INSTITUTE FOR DEFENSE ANALYSES

Mental Model Edit Distance as a Measure of Difficulty for Adaptation to Novelty

Joshua Alspector, Project Leader

March 2023

IDA Non-Standard D-33425

INSTITUTE FOR DEFENSE ANALYSES
730 East Glebe Road
Alexandria, Virginia 22305

The Institute for Defense Analyses is a nonprofit corporation that operates three Federally Funded Research and Development Centers. Its mission is to answer the most challenging U.S. security and science policy questions with objective analysis, leveraging extraordinary scientific, technical, and analytic expertise.

About This Publication

This work was conducted by the IDA Systems and Analyses Center under contract HQ0034-19-D-0001, Project DA-5-4648, "Science and Technology Support for SAIL-ON Program," for the DARPA. The views, opinions, and findings should not be construed as representing the official position of either the Department of Defense or the sponsoring organization.

Acknowledgements

Margaret E. Myers, Arun S. Maiya, Brian A. Haugh

For More Information

Joshua Alspector, Project Leader
jalspect@ida.org, 703-728-6032

Margaret E. Myers, Director, Information Technology and Systems Division
mmyers@ida.org, 703-578-2782

Rigorous Analysis | Trusted Expertise | Service to the Nation

Mental Model Edit Distance as a Measure of Difficulty for Adaptation to Novelty

Joshua Alspector[a]

[a]*Institute for Defense Analyses (IDA), Information Technology and Systems Division, 730 East Glebe Road, Alexandria, VA, 22305-3086, USA*

Abstract

We propose that the amount of change in an agent's mental model required to accommodate novel experiences is a measure of difficulty for a task. We define adaptation to novelty as learning to change and augment existing skills to confront unfamiliar situations. Agents have skills to perform tasks in an environment. Skills are viewed as algorithms to accomplish a range of tasks. If new skills are needed for adapting to a novel situation, these skill programs will need modification or augmentation. The degree of modification is considered as an edit distance in representation space using algorithmic information theory. The representation edit distance (RED) is related to simple, interpretable, and intuitive models in a variety of environments. Simple models almost as accurate as the best complex machine learning models often exist in practical domains as an example of the Rashomon effect. In this paper, we propose that the amount of editing of an effective representation (the RED) used in a set of skill programs in an agent's mental model is a measure of difficulty for adaptation to novelty. A further approximation, the Mental Model Edit Distance (MMED) is a practical, semantic, and intuitive approximation to RED. We present some notional examples of how to use RED for predicting difficulty and how MMED can be used for the semantic representations found in the DARPA SAIL-ON program.

1. Framework for adaptation to novelty

1.1. Introduction

This paper is an expansion of previous work [1] that presents a framework for using Representation Edit Distance (RED) to determine difficulty of adaptation to novel experiences with any machine learning method. Here,

we elaborate on that viewpoint by relating it to simple intuitive measures with applications to domains used in DARPA's SAIL-ON program. For completeness, we reprise the abovementioned framework for agents to learn and adapt to novelties based on Algorithmic Information Theory (AIT) [2] and the Minimal Description Length (MDL) principle [3]. We chose AIT because it is based on a small set of sound principles, because it is widely applicable to all machine learning methods from traditional AI to deep neural nets, and because instantiations within the framework are falsifiable by testing. Its use depends only on information content and generation, their representation structure, and the probability distributions involved. Agents are viewed as information processing machines (computers) whose experiences add to their information content and skills, and, therefore, their complexity. Complexity in the form of algorithmic entropy is useful for determining difficulty of adaptation to novel experiences. However, because AIT must be applied to optimal and non-computable quantities, one must search for methods that can approximate optimality but are still useful to practically predict adaptation. For this, we hypothesize that a RED using near-optimal representations can predict approximate difficulty of adaptation.

In our view, novelty is in the eye of the beholder. To an agent, any sufficiently advanced technology is indistinguishable from magic [4]. But if an agent is familiar with the technology, the novelty, and therefore the magic, disappears. In competitions, conflicts, and confrontations, a competitor strives to achieve surprise through novelty to the adversary. The chessmaster prepares a surprising move, the hacker creates a zero-day exploit, and the warrior uses a newly developed secret weapon. The agent's adaptation to novelty and the difficulty associated with such adaptation is the subject of this paper. Contributions include 1) a framework for using any machine learning method that can adapt to novel experiences; 2) a representation change measure (RED) for determining difficulty of adaptation; 3) a view of novelty as a mismatch between an agent's mental model expectations and its observations; 4) a view of detection, adaptation, and characterization of novelty using ideas in the framework; 5) some notional examples of how to approximate the framework's representations for prediction testing; 6) a justification for using some intuitive, semantic representations as a further practical approximation to RED; and 7) some examples of intuitive representations used in SAIL-ON and how they may be used for determining difficulty of a task that a SAIL-ON agent is asked to perform.

1.2. A general setting for adaptation

Consider a primary agent operating in an environment in which novelty is introduced. Novelty can be seen as a reconfiguration of previously existing environmental elements in a new arrangement, or as one or more elements not previously encountered, or as a previously encountered arrangement of elements in a novel context. It may be a situation newly encountered by the primary agent or a novel situation introduced by a secondary agent. The primary agent possesses a mental model of the environment that may include other agents and their mental models (Fig. 1). The mental model includes a model of the environment composed of built-in knowledge and skills (*priors* in Bayesian terms) and the ability to acquire new knowledge and skills from experience and adapt to novel environmental elements and situations that the agent encounters. Agents have sensors through which they perceive the world, perhaps incompletely and with distortions, and actuators for interacting with their environment. The agent may observe the world passively, enabling the ability, for example, to categorize previously unseen visual object classes. The agent may also be able to act in the world and affect the environment to better adapt to novelty, for example, by moving to get a better view. The ability to quickly and appropriately acquire new skills as a result of experience (including the ability to adapt to novelty) is considered a measure of intelligence by some [5].

The agent senses the elements of the environment through its perceptual system, which is composed of sensors and computational components. The system does not necessarily render the environment perfectly. This imperfect view is modified further by the agent's ability to incorporate the view into its mental model. The mental model view is its interpretation of the environment, which includes an interpretive program for prediction encompassing the agent's skills in the perceived environmental situation. As in Fig. 1, the representation may include distorted or missing entities, attributes, or relations.

A practical agent likely has a built-in model of its environment even if there are multiple domains and if additional exploration is required to specify which domain is appropriate for each instance and task. This model may change as it explores and executes tasks as a result of experiences encountered. If it detects something unexpected or novel or has difficulty executing its tasks, it may adapt to change its model to perform as required. Intuitively, we may view the difficulty of changing its model as a series of steps that result in a set of adapted skills sufficient to perform well on the task at

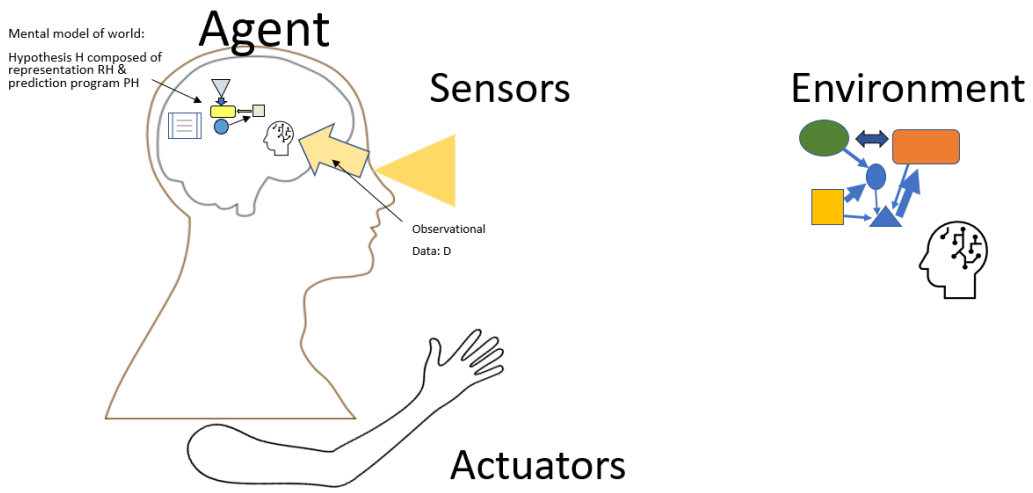

Figure 1: An agent, the environment, and its mental model

hand. In previous work [1], we formulated this process as an edit distance on an internal representation AIT concepts. The setting used previously [1] considers an agent, its mental model, and the environment as it might be represented in a simulation that provides a ground truth. A real-world environment can only be represented in an internal model. Its ground truth is approximated by observations backed by possibly incomplete understanding. The AIT formulation [1] provides an abstract theoretical viewpoint that does not depend on models of the environment or domain but is still able to make predictions. We will repeat the reasoning that justifies the viewpoint here, following Chollet's [5] method for determining a measure of intelligence, but applying it here to difficulty of adaptation to novelty.

We propose a mental model for an agent consisting of a representation portion and a prediction portion. Following the MDL principle [3], the total description length of these two portions, the "structure" of the model, plus a third component describing the observation data, the "random noise" of the data deviating from the "structure" of the model, should be near a minimum. The representation portion may be, for example, a graph of concepts and relationships and their probability distributions. The complexity

of a chosen representation is unlimited as more concepts and relations are learned. An example is the representation of the novel coronavirus, which has become more complex as the world progressed from understanding it as an infectious disease to developing vaccines that require understanding at an atomic level including the complexities of protein folding and human physiology. There are deeper levels of understanding to come. From an AIT perspective [2], we view a mental model of a situation as a hypothesis, H, in a two part form. Part 1 is the representation of the hypothesis, RH, such as a function, a neural network, or a probabilistic graphical model. Part 2 is a prediction program that analyzes observational data, assesses the situation, and generates predictions, PH. $L(H)$ is the length in bits of the hypothesis description and is the sum of the descriptions lengths of the representation and the prediction program. This is the structural part of MDL.

Observation data, D, is encoded with the help of the hypothesis, H, so that $L(D|H)$ is the length, in bits, of the data description for use with the hypothesis model. This is sometimes considered as a noise term. An example would be a model for the prediction of the location of a planet and directions for pointing a telescope at it. An instrumented telescope would record the planet's observation data as a density of light in a location. The density might have a gaussian distribution. If the hypothesis is poor, the light density would not be centered at the predicted location. This would indicate that the hypothesis needs structural change. If the light density is centered at the predicted location but the gaussian spread is larger than the expected parameter encoded in $L(D|H)$, this would indicate that the telescope needs better focus or an improved lens design because it generates a lot of noise in the observation.

The prediction portion of the mental model can be seen as a program that reads the hypothesis representation and operates on it to generate predictions and, perhaps, select a course of action. The complexity of such a program is approximately fixed in size. One may enforce constant program size by using the same generation/prediction program for all representations of a certain type. This is often the case for knowledge engineering methods. Such a fixed size is applicable to neural nets as well. For example, a neural net inference (prediction) program has the same instructions for all neurons and connections in the network graph. $L(PH)$ is the length in bits of the generation/prediction program, while $L(RH)$ is the length in bits of the representation. $L(H)$ is the sum of the two. According to the "crude" two-part MDL principle [2], a good model minimizes the sum $L(H) + L(D|H)$. Our

formulation of an MDL-inspired mental model has three parts because of the two structural portions of the hypothesis (prediction program and representation) plus the noise term describing the data. We argue that the prediction program portion can be ignored in the minimization of code lengths because its code length is close to constant in comparison to the code length of the representation. Thus, once a machine learning and prediction method is chosen, a good representation would be one where its description length plus the description length of observational data is minimized.

2. Novelty detection, characterization, and adaptation

2.1. Detecting novelty

The prediction program described above operates on a representation that is learned, built-in, or, most likely, some combination of the two. This representation may be adapted if the environmental data perceived by the agent does not match the expectation generated by the prediction program. The representation characterizes the perceived state of the world, and if a novel element is introduced or perceived, the mental model adapts its characterization to correct its representation and, potentially, improve performance on tasks using the corrected representation. This ability to adapt the representation can be called *characterizing novelty*.

Novelty and surprise are factors that arouse interest and motivate exploratory or avoidance behavior [6]. Novelty has not been previously experienced or encountered, while surprise refers to something unexpected. In our view, the mechanism for novelty and surprise is the same, stemming from an observation that does not match an expectation. This mismatch is a novelty or surprise detection mechanism. "Surprise" requires introspection to determine if this entity or situation had been previously encountered. If it has, the mismatch could result from an unexpected context for the situation leading to surprise rather than novelty. Finding a live fish in a desert is an example of surprise, or contextual novelty, but not inherent novelty to the agent.

Prediction errors from novelty and surprise also appear in "predictive coding" architectures [7]. These are layered systems going from a hierarchy of sensory input levels to levels encoding information in more abstract form. The bottom-up flow of sensory information to abstract representations is paralleled by top-down information predictions to the lower levels. This allows higher-level stages to receive information only through the information

mismatch between their predictions and sensations, so that higher levels receive only unexpected information. Our view of novelty encountered by an agent depends on a type of predictive coding within the mental model where the representation can be considered a description code in the MDL sense.

2.2. Characterizing novelty in representation

A variety of representations are depicted in Figure 2. In knowledge engineering, a knowledge graph of entities and relations is read by the prediction program and used for inference. In a neural network, nodes and connections create latent representations during the process of learning. These may be difficult to disentangle in the manner of a symbolic knowledge graph, but the simple procedural prediction program produces output inferences just as the program for a knowledge graph does. The third depiction is for regression to a set of data points using, for example, the family of polynomials. Any of these representations plus others such as decision trees, belief nets, and probabilistic graphical models can be cast into a prediction program portion and a representation portion to be minimized along with data for a mental model using MDL techniques. MDL principles [3] can be considered the practical version of AIT [2]. We will motivate our methods using these ideas.

To adapt its mental model to novelty, the agent can utilize priors, P, like prior knowledge and prior mental models along with any new experiences, E, it encounters that seem relevant or valuable to such adaptation. Prior models and knowledge add an inductive bias to the adaptation due to the structure of such knowledge and models. For example, if a prior model is in the form of a fully connected neural net, it will not adapt as well as a convolutional neural net (CNN) to a novelty in the visual domain. If the model has an arithmetic and logic unit, as digital computers do, it can be designed to adapt better to knowledge graph inference or first order logic than a neural net will. The difficulty of adaptation depends strongly on the prior model and the task. Adaptation will also depend on the algorithm that optimizes the model based on the representation, priors, and experience. We will call the ability to generalize prior models to a novel situation the *generalization difficulty*, GD [5], as Chollet does but applied to novelty adaptation rather than a theory of intelligence.

Regardless of the method used, the change in representation resulting from adaptation can be considered characterizing novelty. If the representation is a knowledge graph, an agent's mental model seeking to effectively

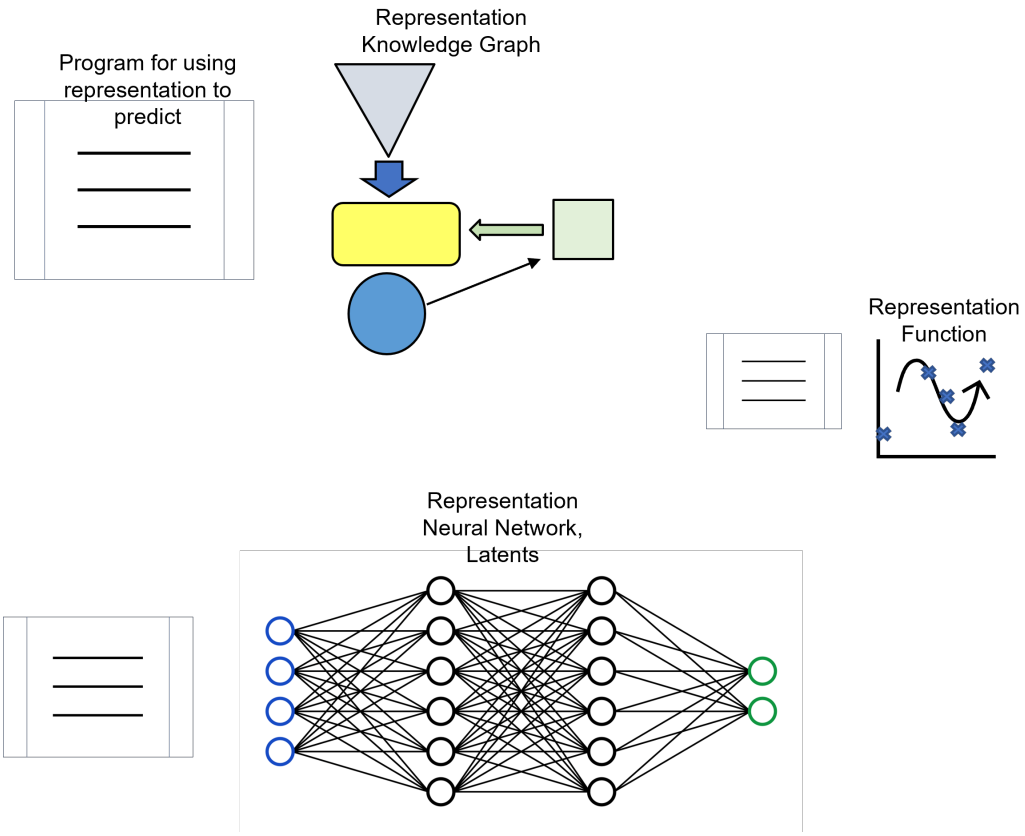

Figure 2: Various machine learning representations and their prediction programs

adapt to a new entity discovered by observation must add the entity and any relationships observed to its knowledge graph. Since the beginning of the pandemic caused by the novel coronavirus, the knowledge graph representing our understanding of the virus has been augmented by its genetic code, its transmission modes, its shape and the shape of variants, the types of infection observed, the effectiveness of various types of vaccines, etc. If the representation is a neural network, characterization may consist of latent features embedded in the network layers and the connection strengths. For regression, characterization consists of functional forms and parameters. The semantics are clearer in a knowledge graph where natural language is most likely used, but each representation type has its own language and semantics.

2.3. Approximate information distance

A useful concept is the information distance between two information objects, say x and y, such as computer programs. It is the size of shortest description that transforms each object into the other [8]. Mathematically, this algorithmic information distance (AID) is the maximum of the conditional Kolmogorov complexities of the transformation programs up to a logarithmic additive term. It can be recast to remove conditional terms.

$$\begin{aligned} AID(x,y) &= max[K(x|y), K(y|x)] \\ &= K(xy) - min[K(x), K(y)] \end{aligned} \tag{1}$$

It is a universal distance function of shorter length than any other distance function and also a metric. Here, xy is the concatenation of the strings. Note that strings x and y may have mutual information to be compressed; for example, they may be a pre-novelty program to be edited for a post-novelty program. To adjust for the AID being larger for longer strings, normalize by the maximum of the string lengths to obtain the normalized information distance

$$NID(x,y) = \frac{AID(x,y)}{max[K(x), K(y)]}, \tag{2}$$

which is a metric.

One can use compression algorithms like gzip to measure distances between objects for a practical substitute for the uncomputable Kolmogorov complexity, K, of a string using a compressed but approximate version, Z. The normalized compression distance (NCD) is

$$NCD(x,y) = \frac{Z\left(xy\right) - min\left(Z(x), Z(y)\right)}{max\left(Z(x), Z(y)\right)}, \tag{3}$$

which may be useful for bit-string representations.

One can use the statistics of web search engines to measure distances between semantic ideas, not bit strings. Such concepts come from human knowledge and require semantic context for their meaning. Using the probability that the search term x appears in a page-indexed search engine G, one can derive the normalized web distance, $G(x,y) = NWD(x,y)$, between terms x and y as

$$NWD(x,y) = \frac{[G\left(xy\right)] - min[(G(x), G(y))]}{max[(G(x), G(y))]}. \tag{4}$$

We view the search engine, G, as a compressor using the Web, and $G(x)$ as the binary length of the compressed version of the set of all pages containing the search term x. This is not a metric because it does not satisfy the triangle inequality. If concept x ("fried") is semantically close to concept y ("chicken") and concept y is semantically close to concept z ("feather"), it is possible that concept x can be semantically very different from concept z. The NWD is useful where semantic concepts, such as those in knowledge graphs are the information objects of interest. Semantic concepts are common in the representations used in the DARPA SAIL-ON program [9].

2.4. Information theoretic measures

Difficulty of generalizing a skill program to novelty

The previously mentioned generalization difficulty, GD, is a measure of how much the pre-novelty computational mental model must change to adapt acceptably to novelty given a set of experiences in a curriculum, C, for learning. It is notionally an edit distance between pre- and post-novelty mental models not unlike the information distances in the previous section. The concept of adapting acceptably must be defined for each task, T, perhaps by a threshold, θ, for a performance measure (e.g., 90 percent correct classification) that the post-novelty computational mental model must reach. Proficiency to reach the acceptable threshold for each task has a value, T, representing the task's subjective importance to the agent expressed as a function, $\omega_T(\theta_T)$.

Algorithmic complexity (AC) measures information content and is also known as Kolmogorov complexity, denoted by K. The AC of a binary string,

$K(s)$, is the length of the shortest program that outputs the string, s. $K(s)$ is not a computable function of s, so an approximate measure [10] will have to suffice for practical use. This allows approximating $K(s)$ with the compressed version obtained with any lossless compressor C, so that $C(s) = K(s) +$ constant. The use of zip-style compression, Z, (for strings) and Google-frequency-style compression, G, (for ideas) as distance measures is motivated by the need for practical measures. We will propose further approximations to Kolmogorov complexity or AC that justify the use of intuitive, semantic representations that are typical in the SAIL-ON program.

The relative algorithmic complexity (RAC), $K(s_1|s_2)$, is the length of the shortest program that, taking s_2 as input, produces s_1. The RAC, like the AC, can also be estimated by compression [11]. To compare the edit distance required to transform one string to another, one must relate the edit distance to the size of some program. For strings of random bits, which have high Shannon entropy, the program size needed, K, and the string edit distance are of the same order. If there are regularities in the bit string, such as repeating bit sequences, the program size may be much shorter in a compression code.

Programs can be compared by RAC to determine, for example, how much a pre-novelty program must be edited or adapted to become a post-novelty program. For an agent, a skill program represents a snapshot of its task-specific capabilities including the ability to adapt to novelty within the task. Following Chollet [5], consider a task T; Sol_T^θ, the shortest of all possible solutions of T of threshold θ (shortest skill program that achieves at least skill θ during evaluation, which may be viewed as the agent's post-novelty program), and $TrainSol_{T,C}^{opt}$, the shortest optimal training-time solution given a curriculum (e.g., the pre-novelty program that has the best adaptability). The (developer-aware) generalization difficulty (GD) is defined as the shortest program that, taking as input the initial system (which may contain built-in priors such as convolutions) plus the shortest possible program that performs optimally over the situations in curriculum C, produces a skill program that performs at a skill level of at least θ during evaluation, normalized by the length of that skill program. Thus, if the developer builds in the solution to the system at $t = 0$, the GD is low, while, if the developer relies only on training data without built-in prior bias, the GD is high. GD represents the edits you would have to make to the shortest training-time solution (what training should do) to obtain the evaluation-time solution while using the initial agent's capabilities without penalty. We will justify the use of

RED as an approximation to GD.

$$GD^{\theta}_{a,T,C} = \frac{K\left(Sol^{\theta}_{T} | TrainSol^{opt}_{T,C,a_{t=0}}\right)}{K\left(Sol^{\theta}_{T}\right)} \tag{5}$$

Note that this quantity is between 0 and 1 by construction. An approximation to GD, like RED, may be used as a measure to judge how difficult a given task would be. If the environment that the agent confronts is a simulated one, a novelty generator can test the adaptation ability A of an agent a to produce a near-optimal skill program. Because such a generator would contain all testable novelties, a developer should be able to produce a program that can adapt well to all of them. Relative complexities can be determined based on the edits needed to turn test agents into the near-optimal agent. We will use MDL to propose a practical, approximate, intuitive version of GD, RED, which is computable.

A task with high GD is one where the evaluation-time behavior needs to differ significantly from the simplest possible optimal training-time behavior in order to achieve sufficient skill. A more generalizable skill program can handle a wider variety of situations than the ones it was trained on. A novelty-adaptable agent is one that can detect, characterize, and adapt to novelty and thus is better able to deal with future uncertainty and novelty.

Built-in priors

Priors (labeled P) of an agent, a, relative to a task, T, and a skill threshold, θ, is the fraction of the AC of the shortest solution of T of skill threshold θ that is explained by the initial system at the start of the training phase. This does not consider the training data for an optimal adaptation program as GD does but only the baseline initial agent at $t = 0$. This is the length (normalized by the AC of the solution) of the shortest possible program that, taking as input the initial agent skill, produces the shortest solution of T that performs at a skill level of at least θ during evaluation.

$$P^{\theta}_{a,T} = \frac{K\left(Sol^{\theta}_{T}\right) - K\left(Sol^{\theta}_{T} \middle|\, a_{t=0}\right)}{K\left(Sol^{\theta}_{T}\right)} \tag{6}$$

If the initial system solves the task, the complexity of the solution conditioned on the initial system is 0 and the priors are at the maximum value of 1. If there is no built-in knowledge, the conditional term is high and the same as the solution complexity, so the priors have value 0. "Priors" can be

interpreted as the "amount of relevant information" embedded in the initial system. Priors may be considered as contained in the performance of a baseline, pre-novelty-adaptation agent that has no training. Components may include a built-in knowledge base or transfer from pre-training on a similar task or other components of core knowledge.

Experience relevant to novelty adaptation

At each step, the agent is exposed to a situation, responds according to its current program, and receives feedback from the environment, thus gaining information that may affect the agent's future responses if the agent is adaptive enough to use it. The increase in experience at time step t is

$$E^{\theta}_{a,T,t} = K\left(Sol^{\theta}_{T} \middle|\, a_t\right) - K\left(Sol^{\theta}_{T} \middle|\, a_{t-1}, data_t\right). \tag{7}$$

We can define the experience, E, accumulated by an intelligent system about a task during a curriculum as

$$E^{\theta}_{a,T,C} = \frac{1}{K\left(Sol^{\theta}_{T}\right)} \sum_{t} E^{\theta}_{a,T,t}. \tag{8}$$

E measures how much easier it is to create the solution program if the agent makes optimal use of the new information at each step, but this is not necessarily the actual use of information by an agent in a task. Experience can be considered the amount of accumulated information that is novel to the agent and relevant to the task. There may be other information that is not relevant or that is repetitive or redundant to what the agent already knows. The AC of an optimal adaptive agent should increase at each step if the information is novel and relevant, but not if the information is irrelevant. Incremental learners like backpropagation-optimized neural nets may require several exposures to the information to achieve the full increase in experience. The potential uncertainty reduction from the data is that for an optimal learner. Learning curves measure the speed of adaptation to a task from data and depend strongly on the optimization procedure. The increase in AC may be a way to measure novel and relevant information gain if we can approximate K with a computable C from an off-the-shelf compressor.

Adaptability

We define the adaptation ability, A, of agent, a, rather than the intelligence [5] of a system, as a measure of its skill-acquisition efficiency over a

scope of tasks, with respect to priors, experience, and generalization difficulty. For a single task, T, adaptability can be expressed as the difficulty of the task to the agent divided by the total information contained in the agent's priors and experience.

$$A_{a,T}^{\theta_T} = \frac{GD_{a,T,C}^{\theta_T}}{P_{a,T}^{\theta_T} + E_{a,T,C}^{\theta_T}} \tag{9}$$

This adaptability can be viewed as a generalized sample efficiency that measures how well the agent turns experience and priors into new skills. Put another way, it measures how well an agent adapts to novelty where GD measures the degree of novelty or difficulty. For a scope of tasks on which the agent is tested, the adaptability is averaged over the tasks and weighted by the perceived task value.

$$A_{a,sc}^{th} = \underset{T \in sc}{Avg} \left[\omega_T(\theta_T) \sum_{C \in Cm_T^{\theta_T}} \left[Pb_c \cdot \frac{GD_{a,T,C}^{\theta_T}}{P_{a,T}^{\theta_T} + E_{a,T,C}^{\theta_T}} \right] \right] \tag{10}$$

Sufficient skill threshold, θ, is the subjective threshold of skill associated with a task, above which a skill program can be said to "solve" the task T. It is a property of a task. The average over tasks has an implied threshold, th. The set of all tasks in scope is sc. The probability of a single curriculum is Pb. The agent's scope is the space of tasks of non-zero value for which the agent is capable of producing a sufficient solution after a training phase.

$P+E$ (priors plus experience) represents the total exposure of the system to information about the problem, including the information it starts with at the beginning of training. This could be approximated by the information in a set of experiences. This should also include any start-up information either in a baseline system or as a knowledge base that may be used as core knowledge.

The sum over a curriculum subspace, weighted by the probability of each curriculum, represents the expected outcome for the system after training. This assumes that agents are exposed to multiple curricula or training sets in a supervised setting for each task. The subjective value placed on achieving sufficient skill at a task T is $\omega_T(\theta_T)$. The mental model's value function captures the relative importance of skill at each task subjectively. For humanlike intelligence [12], it might be high for tasks relevant to humans, but

low for tasks that are not relevant. It would be low for "nuisance novelty" [13], which does not affect task performance.

3. Representation Edit Distance (RED)

3.1. RED as a measure of difficulty

Prediction error can be the stimulus for learning new concepts and relations. Error may be used to define novelty as a new situation that an agent failed to predict accurately. In the real world, any description of the physical environment is necessarily incomplete and correct prediction is the only test of the mental model and its complexity is the only one that is relevant. An exception would be a simulated world where the model of environmental and agent behavior is fully specified. Note that environment complexity in a mental model may be reduced by the task the agent must perform because only a portion of the environmental description is needed for executing the task.

We propose RED as an approximate measure for use in determining difficulty. Defining this metric for a particular architecture requires some thought and will depend strongly on the structure of the representation. The size of an effective representation can be approximated by a type of minimum description length such that the representation "effectively" expresses the behavior of the system it describes in the mental model. For example, if multiple graphical models, such as random forests, can adequately represent the task and skills needed to a threshold accuracy, and if such a representation can be reduced optimally to a few graphs that are accurate enough, the reduced model can be said to effectively represent the system. We define a near-optimal representation as the minimum size model that can be used to approximate an oracle to a threshold accuracy. *Our hypothesis is that edit distances on near-optimal representations can be used for practical information-theoretic measures of difficulty and adaptability.*

3.2. Complexity for knowledge graphs

A knowledge graph (KG) [14] is a structured representation of facts, consisting of entities, relationships, and semantic descriptions. In Figure 2, an abstract knowledge graph is represented as a set of concept nodes containing semantic descriptions linked by relational edges. KGs may be hand coded or learned, and the representation graph may be embedded in a low-dimensional space. One way the KG may be compressed is to represent each entity in the

KG as a vector of discrete codes and compose embeddings from these codes [15]. An approximate complexity may be derived from such techniques.

KGs are meant to create a semantic model of the environment that may be used to perform inference. Inferences that do not match observation indicate that the KG must be changed or edited. This may be due to the introduction of novel concepts or relations. The edit distance from the previous KG to the new one is a measure of the difficulty. These distance calculations should ideally be performed on the most compressed version of the KG, which would be the Kolmogorov complexity of the graph. For practical use, we posit that a compressed version, perhaps using NWD or another intuitive measure, that incorporates semantic concepts is an acceptable substitute. Such a description of the environment and the task context should be reduced to its most complete, correct, and consistent but simplest form using a variety of techniques [14] for determining difficulty.

For an agent, a knowledge and skills graph encompasses concepts, actions, attributes, relations, and other elements useful in characterizing phenomena. An agent's mental model may contain a subset of a larger graph depending on the agent's skill and knowledge level. A useful example is the knowledge and skills needed to adapt to the novel coronavirus. When it first appeared, the previously held knowledge that it is an infectious disease was sufficient to promote the standard mitigation measures of quarantine, disinfection of surfaces, washing hands, and not touching your face. As the virus characterization revealed aerosol transmission, masks and social distancing were added as mitigation measures. At that stage, the level of granularity for characterization in an ontology of hierarchical concepts had limited success due to less-than-perfect following of mitigation procedures, and it became clear that a much finer granularity of knowledge and deeper characterization would be needed to create vaccines. This involved knowledge at a more atomic level. Knowledge of the molecular structure from the virus genome and protein folding physics revealed the structure in sufficient detail for groups around the world to develop vaccines. Even finer granularity of skills was need for vaccine technologies, especially for the mRNA vaccines. The level of characterization granularity reached is the "atomic" level for vaccines or the deepest level of hierarchical and relational knowledge sufficient for executing a task; in this case, developing a vaccine.

Using the refined and reduced semantic knowledge graphs before adaptation and after adaptation to novelty is a matter of replacing the GD of Eq. 5 with the normalized difference in description lengths (code-lengths or

compression lengths, C) of the two KGs.

$$RED^{\theta}_{a,T,C} = \frac{Cpretr\left(TrSol^{\theta}_{T}\middle|\, a_{t=0}\right)[to]Cpost\left(Sol^{\theta}_{T}\right)}{Cpost\left(Sol^{\theta}_{T}\right)} \tag{11}$$

RED, like GD, represents the edits ($[to]$ in the equation, which may be $Cpost - Cpretr$) one would have to make to the shortest training-time solution (what optimal training should do) to obtain the evaluation-time solution while using the initial agent's capabilities without penalty on compressed, near-optimal versions of the knowledge graphs. This means that $Cpretr$ must operate near-optimally not only on the pre-novelty data as $Cpre$ does but also on the training curriculum data. However, it can use the capabilities of $Cpre$ without being charged for edit distance on the $Cpre$ graph. Such an optimal use of training data may indicate that $Cpretr$ uses a post-training graph structure. We posit that the edit distance can be approximated by using the structure of the $Cpost$ graph for the $Cpretr$ graph without the newly learned entities and relations. The edits to add new entities and relations to the post-novelty graph are charged to RED.

This difference normalized by the post-novelty codelength is the effective RED in our framework if careful attention is paid to the correct refinement and reduction of the KGs. To determine difficulty of adaptation, we also need to estimate the potential increase in complexity of the agent when it experiences the learning curriculum and adapts optimally. This represents the experience in Eq. 7. For semantic concepts, this may simply be the addition of new concepts and relations to the KG. For example, the change in the structure of a new variant of the coronavirus expressed in atomic form such as a shape description may be called the approximate effective experience, $Eeff$. An optimal agent (an entire drug company plus consultants and knowledge bases) would need just a single exposure to this shape to learn how to make a vaccine if its prior knowledge included vaccine technology. Of course, this prior knowledge, Eq. 6, must also be estimated. For KGs, this is simply the compressed description length, $Cpre$, of the entire pre-novelty knowledge graph. The compressed description length of the priors, Pd, according to Eq. 6, would then be the length of the post-novelty solution graph, $Cpost$, minus the length, $Cpre$, divided by $Cpost$.

$$Pd^{\theta}_{a,T} = \frac{Cpost\left(Sol^{\theta}_{T}\right) - Cpre\left(Sol^{\theta}_{T}\middle|\, a_{t=0}\right)}{Cpost\left(Sol^{\theta}_{T}\right)} \tag{12}$$

Using Eq. 9 and the approximate effective values for priors, experience, and edit distance, the approximate difficulty of adaptation, $Aeff$, to novelty for the task is then

$$Aeff_{a,T}^{\theta_T} = \frac{RED_{a,T,C}^{\theta_T}}{Pd_{a,T}^{\theta_T} + Eeff_{a,T,C}^{\theta_T}}. \tag{13}$$

3.3. Complexity in regression

MDL is based on finding structure in individual data sequences [2]. Fitting data to probability distributions (models) extracts regularities viewed as representations that express the "structural" meaning of the data. Variations around the structure are the "random" portion of the data in the two-part MDL formulation. The MDL principle promotes the idea that for a given set of hypotheses H and data set D, one should try to find the hypothesis or combination of hypotheses in H that compresses D most. An effective, compressed structure in the MDL sense makes for a good representation.

In Fig. 2, a notional polynomial function fits a set of data points. Assume the family of polynomial functions form the set of hypotheses. The perfect fit polynomial (one degree less than the number of data points) would likely overfit the data and lead to poor generalization to new data from the same source. A third degree polynomial, the "model" as shown, seems like a good enough fit, leaving some random variation to the fitted data but able to generalize to new data. The particular fitted parameters, say $y = 3x^2+7x+2$, of the chosen third-degree model form a "point hypothesis" in the regression fit. Two-part MDL chooses a point hypothesis that minimizes the sum of description lengths of the model, $L(H)$, plus the data encoded with the help of the model, $L(D|H)$. By using the model of third-degree polynomials to fit the data, we extract the "structure" of the data in a compressed code, leaving only the random errors in the data fit to the hypothesis to encode. If the random portion is distributed according to a simply described function such as a normal distribution, the total code-length would be quite short. If, however, the new data fits a sine function better, the noise portion would be large. A large amount of noise indicates the model needs adaptation to the novel data. This is done by choosing a new function or a new family of functions and a new point hypothesis.

The length of the two-part MDL compressed code is $L(H) + L(D|H)$, and this quantity is to be minimized in a model selection and regression procedure. Difficulty of adaptation to novel data can be estimated by using

such a procedure. One may try a family of functions such as polynomials to fit a training set of data points. For the optimization procedure, one would select a degree of the polynomial and perform a regression on the data. The sum, $L(H) + L(D|H)$, would be minimized according to this optimization procedure, which may be a random search guided by a gradient. This would produce a pre-novelty model (the function) whose complexity or code-length can be determined for $Cpre\left(Sol_T^{\theta}\middle|\, a_{t=0}\right)$. After novelty is introduced or discovered in a test set of data, the model selection and regression procedure is repeated to determine $Cpost\left(Sol_T^{\theta}\right)$. The function family may be different or the degree of the polynomial may be higher post-novelty. Choosing the best function family and parameters can be considered as *characterizing* the novelty by finding the *structure* in which it resides. In any case, the priors, $Pd_{a,T}^{\theta}$, can be determined using Eq. 12. The effective experience, $Eff_{a,T,C}^{\theta_T}$, can be determined by assessing those data points that are novel and relevant in the test set by measuring the effect of each point on the model parameters by estimating the goodness of fit. The $RED_{a,T,C}^{\theta_T}$ is the edit distance from pre-novelty point hypothesis to post-novelty. We posit, in analogy with what we did for KGs, that the *Cpretr* near-optimal function for training is that of the *Cpost* function before parameter optimization for determining RED in Eq. 11. Putting these quantities together in Eq. 13 determines difficulty of adaptation for the agent to the task of regression for the new data starting from the pre-novelty agent's mental model.

3.4. Complexity in neural nets

In one view of neural network function [16], the layers of a deep net form a Markov chain of successive internal representations of the input. The information bottleneck (IB) bound characterizes the optimal representations that maximally compress the input for a given mutual information (MI) on the desired output. A trained deep net processes the input through the chain of hidden layers to the predicted output capturing relevant information in the process. Each hidden layer can be viewed as an encoder on the input and decoder to the output forming a representation characterized by the layer's MI to the input and to the output regardless of the features encoded. The MI measures the number of relevant bits that the input contains about the output label on average. The optimal learning problem can be seen as the construction of an optimal encoder of that relevant information via an efficient representation. The IB tradeoff between the compression of the input and the prediction of the output leads to an approximate optimal

representation. Deep learning effectively finds efficient representations in the hidden layers that are approximately optimal, and this plays a large role in accounting for its success including the surprising lack of overfitting in deep learning. However, this claim and the connection between compression and generalization have been disputed [17].

The problem of compressed models in neural nets has been studied in a supervised setting [18], where inputs to a neural model predict outputs after training. The predictive model should generalize well to inputs not in the training set. Using the MDL principle, one could choose, say, a uniform encoding model with K classes so that probability, p, of a class is $1/K$. The codelength gain, given "true" distribution q, compared to a uniform code is limited by the amount of mutual information between input and output. There are advantages [18] in using a prequential (prediction/sequential) code where one predicts the $n+1$ label after already having seen n input-output pairs. First, one trains a default model on a few samples of data, then one trains on the resulting encodings. Next, one uses this model for the next few data samples, retrains on all new encodings, and so on. For neural nets, a trainable deep net encoding data incrementally starting from uniform encoding works well even though model parameters are never encoded explicitly. For a convolutional network of depth 8, the authors [18] achieve a description length with a compression ratio of 50x, with test set accuracy that is better than that of a variational code [19].

To use these ideas practically requires a compressed coding of a neural network such as a uniform encoding of the synaptic weight parameters. Encodings may reduce the precision of the weights from a floating-point value to a fixed-point value, which, in practice, has been studied extensively, usually in the context of hardware implementations [20] [21]. The network adjacency matrix for the graph and the binary coded connections form the string to be edited by the optimization method, which may be supervised learning by back-propagation of errors. This learning can take place at full precision if required. We are interested in the compressed version for evaluation of the RED.

The procedure for determining difficulty of adaptation is much the same as in the previous cases discussed. Use the compressed version of pre-novelty and post-novelty networks for evaluation of $Cpre$, $Cpretr$, and $Cpost$ complexities. If, for example, weight dropout is used during training, $Cpretr$ would use the $Cpost$ net structure after training. If no structure change occurs, then $Cpre$ and $Cpost$ network structures are the same. Determine

the experience in the training curriculum by estimating the increase in complexity for the agent's neural network representation, assuming the agent can make optimal use of the training examples. A way to do this may be to experiment with batch learning of a representative training sample until the prediction performance saturates. The change in the compressed network representation represents the effective (novel and relevant) experience, $Eeff$. The description length of the priors, Pd, is given by Eq. 12. The RED is given by Eq. 11. Eq. 13 then gives the difficulty of adaptation for the agent and task.

3.5. Intuitive representations may work well

In many domains, there may be a multiplicity of representations producing similar accuracy results. This multiplicity has been called "the Rashomon effect" after the classic movie by Akira Kurosawa where the same facts of a crime are interpreted in different ways by a multiplicity of characters [22]. The use of intuitive representations for the calculation of these quantities is viewed as an example of the Rashomon effect [23]. In that study, the authors noted that a model from a simpler class is approximately as accurate as the most accurate model within the hypothesis space in many cases. The reason that simple, yet accurate, models often do exist is that the size of the Rashomon set is often large, where the Rashomon set is the set of almost-equally-accurate models from a function class. Thus the complexity analysis of model descriptions can be reconciled with the notion that a near-optimal representation, perhaps even an interpretable semantic representation, can be edited to provide a RED measure for determining difficulty. A recent paper [23] proposes using the Rashomon ratio, which measures the fraction of the hypothesis space contained in the Rashomon set, to measure the simplicity (or, conversely, the difficulty) of the learning problem. If the Rashomon set is large, many different optimization or learning procedures could lead to a model from the Rashomon set. The Rashomon ratio depends on a loss function, the hypothesis space, and a data set, while other statistical learning theory measures are data-agnostic, such as VC dimension, or depend on model properties, such as the geometric margins of support vector machines, in the space. The authors used data sets from the UCI repository and measured training and generalization performance from several algorithms (logistic regression, CART, random forests, gradient boosted trees, and SVM with RBF kernels) on all data sets. They observed that larger Rashomon ratios led to approximately similar training results across all algorithms (within about 5

percent). The authors show that one can bound the best empirical model from a simpler function class with the true risk of the best model within the more complex class (the union of all algorithms in their experiments), which likely contains a simple one having close to the best performance. The larger the Rashomon sets, the more likely they contain simpler models with good generalization guarantees. It turns out that training performance in simpler hypothesis spaces is correlated with test performance in more complex hypothesis spaces. Also, good training performance in a simpler space correlates with good generalization performance of other models in the more complex space. Their experiments suggest that similar performance across a range of algorithms with different hypothesis spaces is strongly correlated with a large Rashomon set. This makes possible an empirical procedure to determine whether a large number of models have approximately equal performance. The authors estimated the size of the Rashomon set and the associated Rashomon ratio by sampling from decision trees of depth seven. They are easy to sample and can divide an input space arbitrarily finely as tree depth increases and can approximate many other types of hypothesis spaces used by other machine learning methods.

4. Intuitive Representation in the DARPA SAIL-ON Program

4.1. Methods used in SAIL-ON

The approximate or intuitive representations above can be related to the quantities used to determine difficulty by the various novelty generating teams in the DARPA SAIL-ON program. Teams used simulated worlds in action-oriented domains where an agent has to perform a task, usually a planning task, in an environment. Difficulty was judged using easily interpretable intuitive representations. There were also perception-oriented domains relying on image data where the task was passive classification. Robotic domains where perceptions and actions are both required were not represented in the program. The diagram below shows the setting in SAIL-ON and describes, at a high-level, the functions and methods of adaptive agents in the program. Most of the methods used were tailored to the domain at hand but, some, like non-parametric Bayesian analysis, would work with any domain. Methods to predict levels of difficulty for adaptation, which were required by the program, varied from one novelty generating team to another. Agent teams had to adapt to the novelties generated. These methods of novelty generation did not estimate difficulty using AIT as in the above description

but, if both the teams' estimates of difficulty and those calculated using AIT are part of a large Rashomon set, they should give similar estimates.

The mental model in the AIT approach is composed of an executive program of roughly constant size that can use a compact representation of the world. All adaptations due to novelty result in changes to this representation but, we assume, not significantly to the executive program. This assumption means the information distance between the mental model of the world before and after adaptation to novelty is almost entirely within the representation so that RED is the relevant information distance. In the SAIL-ON program, performers' agents did not necessarily adhere to the prescription of separating the mental model into a constant size executive program and a variable size representation. As a result, the distance between pre-novelty and post-novelty mental models for SAIL-ON agents is a combination of elements that may be more complex than simply the compact representation of the world. In general, a pre-novelty and post-novelty adaptation difference of some sort, generally more complex than the compact representation, is used by the novelty generator teams to predict difficulty of adaptation as easy, medium, or hard. We speculate that the set of elements used by the novelty-adaptive agent teams can be reduced to a roughly constant executive program and a compact representation for most if not all such models.

As in Figure 3, the novelty adaptive teams generally use an executive program which may refer to a representation plus new percepts and feedback to make predictions. Teams that adapt to action-oriented domains, like those in the game Minecraft, tend to use symbolic methods. For these mostly symbolic systems, methods of adaptation use a violation of expectations (VoE) including low-level sensor prediction error (within the sensor system) and high-level state prediction error, which triggers mechanisms of novelty detection, characterization, and adaptation in many SAIL-ON agents. Anticipation of a novelty may be part of characterization, prediction, and VoE. Methods of adaptation include search of configuration/parameter/hypotheses space guided by guesswork or gradient/loss. This may include retraining on signals and active experimentation. For this discussion, characterization may change or add specific features or dynamics including novel behavior by external agents in the environment. These features, which may be parameters or semantic concepts, are related to the structure portion of our MDL description. To decide how to proceed, novelty adaptive agents perform internal evaluation by expectation checking and may use distance or error measures including measures of task performance to evaluate adaptations and con-

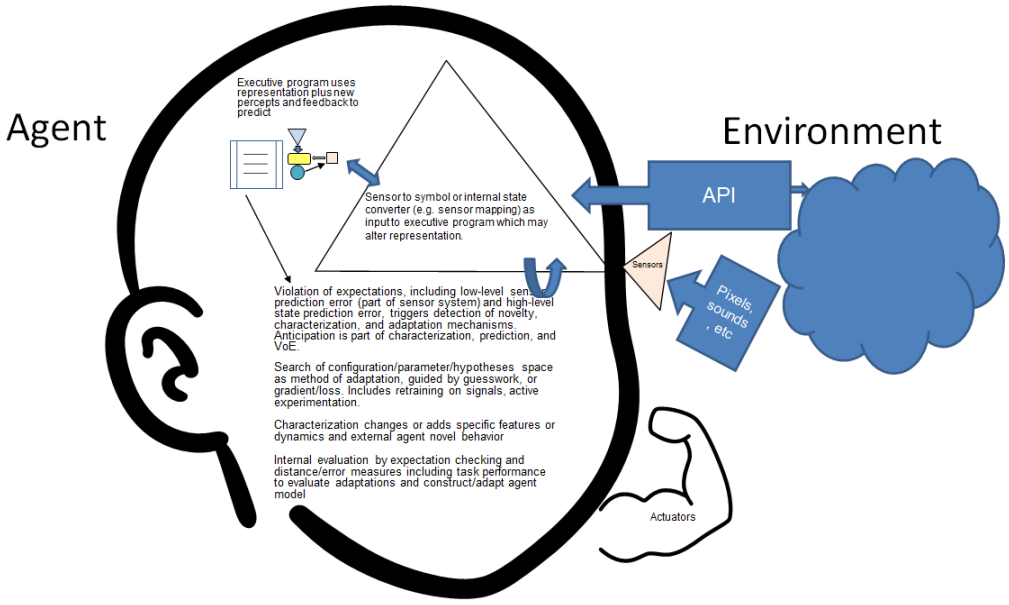

Figure 3: A mental model for novelty processing

struct or adapt the agent model.

Agent teams that adapt to novelty in perception-oriented domains have tasks such as image classification. They tend to use statistical methods such as deep neural network image classifiers. The classifiers can be considered a sensor to symbol or internal state converter (e.g., sensor mapping). These symbols are typically used as input to an executive program, which may alter representation using some of the symbolic methods above in a slow loop. The perceptual system may also have a reactive component for fast novelty processing and reasoning.

In summary, the novelty adaptive agents in the program use the following abstract components of novelty processing: representation, VoE, search of configuration space, characterization, internal evaluation, and sensor to symbol conversion for perceptual domains.

4.2. Difficulty in planning domains

Difficulty of a task for an agent depends on what skills the agent has, what the agent knows, how the skills and knowledge are organized and represented, and how the agent adapts to what it experiences in its environment.

This viewpoint has been discussed in the language of information theory and complexity. In the SAIL-ON program, agents that act in the environment use intuitive and semantic organizations for their world model. Teams are divided into those that create the novelties (generators) in an environment and those (adapters) that prepare an agent to perform a task in the environment and adapt to novelties that the first team generates. The adapters are not informed about the novelties prepared by the generators, so this experimental setup is single-blind.

Perceptual domains typically use neural net methods and will not be discussed here. However, the procedures previously discussed for RED determination in neural nets would be applicable. A task for an action-oriented agent in SAIL-ON involves planning a series of steps to reach a goal state in an environment, so planning software is typically used in such domains. Teams develop agents to respond to novelty in video-game-like, action-oriented environments similar to those in games like Minecraft, Monopoly, and Angry Birds. As an example, agents in a Minecraft-like environment have a goal to build something useful by searching for materials and crafting tools while being confronted with unexpected difficulties such as obstacles and dangers. The baseline agents have an understanding of their environment that they must update based on their experiences. Agents have strategies to achieve the goal based on mental models of the world and expectations of what they will experience according to various plans. Many teams execute these strategies using the classic planning language PDDL [24] and variants.

In Fig. 4, these strategies are listed notionally as plan 1, 2, 3, and so on. Depending on observations, one or another of these plans are executed with varying degrees of probability. Therefore, some plans are rarely executed and, consequently, if they happen to be the ones to accommodate the novelty, may be hard to discover. This has a bearing on the difficulty of novelty adaptation to a task for an agent. In actual implementations [25], these plans may not be stored but rather developed on the fly as the agent takes a snapshot of the world. In that case, the complexity is contained within the action plans box and any edit distance calculation refers to the plan-generating software within, as well as any stored plans. An information theoretic distance measure, like the NCD previously discussed, may work well for software planners [26] used in calculating a plan diversity measure or an edit distance. We propose going beyond NCD for approximating pre- and post-novelty plan distance by explicitly using the semantic descriptions that planners typically employ.

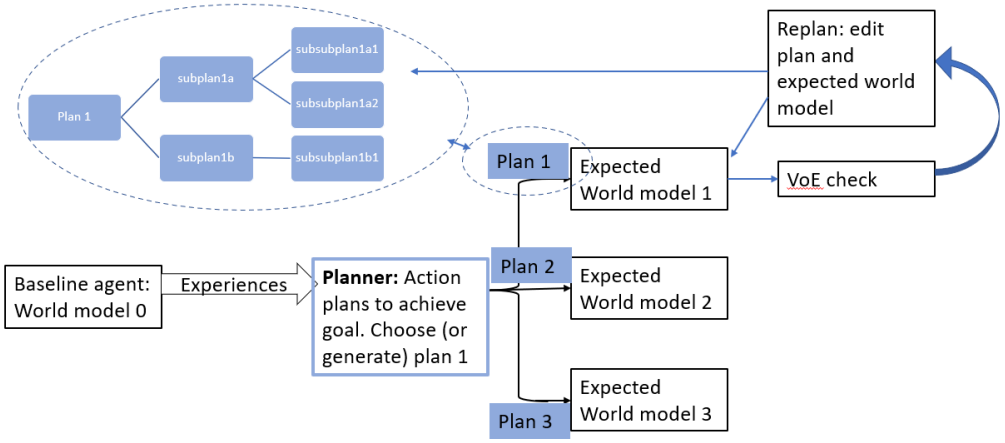

Figure 4: Planning graph

Plans are typically multi-step and hierarchical and are composed of subplans whose probability of choice depends on what the agent experiences. Plans and subplans can be represented in graph form through tools such as Graphplan [27], which can find a plan from its graph using its associated algorithm. The algorithm can be considered part of the executive program of roughly constant size relative to the world model and plan. Novelty adaptation is usually triggered when observations indicate a violation of expected world state. Expectation checking is typical of many SAIL-ON agent strategies. VoE necessitates replanning to avoid an obstacle or danger or simply to get to a world mental model state that meets expectations so that existing plans may be used. Replanning may increase the probability of some plans or sub-plans or may trigger a search for new plans from elements that might be novel in the environment. In any case, it involves an editing of the mental model of the agent. The mental model may consist of a graph with conditionals, branches, and weighted connections if the model has probabilistic elements. The edit distance between pre- and post-novelty graphs would be a candidate for a measure of difficulty in the same way as previously discussed if the graph is part of a large Rashomon set.

To determine the plan graph edit distance, one must find the plan graphs for the baseline agent, as well as the near-optimal plan graph for the novelty adaptive agent. It is assumed that the planner creates efficient plans rather than, say, repeating a plan step multiple times if a single step would have the

same effect. The process is similar to what was discussed for KGs. KGs may be part of the world model description that the agent consults to determine VoE and the needed edits to the world model count toward the edit distance. Furthermore, any edits to the steps in the plans and subplans count as well. Edit steps may be weighted by the inverse probability of the use of the plan or subplan for a task. Graph edit distance (GED) calculation is a well-known procedure that counts additions and deletions to links and has its own large set of techniques to deal with complexity.

The larger graph encompassing the plan, world model, and executive program can be thought of as the mental model of the agent. Note that for perceptual and robotic domains, this would include sensory portions that may be implemented as neural nets and would have a procedure different from graph editing for edit distance, so GED would be augmented by other types of information distances. If the executive program is approximately constant in size compared to the rest of the mental model expressed in graphical form then one can derive a MMED as the GED of the graphical portion that adapts to novelty plus other distances for the non-graphical portion. Testing whether the GED is a good predictor of difficulty also requires determining whether the solution to the novelty accommodation is part of a large Rashomon set, as previously discussed. The evaluation of agents and the number of computational steps they require to adapt can be compared to GED as validation of the procedure.

5. Conclusion and future work

We have described a framework based on AIT that is applicable to all machine learning methods for adapting to novelty. It remains to test these ideas with approximate instantiations that can provide predictions of difficulty and adaptability using this framework. Methods are needed for good, practical approximations to non-computable Kolmogorov measures within the framework. We propose that RED can be used as an approximate measure to predict difficulty. Research to provide theoretical justifications for methods to produce RED and for its use in prediction is needed. Likewise, it is important to provide experimental tests of predictions of the framework. We relate RED to simple intuitive measures, which encompass MMED, in domains used in DARPA's SAIL-ON program. More theoretical and experimental work is needed to establish and verify the framework proposed.

6. Acknowledgments

This work was supported by the Science of Artificial Intelligence and Learning for Open-world Novelty (SAIL-ON) program of the U.S. Defense Advanced Research Projects Agency (DARPA). It has been approved for public release; distribution unlimited. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. It has benefited greatly from discussions with Eric Kildebeck and David Musliner as well as other participants in the program.

References

[1] J. Alspector, Representation edit distance as a measure of novelty, AAAI Spring Symposium Series March 21–23 2022 Designing Artificial Intelligence for Open Worlds (2021).

[2] P. D. Grunwald, P. M. Vitanyi, Algorithmic information theory, homepages.cwi.nl/ paulv/papers/handbooklogic07.pdf (2007).

[3] J. Rissanen, Modeling by the shortest data description, Automatica 14, 465–471 (1978).

[4] A. C. Clarke, Profiles of the future: An inquiry into the limits of the possible, ISBN 978-0-33023619-5 (1973).

[5] F. Chollet, On the measure of intelligence, arXiv preprint arXiv:1911.01547 (2019).

[6] A. Barto, M. Mirolli, G. Baldassarre, Novelty or surprise?, Front. Psychol., 11 December 2013 (2013).

[7] R. Rao, D. Ballard, Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects, Nat Neurosci 2, 79–87 (1999) (1999).

[8] P. M. B. Vitanyi, F. J. Balbach, R. L. Cilibrasi, M. Li, Normalized information distance, Chapter 3 from Information Theory and Statistical Learning, 2009 (2009).

[9] DARPA, Sail-on proposers day (archived), DARPA archives (2019).

[10] D. Cerra, M. Datcu, Algorithmic relative complexity, Entropy 2011, 13, 902-914; doi:10.3390/e13040902 (2011).

[11] R. Cilibrasi, P. Vitányi, Clustering by compression, IEEE Trans. Inform. Theor. 2005, 51, 1523–1545 (2005).

[12] B. Kurdi, S. J. Gershman, M. R. Banaji, Model-free and model-based learning processes in the updating of explicit and implicit evaluations, PNAS 116 (13) (2019) 6035–6044.

[13] T. E. Boult, P. A. Grabowicz, D. S. Prijatelj, L. H. R. Stern, J. Alspector, M. Jafarzadeh, T. Ahmad, A. R. Dhamija, C. Li, S. Cruz, A. Shrivastava, C. Vondrick, W. J. Scheirer, Towards a unifying framework for formal theories of novelty, The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21) (2021).

[14] A. Hogan, E. Blomquist, M. Cochez, C. D'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A.-C. N. Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, A. Zimmerman, Knowledge graphs, arXiv:2003.02320v6 [cs.AI] 11 Sep 2021 (2021).

[15] M. Sachan, Knowledge graph embedding compression, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2681–2691; July 5 - 10, 2020 (2020).

[16] R. Schwartz-Ziv, N. Tishby, Opening the black box of deep neural networks via information, arXiv:1703.00810v3 [cs.LG] 29 Apr 2017 (2017).

[17] A. M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B. D. Tracey, D. D. Cox, On the information bottleneck theory of deep learning, ICLR 2018 (2018).

[18] L. Blier, Y. Ollivier, The description length of deep learning models, Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada 2018 (2018).

[19] G. E. Hinton, D. van Camp, Keeping neural networks simple by minimizing the description length of the weights, Sixth ACM Conference on Computational Learning Theory, Santa Cruz, July 1993 (1993).

[20] S. Gupta, A. Agrawal, K. Gopalakrishnan, P. Narayanan, Deep learning with limited numerical precision, arXiv:1502.02551v1 [cs.LG] 9 Feb 2015 (2015).

[21] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, Y. Bengio, Quantized neural networks: Training neural networks with low precision weights and activations, Journal of Machine Learning Research 18 (2018) 1-30 Submitted 9/16; Revised 4/17; Published 4/18 (2018).

[22] L. Breiman, et al., Statistical modeling: The two cultures (with comments and a rejoinder by the author), Statistical Science 16 (3) (2001) 199–231.

[23] L. Semenova, C. Rudin, R. Parr, On the existence of simpler machine learning models, arXiv:1908.01755v4 [cs.LG] 12 May 2022 (2022).

[24] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, D. Wilkins, Pddl-the planning domain definition language v1.2, Yale Center for Computational Vision and Control Tech Report CVC TR-98-003/DCS TR-1165 October, 1998 (1998).

[25] D. J. Musliner, M. J. S. Pelican, M. McLure, S. Johnston, R. G. Freedman, C. Knutson, Openmind: Planning and adapting in domains with novelty, Proceedings of the Ninth Annual Conference on Advances in Cognitive Systems Submitted 9/2021; published 11/2021 (2021).

[26] R. P. Goldman, U. Kuter, Measuring plan diversity: Pathologies in existing approaches and a new plan distance metric, Published in the Proceedings of AAAI-15, AAAI Press, Austin TX, January 2015 (2015).

[27] A. Blum, M. Furst, Fast planning through planning graph analysis, Artificial intelligence. 90:281-300 (1997).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE (DD-MM-YY)	2. REPORT TYPE	3. DATES COVERED (From – To)
00-03-23	Non-Standard	

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Mental Model Edit Distance as a Measure of Difficulty for Adaptation to Novelty	HQ0034-19-D-0001
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBERS

6. AUTHOR(S)	5d. PROJECT NUMBER
Joshua Alspector	DA-5-4648
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESSES	8. PERFORMING ORGANIZATION REPORT NUMBER
Institute for Defense Analyses 730 East Glebe Road Alexandria, VA 22305	NS D-33425

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR'S / MONITOR'S ACRONYM
Ted Senator DARPA 675 N. Randolph St, Arlington, VA 22203	DARPA
	11. SPONSOR'S / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

Project Leader: Joshua Alspector

14. ABSTRACT

The amount of change in an agent's mental model required to accommodate novel experiences is proposed as a measure of difficulty for a task. Adaptation to novelty is viewed as learning to change and augment existing skills to confront unfamiliar situations. Agents have skills to perform tasks in an environment. Skills are viewed as algorithms to accomplish a range of tasks. If new skills are needed for adapting to a novel situation, these skill programs will need modification or augmentation. The degree of modification is considered as an edit distance in representation space using algorithmic information theory. The representation edit distance is related to simple, interpretable, and intuitive models in a variety of environments. Simple models almost as accurate as the best complex machine learning models often exist in practical domains as an example of the Rashomon effect. In this paper, we propose that the amount of editing of an effective representation (the Representation Edit Distance or RED) used in a set of skill programs in an agent's mental model is a measure of difficulty for adaptation to novelty. A further approximation, the Mental Model Edit Distance (MMED) is a practical, semantic, and intuitive approximation to RED. We present some notional examples of how to use RED for predicting difficulty and how MMED can be used for the representations found in the DARPA SAIL-ON program.

15. SUBJECT TERMS

Mental Model, Novelty, Open World, Algorithmic Information Theory, Machine Learning, Planning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Ted Senator
Unclassified	Unclassified	Unclassified	Unlimited	30	19b. TELEPHONE NUMBER (Include Area Code) 703-526-2816