end{aligned} \tag{15}$$

Here, the *att(x,y)* is the abbreviation for *x* R_{att} *y*. Figures 6, 7 and 8 show this process. Whenever the hypothesis *R* → *Push* enters the AF unit, since it has the same precondition but different post-condition with respect to the existing hypothesis *R* → *Alt* in *AF*, they will bidirectionally attack one another.

Each time a new data d_t enters the AF unit for the first guess generation, the *grounded extension* called GE_{t+1} is computed. Using $Combs_{t+1}$, the Best matching Hypothesis BH_{t+1} is chosen to generate the first guess in the following way.

$$BH_{t+1} = \big\{A \in H_{t+1} | B \in H_{t+1}, A.weight \geq B.weight\big\} \tag{16}$$

where

$$H_{t+1} = \big\{h \in GE_{AF_{t+1}} \mid h.pre \in Combs_{t+1}\big\} \tag{17}$$

This means that only the hypothesis with the highest weight can be selected as the best matching hypothesis. Subsequently, the first guess is the post-condition of the current best hypothesis:

$$FG_{t+1} = BH_{t+1}.post \tag{18}$$

I. Why not Reinforcement Learning?

Reinforcement Learning (RL) techniques learn by interacting with the environment. Like our proposed method, these methods effectively learn with trial and errors, by performing actions and remembering their consequences [37]. Traditional tabular reinforcement learning methods are inefficient for large state spaces [38]. Moreover, most traditional tabular reinforcement learning techniques do not take the similarity of the features of each state into account, which is needed for the robotic scenarios in this paper. However, there are a few exceptions. Some more recent tabular RL techniques have the generalization capability and take the similarity of features into account [39], [40]. In order to include the generalization capability into traditional tabular RL techniques, a non-linear function approximation technique like artificial neural networks is incorporated to handle the large state space and account for the similarity of the features among the states. However, the robotics scenarios in this research have the following properties which make these RL methods behave similarly to a neural network:

- In these scenarios, the next state is not dependent on the current state and the current action because the simulated failure states are generated randomly in the experiments.
- As a consequence of independence between two consecutive states, there is no delayed reward in the corresponding robotic scenarios. Therefore, only the instant rewards, that are dependent on the success of choosing the best recovery behavior at each state, are enough for the formulation.

Considering these properties, the function approximation of the reinforcement learning approach is like a neural network which takes the current state and the current action and outputs the instant rewards. Using such a neural network, the next step is to find the action with the highest instant reward for that state to be selected as the best recovery behavior. This is similar to having a neural network which takes the current state as the input and outputs the best recovery behavior in that state. This network has been implemented as an MLP neural network in the results section. Moreover, we have compared our method with contextual bandit algorithms.

J. Contextual Bandits

Contextual bandits or associative reinforcement learning techniques have been used for scenarios similar to those studied in this paper. Therefore, we compare the performance of the proposed ABL technique with various online contextual bandit algorithms.

Contextual bandit is defined as follows. There is an agent who can choose between a number of choices (known as “arms”), which can lead to stochastic rewards. In each round, the current state is generated, which is a set of features of a fixed dimensionality that is known as “context”. The agent chooses an arm at each round and the corresponding reward for that action in that specific context is returned as a feedback to the agent. The ultimate goal of the agent is to find a policy that maximizes the long-term rewards using the history of previous actions.

Most research on finding an efficient algorithm for contextual bandit problems in the last decade can be divided into two categories, namely Upper Confidence Bounds based algorithms (UCB) [41], [42], [43], [44] and Thompson Sampling algorithms (TS) [41], [45], [46], [47]. Zhou el al. [48] proposed an offline multi-action learning approach which can take constraints on the learning policy into account, for instance budget constraints. In Section V, we will compare our method with both UCB and TS approaches.

K. Generalizing ABL to Other Real-Word Scenarios

So far, we have assumed that at each failure state only one recovery behavior is successful and the others fail. However, this assumption might not be the case in all the real-word scenarios. Therefore, in the following paragraphs, we explain how we can generalize the *ABL* method to handle multiple successful behaviors.

Like Reinforcement Learning (RL) techniques, each action (i.e. a recovery behavior in our case) must have a reward reflecting how good it is. For example this reward can be a function of the run-time of that recovery behavior where the lower run-time leads to higher reward. This reward function for each recovery behavior can be formulated as follows.

$$R = \begin{cases} 0 & \text{Failure} \\ \frac{1}{\text{run-time}} & \text{Successful Recovery Behavior} \end{cases} \tag{19}$$

Using the epsilon-greedy algorithm [49] for choosing the different recovery behaviors at each failure state, we are able to have a trade-off between the exploration of the new recovery behaviors and the exploitation the previously successful recovery behaviors. When a new recovery behavior is explored and it is successful, then the BAF unit in *ABL* should generate a new set of hypotheses based on that recovery behavior and its run-time.

Furthermore, the hypothesis format in the AF unit of the *ABL* method should be changed. The new hypotheses have the form $pre \rightarrow post : weight : reward$. Subsequently, the set of hypotheses with the highest rewards in the grounded extension of the AF unit is found for choosing the best recovery behavior. If the hypotheses used in the previous step have the same rewards, the *weight* is used to choose the best recovery behavior as before.

The required changes in the *ABL* algorithm are listed below:

- The line “choose the Best Recovery Behavior node called **BRB**” in the Algorithm 4 should change to “choose the Best Recovery Behavior node called **BRB** based on the epsilon-greedy algorithm ($\epsilon = 0.05$)”
- The Reward is added to the format of each hypothesis.
$$pre \rightarrow post : weight : reward$$
- The Best Recovery Behavior from the AF unit is chosen based on the grounded extension, the reward and the weight of each hypothesis.

This methodology is only needed when we don’t know the rewards of each recovery behavior (action) in advance. If we previously know the rewards for each of the recovery behaviors (as in our experiments), with the following modification, we can use the same *ABL* method as before. At each point a trial and error procedure takes place based on the ordering of the recovery behaviors from the one with the highest reward (lowest run-time) to the one with the lowest reward (highest run-time). This guarantees that the first successful recovery behavior is always a best choice.

V. Experiments

In this section, we compare the performance of our proposed *ABL* method with other incremental learning techniques and contextual bandits algorithms. The survey by V. Losing et al. compared a broad range of incremental online machine learning techniques [27]. Using the key methods in their survey, we are also comparing the proposed method with

Incremental Support Vector Machine (*ISVM*) [28], [50], [51], incremental decision tree based on *C4.5* [52] and ID3, incremental Bayesian classifier [53], Online Random Forest (*ORF*)[31] and Multi-Layer Neural Networks for classification with localist models like Radial Basis Functions (*RBF*) which work reliably in incremental settings [54], [55].

Cortes has recently compared the performance of different contextual bandit algorithms in his paper [56]. He adapted Multi-Arm Bandits (MAB) policies to contextual bandit algorithms. We have also compared our proposed ABL technique to various online contextual bandit approaches.

A. Performance Measure

The mean performance of each method is calculated over 1000 independent runs. Each run for the robotic scenarios consists of 200 failure recovery attempts. Since the order of the failures has a direct effect on all the open-ended online incremental learning methods like ours, the order of failures is randomized for each run in which there is an equal uniform probability for each solution to be a success.

We are interested in knowing whether the method picked the best recovery behavior or not for a given failure state.

For the third test scenario, we randomly choose 40 data instances for online train and test procedure.

Notice that all the methods use the same randomly generated data set compatible with the conditions mentioned in the test scenarios.

Furthermore, the mapping of each state to the best recovery behavior (a table like Table I), which is used for testing the performance of the model, is randomly generated at each of the 1000 independent experiments.

B. Comparison criteria

In the robotic scenarios, we need a learning approach which can quickly learn to recover from failure states in a low number of attempts. Moreover, for other test scenarios, the goal is to incrementally learn from a lower number of training instances. Therefore, the increase in learning precision in a lower number of attempts is one important criterion (which we call *learning speed*) to evaluate the efficiency of the method [57]. Therefore, learning curves with the highest steepness in a smaller number of attempts are desirable. Furthermore, the *final learning precision* is also an important criterion.

C. Comparison Methods

The first method utilized for comparison is a incremental naive Bayesian classifier [58]. We use exactly the same parameters as [58] in the experiments. The second categories of methods that are used for comparison are rule extraction and decision-tree based methods. The *PART* algorithm is based on the *C4.5* decision tree classification method [59]. *PRISM* is an algorithm for inducing modular rules [60], [61]. The *ID3* algorithm constructs an unpruned decision tree for classification [62]. The *J48* algorithm is also based on a pruned or unpruned *C4.5* decision tree [52]. The incremental version of decision tree algorithm is discussed in [63]. We used the standard Weka[1] machine learning software for the implementation of these methods.

The incremental version of the random forest algorithm is called Online Random Forest (*ORF*) [31]. We have used the same parameters in the implementation of the online random forest method. The Multi-Layer Perceptron (*MLP*) neural network is also used for the comparison. An extensive set of experiments has been conducted to find the best number of layers and the best number of nodes at each layer. Notice that a high number of hidden units and nodes leads to over-fitting of the model in the initial iterations and a low number of hidden units leads to under-fitting of the model and low learning capacity of the model in the final iterations. Therefore, we chose four hidden layers with 10 nodes in each layer which had good results in our experiments. The final algorithm for the comparison is Incremental Support Vector Machine (*ISVM*). We tried different non-linear kernel types for the *ISVM* method, namely, the polynomial kernel functions, sigmoid kernel function and the radial basis kernel function. Consequently, we have chosen Radial Basis Function (RBF) for conducting all the experiments.

We have utilized several online contextual bandit approaches for our comparisons including bootstrapped upper confidence bound [56], [64], bootstrapped Thompson sampling [56], [65] and some other methods from [56], including epsilon greedy, adaptive greedy, explore-then-exploit, exploration based on active learning, softmax explorer and exploration based on active learning approaches.

Notice that to fairly compare the ABL approach with all the contextual bandit approaches, we have used the same procedure as ABL for training the contextual bandit models. This means that we have also used the best choice of action at each state to update a contextual bandit model if it fails to predict the correct action at that state.

In addition to the three scenarios introduced in this paper, we have also included the mushroom dataset from the UCI machine learning repository [32] that has been used in contextual bandits research [66], [67], [68]. The mushroom dataset includes descriptions of hypothetical samples corresponding to 23 species of mushrooms divided into two classes (edible and poisonous). For this experiment, the dataset has been randomly shuffled in each iteration and the first 500 instances of the shuffled data have been chosen.

D. Results

As one can see in Fig. 10, Fig. 11 and Fig. 12, the proposed Argumentation-Based Learning (*ABL*) method outperforms all the other methods in both the comparison criteria used for this research, namely, the final learning precision and the learning speed. The steepness of the learning curve shows that the *ABL* learns faster in a lower number of iterations.

For the first test scenario, after observing 30 failure states, *ABL* achieves 74% precision, while the best method among others has 60% precision. The final precision of *ABL* is 95%, while the best final precision among other methods is 90%. For

[1] https://www.cs.waikato.ac.nz/ml/weka/

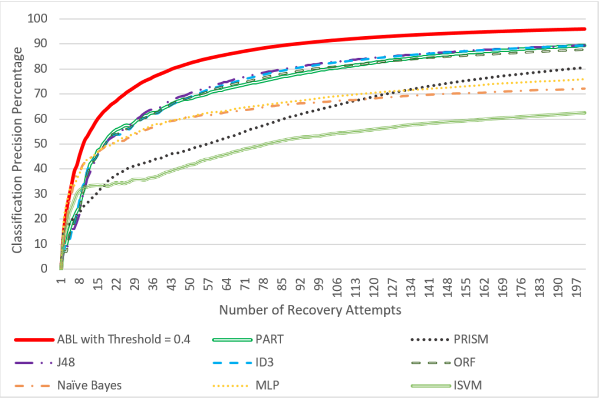

Figure 10: Comparison of Argumentation-Based Learning (*ABL*) with key methods for incremental online learning [27] using the first test scenario.

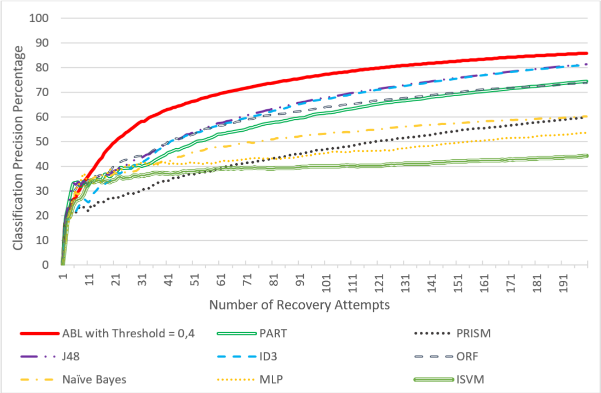

Figure 11: Comparison of Argumentation-Based Learning (*ABL*) with key methods for incremental online learning [27] using the second scenario.

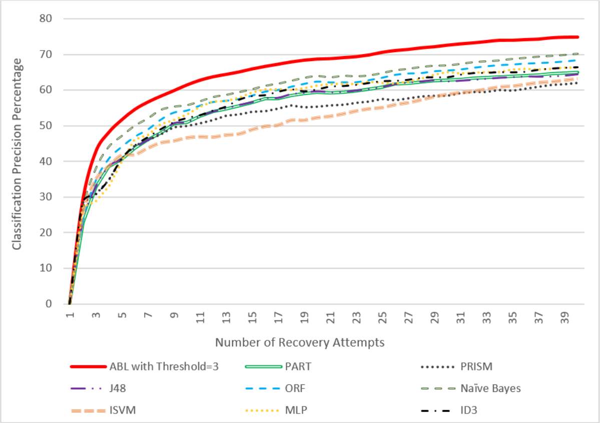

Figure 12: Comparison of Argumentation-Based Learning (*ABL*) with key methods for incremental online learning [27] using the third scenario. This non-robotic scenario emphasizes that the proposed ABL method is generalizable to other online incremental learning scenarios.

the second test scenario, after 30 observations, the *ABL* has 58% precision while J48 as the best performer among all other methods has 42% precision. Moreover, the final precision of *ABL* for the second test scenario is 85% while J48 and *ID3*, the best among all others, achieve almost 80% final precision.

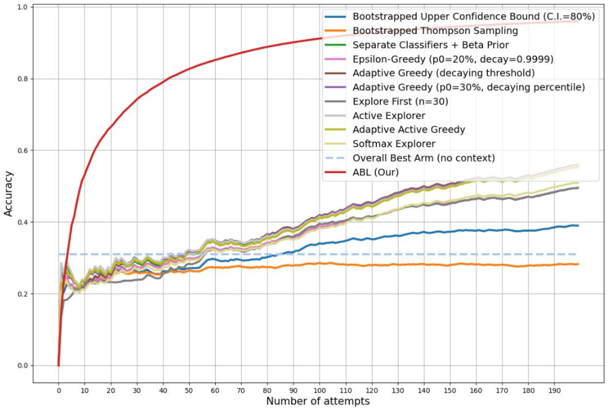

Figure 13: Comparison of Argumentation-Based Learning (*ABL*) and some contextual bandit methods [56] for the first scenario. The red curve shows the accuracy of ABL.

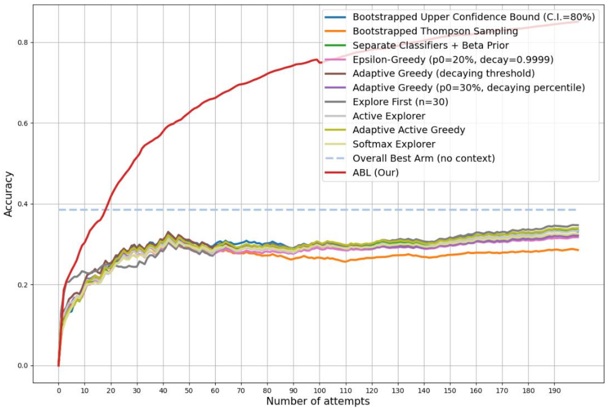

Figure 14: Comparison of Argumentation-Based Learning (*ABL*) and some contextual bandit methods [56] for the second scenario.

In the third scenario, which differs from the two prior scenarios in context, *ABL* repeatedly outperforms all the other methods in both of the comparison criteria. Among other methods, incremental naive Bayes and incremental random forest (*ORF*) have better results. The final learning precision of *ABL* in this scenario is 75% while it is 70% for the incremental naive Bayes method. The slope of the learning curve also shows the faster learning speed of *ABL* with respect to all of the other methods.

Figure 13, 14 and 15 show the comparison of ABL with contextual bandits using the first and second scenario as well as the mushroom dataset. In all experiments, ABL outperforms the other approaches considerably, both in terms of learning speed and in terms of final learning precision. The explorative nature of contextual bandit algorithms may lead to this difference in performance.

VI. Discussion

A key reason that the proposed method works better than Naive Bayes originates from the independence assumption between all features in the Naive Bayesian formulation. In the case of neural networks, considering that there is only a small number of training data instances, a complex neural network tends to over-fit and a small neural network leads to

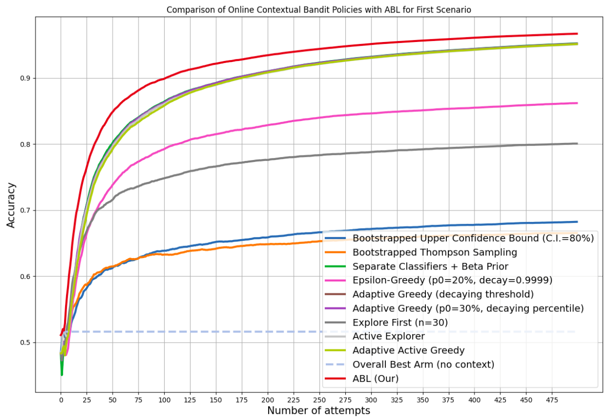

Figure 15: Comparison of Argumentation-Based Learning (*ABL*) and some contextual bandit methods [56] for the mushroom dataset.

under-fitting. Choosing the best neural network architecture dynamically according to the number of visited data is also a challenging task. On the other hand, decision-tree based techniques fail at the initial recovery attempts and then gradually learn the best recovery behavior. This is because of the change in entropy or information gain when new unforeseen data updates the decision tree. This is also the case with the Online Random Forest (*ORF*) method. Furthermore, *ISVM* does not perform well in circumstances where only a few features are associated with predicting the class label. In all the above cases, the suggested *ABL* approach performed better as it considers any possible dependence between features and it can immediately focus on features which are most relevant for the optimal decision.

Moreover, *ABL* leads to an explicit representation of the learning process understandable for humans, as is also the case with decision-tree based techniques. In contrast, neural networks, support vector machines and Bayesian techniques are all black boxes [69] (this means that the trained models are not easily interpretable and explainable) for the humans. This explicit representation of the learning process can be utilized in combination with human-robot interaction. Employing this property, *ABL* can be used in multi-agent scenarios where agents can transfer their knowledge to each other.

The proposed *ABL* method has a limitation. It handles data sets with discrete feature values. This limitation can be addressed in future works.

Consequently, the proposed argumentation-based incremental learning algorithm could learn in fewer attempts with higher precision than other algorithms used for comparison. The results have also shown that ABL outperforms contextual bandit algorithms in terms of learning precision. Moreover, ABL extracts an explicit set of rules that explain the knowledge acquired by the agent over the interaction with the environment. This feature makes the method more explainable and easy to debug by an expert.

Therefore, this method can be a good alternative when the feature values are discrete. Although we have shown that the current ABL approach is working well for the aforementioned scenarios in this paper, these results are limited to datasets with discrete feature values that are not high-dimensional. To make ABL more efficient for higher dimensional problems, we have introduced Accelerated Argumentation-Based Learning (AABL) [70] to improve the space and computational complexity of the method.

VII. Conclusion

General purpose service robots should be able to recover from unexpected failure states caused by environmental changes. In this article, an argumentation-based learning (*ABL*) approach is proposed which is capable of generating relevant hypotheses for online incremental learning scenarios. This set of hypotheses is updated incrementally when unforeseen data enters the model. The conflicts among these hypotheses are modeled by Abstract Argumentation Frameworks.

The performance of *ABL* has been evaluated using both the robotics and the non-robotics incremental learning scenarios. The third scenario, which has a non-robotic context, is a publicly accessible dataset from the UCI machine learning repository. This scenario shows the fact that the proposed *ABL* method can be used in any online incremental learning application with discrete feature values. Moreover, we have also compared the performance of different contextual bandit algorithms with ABL. According to these experiments, the proposed method learns faster and with higher ultimate classification precision than various state-of-the-art online incremental learning methods.

Acknowledgment

This work is conducted at the center of Data Science and Systems Complexity (DSSC) and sponsored by a Marie Skłodowska-Curie COFUND grant, agreement no. 754315.

References

[1] V. N. Lu, J. Wirtz, W. H. Kunz, S. Paluch, T. Gruber, A. Martins, and P. G. Patterson, "Service robots, customers and service employees: what can we learn from the academic literature and where are the gaps?," *Journal of Service Theory and Practice*, 2020.

[2] M. Mende, M. L. Scott, J. van Doorn, D. Grewal, and I. Shanks, "Service robots rising: How humanoid robots influence service experiences and elicit compensatory consumer responses," *Journal of Marketing Research*, vol. 56, no. 4, pp. 535–556, 2019.

[3] S. Schneider, F. Hegger, A. Ahmad, I. Awaad, F. Amigoni, J. Berghofer, R. Bischoff, A. Bonarini, R. Dwiputra, G. Fontana, *et al.*, "The RoCKIn@Home Challenge," in *ISR/Robotik 2014; 41st International Symposium on Robotics*, pp. 1–7, June 2014.

[4] Y. Jiang, N. Walker, J. Hart, and P. Stone, "Open-world reasoning for service robots," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, pp. 725–733, 2019.

[5] A. Kuestenmacher, N. Akhtar, P. G. Plöger, and G. Lakemeyer, "Towards robust task execution for domestic service robots," *Journal of Intelligent & Robotic Systems*, vol. 76, no. 1, pp. 5–33, 2014.

[6] K. Talamadupula, G. Briggs, M. Scheutz, and S. Kambhampti, "Architectural mechanisms for handling human instructions for open-world mixed-initiative team tasks and goals," *Advances in Cognitive Systems*, vol. 5, pp. 37–56, 2017.

[7] P. Schermerhorn and M. Scheutz, "Using logic to handle conflicts between system, component, and infrastructure goals in complex robotic architectures," in *2010 IEEE International Conference on Robotics and Automation*, pp. 392–397, 2010.

[8] F. H. Van Eemeren, B. Garssen, E. C. Krabbe, A. F. S. Henkemans, B. Verheij, and J. H. Wagemans, *Handbook of Argumentation Theory*. Dordrecht: Springer, 2014.

[9] L. Rizzo and L. Longo, "An empirical evaluation of the inferential capacity of defeasible argumentation, non-monotonic fuzzy reasoning and expert systems," *Expert Systems with Applications*, vol. 147, p. 113220, 2020.

[10] A. Vassiliades, N. Bassiliades, and T. Patkos, "Argumentation and explainable artificial intelligence: a survey," *The Knowledge Engineering Review*, vol. 36, 2021.

[11] K. Atkinson, T. Bench-Capon, and D. Bollegala, "Explanation in AI and law: Past, present and future," *Artificial Intelligence*, vol. 289, p. 103387, 2020.

[12] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial Intelligence*, vol. 77, no. 2, pp. 321–357, 1995.

[13] L. Amgoud, C. Cayrol, M.-C. Lagasquie-Schiex, and P. Livet, "On bipolarity in argumentation frameworks," *International Journal of Intelligent Systems*, vol. 23, no. 10, pp. 1062–1093, 2008.

[14] O. Cocarascu and F. Toni, "Argumentation for Machine Learning: A Survey," in *COMMA*, pp. 219–230, 2016.

[15] M. Mozina, J. Zabkar, and I. Bratko, "Argument based machine learning," *Artificial Intelligence*, vol. 171, no. 10-15, pp. 922–937, 2007.

[16] P. Clark and T. Niblett, "The CN2 Induction Algorithm," *Machine Learning*, vol. 3, no. 4, pp. 261–283, 1989.

[17] L. Amgoud and M. Serrurier, "Agents that argue and explain classifications," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 2, pp. 187–209, 2008.

[18] L. Carstens and F. Toni, "Using argumentation to improve classification in natural language problems," *ACM Transactions on Internet Technology (TOIT)*, vol. 17, no. 3, p. 30, 2017.

[19] N. Kotonya and F. Toni, "Gradual argumentation evaluation for stance aggregation in automated fake news detection," in *Proceedings of the 6th Workshop on Argument Mining*, (Florence, Italy), pp. 156–166, Association for Computational Linguistics, Aug. 2019.

[20] M. Moens, "Argumentation mining: How can a machine acquire common sense and world knowledge?," *Argument and Computation*, vol. 9, no. 1, pp. 1–14, 2018.

[21] S. Eger, J. Daxenberger, and I. Gurevych, "Neural end-to-end learning for computational argumentation mining," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Vancouver, Canada), pp. 11–22, Association for Computational Linguistics, July 2017.

[22] M. Bishop, C. Gates, and K. Levitt, "Augmenting machine learning with argumentation," in *Proceedings of the New Security Paradigms Workshop*, NSPW '18, (New York, NY, USA), pp. 1–11, ACM, 2018.

[23] H. Ayoobi, M. Cao, R. Verbrugge, and B. Verheij, "Local-HDP: Interactive Open-ended 3D Object Category Recognition in Real-Time Robotic Scenarios," *Robotics and Autonomous Systems*, 2021.

[24] H. Ayoobi, M. Cao, R. Verbrugge, and B. Verheij, "Handling Unforeseen Failure Conditions using Argumentation-Based Learning," in *International Conference on Automation Science and Engineering*, IEEE, 2019.

[25] A. Tarski, "A lattice-theoretical fixpoint theorem and its applications.," *Pacific Journal of Mathematics*, vol. 5, no. 2, pp. 285–309, 1955.

[26] A. Pazienza, S. Ferilli, and F. Esposito, "On the gradual acceptability of arguments in bipolar weighted argumentation frameworks with degrees of trust," in *International Symposium on Methodologies for Intelligent Systems*, pp. 195–204, Springer, 2017.

[27] V. Losing, B. Hammer, and H. Wersing, "Incremental on-line learning: A review and comparison of state of the art algorithms," *Neurocomputing*, vol. 275, pp. 1261–1274, 2018.

[28] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in *Advances in Neural Information Processing Systems*, pp. 409–415, 2001.

[29] A. Soula, K. Tbarki, R. Ksantini, S. B. Said, and Z. Lachiri, "A novel incremental Kernel Nonparametric SVM model (iKN-SVM) for data classification: An application to face detection," *Engineering Applications of Artificial Intelligence*, vol. 89, p. 103468, 2020.

[30] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *Journal of Machine Learning Research*, vol. 6, no. Sep, pp. 1579–1619, 2005.

[31] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, "On-line random forests," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pp. 1393–1400, IEEE, 2009.

[32] D. Dua and C. Graff, "UCI machine learning repository," 2017.

[33] S. Sen, M. Das, and R. Chatterjee, "Estimation of incomplete data in mixed dataset," in *Progress in Intelligent Computing Techniques: Theory, Practice, and Applications*, pp. 483–492, Springer, 2018.

[34] H. Fujita and Y.-C. Ko, "A priori membership for data representation: Case study of spect heart data set," in *Recent Advances in Intelligent Engineering*, pp. 65–80, Springer, 2020.

[35] T. Do Van, H. Do Duc, and G. C. Nguyen, "Classify high dimensional datasets using discriminant positive negative association rules," in *2018 5th Asian Conference on Defense Technology (ACDT)*, pp. 1–7, IEEE, 2018.

[36] K. Polat, "Similarity-based attribute weighting methods via clustering algorithms in the classification of imbalanced medical datasets," *Neural Computing and Applications*, vol. 30, no. 3, pp. 987–1013, 2018.

[37] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*. Adaptive computation and machine learning series, Cambridge, Massachusetts: The MIT Press, second edition ed., 2018.

[38] R. Bellman, "A Markovian decision process," *Journal of Mathematics and Mechanics*, pp. 679–684, 1957.

[39] S. Dong, B. V. Roy, and Z. Zhou, "Provably efficient reinforcement learning with aggregated states," *arXiv preprint arXiv:1912.06366*, 2020.

[40] Z. Zhou, M. Bloem, and N. Bambos, "Infinite time horizon maximum causal entropy inverse reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 2787–2802, 2018.

[41] M. Dimakopoulou, Z. Zhou, S. Athey, and G. Imbens, "Balanced linear contextual bandits," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3445–3453, 2019.

[42] Z. Zhou, R. Xu, and J. Blanchet, "Learning in generalized linear contextual bandits with stochastic delays," in *Advances in Neural Information Processing Systems*, pp. 5197–5208, 2019.

[43] Y. Han, Z. Zhou, Z. Zhou, J. Blanchet, P. W. Glynn, and Y. Ye, "Sequential batch learning in finite-action linear contextual bandits," *arXiv preprint arXiv:2004.06321*, 2020.

[44] K.-S. Jun, A. Bhargava, R. Nowak, and R. Willett, "Scalable generalized linear bandits: Online computation and hashing," in *Advances in Neural Information Processing Systems*, pp. 99–109, 2017.

[45] T. Lattimore and C. Szepesvári, *Bandit algorithms*. Cambridge University Press, 2020.

[46] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *International Conference on Machine Learning*, pp. 127–135, 2013.

[47] M. Abeille, A. Lazaric, *et al.*, "Linear Thompson sampling revisited," *Electronic Journal of Statistics*, vol. 11, no. 2, pp. 5165–5197, 2017.

[48] Z. Zhou, S. Athey, and S. Wager, "Offline multi-action policy learning: Generalization and optimization," *arXiv preprint arXiv:1810.04778*, 2018.

[49] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[50] B. Gu, V. S. Sheng, K. Y. Tay, W. Romano, and S. Li, "Incremental support vector learning for ordinal regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 7, no. 26, pp. 1403–1416, 2015.

[51] B. Gu, V. S. Sheng, Z. Wang, D. Ho, S. Osman, and S. Li, "Incremental learning for ν-support vector regression," *Neural Networks*, vol. 67, pp. 140–150, 2015.

[52] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[53] R. Agrawal and R. Bala, "Incremental Bayesian classification for multivariate normal distribution data," *Pattern Recognition Letters*, vol. 29, no. 13, pp. 1873 – 1876, 2008.

[54] P. Reiner and B. M. Wilamowski, "Efficient incremental construction of RBF networks using quasi-gradient method," *Neurocomputing*, vol. 150, pp. 349–356, 2015.

[55] J. Lu, F. Shen, and J. Zhao, "Using Self-Organizing Incremental Neural Network (SOINN) for radial basis function networks," in *2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 2142–2148, IEEE, 2014.

[56] D. Cortes, "Adapting multi-armed bandits policies to contextual bandits scenarios," *arXiv preprint arXiv:1811.04383*, 2019.

[57] H. Ayoobi and M. Rezaeian, "Swift distance transformed belief propagation using a novel dynamic label pruning method," *IET Image Processing*, vol. 14, no. 9, pp. 1822–1831, 2020.

[58] S. Ren, Y. Lian, and X. Zou, "Incremental naïve Bayesian learning algorithm based on classification contribution degree.," *JCP*, vol. 9, no. 8, pp. 1967–1974, 2014.

[59] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, "Chapter 6 - trees and rules," in *Data Mining (Fourth Edition)* (I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, eds.), pp. 209–242, Morgan Kaufmann, fourth edition ed., 2017.

[60] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2019.

[61] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, "Chapter 4 - algorithms: The basic methods," in *Data Mining (Fourth Edition)* (I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, eds.), pp. 91–160, Morgan Kaufmann, fourth edition ed., 2017.

[62] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81–106, Mar 1986.

[63] M. Wozniak, "A hybrid decision tree training method using data streams," *Knowledge and Information Systems*, vol. 29, no. 2, pp. 335–347, 2011.

[64] B. Hao, Y. Abbasi Yadkori, Z. Wen, and G. Cheng, "Bootstrapping upper confidence bound," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, pp. 12123–12133, Curran Associates, Inc., 2019.

[65] D. Eckles and M. Kaptein, "Bootstrap Thompson sampling and sequential decision problems in the behavioral sciences," *SAGE Open*, vol. 9, no. 2, p. 2158244019851675, 2019.

[66] Z. Wang and M. Zhou, "Thompson sampling via local uncertainty," in *International Conference on Machine Learning*, pp. 10115–10125, PMLR, 2020.

[67] C. Riquelme, G. Tucker, and J. Snoek, "Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling," in *International Conference on Learning Representations*, 2018.

[68] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *International Conference on Machine Learning*, pp. 1613–1622, PMLR, 2015.

[69] B. Zhou, D. Bau, A. Oliva, and A. Torralba, "Interpreting deep visual representations via network dissection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, pp. 2131–2145, Sep. 2019.

[70] H. Ayoobi, M. Cao, R. Verbrugge, and B. Verheij, "Argue to learn: Accelerated argumentation-based learning," in *20th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

Hamed Ayoobi is a Ph.D. researcher in the artificial intelligence department of the University of Groningen, Netherlands. He investigates topics in Argumentation Theory, Machine Learning, Explainable Artificial Intelligence, Computer Vision and Robotics. He received the MSc degree in Artificial Intelligence and Robotics in 2018 from Yazd University, Iran. He graduated as a top student in the master program. He was a research intern at Linnaeus University, Sweden working on verification of concurrent machine code using formal methods.

Ming Cao is a professor of networks and robotics with the Engineering and Technology Institute at the University of Groningen, the Netherlands. He received the Bachelor degree in 1999 and the Master degree in 2002 from Tsinghua University, China, and the Ph.D. degree in 2007 from Yale University, USA, all in Electrical Engineering. He was a research associate in 2008 at Princeton University, USA and a research intern in 2006 at the IBM T. J. Watson Research Center, USA. He is the 2017 recipient of the Manfred Thoma medal from the International Federation of Automatic Control (IFAC) and the 2016 recipient of the European Control Award sponsored by the European Control Association (EUCA). He is a Senior Editor for Systems and Control Letters, and an Associate Editor for IEEE Transactions on Automatic Control. His research interests include autonomous agents and multi-agent systems, complex networks and decision-making processes.

Rineke Verbrugge is professor of Logic and Cognition at the University of Groningens Bernoulli Institute of Mathematics, Computer Science and Artificial Intelligence. She is the leader of the Multi-Agent Systems group. Verbrugge has led the NWO Vici-project Cognitive systems in interaction: Logical and computational models of higher-order social cognition and is currently co-leader of the national Gravitation project Hybrid Intelligence. Verbrugge is associate editor of the Journal of Logic, Language and Information and is an elected member of the Royal Holland Society of Sciences and Humanities. Her research interests include logics for multi-agent systems and computational cognitive models of intelligent interaction.

Bart Verheij investigates the theoretical, computational and empirical connections between knowledge, data and reasoning, as a contribution to explainable, responsible and social artificial intelligence, aligned with human values. His research focuses on computational argumentation and AI & Law. He is head of the Artificial Intelligence department at the University of Groningen (Bernoulli Institute of Mathematics, Computer Science and Artificial Intelligence), holding the chair of artificial intelligence and argumentation as associate professor.