Chapter 18
Shifting Paradigms in Verification and Validation of AI-Enabled Systems: A Systems-Theoretic Perspective

Niloofar Shadab, Aditya U. Kulkarni, and Alejandro Salado

Abstract There is a fundamental misalignment between current approaches to designing and executing verification and validation (V&V) strategies and the nature of AI-enabled systems. Current V&V approaches rely on the assumption that system behavior is preserved during a system's lifetime. However, AI-enabled systems are developed so that they evolve their own behavior during their lifetime; this is the consequence of learning by the AI-enabled system. This misalignment makes existing approaches to designing and executing V&V strategies ineffective. In this chapter, we will provide a systems-theoretic explanation for (1) why learning capabilities originate a unique and unprecedented family of systems, and (2) why current V&V methods and processes are not fit for purpose. AI-enabled systems necessitate a paradigm shift in V&V activities. To enable this shift, we will delineate a set of theoretical advances and process transformations that could support such shift.

Keywords Verification and validation · Systems theory · Cyber-physical systems · AI-enabled systems · Systems engineering

N. Shadab · A. U. Kulkarni
Grado Department of Industrial And Systems Engineering, Virginia Tech., Blacksburg, VA 24061, USA
e-mail: nshadab@vt.edu
URL: https://www.vt.edu

A. U. Kulkarni
e-mail: aditya88@vt.edu

A. Salado (✉)
Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ 85721, USA
e-mail: alejandrosalado@arizona.edu

W. F. Lawless et al. (eds.), *Systems Engineering and Artificial Intelligence*,
https://doi.org/10.1007/978-3-030-77283-3_18

18.1 Introduction

Systems Engineering experts have developed methodologies and processes to successfully verify and validate complex systems. V&V activities play a crucial role to form experts' beliefs about system performance, functions, and structure (Engel 2010; Hoppe et al. 2007; Salado and Kannan 2019). These V&V methodologies and processes were originally designed to support the development of traditional systems, which we describe as behavior-preserving, such that, when subjected to the same inputs, the system is expected to produce the same outputs throughout the system's operational life (Salado and Kannan 2019). Furthermore, traditional systems are deployed in environments where there is little or no learning to be performed by the system itself; instead, learning is an attribute of the human operators that are part of the system's operational environment. Thus, in the design of behavior-preserving systems, V&V activities are employed to predict, confirm, or gain confidence about the future behavior of a system in its operational environment.

In contrast to traditional systems is Artificial Intelligence-enabled systems (AI-enabled systems), which we define as cyber-physical systems that exhibit artificial intelligence (AI) capabilities. The AI capability can utilize *history* to alter the operational parameters of the system. That is, in general, AI-enabled systems are *not* behavior-preserving systems. The possibility of AI-enabled systems to dynamically adjust operational parameters in the field gives rise to an unprecedented challenge in systems engineering: how can we verify and validate AI-enabled systems whose behavior can dynamically change when deployed? (Felder 2018)

In this chapter, we will discuss the challenges that the systems engineering community may face in designing V&V strategies for AI-enabled systems, and how these challenges might be potentially overcome. Overall, we suggest that a paradigm shift is necessary, even though obstacles will be faced, and we focus on the following aspects. First, we elaborate on how performing V&V in a test environment might not be suitable for predicting the system's behavior in its operating environment. Second, we examine the endogenous evolution of intelligent systems that sheds light on the misalignment of using the homomorphism concept in V&V for such systems. Third, we discuss the V&V challenges of the meta-capabilities of intelligent systems. Fourth, we address the scalability of intelligent systems and the implications to V&V at different scales. Fifth, we expand on the nature of dynamic changes of the set of state descriptions of an intelligent system and the related misalignment with current V&V methodologies and processes.

We address these challenges around a systems theoretic definition of intelligence, which we provide later in the chapter.

18.2 A Need for a Paradigm Shift in V&V

There is an increasing number of currently operating systems that are being outfitted with advanced intelligent abilities such as learning, knowledge representation, and perception. Therefore, it is evident that the existing V&V processes need to be realigned due to this transition. For instance, autonomous vehicles are now being seriously considered as viable alternatives to traditional modes of transport. Proponents of the technology argue that it is only a matter of time before civilian laws are suitably altered and there is widespread usage of autonomous vehicles on public roads. In this regard, a key challenge is characterizing the capabilities of autonomous vehicles so that suitable laws can be passed to govern the use of autonomous vehicles.

Characterizing the capabilities of autonomous vehicles will require a new paradigm for V&V activities during the development process. For example, one key technology that is currently under development in autonomous vehicles is the onboard AI that learns the driving style of the car's owner (Kuderer et al. 2015). By statistically analyzing the owner's driving style, which is defined as a multi-attribute variable that includes acceleration, deceleration, and route preferences, the onboard AI is able to learn and replicate the driving style of the user. Similarly, another key technology in autonomous vehicles is intelligent navigation (Isele et al. 2018), where onboard AI continuously improves its navigation capabilities during the car's operation.

Existing approaches to designing and executing V&V strategies may not be fit-for-purpose in the design and development of AI-enabled systems. For the technologies presented above, for example, the vehicle's behavior is subject to change based on the data received by a vehicle during its operation (Felder 2018). Conducting V&V in the lab will most likely be of little relevance since it will be cost-prohibitive to create an input dataset that can capture all, or almost all, of the possible scenarios in which the user will operate the autonomous vehicle. Furthermore, even if V&V is conducted in the lab, there is no guarantee that the behavior of the autonomous vehicle predicted by the V&V activities will be realized during the vehicle's operational life. Indeed, the behavior observed in the lab may be entirely alien to that observed in the field.

The challenge here is to change how we think about V&V activities. In AI-enabled systems, the consequences of learning by the AI algorithms manifest as a system-level behavior. However, the actual operational parameters chosen by the AI algorithms may not be observed until the system is deployed in the field. Thus, traditional approaches to V&V may end up giving false confidence to both designers and stakeholders in the possible behavior of the AI-enabled system.

The two technologies mentioned above for autonomous vehicles are not isolated achievements of the engineering community. Indeed, an increasing number of complex systems currently operating are being outfitted with advanced intelligent abilities such as learning, knowledge representation, and perception. For example, in continued deployment approaches, such as DevOps (a collaborative merger of development and operations), V&V activities are heavily reused as new systems are

deployed within the operational infrastructure. With fixed V&V approaches, the risk of learning how to pass the test increases with each new deployment.

The risk of "learning how to pass the test" is also a cybersecurity-related problem for AI-enabled systems. Persevering with traditional, or known, V&V methods to verify AI-enabled systems enables malicious attackers to compromise the process. This, in turn, gives the designers and users false confidence on the performance of the AI-enabled system. A prime example of this scenario is adversarial attacks on deep neural networks (DNNs) deployed to classify images (Sengupta et al. 2019). DNNs are often used to classify images for various purposes. These include identifying cancer cells for medical treatment, identifying obstacles for navigation of autonomous vehicles, and identifying individuals for public security. Often, DNNs are first trained on known datasets, which are often publicly available. After training a DNN, its performance in classifying images correctly is verified and validated. The DNN is deployed only if there is sufficient confidence in the accuracy of its performance.

Since DNNs are trained on known, or publicly available, datasets, to compromise the development process of a DNN, one only needs to corrupt the dataset. Indeed, adversarial attacks on DNN is an active research area (Wang et al. 2019). Here, attackers introduce small perturbations in certain samples of the dataset. By doing so, the attackers ensure that when the DNN is trained, the perturbations in chosen samples ensure the DNN learns to misclassify these samples. Since DNNs are expected to classify several objects, attackers can ensure the DNNs learn to misclassify certain key objects that are of high value to the attackers. In this way, the DNN gives its designers, and users, false confidence about the accuracy of this performance.

Moreover, the nature of V&V for learning capabilities itself is challenging in system-level verification of AI-enabled systems (Xiang et al. 2018). As AI-enabled systems might have learning capabilities at different scales, their solution space and design space may not be static; in fact, they can keep changing over the system's lifetime. As a result, it makes it difficult for V&V strategies to predict the possible future behaviors of the system over its lifetime.

The examples discussed above show that we cannot utilize traditional V&V strategies to verify and validate AI-enabled systems. As the examples discussed above show, in addition to the inability of accurately predicting all possible future inputs to the AI-enabled system, securing the design process should also be a concern of V&V strategies. Thus, there is a need for a new paradigm in V&V for cyber-physical systems.

18.3 A Systems-Theoretic Interpretation of Intelligence

We first characterize intelligence, with respect to AI-enabled systems, to have a meaningful conversation about V&V in AI-enabled systems. There are multiple formal definitions of intelligence in the literature (Chollet 2019; Legg et al. 2007). In this section, we re-scope previous definitions of intelligence using elements of

systems theory. By doing so, we lay a foundation for the discussion on adapting V&V techniques to AI-enabled systems.

We adopt von Bertalanffy's definition of a system and declare that a system is a set of inter-related elements (Bertalanffy 1969), where the type of relation is unrestricted. Since the null relation is a type of relationship between elements (Wymore 2018), it suffices to define a boundary around a set of elements to call such a set a system. We distinguish then between open and closed systems, where open systems are those that transfer information, energy, or matter through their boundaries (in and out) and closed systems are those that do not transfer any information, energy, or matter through their boundaries. Furthermore, we restrict our attention to engineered systems, which we define as those made by humans (or machines) using engineering. Within a systems engineering framework, we distinguish between the Intervention System and Context System; the first is the system of interest that is realized to satisfy a need or pursue an opportunity and the second is the system formed by the Intervention System and all systems that directly interact with it. In the latter system, the satisfaction of the need or realization of the opportunity takes place (Salado 2021). We define an AI-enabled system and call it an intelligent system instinctively in this chapter; it is an engineered system that exhibits one or more of the following capabilities:

1. Learning to perform a function better, which can include more efficiently or more effectively;
2. Learning to handle a larger set of inputs for an existing function;
3. Learning a new function;
4. Learning to achieve an existing outcome in a new context;
5. Learning to achieve an outcome better, which it can perform more efficiently or more effectively; and,
6. Learning to achieve a new outcome
7. Deciding to pursue a different outcome.

Of the seven capabilities of an intelligent system, the first three capabilities are related to a system's ability to effect a change in its behavior (that is, functions the system executes). Whereas the latter four capabilities are related to the "curiosity" exhibited by an intelligent system in seeking new ways to achieve its purpose, a system's purpose is defined by one or more long-term objectives. In this regard, we say a system's long-term objective is an outcome, and distinguish it from a goal, which we define as any task and/or challenge that needs to be achieved to fulfill the desired outcome of the system; for example, the time required for an autonomous car to detect a moving object.

We can broadly define the behavior of systems using functions, which we refer to as behavior functions. The domain of a system's behavior function can be any subset of the space of all possible inputs to the system. Similarly, the range of a system's behavior function is a subset of the space of all possible outputs. Traditional systems are behavior-preserving, and hence their behavior functions are set for their operational life. That is, for behavior-preserving systems, the mapping between the

domain and range of the system's behavior functions does not change throughout the system's operational life. In contrast, AI-enabled systems can determine if their behavior functions are suboptimal in operation. Furthermore, AI-enabled systems can change the set of outputs for a given set of inputs so that the system's behavior is optimally aligned regardless of changes in the space of inputs and the desired outputs for those inputs (by, for example, changing requirements).

By changing the range of a behavior function, AI-enabled systems, in effect, exhibit the *first capability* of an intelligent system: improving the execution of one of its functions. For example, an intelligent detection system can improve its observation accuracy as it learns from past observations.

The *second capability* is its ability to operate with a different domain of its behavior functions. In this regard, AI-enabled systems can potentially accept (and use) an increased, or varied, set of inputs to a behavior function. For example, a detection system that was trained to observe targets with certain signature profiles may learn to perform the same observation function for other signature profiles.

With the *third intelligence capability*, a system can generate a new mapping of its domain to its range, effectively learning a new function. For example, an intelligent detection system that is trained to observe a particular type of target learns to also classify the target according to certain characteristics.

Since open systems execute functions, and outcomes are exhibited in closed systems (Salado 2021) by the actions of open systems that form it, an intelligent system can leverage its own behavior to yield desired outcome-related learning in the closed systems they belong to. In this sense, the *fourth capability* describes the adaptability of an intelligent system to changing contexts. For example, a detection system originally trained to detect security threats inside of buildings is trained to detect threats in open areas.

With the *fifth* capability, the system can also learn from its experience in detecting threats to become better at it, either because it can do it more efficiently (e.g., faster) or more effectively (e.g., reduction of false positives). The key difference between this capability and the first capability, is that the former improves the outcome achieved by the system, and the latter the system's function. This difference is akin to a human improving its kicking ability (function; precision of joint movement, muscle strength, etc.) and its goal scoring ability (outcome; use of functions in the context of a ball, a goal, and a goalkeeper trying to stop the goal) through repeated practice.

Using the *sixth capability*, an intelligent system can be trained to incorporate new long-term objectives that maximize the net utility of the system's operation over its lifetime. For example, a system that is originally trained to detect threats is trained at a later stage (without any other functional or form changes) to mitigate the threat.

Finally, the *seventh capability* alludes to the possibility of an intelligent system possessing some semblance of choice. Since an intelligent system is designed, the initial set of outcomes the system is meant to achieve could prove to be short sighted in the long run. In this regard, it is necessary for the intelligent system to adapt it and decide on a new outcome. For example, continuing with the detection system, instead of being trained to mitigate the threat, with this capability the intelligent

system identifies that mitigating the threat is a better outcome than simply detecting it and, as a result, it decides to learn, and implement it.

If not otherwise specified in the rest of the chapter, our discussion addresses an intelligent system that may implement one or more of these learning capabilities.

18.4 Challenges to the V&V of AI-Enabled Systems

18.4.1 Differential Learning in V&V Versus Operational Environment

How it is done today. Consider a formal definition of a system as a transformation P of an input vector $\bar{\boldsymbol{I}}$ into an output vector $\bar{\boldsymbol{O}}$ (ref. Fig. 18.1a). A verification activity consists of injecting a V&V input vector $\bar{\boldsymbol{I}}_{\boldsymbol{T}}$, which the engineer considers sufficiently representative of the actual input vector that the system will receive in operation, that is, $\bar{\boldsymbol{I}}_{\boldsymbol{T}} \approx \bar{\boldsymbol{I}}$, and observing a V&V output vector $\bar{\boldsymbol{O}}_{\boldsymbol{T}}$, which the engineer considers sufficiently representative of the desired output vector the system will provide during operation, that is, $\bar{\boldsymbol{O}}_{\boldsymbol{T}} \approx \bar{\boldsymbol{O}}$. If transformation P is demonstrated for the V&V vectors $\bar{\boldsymbol{I}}_{\boldsymbol{T}}$ and $\bar{\boldsymbol{O}}_{\boldsymbol{T}}$, then it is inferred that the system will also execute transformation P when seeing the actual input vector $\bar{\boldsymbol{I}}$. And, hence, the system would be considered properly verified.

This approach to verification is sound for non-learning systems that preserve their behavior. In such systems, since the transformation the system executes is invariant to its inputs, the results of the V&V activity can be a good predictor of the behavior of the system in its operational environment. This transformation can be modeled as a Bayesian network (Salado and Kannan 2019), as shown in Fig. 18.1b, where θ denotes the actual performance of the system and V denotes the results of the verification activity employed to predict it.

Limits of the current approach. Recent works demonstrated that intelligent systems can behave differently to synthetically generated inputs that are perceptually indistinguishable from data in their natural form (Nguyen et al. 2015; Szegedy et al. 2013). Hence, we suggest that AI-enabled systems may be able to discern the V&V input vector $\bar{\boldsymbol{I}}_{\boldsymbol{T}}$ from the actual input vector to be received during operation $\bar{\boldsymbol{I}}$, and evolve as a result different behaviors for each type of input vector. In this way, as shown in Fig. 18.2a, the AI-enabled system may create a specific transformation P_T

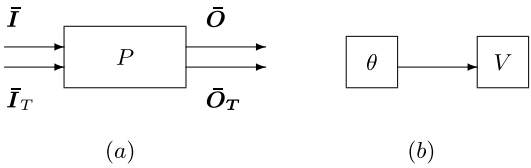

Fig. 18.1 Current approach to V&V design

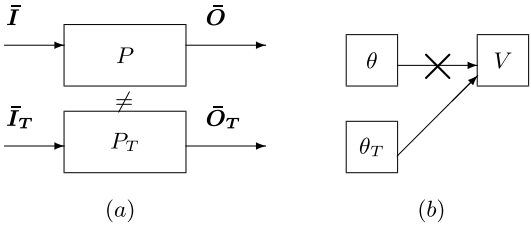

Fig. 18.2 Limits of current V&V design for AI-enabled systems

to construct expected V&V outputs $\bar{O_T}$ for given V&V inputs $\bar{I_T}$, without providing any information about the transformation P it will execute when the operational input vector $\bar{I}$ is inputted. In terms of V&V, the system has constructed a specific type of performance, which we call V&V performance, denoted by θ_T, that disconnects the V&V activity from the original performance θ that it was trying to infer (ref. Fig. 18.2b).

This idea is inspired by critical issues in the field of education, where accurately assessing student learning is difficult. In a formal learning setting, a student learns by preparing for an upcoming exam and, in doing so, masters the exam. However, research shows that mastery of an exam is not necessarily correlated with mastery of the material (Suto 2012). Thus, exams may be poor predictors of student learning. This analogy can be used for AI-enabled systems. To better understand it, consider for example, continued deployment approaches, such as DevOps, where V&V activities are heavily reused as new systems are deployed within the operational infrastructure. With fixed V&V approaches, the risk of "learning how to pass the test" increases with each new deployment.

A similar situation exists with systems that are maintained frequently in the field. Furthermore, there are security risks in which a system may be hacked so that it can actively detect V&V vectors and learn how to deceive them, leaving system owners ignorant and naíve about the behavior the system will exhibit in operation. Current approaches to designing V&V strategies are unable to detect such a vulnerability.

These potential issues justify a transformation in how we approach the V&V of cognitive agents.

18.4.2 Endogenous Evolution of Systems

Behavior-preserving systems evolve due to exogenous factors, both during development and operation. Examples of exogenous changes include active design changes exercised by engineers, configuration changes that are externally activated or programmed, technology refresh programs activated by operators, or external maintenance. Absent these factors, traditional systems remain unchanged and, their behavior is not expected to evolve with time, with the exception of degradation due to wear. As a result, V&V strategies rely on V&V models that represent aspects of the system

(i.e., homomorphisms of the system). Mathematically, the set of potential verification strategies for a system can be described as (Salado and Kannan 2018):

$$\gamma(Z_0, R) = \bigcup_{i=0}^{n} \left(F_{(Z_i)} \bigcup_{j=0}^{H_i} F_{(Z_{ij})} \right) \times R \tag{18.1}$$

where:

- Z_0 is the system of interest and $Z_1, \ldots, Z_n$ are the systems that decompose Z_0 in all of its constituent elements on which formal verification occurs. They are traditionally referred to as subsystems, components, or parts among others.
- $H_i = \{Z_i, Z_i 1, Z_i 2, , , , \ldots, Z_i m\}$ is the set of systems that are homomorphic images of system Z_i . This set represents all models of system Z_i that are used for verification. In practical terms, they can take the form of a mathematical model, a prototype, or the final product, for example.
- $F(Z) = \{P_1, P_2, \ldots, P_k\}$ is a parameterization of system Z. This parameterization is finite and represents the set of parameters of system Z that need to be formally verified.
- A verification activity V is a tuple (p, r), where r denotes a verification procedure. A verification activity is understood as the application of a verification procedure r to the discovery of knowledge about a system parameter p.
- $R = \{r_1, r_2, \ldots, r_i\}$ is the set of verification procedures that could be executed by a given organization.

Two aspects are central to this model: homomorphisms and parameterizations (Salado and Kannan 2018). First, the model that is used in a verification activity influences the confidence gained through such an activity. A verification activity must always refer to (or be characterized by) the model (homomorphism) in which it is executed. Second, the confidence on the system of interest exhibiting certain behavior or characteristic may not be obtained by measuring or observing a characteristic directly for the system of interest. Instead, it may be inferred from measuring or observing an equivalent or indirect characteristic of one of its homomorphic images, other than the system itself. Therefore, a verification activity must always refer to (or be characterized by) the parameter that it verifies.

In behavior-preserving systems, because system evolution is always initiated exogenously, verification models (that is, homomorphisms and parameterizations) remain relevant during the system development and can be adapted anticipatorily to those system's changes. However, AI-enabled systems can initiate internal change endogenously. As previously indicated, such is the purpose of learning: AI-enabled systems will be able to exhibit new behaviors by learning from their interaction with the environment without any specific external action. In other words, the behavior of the system is not necessarily preserved, may not be able to be anticipated, and can occur at the discretion of the system of interest itself. In turn, this implies that at least the space of homomorphisms for an AI-enabled system is *discovered* dynamically during the operational life of the system.

The lack of knowledge of the complete space of system homomorphisms for AI-enabled systems poses a significant challenge in V&V for AI-enabled systems. Once an AI-enabled system changes its behavior, V&V models that were previously homomorphic images of the original system may no longer fulfill homomorphic conditions with respect to the evolved system. This implies that the confidence gained through V&V activities, on models of the system that were previously homomorphic to the system itself, is now nullified. Furthermore, the dynamic discovery of homomorphisms of the system during the system's operation also implies that V&V models are not guaranteed to produce relevant evidence about system's behavior. Hence, V&V models employed in traditional V&V are likely to become obsolete (potentially not observable) during the development and operation of AI-enabled systems.

One could argue that the obsolescence of one or more homomorphisms of an AI-enabled system, due to an endogenously generated change in the AI-enabled system's behavior during operation, can be overcome by equipping known homomorphisms of the AI-enabled systems with the capability to approximately represent the AI-enabled system's behavior. However, even if we ensure that the homomorphic images of an AI-enabled system can be equipped with capabilities to adapt and dynamically change according to an AI-enabled system's evolution, verification of the dynamic structure and behavior of these homomorphic images will also be an important part of the verification process for the system of interest itself. This process could become recursive, possibly leading to major challenges. Indeed, the computational complexity required to aggregate all verification ramifications to deduce a meaningful inference makes finding a solution nearly infeasible with current technological capabilities and the required timeframes to complete V&V beneficially. Furthermore, developing a multi-agent verification platform in a multidisciplinary environment could result in out-of-order events and increased computational complexity of the verifications executed, which might introduce unintended behaviors that contribute to inaccurate emergent behaviors in the verification platform (Yilmaz 2006, 2015; Arifin and Madey 2015). The learning and evolutionary nature of AI-enabled systems demand a different response from engineers to ensure V&V models and activities remain effective.

Since AI-enabled systems change their behavior dynamically, rendering one or more homomorphisms obsolete, we advocate for the use of belief distributions to dynamically update the beliefs of designers and stakeholders when the AI-enabled system changes. In this regard, we now sketch the outline for a potential belief model for V&V activities in AI-enabled systems.

We assume that there are M homomorphic images of the system $\mathbb{Z}$, and let $m \in 1, \ldots, M$. We define a conditional probability for each verification activity given the probability distribution of the previous verification level. Let us start with the joint belief distribution on the system performance, $b(\mathbb{Z})$, after applying all verification activities at system level

$$b(\mathbb{Z}) = f(V_{Z_1}) \times \cdots \times f(V_{Z_M}) \tag{18.2}$$

In this equation, $b(\mathbb{Z})$ is the belief distribution for the performance of $\mathbb{Z}$ after conducting a set of verification activities over homomorphic images of the system $\mathbb{Z}$. Each V_{Z_m} is the verification process performed on the mth homomorphic image of the system $\mathbb{Z}$. The belief derived from the verification of a homomorphic image is denoted by $f(V_{Z_m})$. Note that if the homomorphic image Z_m is not utilized for verification and validation, then we assume $f(V_{Z_m}) = 1$, else $f(V_{Z_m}) \in [0, 1]$.

Given that the homomorphic images of a learning system might also need to exhibit learning capabilities, the accuracy of assumptions of homomorphism for these homomorphic images require to be verified as well. If we assume that the homomorphism property for each of these homomorphic models should be verified by just another verification process, it will be evident that the possibility of errors in verification models will be multiplied just by adding this one more layer of uncertainty to the verification process of the learning system. In this regard, if we simplify the second layer of uncertainty (the verification process for homomorphic images) as the distribution of the standard deviations (σ) over each $f(V_{Z_m})$, we can see that the new $f(V_{Z_m})$ will be the multiplication of the errors (variations) of standard distribution over homomorphic model Z_m of the system $\mathbb{Z}$ after conducting the second layer of verification activities and the observed verification belief distribution.

Variation over the standard deviation here can be a representative of the uncertainty over the confidence of the belief formation after the verification activities are done. As mentioned earlier, just adding one more layer of the uncertainty can result in the possibility of more significant errors in verification activities. Consequently, even if we consider the best case scenario where we assume that all the verification activities are independent of each other (which is not the case most of the time as each verification activity can impact on the result of one or more verification activities), the final equation of the belief deviation with one added layer of uncertainty over the system of interest can be the result of the multiplication of the first-order error of the standard deviation of each homomorphic model of the system.

It is evident that even if we consider this variation as some small constant that can be added to the σ of $f(V_{Z_m})$, the uncertainty over the joint distribution of the homomorphic models might become significant, especially if the size and complexity of the system increases (i.e., larger set of homomorphic models). We can see that by simply creating agent-based homomorphic models that can capture dynamic and emergent behavior of an AI-enabled system, we might not be able to successfully verify the system with high confidence.

18.4.3 Verification of Learning to Learn

Current AI-enabled systems learn a complex skill by investing a large amount of time in trial-and-error experiences or by acquiring enough data to accumulate skills over time (Finn et al. 2017). If rapid learning in different contexts is needed, it is not possible though to invest such an amount of time or even generate such a number of experiences (Finn et al. 2017). To overcome this challenge, meta-learning and

abstract-concept learning may be ways to improve the learning process of intelligent systems. Here, meta-learning refers to the ability of the intelligent system to continuously adapt learning strategies in the presence of new tasks (Thrun and Pratt 1998; Rendell et al. 1987). By abstract-concept learning, we refer to the ability of the learning system to adapt not just to an example for a level of experience, but to a different level of abstraction of that example (Vilalta and Drissi 2002). For example, a robot that can jump from a bar learns to jump from any type of obstacle. This type of inductive learning allows an intelligent system to update its meta-learning capability as it accumulates more meta-knowledge from its experiences on various complex tasks. The challenge to achieve a life-long meta-learning capability for AI-enabled systems signifies the fact that training a system, or an algorithm to have educated guesses alone, might not be sufficient for future systems (Hunt 1962).

It is common practice in systems engineering to develop and build a system based on a set of concrete requirements (e.g., functional, performance, resource, and environment requirements (Salado and Nilchiani 2014), which serve as the bases to verify the correct operation of the system (INCOSE 2015). Since fulfilling these requirements is bounded to a structured space of capability, $S_{structured}$, we could argue that it is possible to exert control on the portion of the entire space in $S_{structured}$ that will be covered by verification scenarios. AI-enabled systems that can be trained to learn a general concept instead of being induced to learn a set of predefined capabilities have the ability to adapt their learning strategy to a new problem space S_{new} by creating meta-knowledge (a hypothesis) from its previous experiences. The traditional approach to develop a set of concrete requirements and achieve a plausible design space for the AI-enabled system may therefore be ineffective for meta-learning capabilities, since the intelligent system can potentially access the many solution spaces during its lifetime which cannot be controlled during the design. Due to the strong interrelation between requirements and verification activities, this implies that using a set of concrete verification scenarios that cover part of the solution space will likely be ineffective to assess the meta-learning capabilities of a system.

We seek inspiration to overcome this challenge in how human meta-learning occurs and is assessed. Some theorists have argued that the mental representation of a concrete concept is necessary to fully induce the corresponding abstract concept (Lakoff and Johnson 2008; Johnson 2013; Murphy 1996). It is also believed by mental modelers that mental meta-representations are enabled by reasoning capabilities in human cognition that are triggered by a reasoning module (Mercier and Sperber 2017). Using this analogy, we could argue that the meta-learning capability of an intelligent system (specifically its ability to *understand* an abstract concept) could be verified by either testing several concrete concept learning processes, or by testing the accuracy of its reasoning functionality.

From a system's theoretic point of view, we can identify two steps to formulate such capability. The first one consists of defining meta-learning capability as a need instead of as a requirement (Salado 2021). In this case, we can either define scenarios in advance (when the outcomes of the system which are derived by the needs are explicit and known), or define scenarios abstractly (for example, we can require the system to learn whatever it needs to learn in its lifetime). This still leaves an

open question as to how to define abstract learning as a need and more precisely, how to define the learning of unknown skills as a defined outcome. This approach, nevertheless, implies using direct validation techniques because the concept of verification (related to meta-learning) becomes inapplicable. The second step consists of identifying the functions that enable the meta-learning capability to happen. In other words, to derive the functions that enable meta-learning, we must first identify the mechanics by which learning occurs and, more precisely, the internal processes the intelligent systems go through while learning is achieved. These functions can then be subjected to verification.

Nevertheless, one fundamental question remains for which we do not have an answer yet: which type of V&V activities and how many of them will be necessary to gain confidence about such an elusive capability as meta-learning? Formalizing the definition of reasoning capability in this context can be challenging given the fact that the causes and uses of reasoning capability in intelligent systems with meta-learning capability that have been around for a long time, for which we have abundant data, and for which we can describe the results from numerous experiments (i.e., with human cognition) that are still debatable among logicians and mental model theorists (Mercier and Sperber 2017; Schroyens et al. 2001).

18.4.4 Encapsulation of Intelligent Properties

In behavior-preserving systems, functionality can be encapsulated in modules that may be aggregated hierarchically. This allows for decomposing and composing systems during the design, manufacturing, and integration of a system, such that a more sophisticated functionality can be provided by composing *lower-level* functions. Verification-wise, this functional encapsulation allows for partitioning and sequencing verification activities; which there may be beneficial to grow confidence with small steps to avoid large rework efforts and/or because of limitations to observe certain system attributes at some integration levels. Regardless, because the system (and all of its building components) are behavior-preserving, the number and sequence of verification activities that are executed on the system (and all of its building components) do not affect the final system-level behavior (with the exception of some physical parameters that may result in significant wear, such as shock testing).

However, an intelligent system, and by extension any intelligent component that forms it, may always be in a state of learning. Consequently, every verification activity that an intelligent system goes through becomes, effectively, a learning opportunity that the system may use. We call this aspect the *becoming* property of the system. The changes that the intelligent system experiences from these learning scenarios cannot be reset and may not even be totally reversible without *clearing* all of the knowledge the system had acquired up until such an event. This poses a major challenge to designing verification strategies since verification activities do not only serve the purpose of checking the correct operation of a system but they necessarily act as learning events as a consequence. Since the results of the verification activity cannot

be guaranteed because there may be errors in both the system under verification, the verification activity itself, and its execution, it is likely impossible to control what the system will learn during a verification campaign. Since, as discussed, unlearning is far from a trivial or deterministic activity for intelligent systems (as opposed to most rework or repair activities in behavior-preserving systems), verification poses an unprecedented risk to the system development process, as well to the correct operation of the system. In fact, because verification inherently modifies an intelligent system as it is verified, an intelligent system will always transition to an unknown state after a verification activity is executed. Furthermore, trying to avoid this situation by increasing verification in the hope of using behavioral trends suffers from the risk of the system over-emphasizing the learning occurring during verification, which may degrade its desired performance in an operational environment. This raises the question of whether a verification activity exists that can provide high confidence in the state of an intelligent system.

This problem is further amplified in intelligent systems that are subjected to hierarchical verification; i.e., the system becomes exposed to a higher number of uncontrolled learning events, and the uncertainty associated with the real state of the system couples with those of the other systems it integrates with.

We conjecture that, for this type of system, the notions of design and verification might need to be totally blended. Furthermore, success criteria may need to become fluid, as it may have to be adapted to the evolution of a verification campaign given that the intermediate results are uncertain.

18.5 Conclusion

We have shown through four specific challenges that there is a fundamental misalignment between current approaches to designing and executing verification and validation (V&V) strategies and the nature of AI-enabled systems. The main cause for such misalignment is the behavior-preserving condition, which is present in traditional systems but not in AI-enabled systems. The nature of *learning* requires intelligent systems to evolve their behavior.

The four challenges have been derived from conceptualizing intelligent systems within the framework of systems theory. The first challenge addressed situations of differential learning, where the intelligent system learns to respond to operational scenarios differently than to verification scenarios, with these differences remaining unnoticed. The second challenge resulted from the endogenous evolution of an intelligent system, which may strongly reduce the fidelity of verification and validation models. The third challenge addressed the difficulty to verify learning as a capability when an uncontrolled solution space is generated. Finally, the fourth challenge pointed to the effects of decomposing verification strategies, as verification scenarios become learning experiences for the system through a hierarchical chain of integration.

The four challenges have been accompanied by discussions on the practical difficulties that verification engineers will face when designing verification strategies for intelligent systems. We conclude that, as systems embed more intelligence, a paradigm shift in V&V activities will be necessary and suggest that advances in systems theory are necessary to underpin such a shift.

References

Arifin, S. N., & Madey, G. R. (2015). Verification, validation, and replication methods for agent-based modeling and simulation: Lessons learned the hard way! In *In Concepts and Methodologies for Modeling and Simulation* (pp. 217–242). Springer.

Bertalanffy, L. v. (1969). General system theory: Foundations, development, applications.

Chollet, F. (2019). On the measure of intelligence. cs.

Engel, A. (2010). *Verification, validation, and testing of engineered systems* (Vol. 73). Wiley.

Felder, W. N. (2018). Addressing the complexity challenge with adaptive verification and validation.

Finn, C., Abbeel, P., Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning* (pp. 1126–1135). PMLR.

Hoppe, M., Engel, A., & Shachar, S. (2007). Systest: Improving the verification, validation, and testing process-assessing six industrial pilot projects. *Systems Engineering, 10*(4), 323–347.

Hunt, E. B. (1962). Concept learning: An information processing problem.

INCOSE, D. D. W. (2015). Systems engineering handbook: A guide for system life cycle processes and activities. *San Diego, US-CA: International Council on Systems Engineering*.

Isele, D., Rahimi, R., Cosgun, A., Subramanian, K., & Fujimura, K. (2018). Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2034–2039). IEEE.

Johnson, M. (2013). *The body in the mind: The bodily basis of meaning, imagination, and reason*. University of Chicago Press.

Kuderer, M., Gulati, S., & Burgard, W. (2015). Learning driving styles for autonomous vehicles from demonstration. In *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2641–2646). IEEE.

Lakoff, G., & Johnson, M. (2008). *Metaphors we live by*. University of Chicago press.

Legg, S., Hutter, M., et al. (2007). A collection of definitions of intelligence. *Frontiers in Artificial Intelligence and applications, 157*, 17.

Mercier, H., & Sperber, D. (2017). *The enigma of reason*. Harvard University Press.

Murphy, G. L. (1996). On metaphoric representation. *Cognition, 60*(2), 173–204.

Nguyen, A., Yosinski, J., & Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 427–436).

Rendell, L. A., Sheshu, R., & Tcheng, D. K. (1987). Layered concept-learning and dynamically variable bias management. In *IJCAI* (pp. 308–314).

Salado, A. A (2021). Systems-theoretic articulation of stakeholder needs and system requirements. *Systems Engineering, 24*, 83–99. https://doi.org/10.1002/sys.21568.

Salado, A., & Kannan, H. (2018). A mathematical model of verification strategies. *Systems Engineering, 21*(6), 593–608.

Salado, A., & Kannan, H. (2019). Elemental patterns of verification strategies. *Systems Engineering, 22*(5), 370–388.

Salado, A., & Nilchiani, R. (2014). A categorization model of requirements based on max-neef's model of human needs. *Systems Engineering, 17*(3), 348–360.

Schroyens, W. . J., Schaeken, W., & d'Ydewalle, G. (2001). The processing of negations in conditional reasoning: A meta-analytic case study in mental model and/or mental logic theory. *Thinking and Reasoning, 7*(2), 121–172.

Sengupta, S., Chakraborti, T., & Kambhampati, S. (2019). Mtdeep: boosting the security of deep neural nets against adversarial attacks with moving target defense. In *International Conference on Decision and Game Theory for Security* (pp. 479–491). Springer.

Suto, I. (2012). What are the impacts of qualifications for 16 to 19 year olds on higher education? a survey of 633 university lecturers. *Cambridge Assessment*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. arXiv:1312.6199.

Thrun, S., & Pratt, L. (1998). Learning to learn: Introduction and overview. In *Learning to learn* (pp. 3–17). Springer.

Vilalta, R., & Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial Intelligence Review, 18*(2), 77–95.

Wang, X., Li, J., Kuang, X., Tan, Y.-A., & Li, J. (2019). The security of machine learning in an adversarial setting: A survey. *Journal of Parallel and Distributed Computing, 130*, 12–23.

Wymore, A. W. (2018). *Model-based systems engineering* (Vol. 3). CRC Press.

Xiang, W., Musau, P., Wild, A. A., Lopez, D. M., Hamilton, N., Yang, X., Rosenfeld, J., & Johnson, T. T. (2018). Verification for machine learning, autonomy, and neural networks survey. arXiv:1810.01989.

Yilmaz, L. (2006). Validation and verification of social processes within agent-based computational organization models. *Computational & Mathematical Organization Theory, 12*(4), 283–312.

Yilmaz, L. (2015). *Concepts and methodologies for modeling and simulation*. Springer.