Context and event-based cognitive memory constructs for embodied intelligence machines

Navin Ipe [1]

[1]M.S.Ramaiah University of Applied Sciences

October 30, 2023

Abstract

Any algorithm that needs to "understand" information to be capable of taking "intelligent" decisions, needs to access a lifetime of memories and experience the world as an embodied consciousness. This paper emphasizes these concepts and proposes a few fundamental constructs that provide algorithms with the capability to understand the human world, build larger sets of cooperative machines and perform causal inferences without requiring human intervention.

Context and Event-based Cognitive Memory Constructs for Embodied Intelligence Machines

Navin K Ipe

***Abstract*—Any artificial intelligence algorithm that is required to "understand" the information it processes, requires the capability to store a lifetime of memories and to physically experience the real world as an embodied consciousness. This paper proposes solving the challenges of self-supervised learning by creating concepts like the universal event model, questioning constructs, a partial-context-based event storage datastructure and an architecture that allows an embodied algorithm to represent the perceived world, access contextually relevant memories, derive conclusions about the cause and effects of phenomena, label sets of events to generalize them and create objectives to pursue. The proposed constructs and implementation are currently in a rudimentary state and are severely limited by current technology, but even a simplistic implementation of context-based causal-decision-making was proven to be more effective than neural netwoks, when implemented with an embodied robot in a 2D physics environment. The remaining concepts and modes of use of the constructs are presented theoretically, along with a proposal of the fundamental building blocks identified for creating an intelligent machine.**

***Index Terms*—self-supervised learning, embodied consciousness, artificial intelligence.**

I. Introduction

THIS paper succeeds the work that began investigating the basis of intelligence [1] and the conclusions made about Artificial Intelligence (AI) requiring an event-based memory that stores a lifetime of experiences which provide contextual and causal information that provide the necessary intelligence even for trivial decision-making tasks. Artificial Neural Networks (ANN), Computational Intelligence (CI) algorithms and other contemporary machine learning (ML) algorithms utilize probabilities, weights and symbolic representations to store memories, which are insufficient to account for the vast complexities of the real world [2] [3] [4] [5] [6]. The methodology presented in this paper takes a stand on not utilizing conventional approaches, and presents a rudimentary attempt at designing a datastructure that stores sensory inputs from a legged robot, as events and partial-contexts which gives the robot the capability to imagine, conceptualize, correlate, generalize and plan activities before performing them. The datastructure is designed for quick access and abstraction of raw data. This approach creates a bridge between the software world and the real world. Training an ANN with pixels of many apples can help it identify similar clusters of pixels, but it would not know what it means to throw an apple or to eat an apple. This kind of an understanding and meaning is necessary if AI/ML algorithms are required to understand written words, because when humans read something, it triggers imagery and events in the brain that get correlated with past memories and events. This paper proposes the use of "names" (labels) for sensory events, which are abstracted into names that describe even sequences of events. When these names are used to represent events and actions in a physical world, the AI, when given a physical body, will be capable of observing similar patterns in real-world events and assign a name to it. For example, if it detects a person's body moving up and falling down, it would realize that it is the same kind of motion it is capable of doing, and it knows that in its memory the motion is named "4fo34n338d3in". When a human informs the AI that the motion is called "jump", the AI actually "understands" what a jump is, and will then correlate "jump" with "4fo34n338d3in" and also classify similar motion patterns under the name of "jump". This capability not only helps humans talk to robots, but also helps the robot understand the physical world with respect to the software world, by relating words with classes of actions (this is how true Natural Language Processing should be done). Right from the Abacus to Unix, computing systems were designed to utilize resources efficiently and ease the work of humans, rather than be intelligent. A biological consciousness constantly compares objects, actions and plans to its own perception of the universe. This perceived universe and space is called an Umwelt [7] that consists of multiple dimensions. Classification algorithms, regression algorithms, ANN's, etcetera do not understand the task they are programmed to perform because their umwelt does not incorporate facets of the human umvelt. For a machine to be able to think ethically or morally, it is necessary to give it the capability to understand consequences of actions. To experience love or emotion, it is again the consequences that play a role. It is necessary for science to move on from conventional thinking and explore new angles of thought. For example, a new angle of thought that 0,1,2,3 are not numbers, but are counting actions. Prior work on embodied consciousness has either often veered off the right path or has focused on achieving results that fit generally accepted scientific representations. This paper proposes solutions with the hope that research on embodied consciousness would be pursued henceforth with the realization that the very fundamentals of technology that make up our computing systems, need to be altered to cater to a multi-dimensional analog world.

The remainder of this paper is organized as follows: Section II presents the literature review performed to examine similar

Author acknowledges the support received from M S Ramaiah University of Applied Sciences, Bengaluru, India. N. K. Ipe is with the Department of Computer Science and Engineering, M S Ramaiah University of Applied Sciences, Bengaluru, India (e-mail: navinipe@gmail.com).

Manuscript received July 9, 2020

concepts, Sect. III presents the objectives that influenced the design of the architecture, Sect. IV presents how a simplistic implementation of the proposed concept fared better than conventional algorithms, Sect. V presents theoretical aspects of how the the proposed constructs can be used, and the paper concludes with Sect. VI.

II. Related work

The literature review for this paper encompassed work on embodied consciousness, legged robots, pattern recognition and storage of sensor signals.

A paper titled "The poverty of embodied cognition" [8] aptly summarized the lack of scientific contribution to the field of embodied cognition, but is biased due to conventional perception of computing. Survey papers [9] [10] have reviewed work on cognitive architectures built during the past forty years. The Stanford Encyclopedia presents a curated collection of various facets of research on embodied cognition [11] which theorize cognition as a capability that is housed in the agent's body but also extends to the world they live in and build, and also into the social communities they create. Wilson's work [12] outlines four steps for any research on embodied cognition. Albus [13] proposes a real-time control system architecture to represent intelligence and model the world. A work that compares the representative and associative theories of learning [14], analyzes various angles of thought from the perspective of neuron signals, spatial perception, time perception and critically assesses if a modern computer architecture could capture biological information. Allen's work [15] performs a thorough and critical examination of prior literature and concludes that characteristics required by a consciousness are ergodicity, Markov blankets, active inference and autopoiesis. Pathak [16] attempted determining causal relationships.

Machine learning for legged robots include ragdoll gaits using genetic algorithms [17], ANN's for hexapod robots [18] which arrange limb positions to define a gait and avoid obstacles, action-recognition using embodiment [19] and DeepMind's walker which uses reinforcement learning and vast memory storage [20].

To detect patterns in sensory signals, wavelets were initially considered, but the slow windowed wavelet matching approach and the need for pre-defined wavelet patterns did not fit the requirement of generality for detecting a varied set of patterns and matching them to causal events. Dynamic time warping [21] was considered because signals from sensors would vary in speed, but due to time complexity and the fact that it could be simplified into a distance measure comparison, the approach was discarded. Cross-correlation was discarded due to implementation complexity. The QRS wave technique [22], where waves and their generalized thresholds are compared, was a more promising technique when combined with wavelets, but wavelets were not favorable. Distinguishing whether a body part moved on its own or was moved by an external agent, requires distinguishing between signals from the surroundings versus signals generated by the consciousness. Such studies were conducted on electric eels, where the signals emitted by the creature got subtracted by a negative image of the same signal. Such circuits help the eels generalize what is learnt and apply it to other situations [23].

Since human memory works similar to how an Internet search engine retrieves data based on the context of the words, the concept of indexing and the suffix tree were explored [24]. The Knowledge Graph concept [25] was also considered. Various types of learning were considered: supervised, unsupervised, semi-supervised, weakly-supervised, transfer learning, reinforcement learning, active learning, one-shot learning etc. The Differential Neural Computer model (it augments a neural network with memory) [26], structural equation models [27], correlations with p-value testing, Granger causality tests and an abductive reasoning approach (logical inferencing that starts with one or more observations and searches for the simplest explanation for it) were also considered.

Overall, it was observed that although some research tended to proceed in the right direction, there is a clear need for an event-based datastructure that stores a lifetime of memories and the need for faster and more generalized pattern matching techniques.

III. Design Decisions

For any artificial intelligence to be self-reliant, it needs to use self-supervised learning, which is a method of using existing signals to train the algorithm with almost zero intervention from humans. This is accomplished by creating and using self-generated labels which help understand relationships between signals and how the actions of the embodied consciousness or external factors affect those signals. Automated machine learning to date, has focussed on automation of data cleaning, engineering features, selection of models, tuning hyperparameters, selecting a pipeline, evaluation metrics, analyzing and visualizing results and checking for problems, while largely ignoring the need for creating primitive constructs that enable automated integration of one intelligent entity with another, to accomplish objectives. The true scope of this investigation involves building physical robots with at least many hundred terabytes of data storage, where the robots not only learn causal relationships from each of the sensors they have, but they also communicate with each other to form cooperative entities that form one body that pursues objectives. It is very evident in any animal brain that even for seemingly simple tasks, there is a huge amount of communication that happens between various cells. This is in stark contrast to the overtly simple mechanisms created in conventional AI/ML algorithms.

A. Embodiment and environment

Storing a lifetime of memory and using abductive and deductive reasoning could potentially build an Explainable AI that understands information and is capable of continued processing of information, even if it does not immediately comprehend something, much like how humans do. The challenges are multi-fold: the design of generic behavioural patterns, classification techniques and pattern matching techniques which understand data patterns under various contexts,

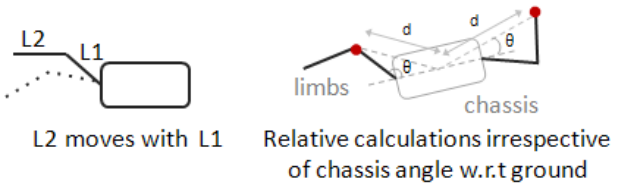
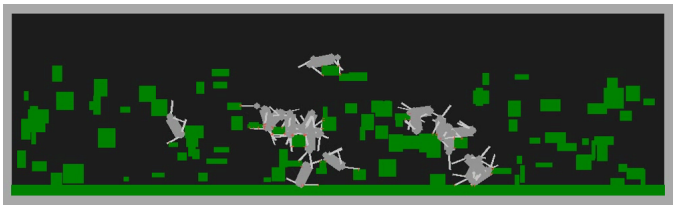
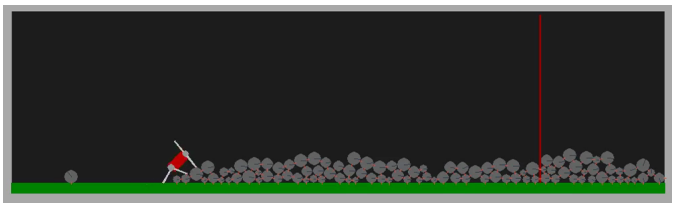


Fig. 1: Simulated robot



(a) Min-max world



(b) Spheres world

Fig. 2: 2D environments

overcoming hardware latencies when dealing with data storage and designing "intelligence" and "understanding" without using probability values or neural weights. To attempt this, a simulated legged robot weighing $10kg$ was initialized in a 2D physics environment named PyMunk, as shown in Figs. 1 and 2. The robot limbs are attached with motors capable of rotary motion at any angle. The robot possesses a movement sensor that registers 2D movements in directions dx and dy, a body angle sensor, body angle stability sensor, motor rate sensor, energy sensor, limb tip position sensor, a tactile sensor and an impact sensor. The robot is run in environments with acceleration due to gravity at $9\,\mathrm{m/s}$. Multiple robots were first run in a min-max world (Fig. 2a) with random limb motions, to obtain all possible minimum and maximum possible sensor values, to determine the range of values possible for each sensor. The range was used for normalization of values.

B. Designing the storage of events

Various factors needed to be considered to design datastructures that could store a lifetime of memories and yet efficiently access it. Every experience of the embodied consciousness is considered an event, and rather than use a graph model to store memories, a soft-connections model was designed keep senses separate and uniquely identifiable and yet be integrated to form a new memory (Fig. 3). This model helps immensely when building a hierarchical division-of-labor AI or a growth-model AI (where an intelligent body is formed from multiple cooperative intelligent entities).

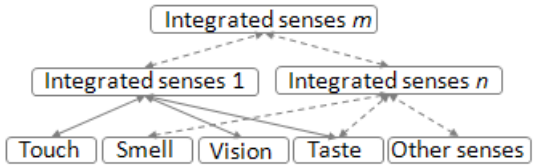


Fig. 3: Soft-connections model

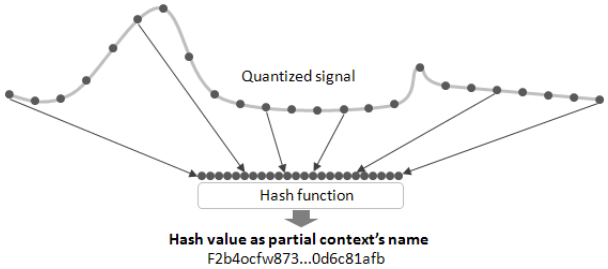


Fig. 4: Quantization and hashing process

Cognitive architectures conventionally possess functionality specific to perception, attention, action selection, motivation, actuation, memory, learning, reasoning, meta-reasoning, social interaction, planning, emotion and creativity. However, building a generalized intelligence, requires first building a fundamental capability to:

1) Take into account context and ask questions about phenomena.
2) Play with available matter or phenomena to deduce causal relations (researching behavior).
3) Utilize matter or phenomena to enhance existing capabilities (building new sensors or a stronger body) and to continue pursuing core objectives.

The Cognitive Memory Construct (CMC) was conceptualized with these needs in mind: quick access, partial-context matching, causal relationship detection and lifetime storage. Every sensor of the robot provides a constant stream of values every second. For simplicity, events are considered as time slices of one second length, discretized into 24 slices of length dT. The signals from each sensor are a partial context of the entire context of events. Quantized signals are hashed, and the hash serves as the name of the partial context (Figs. 4 and 5).

Figure 5 depicts how each sensor signal's values are considered a partial context combined in order to form a complete context that forms an event. Each consecutive event is stored as

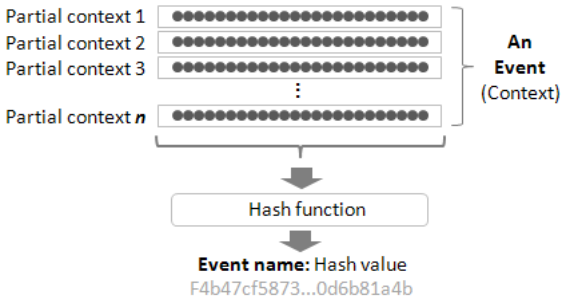
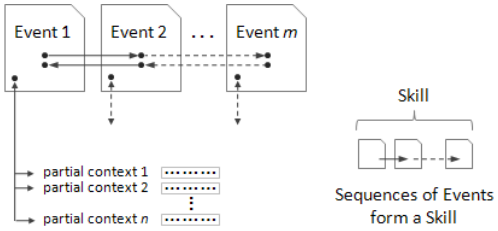


Fig. 5: Full context of an event



(a) Doubly linked events (b) Skill

Fig. 6: Event storage format

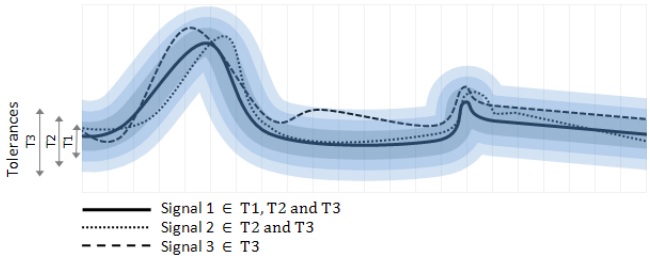


Fig. 7: Signal classification by tolerances

a doubly linked list as shown in Fig. 6. The event datastructure holds the event name, pointers to the next and previous event and also doubly linked pointers to each partial context (sensor's signal values). This is a rudimentary storage method that needs to be improved.

C. Classifying the signals

Each signal CMC receives, is first classified into start context categories based on the first value of an event signal, as depicted in Fig. 9a. This initial classification reduces the iteration complexity when searching for patterns. The start context is followed by the signal being classified based on the pattern of all values of the signal that fit into tolerance bands. For example, tolerances for the events layer are: $T_1 = 6\%$, $T_2 = 10\%$, $T_3 = 20\%$ as roughly represented in Fig. 7. While any number of tolerance levels can be chosen (T_1, T_2...T_A), this paper considers $A = 3$. If a sensor can generate values ranging from a minimum of -1 to a maximum of 1, the tolerance T_1 would be ± 0.06. When comparing signals, each of the 24 values of an incoming signal v_1 is compared with a stored signal value v_2. If the $M.S.E = \frac{\left(\sum_{i=1}^{24}(v_{2i}-v_{1i})^2\right)}{24}$ does not exceed the tolerance band of 6%, v_1 is stored in the same class as v_2 (Fig. 8). Tolerance bands can be adjusted based on the accuracy desired, but once decided, the values should not be changed.

As evident in Fig. 7, signals 1, 2 and 3 have almost the same kind of pattern. However, if their quantized values were hashed, it would be impossible to identify similarity of the signals based on hash values. To overcome this, a decision was taken to categorize the hashes into a hierarchy that matched signals based on desired accuracy. Figure 8 depicts the event hash storage hierarchy, where $Hash_1$ at level T_1 is the first signal of a classification hierarchy. This first signal's hash gets stored across levels T_1 to T_A. The next signal that falls within the tolerance limit T_3 of $Hash_1$, gets stored at T_3

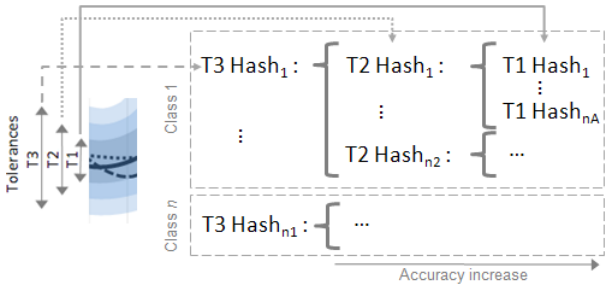
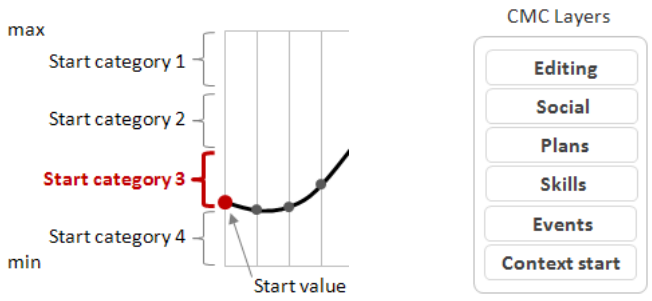


Fig. 8: Classification of tolerance hashes

. If it falls within T_2 and T_1 it also gets stored in those categories. Storage operations have a worst-case complexity of $\sum_1^A O\left(log\left(n_A\right)\right)$, where n_A is the number of hashes in each category A. Similar signals are thus classified into classes of signals, where the entire signal is compared with a stored signal using M.S.E to check for similarity. An advantage of this storage technique is that if an exact match of any hash is found, it is not necessary to iterate the hierarchy. Instead it can safely be assumed that if T_2 was found, then it definitely falls within the bounds of all hashes in T_3. Exact matches have a search complexity of $O(1)$, while searches of the highest accuracy have a worst-case complexity of $O(A \times n_A)$, when stored in a hashmap.

Indexing techniques and the suffix tree were considered for storage, but hashes provided faster lookups and also had the advantage of providing a unique name to any observed phenomenon. Since the size of inputs to SHA-256 can be expressed as a 64 bit number, the maximum input size of a string that can be hashed is $\frac{\left(2^{64}-1\right)}{8}$ bytes $\approx$2091752 TB. Pattern detection is not only about onset detection or detecting parts of a pattern in a larger pattern (windowed techniques and convolution). Pattern matching in an intelligent entity involves matching patterns irrespective of the time it gets stored (the way people remember last year's new year celebration as if it were yesterday), matching patterns across sensors (synesthesia), partial patterns triggering memories of complete patterns, predicting/estimating patterns before and after a pattern (requires moving back and forth in a memory) and most importantly, being able to combine partial patterns from various facets of memory to create new patterns (imagination, dreams, creativity). Such functionality necessitates the storage of memory "as-is", and cannot be generalized into neural weights in the way ANN's do.

CMC is designed with a capability hierarchy consisting of multiple layers (Fig. 9b), where each layer has a higher level of abstraction (using integrated senses) than the layer below it. The context start layer (proposed and implemented) stores the start values (Fig. 9a) of partial context's in categories that help quickly locate relevant events that these start values represent. The event layer (proposed and implemented) stores the signal in its most raw form as partial contexts, and performs pattern matching for any event signal. The skill layer (proposed and implemented) selects events in a certain order so that it can later identify that specific sequence of events with a single name, and a skill could be performed without having to re-



(a) Start context (b) Capabilities

Fig. 9: Start categorization and capability hierarchy

access vast stores of event memories (for example, events that need to be executed to climb a specific kind of obstacle is a skill). The plans layer (proposed, but not implemented) is a sequence of skills and actions that are planned for effective robot navigation. For example, when the robot has knowledge of what the terrain ahead is like, and possesses a set of skills to be able to climb each type of obstacle in the terrain, the planning layer stores the sequence of skills that can be used to navigate such a terrain. The social layer (proposed, but not implemented) allows creation, editing and storage of planned sequences which can be used to cooperate with or combat other robots. The editing layer (proposed, but not implemented) enables the robot to create or destroy physical or virtual entities. The capability hierarchy is not a complete representation of the architecture required to build an intelligence. Details like object recognition, reproduction, self-recognition, meta-cognition, self-preservation and many more features need to be incorporated, to create an embodied cognition.

D. Comparing ANN with CMC

A simple algorithm was created to compare CMC with ANN. Thirty robots were assigned random motor rates and run for multiple generations using Particle Swarm Optimization to determine and improve the motor rate combinations that yielded motion in the $+dx$ direction. A Pareto front was used to select the best robot based on multiple objectives like minimal impact to the chassis and highest dx movement. The goal of the robot was to reach a red line at the extreme right of the environment. A Multi Layer Perceptron (MLP) was trained with the normalized sensor values as input and the best corresponding motor rates (which move the robot rightward) as output. Various trials were conducted with variations in the number of hidden nodes, activation functions, loss functions etc., but the ANN showed no significant improvement in learning the combination of motor rates under various body positions, that would lead to a $+dx$ movement instead of a stationary or $-dx$ movement. Figure 10 shows a screen-shot of the thirty robots at the dnd of a PSO epoch, and the two-dimensional objective fulfillment of each robot depicted as a circle on a graph. The circles in blue depict the optimal Pareto front (best robots), among which green is the selected optimal robot.

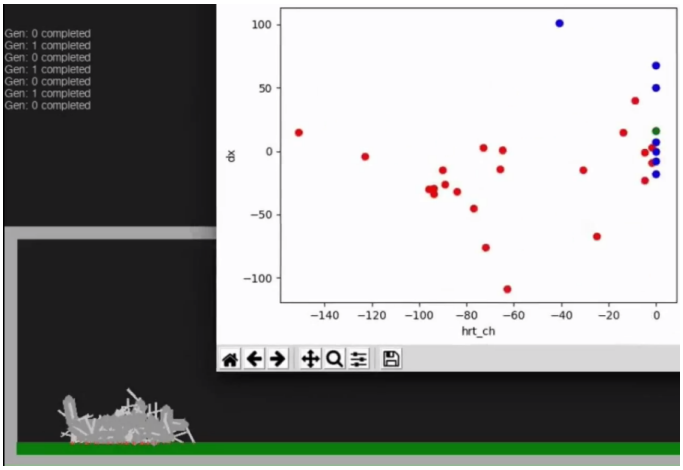


Fig. 10: Pareto front from ANN robot's PSO trials

Algorithm 1 CMC robot algorithm

1: Initialize simulation environment and robot.
2: If min max values not present, generate it.
3: Check objective to achieve and check if methods of achieving objective are stored in memory. If not, activate baby learning mode for $300s$ and determine events that fulfil focus objective.
4: If objective not fulfilled, get sensory inputs.
5: If at least two legs not touching terrain, activate feeler mode with low torque motor until legs touch terrain.
6: Check memory for partial contexts similar to current context. Determine best motor rates based on Pareto front, consequence of event or the relationship data.
7: Execute motor motion for $1s$.
8: Goto Step 4.

Partial contexts were implemented to identify causal relationships between signals. Causal relationships are not always linear or predictable and should typically be examined over a large span of time, but for simplicity, this paper examines relationships with a tolerance of $\pm dT$. As mentioned in Algorithm 1, the robot is first subjected to a baby-learning mode, where causal relationships are derived from sensor values. As an example, Fig. 11 shows how simultaneous variation in signals helps the robot learn that its chassis angle changes only when a limb moves.

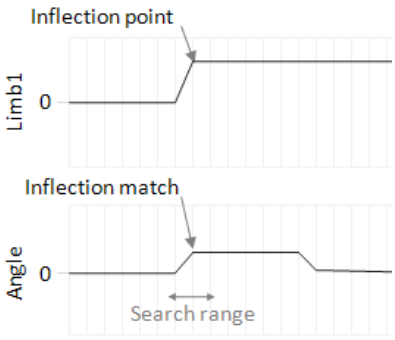
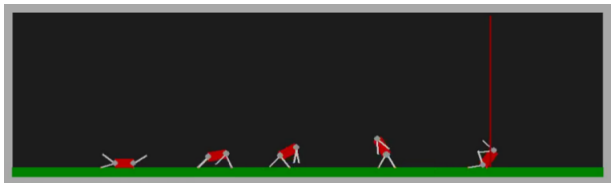
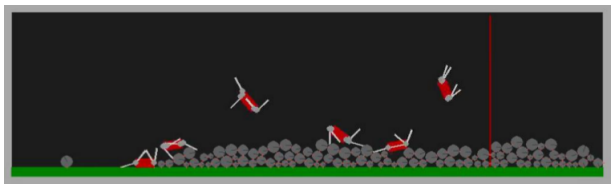


Fig. 11: Inflection points



(a) Flat ground



(b) Spheres terrain

Fig. 12: CMC robot time-lapse

TABLE I: Robot movement with ANN

Epoch	Training data	Layers	Hidden nodes	Training accu.	Fd.,Bk. movt.	Finish time (s)
2000	35	4	120	0.4	149,135	282.15
2000	58	4	120	0.5	41,28	68.43
2000	76	4	120	0.57	109,62	171.23
2000	96	4	120	0.32	113,91	204.38
2000	118	4	120	0.65	54,35	88.13
100	1470	4	120	0.42	46,32	77.92
1000	1526	6	240	0.57	298,415	704.11
1000	1616	6	240	0.58	228,237	460.78
1000	1702	7	480	0.52	45,29	475.85
1000	28	7	480	0	82,52	133.62
1000	57	7	480	0.62	91,77	166.43
1000	82	7	480	0.5	310,327	629.51

Although CMC performed better than ANN on flat terrain, the motor torque caused an undesirable jittery/jumping motion on more complex terrains, as shown in Fig. 12b. Figure 12 depicts a time-lapse showing robot motion from left to right, until it reaches the red finish line. Results are listed in Sect. IV.

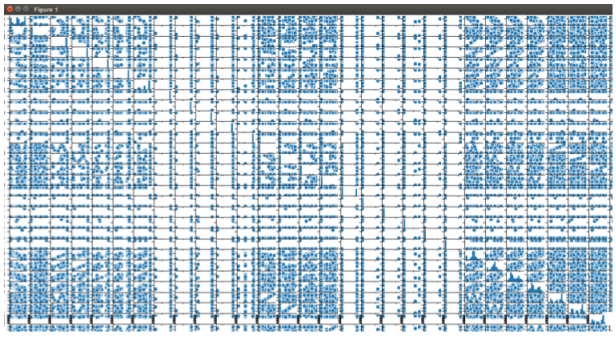
IV. Results

The primary aim of this paper was not the comparison of ANN and CMC. The aim was to design a datastructure that could store a lifetime of events as partial contexts, generalize them and access the data fast, to perform causal inferences.

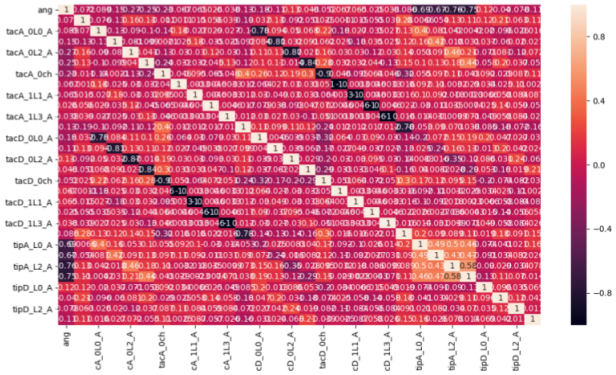
A. ANN results

Table I shows how the number of training instances, a deeper or fatter network or even the number of training epochs had no predictable effect on the accuracies or the time taken for the ANN robot to reach the finish line. The "Fd.,Bk. movt." column depicts "number of forward movements, number of backward movements", and it's clearly visible that the ANN trained robot suffers from a huge number of backward movements which is undesirable.

Investigation into the reason for ANN's lack of ability to learn, was conducted by plotting the pair plot and heat-map (Fig. 13) for all sensor values. This revealed that the issue



(a) Pair plot of sensors



(b) Heat map of sensors

Fig. 13: ANN analysis

was with a lack of a clear decision boundary. There were very few positive correlations between variables, due to a large number of overlaps in limb positions at various chassis angles and corresponding motor values, which prevented the ANN from distinguishing between forward and backward motion. Additionally, the total number of permutations of sensor states were in the order of 10^{28}. An extremely large neural network would be required to capture such high-dimensional information.

B. CMC results

Even with a very limited use of CMC's database and capabilities, a more reliable pattern of finishing times and forward motions were demonstrated for flat terrain, due to the algorithm being able to utilize partial contexts and relationships to determine if a certain motor rate combination would move the robot forward. As the robot was still in the process of building the CMC memories, some backward motions were inevitable as shown in Table II.

Advantages of CMC

1) CMC allows searching for any one partial context and finding which event it belongs to. This is a powerful method of pattern matching because, when two or more patterns are found to match the partial contexts of multiple events, finding the appropriate event to choose, is simply a matter of finding the intersection of the event names. Example: When searching for an event where the robot's dx changes in the positive direction with

TABLE II: Robot movement with CMC

Fd.,Bk. movt.	Time to finish (s)
58,14	84.42
59,17	90.79
62,18	91.27
69,24	106
90,22	131.16
68,27	111.25
70,24	111.13
51,9	73.19
45,14	72.23
51,11	77.84

a particular pattern of motion and where the chassis collision impact is zero, there will be many partial context's that match dx events E_1, E_{1003}, E_{3424} and E_{32}. The chassis collision impact being zero, may match events E_1, E_{452}, E_{57} and many more events. To find the relevant event, it's simply a matter of performing an intersection operation on the event hashes as: $\{E_1, E_{1003}, E_{3424}, E_{32}\} \cap \{E_1, E_{452}, E_{57}, ...\} = \{E_1\}$. The cost of the operation is $O(n)$, where n is the length of the smallest set.

2) Easily add, edit and prune data.
3) The possibility of event sequences that are completely disconnected from other event sets (for example, there will be no memory continuity if the robot is switched off and switched on), but the discontinuity does not prevent partial context matching.
4) Creates an explainable AI. It is possible to drill down and find out the reason for every action, unlike the obfuscation of data in ANN's. Memories can be referred based on context, thus giving the robot the experience needed for intelligence, understanding and generalization.

Disadvantages of CMC

1) The amount of storage space required.
2) Caching and hardware latencies.
3) The complexity of the datastructure.

V. Event model and questioning constructs

As mentioned in Sect. IV, the objective of this paper is not to compare CMC with ANN, but to conceptualize a new datastructure and methodology for achieving intelligence via algorithms. A key missing element in AI/ML algorithms is the algorithm's ability to ask questions about phenomena to learn more. This led to the creation of the following:

A. Proposed Universal Event Model

The Event Model paradigm assumes everything in the universe which the embodied consciousness encounters, is composed of events E_U, detected at time t, and each E_U is composed of multiple partial contexts C_p. These events may be represented in various forms like "Purpose", "Question", "Skill", "Social" etcetera. Even solid objects and the embodied consciousness itself are perceived as events, because their existence is defined by signals from sensors. $E_U = \sum_{i=0}^{N} E_i$, where E_i is one among N events.

- E_{CMC}'s are events stored in CMC (these include the attention and purpose constructs introduced in Sects. V-B and V-C).
- E_{soft}'s are combinations of basic event types that form a generalized higher-level event (soft representations). Once created, E_{soft}'s are identified and used as an E_{CMC}.

B. Proposed Attention Construct

- **Attention (A_{span}):** Simulates an attention span with a duration that can vary according to the attention span desired at any point of time. A_{span} can be considered as a primary consciousness that constantly checks CMC for purpose of existence, body boundaries, relationship between signals etc. The length of any stored event can be made dependent on the length of A_{span}.

C. Proposed Purpose Construct

Purpose constructs are composed of one or more events, composed of C_p's and events that visualize an abstract pattern of objectives to be achieved.

- **Basic purpose (P_{basic}):** The purpose of existence of the embodied consciousnesswhich is defined by the creator, and cannot be altered by the embodied consciousness unless it learns how to modify its own programming. These include the core facets of an AI like self-preservation, community preservation, ensuring self-nutrition, replication, gaining knowledge, growing larger etc. The embodied consciousness can choose to focus on one or more of these purposes at any given time and ignore any of these purposes too, based on the context of the sensory inputs it encounters.
- **Temporary purpose (P_{temp}):** These purposes/objectives are defined and created by the embodied consciousness, based on the sensory inputs it encounters or based on the final objective it seeks to achieve in any situation.

D. Proposed Questioning Constructs

The questioning constructs enable the recognition of self, implementation of curiosity and exploration capability. The interactions of various constructs are concisely described via special operators listed in Table III. As an example, $Q_{what} \lozenge P_{temp}$ is read as "What temporary purpose needs to be fulfilled"? The $\lozenge$ operations are searches performed to estimate long-term or short-term consequences between multiple events E_i and whether these consequences fulfill the abstract definition of a purpose P. For example, a purpose can be "*I want to eat healthy food everyday*", and there is no single way to define the achievement of this purpose, but it can be done via multiple comparisons from experiences of healthy foods eaten in the past and the consequence of eating them under various contexts. Such comparisons are what constitute common-sense. The $\approx$ operator on the other hand, is a simple calculation like estimating an M.S.E. Conventional ML/AI algorithms have faltered when defining objective fulfilment

TABLE III: CMC operators

Operators	Meaning
$\Diamond$	Achieves or fulfills
$\approx$	Approximately similar
$\neq$	Dissimilar
==	Exact match
$\rightarrow$	Consequence or cause
$\sum$	Ordered composition of

via hard rules or fuzzy logic which fail to work under various conditions.

- **What (Q_{what}):** Used for searching within CMC and giving the embodied consciousness "free will", by choosing a P_{temp} or P_{basic} to follow. Some examples: $Q_{what} == E_i$ is an identification task that searches for an exact pattern match in CMC. $Q_{what} \approx E_i$ is an identification task that searches for approximate pattern matches in CMC. $Q_{what} \neq E_i$ is an elimination task that ignores dissimilar patterns in CMC. $Q_{what} \Diamond P_{temp}$ or $Q_{what} \Diamond P_{basic}$ searches for purposes to fulfill.
- **Which (Q_{which}):** Used for partial context searching, causal reasoning and estimating consequences. These are soft-representations of other events. Some examples: $Q_{which}\, E_{CMC} \neq E_i$ or $Q_{which}\, E_{CMC} \neq E$ performs partial context matches or searches for mismatches between stored senses and experienced senses. $Q_{which}\, E_{CMC} \rightarrow E_{CMC}$ imagines consequences of executing an event (execution of which E_{CMC} could result in a particular E_{CMC}?). Also used for asking "Why" or "How" an event led to another event. It can internally be realized via $Q_{what} \approx E_i$ and $Q_{what} \neq E_i$, but the "Which" constructs were created distinctly to emphasize the importance of continuous and extensive partial context matching and consequence searches even for small decisions that the embodied consciousness needs to take based on context, rather than probability values or weights.

E. Proposed Intelligence Machine

Intelligence Machine (I_m): Figure 14 conceptualizes the essential components/building-blocks that constitute any intelligence. Multiple such units possessing a similar equilibrium of memories can cooperate to form larger I_m entities, as shown in Fig. 15. The splits on the borders of A_{span} in Fig. 14 indicate that the event can have a variable length.

Effectors and receptors can be one or more components that belong to the "body" of I_m, and can be located all around or even inside I_m. Effectors are used for performing actions (which are also generated as events, by the I_m), for establishing connections with other I_m's and for sending response signals to any external or internal agent. The structural support of the "body" can be composed of a combination of effectors, receptors and other suitable materials sourced from the environment. All such information about energy, materials, structure and composition of the body etc. are stored as partial context's in memory. The "survival instinct" of I_m is to store, preserve and replicate these memories.

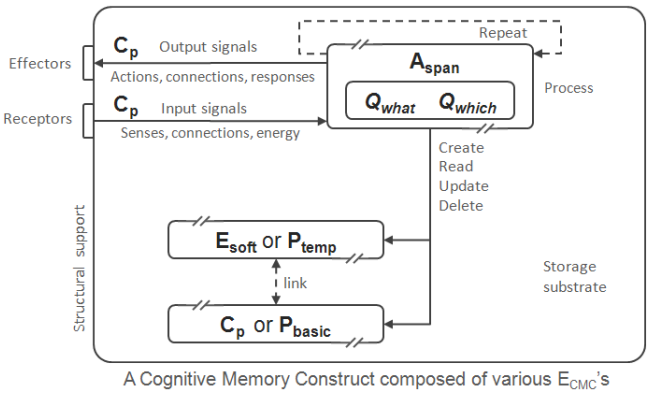


Fig. 14: Components of an intelligence machine

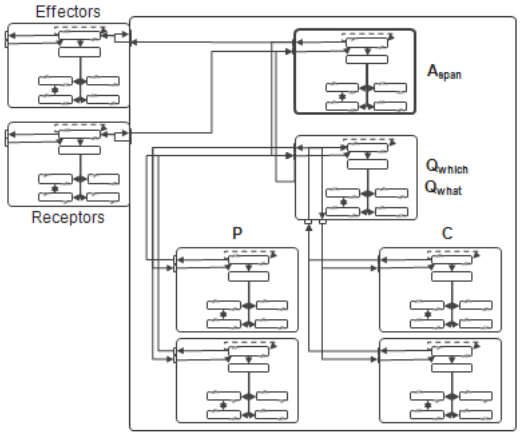


Fig. 15: Cooperative machines forming larger body

F. Proposed decision making via equilibrium and causality

Every decision taken by an intelligent entity is a result of a need to maintain equilibrium with an objective or the environment. When hungry, the chemicals in an animal are in a state of non-equilibrium, and to maintain equilibrium it creates a temporary purpose $P_{temp}[find\, food]$ in memory. Social skills and plans are executed to find and eat food to restore equilibrium. An I_m is thus in a constant state of ensuring equilibrium to fulfill various purposes. Intelligence does not have to be an efficient process, but it has to be effective in fulfilling one or more purposes.

Equilibrium (ϕ): Given an event E = $\{C_{p1}, C_{p2}, \ldots, C_{pn}\}$, a state of equilibrium is achieved if in a system of E_{CMC}'s = $\{E_1, E_2, \ldots, E_i\} \in I_m$, the input $E \Diamond P_{temp}$ (read as "the event fulfills a temporary purpose") and/or $E \Diamond P_{basic}$ such that by the action, no other P_{temp} or P_{basic} shall be put into a permanent state that prevents it from achieving a state of fulfillment (the permanence is verified by evaluating E_{CMC}'s that have led to $E \Diamond P$, so an inexperienced I_m will be incapable of performing the evaluation until it acquires at least a few C_p's which enable it to perform context and consequence constructions (imagination) using partial contexts relevant to the specific context). Every context can have its own ϕ with respect to related contexts, and links between events can be disassociated if the link would cause non-equilibrium in a new context (the same reason the human brain

forgets information but remembers it when necessary).

Example: Algorithm execution pipeline: As an example of how the concept of equilibrium is realized, and how it differs from conventional programming, the following steps describe how a dormant I_m (similar to a plant seed or an organic virus) begins functioning and maintaining equilibrium. The steps depict how self-supervised learning is achieved as a consequence of reinforcement learning, where I_m attempts to maintain a state of equilibrium.

Step 1: I_m is created with pre-stored information about its body as $E_{CMC} = \sum C_p$ and the objective of acquiring more knowledge with each time-step t is specified as $E_{CMC}(t) > E_{CMC}(t-1) \rightarrow P_{basic}[knowledge]$. The consciousness A_{span} remains inactive. I_m is dormant, in a state of non-equilibrium.

Step 2: Receptors inadvertently make contact with an energy source, receive energy $C_p[energy]$ and A_{span} activates, performs checks for what purposes need to be fulfilled: $Q_{what} \lozenge P_{basic}$ and $Q_{what} \lozenge P_{temp}$. One purpose is knowledge acquisition. Another purpose is energy acquisition. Since energy was acquired, A_{span} notes which $C_p[energy] \lozenge P_{basic}[energy]$. (Note: The signal pattern of $C_p[energy]$ has to be hard-coded into CMC). The remaining C_p's provide contextual knowledge of the situation that made energy available and A_{span} stores these C_p's too and evaluates them with respect to similar energy acquisition events in memory. Thus $E_{CMC}(t) > E_{CMC}(t-1)$, which fulfills knowledge acquisition purpose: $C_p[energy\,context] \lozenge P_{basic}[knowledge]$, and equilibrium is maintained.

Step 3: A_{span} now has a relationship pipeline stored in CMC, that $\lozenge P_{basic}[energy] \rightarrow \lozenge C_p[energy\,context] \rightarrow \lozenge P_{basic}[knowledge]$ (*read as "fulfillment of a basic purpose for energy acquisition is a consequence of fulfillment of being in a certain context where energy is available and that causes fulfillment of the basic purpose of knowledge acquisition*"), and it needs to maintain ϕ of $P_{basic}[energy]$ and $P_{basic}[knowledge]$. To continue obtaining knowledge, A_{span} has to continue acquiring energy, so it analyzes the relationship pipeline from R.H.S to L.H.S, searching for a C_p to fulfill: $Q_{which}\, C_p \lozenge P_{basic}[energy]$ (which situations provide energy) and $Q_{which}\, P_{basic} \lozenge C_p[energy\,context]$ (which purposes can be fulfilled by a particular situation). A_{span} locates the linked C_p's and is primed at a state of equilibrium that is receptive to any input C_p which matches the pattern of one or more $C_p[energy\,context]$, since such situations/contexts provide access to more $C_p[energy]$, and will enable I_m to continue fulfilling purposes and maintaining equilibrium.

A P_{basic} can also be defined for controlling the effectors, which in turn would create causal and contextual relationship pipeline events for how a signal to an effector affects the state of equilibrium for any P_{basic} or P_{temp}. The creation of P_{temp}'s is performed via an ordered composition of C_p's that lead to a desired state of equilibrium: $\sum C_p \rightarrow \phi$. For example, since a $C_p[energy\,context]$ leads to acquisition of $C_p[energy]$, the I_m can, after noting similar patterns in CMC over a period of time, create a $P_{temp} = \sum C_p[activate\,effector] \rightarrow \lozenge C_p[energy\,context] \rightarrow \lozenge P_{basic}[energy]$, which is a set of multiple pipelines of various types of linked E_{soft} effector actions under various C_p's that lead to energy acquisition. Similar to how humans create multiple ways of moving limbs to reach a canteen or store to purchase food and then create multiple ways of cooking and eating food. It is important to note that the selection of events are performed by examining consequences under specific contexts.

G. How and why algorithms need to understand information

Number prediction with ANN: Consider a simple ANN trained to predict a number in the set S = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}. After normalization to values S_N = {0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}. Values are fed to the ANN as input-output pairs $\{input_1, input_2 : output\}$ = {[0, 0.1: 0.2], [0.1, 0.2: 0.3], [0.2, 0.3: 0.4], [0.3, 0.4: 0.5], [0.4, 0.5: 0.6], [0.5, 0.6: 0.7], [0.6, 0.7: 0.8], [0.7, 0.8: 0.9], [0.8, 0.9: 1]}, to enable the ANN to predict a number, given two past numbers. However, if the sequence of numbers is changed or if a number is missing from the set, the predictions of the MLP will be incorrect. This happens because the ANN has not "understood" what a number is or what a sequence of numbers is. The ANN does not even know what objective the human wants to accomplish. Be it a small problem such as this or a complex problem like humanoid robots, the core issue that prevents fault-tolerance and a lack of understanding or intelligence in conventional AI/ML algorithms is the simple fact that the "intelligence" is still in the human brain, and has not been transferred to the algorithm.

Number prediction with CMC: The creation of the CMC algorithm as an embodied consciousness begins as presented in Sect. V-F, where an equilibrium is built for every knowledge acquisition, associated contexts and a pipeline of events and consequences are created with a contextual equilibrium established with the corresponding purpose of accomplishment. Once lower-level contexts are created and abstracted to higher-level contexts, the I_m deduces higher-level features to recognize objects. Once these basic capabilities are achieved, the CMC is in a position to "understand" very high-level tasks like number sequence recognition. Numbers are events that are counting actions. Humans learn the importance of counting on their fingers and by counting objects.

Number recognition: CMC is first trained on correlating the number of objects it perceives $E_{soft}[n\,objects]$, with the numbers it is visually shown $E_{soft}[font]$ and the numbers it is fed as binary digits $C_p[n]$. It is important to note that the training has to be performed as a result of maintaining the equilibrium (ϕ) of acquiring knowledge $P_{basic}[knowledge]$ or via a P_{temp} that incentivizes better consequences of learning. Recognition of negative numbers are a more complicated process, involving training CMC on the concept of "lending" and "borrowing", since negative numbers represent the concept of "debt".

Number sequence recognition: Next, CMC is trained on recognizing the sequence of numbers as an event pipeline $E_{soft}[1] \rightarrow E_{soft}[2] \rightarrow E_{soft}[3] \rightarrow E_{soft}[4] \rightarrow E_{soft}[5] \ldots E_{soft}[10]$, where each $E_{soft}[number] \rightarrow E_{soft}[n\,objects]$ (meaning, each number memory is associated

with the memories of counting actions of those many objects). The event pipeline assists with remembering the order of numbers.

Fault-tolerance: If a trained CMC is presented with an incorrect sequence of numbers $S = \{0, 1, 2, 4, 3\}$, it develops a need to maintain equilibrium of new C_p's (the incorrect sequence) with existing $E_{soft}[numbers]$ (the correct sequence) via $Q_{which}\, C_p[numbers] \approx E_{soft}[number]$ and traversing the pipeline searching for the next expected event $Q_{which}\, E_{soft}[1] \rightarrow E_{soft}[2]$, $Q_{which}\, E_{soft}[2] \rightarrow E_{soft}[3]$, ... until CMC detects a difference in the expected context and the actual context, thus sending A_{span} into a state of non-equilibrium. As this point, CMC can either accept the incorrect sequence as new knowledge or if CMC has been trained to recognize and interact with the human who trained it, CMC can now question the human via effectors, asking if the sequence is valid, via $Q_{what}\, E_{soft}[4] \rightarrow E_{soft}[3] \lozenge E_{soft}[3\, objects] \rightarrow E_{soft}[4\, objects]$.

Generalization capability: Given a new sequence of numbers as Roman numerals, S = {I, II, III, IV, V, VI, VII, VIII, IX, X}, with a C_p indicating that the numerals are counting actions, CMC finds partial context matches with the decimal number sequence pipeline $Q_{which}\, C_p[numbers] \approx E_{soft}[number]$. Now equilibrium is fulfilled ($\phi \lozenge$) with $P_{basic}[knowledge]$ and an equilibrium is also established between the decimal number pipeline and the Roman numeral pipeline, where the equilibrium relationship is specified as $E_{soft}[numeral] \rightarrow E_{soft}[n\, objects]$, since both pipelines are counting actions.

Conjecture on simulating intelligence

Given one or more embodied intelligence machines I_m = $\{I_{m1}, I_{m1}, \ldots, I_{mn}\}$ that share common purposes P_{basic} and can exist in equilibrium with each other or exist as a single entity, such Turing-complete machine(s) are capable of intelligent processing, intelligent action and capable of understanding the human world, when given the capability to store all partial-contexts as events it accumulates in its lifetime, abstracting sets of events into higher-level events, associating events using context and performing decision-making actions via questioning constructs, to establish the relevance of partial contexts and events with each other and their role in fulfilling the basic and temporary purposes of I_m such that every purpose and context in I_m is maintained in a state of equilibrium, failing which, associations between events may be severed or re-established in the interest of maintaining equilibrium.

VI. Conclusion

Hardware limitations or funding constraints should not dictate the ideation of intelligent architectures. Once the right concept is created, enterprising individuals will build the necessary hardware. This paper presented a simple concept of how a universal event model can provide a common foundation to represent any sensory experience or thought-process of an embodied consciousness as an event-based cognitive memory construct that is capable of generalizing information hierarchically, using partial contexts with algorithms that equip it to ask questions about phenomena, thus giving it an understanding which makes the intelligence machine tend toward a state of possessing intelligence. Although the datastructures and concepts presented are rudimentary and need to be improved, they present an unconventional methodology (which may require a new kind of hardware) that stresses on moving beyond current methods of logic representation, neural weights, symbolism and probabilities, and demonstrates the need for even simple intelligence machines to possess the following:

- The concept of the questioning constructs, attention span and abstract purpose definitions based on context and consequences.
- The universal event model that simplifies representational complexity and allows embodied consciousnesses to have a common foundational language.
- The basic components required for creating a machine capable of intelligence.
- The storage of a lifetime of memories as partial contexts, that enable decision-making by utilizing contextual information, rather than probabilities or weights.

Although only a small subset of the theory and functionality could be presented via this paper, it was observed that the most simple utilization of partial contexts was sufficient to demonstrate superiority over ANN, when tested with a robot that needed to reach a goal in a 2D physics environment. More complex implementations can give the algorithm even more powerful features. In order to improve intelligent algorithms, data needs to be stored in more dimensions, hardware needs to be able to utilize an analog storage substrate to represent real-world data better, and physics simulations need to be refined to allow simulated robots to touch their own body to realize their existence and boundaries without the jittery motion that happens when two bodies make contact in physics environments. The current manner in which events are discretized and split across nodes, makes pattern matching difficult, so layered methods of incremental resolution of data storage (like the layers of convolutional neural networks) may hold promise. More importantly, the core aspects of intelligence require continued investigation and radically new methods of data storage and computation need to be discovered. It is a humble wish that the concepts presented in this paper could inspire a more concerted effort at conceptualizing and building viable intelligence machines.

References

[1] N. Ipe, "Facts and anomalies to keep in perspective when designing an artificial intelligence," 2020. I

[2] A. Roy, "Artificial neural networks: a science in trouble," *ACM SIGKDD Explorations Newsletter*, vol. 1, no. 2, pp. 33–38, 2000. I

[3] F. Crick, "The recent excitement about neural networks," *Nature*, vol. 337, no. 6203, pp. 129–132, 1989. I

[4] K. Warwick, "A critique of neural networks for discrete-time linear control," *International Journal of Control*, vol. 61, no. 6, pp. 1253–1264, 1995. I

[5] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 427–436. I

[6] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. I

[7] C. H. Schiller, "Instinctive behavior; the development of a modern concept." 1957. I

[8] S. D. Goldinger, M. H. Papesh, A. S. Barnhart, W. A. Hansen, and M. C. Hout, "The poverty of embodied cognition," *Psychonomic bulletin & review*, vol. 23, no. 4, pp. 959–978, 2016. II

[9] I. Kotseruba, O. J. A. Gonzalez, and J. K. Tsotsos, "A review of 40 years of cognitive architecture research: Focus on perception, attention, learning and applications," *arXiv preprint arXiv:1610.08602*, pp. 1–74, 2016. II

[10] P. Ye, T. Wang, and F.-Y. Wang, "A survey of cognitive architectures in the past 20 years," *IEEE transactions on cybernetics*, vol. 48, no. 12, pp. 3280–3290, 2018. II

[11] R. A. Wilson and L. Foglia, "Embodied cognition," in *The Stanford Encyclopedia of Philosophy*, spring 2017 ed., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2017. II

[12] A. D. Wilson and S. Golonka, "Embodied cognition is not what you think it is," *Frontiers in psychology*, vol. 4, p. 58, 2013. II

[13] J. S. Albus, "A reference model architecture for intelligent systems design," *An introduction to intelligent and autonomous control*, pp. 27–56, 1993. II

[14] C. R. Gallistel, "The nature of learning and the functional architecture of the brain," *Psychological science around the world*, vol. 1, pp. 63–71, 2006. II

[15] M. Allen and K. J. Friston, "From cognitivism to autopoiesis: towards a computational framework for the embodied mind," *Synthese*, vol. 195, no. 6, pp. 2459–2482, 2018. II

[16] D. Pathak, *Learning to Generalize via Self-Supervised Prediction*. eScholarship, University of California, 2019. II

[17] S. Ray, V. S. Gordon, and L. Vaucher, "Evolving qwop gaits," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 2014, pp. 823–830. II

[18] I. Sarda and A. Johannet, "Behaviour learning by a reward-penalty algorithm: From gait learning to obstacle avoidance by neural networks," in *Artificial Neural Nets and Genetic Algorithms*. Springer, 1995, pp. 464–467. II

[19] R. L. McIntyre, "Recognizing actions using embodiment & empathy," Ph.D. dissertation, Massachusetts Institute of Technology, 2014. II

[20] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017. II

[21] V. Tuzcu and S. Nas, "Dynamic time warping as a novel tool in pattern recognition of ecg changes in heart rhythm disturbances," in *2005 IEEE international conference on systems, man and cybernetics*, vol. 1. IEEE, 2005, pp. 182–186. II

[22] K. Bert-Uwe, H. Carsten, and O. Reinhold, "The principle of software qrs detection-reviewing and comparing algorithms for detecting this important ecg waveform," *IEEE Engineering in Medicine and Biology*, vol. 42, 2002. II

[23] C. Dempsey, L. F. Abbott, and N. B. Sawtell, "Generalization of learned responses in the mormyrid electrosensory lobe," *Elife*, vol. 8, p. e44032, 2019. II

[24] S. M. Zu Eissen, B. Stein, and M. Potthast, "The suffix tree document model revisited," in *Proceedings of the 5th international conference on knowledge management*, 2005, pp. 596–603. II

[25] P. M. Ryu, M. G. Jang, H. Kim, Y. Hwang, S. Lim, J. Heo, C. H. Lee, H.-J. Oh, C. Lee, M. Choi *et al.*, "Apparatus and method for knowledge graph stabilization," Mar. 26 2013, uS Patent 8,407,253. II

[26] J. Franke, J. Niehues, and A. Waibel, "Robust and scalable differentiable neural computer for question answering," *arXiv preprint arXiv:1807.02658*, 2018. II

[27] K. A. Bollen and J. Pearl, "Eight myths about causality and structural equation models," in *Handbook of causal analysis for social research*. Springer, 2013, pp. 301–328. II