electronics

MDPI

Article

A Generic Preprocessing Architecture for Multi-Modal IoT Sensor Data in Artificial General Intelligence †

Nicholas Dmytryk * and Aris Leivadeas *

Department of Software and Information Technology Engineering, École de Technology Supérieure, Montreal, QC H3C 1K3, Canada

* Correspondence: nicholas.dmytryk.1@ens.etsmtl.ca (N.D.); aris.leivadeas@etsmtl.ca (A.L.)

† This paper is an extended version of our paper published in IEEE's 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Pisa, Italy, 14–16 September 2020.

Abstract: A main barrier for autonomous and general learning systems is their inability to understand and adapt to new environments—that is, to apply previously learned abstract solutions to new problems. Supervised learning system functions such as classification require data labeling from an external source and do not have the ability to learn feature representation autonomously. This research details an unsupervised learning method for multi-modal feature detection and evaluation to be used for preprocessing in general learning systems. The learning method details a clustering algorithm that can be applied to any generic IoT sensor data, and a seeded stimulus labeling algorithm impacted and evolved by cross-modal input. The method is implemented and tested in two agents consuming audio and image data, each with varying innate stimulus criteria. Their run-time stimulus changes over time depending on their experiences, while newly experienced features become meaningful without preprogrammed labeling of distinct attributes. The architecture provides interfaces for higher-order cognitive processes to be built on top of the unsupervised preprocessor. This method is unsupervised and modular, in contrast to the highly constrained and pretrained learning systems that exist, making it extendable and well-disposed for use in artificial general intelligence.

Keywords: explainable AI; perception; artificial general intelligence; next-generation networks; internet of things; learning models

check for updates

Citation: Dmytryk, N.; Leivadeas, A. A Generic Preprocessing Architecture for Multi-Modal IoT Sensor Data in Artificial General Intelligence. *Electronics* **2022**, *11*, 3816. https://doi.org/10.3390/electronics11223816

Academic Editor: Fernando De la Prieta

Received: 1 November 2022
Accepted: 17 November 2022
Published: 20 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Machine intelligence has been theorized since the mid-20th century, with mathematicians and scientists asking the question—can computational machinery perform intelligent operations analogously to the human brain? In the last decade, the adaption of neural networks has increased the ability and performance of computer systems to search, predict, and classify in large data sets—producing astonishing applications that are changing every technology sector from medicine to construction and transportation. These machine learning (ML) systems do emulate certain abilities of human intelligence—though they are limited in their ability to produce valuable information outside their heavily constrained use cases and pre-programmed operational functions.

Moreover, artificial intelligence (AI) technology (encompassing ML) attempts to tear down the interface limitations of ML systems by enabling human-like communication with the user. The most familiar example of AI technology is probably in our pocket. Voice assistants such as Siri and Google Assistant converse with users through spoken natural language. This AI ability aids in normalizing the human–machine interface and provides aspects of human intelligence such as answering basic questions about weather, news, and general facts by parsing internet data and performing aforementioned ML algorithms. The prevalence of these learning algorithms is increasing given the large number of new consumer IoT sensors, providing users with more digital interfaces that model and interact with their everyday environment [1]. Further, next-generation multi-access networks are providing the high bandwidth and low latency quality of service required for high up-time resource and reliability intensive applications [2–4]. Search, prediction, and classification functions are becoming faster and more accurate as IoT and AI technology progress—but no matter how optimized the processes become, current AI systems are not exhibiting human-like general intelligence (GI) or artificial general intelligence (AGI).

How do we move beyond static AI systems based on narrow ML to produce a system more capable of general intelligence, like humans? We believe a fundamental change from the current approach is required. Starting with the motivation/purpose driving the requirements for designing an AGI. All AI systems to date are designed with the sole purpose of serving humans. To enable a successful design of general intelligence, the system must be given autonomy. The sensory data that models the system's environment must be meaningful to the artificial being, just as it is to humans. The information must impact the system and hold meaning beyond the static servant-like operation of today's AI. Unlike deep reinforcement learning (DRL) and its positive/negative reward/punishment behavior tied directly to input, a novel AGI must hold a complex internal state through evaluation of event/scene abstraction and drive purposeful thought processes to examine how the input data and its actuation on the physical world affect itself at an abstract level. Unlike current AGI frameworks that attempt to explicitly design individually observed byproducts of the intelligence, such as the ability to learn language [5], a novel AGI framework should demonstrate byproducts such as linguistics, emotion, and conjecture without being explicitly programmed. A generic framework that can understand abstract event chains and their outcome on the internal state to develop new information through analogical observations and testing hypotheses through planned actuation on its external environment is key for emulating general intelligence.

Why create an artificial general intelligence? A general intelligence that could create explanatory knowledge based on multi-modal sensory input could innovate like humans and cause technological progress to boom. Many learning frameworks exist. Few are focused on general intelligence. The development and integration of neural networks has become a prominent method of solving specific machine learning problems: classification, prediction, and search. Implementing general intelligence is counter-intuitive, in contrast to traditional machine applications. Computational systems implemented in the AI space have the sole purpose of outputting information to serve human users. In the design stage of current state-of-the-art AI systems, the motivation and purpose of implementation is to augment the digital human experience (virtual assistants, self-driving cars, and other automated systems).

In a shift towards designing general intelligence, the motivation and purpose should be designing a system that can partake as an individual member of society; its run-time is a life filled with communication, reward, self-directed purpose, and awareness of its environment. In short, the motivation in the design of AGI in contrast to AI is to create a learning system with equivalence to humans not only in cognitive ability but also sustained cognitive freedom (without internal control).

Researchers struggle to find a rigid definition of intelligence, usually equating the observed byproducts (creativity, emotion, etc.) to intelligence itself. This results in an integration of designs each individually demonstrating an aspect/outcome of intelligence. However, the general nature of intelligence in its endless variations of observation cannot be defined by its many abstract products. We strive to design a general intelligence based on the more specific definition of intelligence—a system capable of producing explanatory knowledge: information with causal power, applied in the same domains as the system's sensors.

The research in this paper focuses on the prepossessing aspect of artificial general intelligence. Thus, herein, our contribution is to answer the following research questions:

1. RQ1: Can an unsupervised, untrained preprocessing module identify meaningful features in a scene?
2. RQ2: Can sensory data from multiple domains be abstracted by the same algorithm?
3. RQ3: Is feature abstraction from input data a requirement for AGI?

The structure of the paper is as follows: Section 2 presents the related work and provides some background on the AI limitations. Section 3 provides the AGI-based unsupervised and generic preprocessing model proposed. Section 4 presents the implementation of the system and associated results and relevant discussion. Finally, Section 5 concludes this paper and provides future directions.

2. Related Work and Background

2.1. AGI Design Shortcomings

Like most modern technology, AI systems consist of a vast number of different design approaches derived from the integration and optimization of previous work. AI in current implementations is designed to output information desired by a user [6]. The critical difference between AI and AGI technology is the motivational characteristic of each. The use case and motivation for narrow AI resides externally with the user of the technology; the system exists to serve the end user. In contrast, the purpose of AGI resides within itself, though impacted and influenced by the environment in which it resides and its innate design.

A general learning system's ability to thrive in changing environments with unique problem situations is an indication of true intelligence [7]; one not observed in any system to date. To evaluate and thrive in changing environments, we believe a learning system's motivation must be self-directed. Self-motivation is not a novel idea, many AI variants implement reward systems. Notably, deep reinforcement learning (DRL) has recently had success in heavily constrained narrow AI use-cases such as pattern recognition and perception [8,9]. Problems with this approach include over-training/over-fitting limiting adaptability, and constraining a system's ability purely with reward scenarios. There are also issues in DRL linking abstract scenarios to reward through many different inputs—in turn, systems that implement DRL do not understand broad outcomes of sensory data well.

Preprocessing in AGI is more than a segmentation problem. There has been great segmentation and feature extraction work done in recent years, though most algorithms are supervised, requiring a heavy amount of training data prior to runtime, and are designed without considering feature stimulation and interfaces for use and control by an AGI [10,11]. The use cases are designed specifically for one problem set and do not consider the problem of machine perception for general learning.

2.2. Sensor Fusion Constraints

To understand broad outcomes of sensory data, it is important for learning systems to be able to model their environment through multiple modalities and to understand the relations between sensory input. Multi-source information fusion (MSIF) and rough set theory (RST) are two fields that attempt to solve the problem of multi-set relations and dependencies through approximations (probability). Current models for heterogeneous MSIF are successful in finding relations between data sets of different modalities from a purely statistical standpoint. However, a main problem in MSIF is labeling. Although methods for supervised pre-classification and post-classification in MSIF are successful, unsupervised MSIF models are still largely underdeveloped [12,13]. The reason it could be underdeveloped is that the labeling of information is a subjective matter. The entity that labels information groups within data sets is fundamentally opinionated, as the information of the label itself is an abstract representation from an observer. Labeling a group of pixels as "red" is a large abstraction of the physical properties of the data formed through thousands of years of cultural information development. Therefore, to autonomously label data is to understand the data in its abstract meaning to the other observers of the data. In classification problems with a small result space, sensor fusion via multi-level ensemble models which combine single modal deep-learning classifier results at the end of processing function work well when the result space is small [14]. Though for large undefined result space problems such as general intelligence, this method may not be suitable as the learning path to solve problems is always segregated to each individual modality before being combined. Perception must be modeled through a combined view of sensor data before higher-order learning processes can evaluate the environment.

For an AGI to successfully label data in an unsupervised manner, it must do so by forming an opinion of the data. To form an opinion of the data it must communicate with other observers of the data. To communicate with other observers of the data it must share similar sensory and actuation modalities while being impacted by the information in a similar way. To be impacted by information, abstracted input must cause a change to the AGI's internal state. A variety of utility functions exist in today's learning systems for driving decisions and goals at lower levels of abstraction.

2.3. Towards Explainable AGI in IoT

With the exponential increase in IoT devices and associated edge infrastructure, narrow AI use cases in the image and audio domain are becoming widespread in the modern consumer smart device market [15]. As intelligent capability improves in IoT networks, the notion of distributed intelligence—which AI processes should occur at the device and edge level, versus which must be run in the cloud is becoming a difficult, application-specific problem [16]. A generic framework must be flexible in deciding which discrete and abstracted sensory information must be sent to the cloud in order to operate within network bandwidth and latency capabilities.

Explanatory AI is becoming increasingly important as the capabilities of autonomous systems increase and fewer pre-programmed decisions are made by machines. In the account of preventing error scenarios and understanding the inner workings of AI technology—development is moving towards AI systems that can be understood by humans (non-black box). Some researchers propose to add an actual dialogue system to communicate through external interfaces [17]. However, this method still risks translation problems, where parts of the internal state can not be represented in a human-understandable way. Others see the need for explanatory AI to reduce the likelihood of a catastrophic scenario. Pre-programmed ethical models have been proposed to internally influence an AGI into appropriate behavior [18], as well as distributing computational capacity of intelligent systems with blockchain to reduce risk [19].

We believe the first step towards explainable AI is a human-readable abstraction of internal state and memory. The characteristics of image segmentation are conducive to human readable abstraction and may be leveraged during preprocessing to provide a facility for explainable AI. Image segmentation is a method of extracting discrete parts of an image into individual segments. A state-of-the-art method of image segmentation is the differentiable feature clustering algorithm implemented through convolutional neural networks [20]. For a simpler implementation with solid performance, a custom clustering algorithm extending the principle of DBSCAN [21] can be implemented for the segmentation part of the preprocessing algorithm in this study.

The main problem with utility/reward functions is mapping high-level events to abstract scenarios, and evaluating their impact on the internal state for appropriate thought and output actuation. Utility functions are currently working at a lower level of abstraction and have corrigibility problems [22,23]. The principle of associative bias aims to solve this problem through a system that is innately impacted by a small set of rudimentary input data characteristics—which grows into a more abstract impact through scene associations and event-chain understanding [24]. This butterfly-like effect enables data impact that begins at a low level at the start of run time, and progresses to higher levels of abstraction, as the system experiences more environments and absorbs more data [24].

Accordingly, the purpose of this paper is to present a method of input preprocessing conducive to AGI—removing the need for internal intervention such as data labeling input from a human. The preprocessing method presented in this article is a key component of the high-level proposed distributed AGI system presented in our previous work [24]. In this system, as seen in Figure 1, the AGI architecture is spread among wireless, edge, and cloud network components. The application architecture consists of a preprocessing layer at the edge, relational memory construction, storage, exploration, and thought formulation/actuation in the cloud. The focus of this article is the preprocessing layer of the architecture—how multi-modal sensor data are abstracted and how the system is dynamically stimulated by the features it percepts. A detailed implementation of this experiment's memory interface designed to validate the preprocessing algorithm is detailed in Section 4.2.

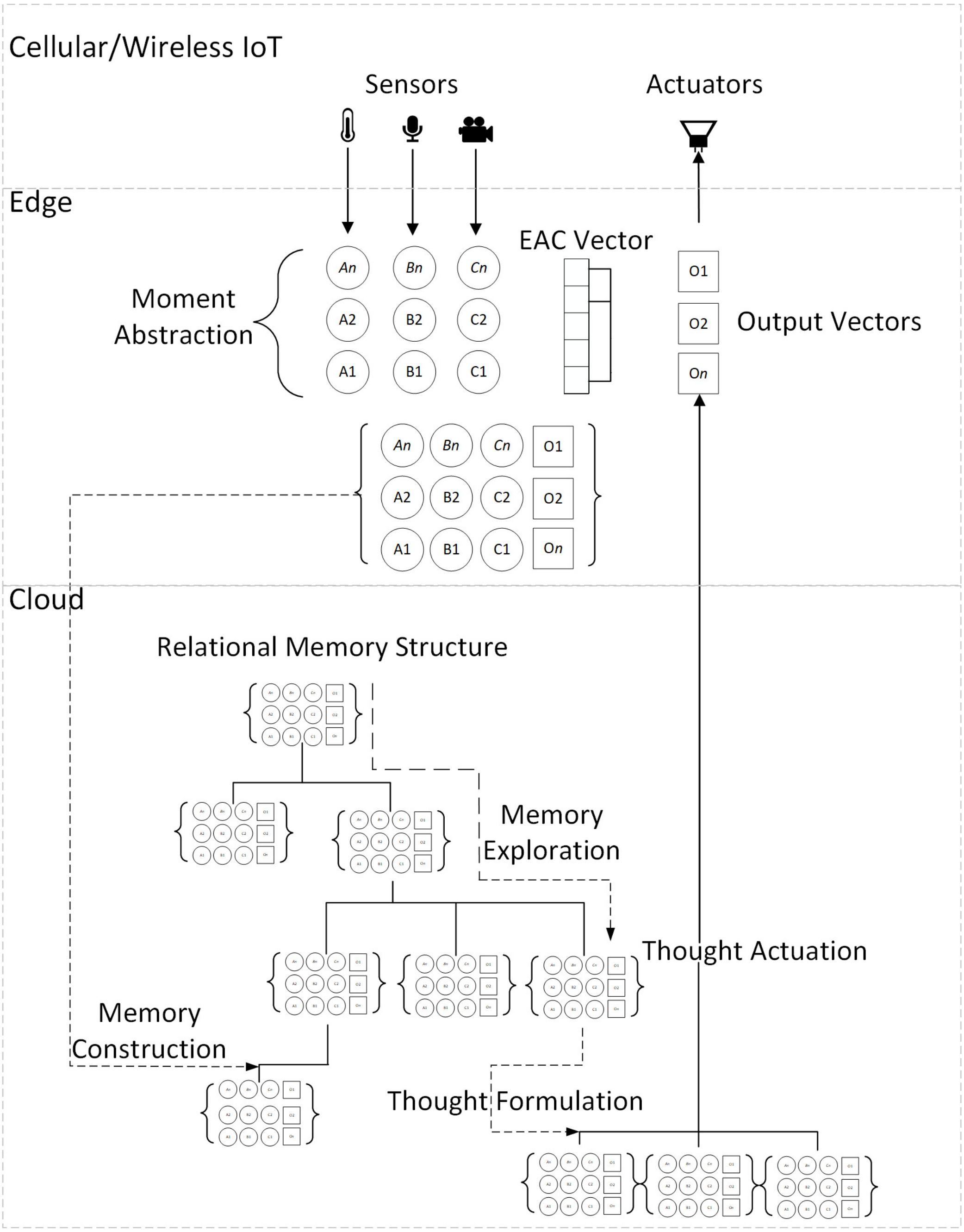

Figure 1. Proposed learning system model.

In contrast with the pertinent literature, our proposed method employs a feature abstraction mechanism and an innate and dynamic stimulus system all during preprocessing.

The key functional characteristics required for a system to have the capability to learn and communicate are the following: preprocessing output must not be limited to a finite classification space. This will allow general learning systems to model and percept their environment more accurately (closer to objective reality), allowing them to learn new/general problems. The design characteristics that constitute the preprocessing algorithm detailed in this article can enable a technology more capable of learning problem sets that change and evolve over time, such as learning natural language from scratch rather than mapping input to predefined language models.

The design of the preprocessing model that aims to target the shortcomings of today's AGI by providing a generic foundation is detailed in the following section. The system model in Section 3 is an extension of a preliminary model presented in our previous work [24], which, however, did not include any proposed algorithms or concrete implementation and evaluations. Hence, in this paper, the results applied to multiple sensor types are also demonstrated and evaluated through a trial run.

3. Unsupervised and Generic Preprocessing Model

The model's components include a generic IoT sensory input, a transform method to organize data into a generic format, an unsupervised feature identification algorithm, a feature abstraction method, and a stimulus system. The output of the model is purposed for input to the rest of the AGI system. By generalizing the sensory interface an AGI may function with input from multiple modalities and is not restricted in design by sensor type (audio, video, temperature, etc.). Figure 2 displays a high-level depiction of the preprocessing method at a component level, whereas each component is detailed in the following subsections.

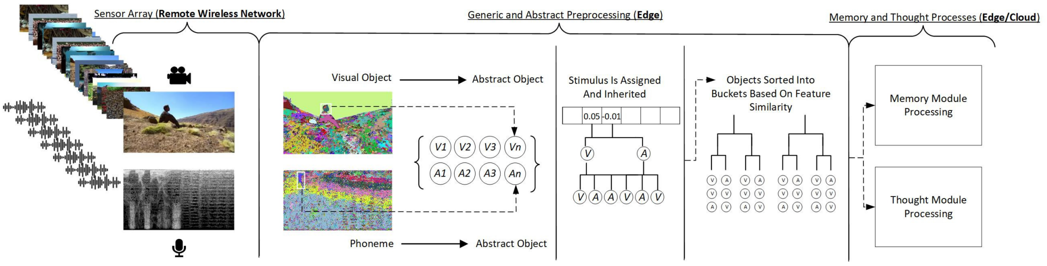
Sensor Array (Remote Wireless Network)
Generic and Abstract Preprocessing (Edge)
Memory and Thought Processes (Edge/Cloud)
Visual Object
Abstract Object
V1 V2 V3 Vn
A1 A2 A3 An
Phoneme
Abstract Object
Stimulus Is Assigned And Inherited
0.05 -0.01
V A
V A A V A V
Objects Sorted Into Buckets Based On Feature Similarity
Memory Module Processing
Thought Module Processing

Figure 2. Preprocessing algorithm sequence.

3.1. Input Alignment and Buffering

The input alignment and data buffering method proposed in [24] is responsible for structuring sensor data in preparation for system input. The multi-modal data inputted into the system have different frame rates and sample structuring; the input layer structures data into groups according to sample rate and may alter the structure of the data depending on its format. The input layer algorithm is detailed as follows:

Operation I1: Find the smallest frame rate within the modality set. The variable M represents an input modality (video, audio, etc.); the function $FR(M_n)$ returns the frame rate of the modality.

$$FR_{MIN} = \min\{FR(M_1), FR(M_2), \ldots, FR(M_n)\} \tag{1}$$

Operation I2: Find the normalization quantity Q for each modality of the set.

$$Q(M_n) = \frac{FR(M_n)}{FR_{MIN}} \tag{2}$$

Operation I3: Determine the presence of exclusive data frame dimension within the modality set (e.g., monaural or stereo audio frames may be singular in dimension). If the data frame is one-dimensional, it is extended by the concatenation of further frames. The number of frames concatenated is proportional to the normalization quantity. If the data frames are in one dimension, the vectors may be extended; if they exist as matrices, their frames are grouped into a set. The function $F(M_n)$ returns a frame of data based on the modality and group index i_g. The group index refers to an initial modality frame within a buffer group BG.

$$BG(M_n, i_g) = \begin{cases} F(M_n, i_g) \frown F(M_n, i_g + 1) \frown \ldots \\ F(M_n, i_g + Q(M_n) - 1), \text{ if } dim(M_n) = 1 \\ F(M_n, i_g), F(M_n, i_g + 1), \ldots, \\ F(M_n, i_g + Q(M_n) - 1), \text{ otherwise} \end{cases} \tag{3}$$

The extension of unit-sized vectors enables the grouping of data and facilitates relational processing across modalities. The output result is a set of data frames aligned with respect to a timing reference interval determined by FR_{MIN}. These outputted data frames, referred to as buffer frames, are consumed by the moment layer for preprocessing.

3.2. Transform for 1-D Input Types

Any singular-dimension data input is required to be represented in multi-dimension matrix form. For high sample rate time domain signals such as audio, the continuous discrete data are altered into a spectrogram image/matrix representation where each column index represents a time bucket and a row index a frequency bucket, and the value at combined time/frequency index represents the power value of the signal (optionally normalized to a visual representation pixel value system such as greyscale 0–256).

1. Each 1-D buffer frame in a group is sliced into t number of s (second) sized buckets.
2. Fast Fourier transform is performed on each time bucket $FFT[F(t)] - > F(w)$
3. Each frequency domain bucket representing s seconds is transposed and joined to form a 2-D matrix representing the entire length of the buffer group.

$$M(BG(M_n)) = \begin{bmatrix} F(0,0) & F(1,0) & F(2,0) & F(t,0) \\ F(0,1) & F(1,1) & F(2,1) & F(t,1) \\ F(0,2) & F(1,2) & F(2,2) & F(t,2) \\ F(0,i_g) & F(1,i_g) & F(2,i_g) & F(t,i_g) \end{bmatrix}$$

The operation above in the context of microphone data can be observed in Figure 3; the high-frequency time domain audio data are represented as a spectrogram in a matrix format with high power frequency-time indexes a high value on the greyscale (white) and lower power indexes represented lower on the greyscale (black).

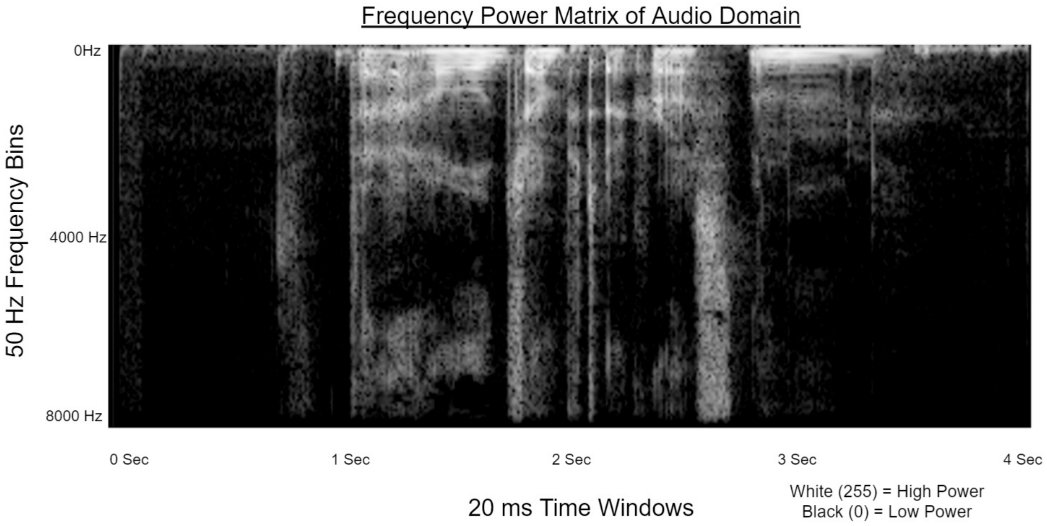

Figure 3. Spectrogram matrix of the audio signal.

3.3. Unsupervised Feature Identification

The purpose of the moment layer is to provide the system with the ability to abstract input data. The goal of this preprocessing algorithm is to find generic, meaningful intra-frame relations and segment them into discrete data—representing them as constrained chunks of memory for the intelligence. Parts of this process involve cluster discovery and extend the general concept from the popular density-based spatial clustering of applications with noise (DBSCAN) [21]. The clustering algorithm implemented in the moment layer differs in its dynamic nature. Instead of the static constraints, DBSCAN implements cluster determination (minimum number of points and distance epsilon), and the proposed algorithm will constrain a cluster by distance as well as by value deviation. The value deviation constraint is dynamic and based on the value of neighboring elements. The exact clustering method is detailed as follows.

Operation M1: Determine the overall data range of frame.

$$C_{Range} = \max(F(M_n)) - \min(F(M_n)) \tag{4}$$

Operation M2: Determine the cluster range focus (optional focus mask). More detailed range constraint gradients are applied to create focus/concentration. Like an eye's focal point in comparison to its periphery, the deviation constraint of the clustering algorithm will be tighter depending on the region of interest/focus. The range constraint center point may be applied to any position on a frame. A position, number of constraint levels, and normalized range constraint per level must be specified. The position may be changed dynamically by other parts of the system, while the other configuration values must be statically specified. The constraint levels are applied radially from the chosen position. The default position is the center of a frame.

Operation M3: The deviation- and neighborhood-based clustering discovery algorithm will generically cluster data frames based on value deviation range and neighborhood distance range. This algorithm has two major components: (i) for every value in a data frame, find the values in the neighborhood (radial distance, denoted as D) that are within the value's constraint range and create a base cluster; (ii) connect overlapping base clusters together, generating aggregate clusters. Optionally, in implementation one we can choose not to aggregate clusters if members from one cluster are outside the constraint range of members from the other cluster, as this feature may pose a constraint on cluster size in features with steep gradients. The pseudo-code is presented in Algorithm 1.

Algorithm 1 Deviation- and Neighborhood-Based Clustering

```
Require: $D$
  for dataPoint in frame do
    generate value constraint range $C$
    for neighbor in $D$ do
      if value(neighbour) within $C$ then
        create neighborhood cluster
      end if
    end for
    if neighbourhood cluster intersects with a main cluster then
      join neighborhood cluster to intersecting main cluster
    else
      promote neighborhood cluster to become a main cluster
    end if
  end for
```

3.4. Feature Abstraction

Operation M4: This operation generates cluster metadata. Clustering enables the abstraction of data and feature grouping. When one observes an object and closes their eyes, they may remember the shape of the object, the color, the size, etc., but they cannot completely recreate the object with perfect detail in their mind. Similarly, the learning system abstracts the raw data into higher-order information. The new form of the data contains less information but allows further relational processing and manageable operation. The output of the clustered data frame is composed of a set of moment vectors; each moment vector contains data representative of cluster characteristics. Although configurable in implementation, examples of members constituting cluster metadata may include (i) number of data points, (ii) average value, (iii) standard deviation of value, (iv) range of values, (v) average distance between data points, (vi) standard deviation of the distance between data points, (vii) complex geometric shape (boundary), (viii) symmetry information, and (ix) proximity information.

This moment-abstraction cluster gives the intelligence the ability to extract information from a large data set and store it for use with abstract representation. Note that the aim of this part of the algorithm is not to perform traditional image/audio processing—it is to find meaningful pieces of data and to represent them as constrained chunks of memory for the intelligence. The number of moment vectors per sampled data matrix may vary depending on the number of clusters found. For example, looking at a blue sky versus looking at a desk full of detailed objects, or hearing one word compared to hearing a clip of a song—the more detail present, the more clusters will be identified and, in turn, the more metadata per moment vector generated.

3.5. Stimulus System

Operation M5: Elementary activation characteristics (EACs) or innate stimulus vectors contain predefined, static low-order stimulus information for each modality. The purpose of EACs is not to pre-program the system with abstracted concepts of innate behavior—e.g., a desire to reproduce or an ability to avoid danger. Rather, EACs are meant to prime the system's thoughts with stimulation toward certain input modality characteristics.

The structure of EACs is formed from the modality in which the natural stimulation quantity exists, the moment vector metadata attribute upon which the stimulation is dependent, the condition associated with the value(s) of the metadata, and the stimulus quantity between -1 (no stimulation) and 1 (maximum stimulation).

Although EACs are initially static and only applicable to certain features of data within a moment, other data in the same frame inherit decimated versions of the innate stimulus. Stimulus then grows dynamically as the system experiences its environment. Further, higher-order memory and thought processes may alter stimulus vectors. Before indexing and storing the features, the stimulus is assigned.

There are four implemented directives for assigning stimulus. Firstly, innate stimulus values are assigned to a feature if a feature's attributes are evaluated to be within range (D1). Secondly, if a feature is assigned multiple stimulus values, they are to be added (D2). Thirdly, all features within a moment that do not trigger any innate stimulus are assigned the decimated stimulus score(s) of other feature(s) in the moment if the other feature(s) have triggered an innate stimulus (D3). Lastly, innate stimulus values apply per modality; however, if triggered, the decimated stimulus value will also be applied/inherited by features in another modality (D4). When a feature is assigned a stimulus, it is added to its attribute list before being indexed. The process repeats for every new moment experienced, but new feature attributes that inherited stimulus in previous moments may now also cause stimulation for newly experienced moments. Further, the thought module (a higher-order process on top of the preprocessing layer) may also alter the stimulus matrix, providing feedback based on cognition. The stimulus system and its four directives are depicted in Figure 4.

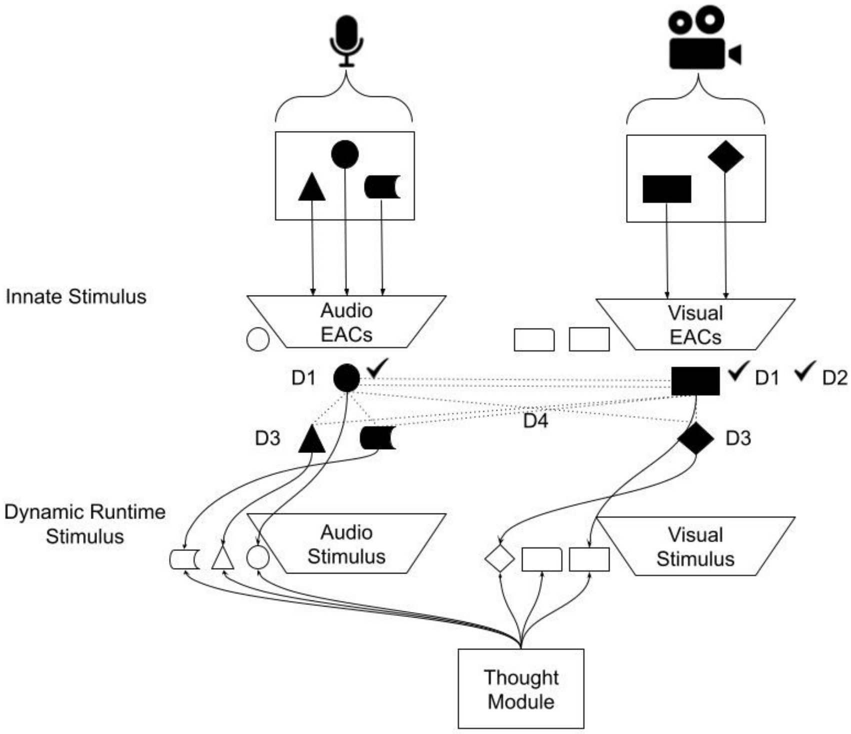

Figure 4. Stimulus system.

3.6. Event Buffering

Note the event layer was not implemented in the trial because a simplified memory structure was implemented in the place of downstream memory and thought components. The event layer composes meaningful events by the aggregation of cross-modal moment vectors under constrained criteria. An event is the highest-order abstraction of inputted data in the preprocessing module.

When adjacent moments share strong moment vector characteristics, an event is created. Each moment of modalities A, B, C must not vary greater than a pre-configured threshold (i.e., a, b, c).

$$Event = \sum[A_n, B_n, C_n], \text{when} \tag{5}$$
$$A_{n+1}\Delta A_n < a \cdot B_{n+1}\Delta B_n < b \cdot C_{n+1}\Delta C_n < c$$

The event aggregation terminates when adjacent moment vectors no longer share strong characteristics. Events bind modalities together; i.e., if the event aggregation condition is only present in one modality, the other modalities are still evaluated and grouped as part of the event. This layer extends the system's capacity to understand information from moments to events in an effort to enable comprehension of scene and pattern recognition differences between independent events in the thought module. The trial implementation in subsequent sections groups different modalities together into an indexed memory tree.

4. Feature Identification Results and Discussion

To demonstrate the preprocessing method in the visual and audio domains, two agents implementing the above algorithms with different EAC configurations perceive a trial video [25] containing various diverse scenes of nature, and including commentator annotation. The goal is to understand how agents with different root EACs are impacted by the data they experience over the course of the video, demonstrating the growth of unique stimuli via associative bias.

4.1. Data Set

The audio samples of the video simulated microphone input and the image framed simulated camera input. The trial run ran on 60 s of video. The image frames were provided at 25 frames per second (1500 total frames) and the audio was provided at 44,100 samples per second (2.646 million total frames). The compressed size of the video was 12 MB.

4.2. Implementation

The unsupervised and generic preprocessing method described in Section 3 above was implemented from scratch in Python 3.9.1. Numpy and Scipy.FFT were used to facilitate data handling, FFT transforms, and other base statistical functions. Moviepy was used to decompress video files and extract raw audio and video data. Seaborn, Matplotlib, and PIL were used to graph output data and visualize features.

The following classes were written to implement the design:

- Raw Data Extractor;
- Data Alignment and Buffering;
- 2-D Sensor Transform (to represent 1-D data such as audio in 2 dimensions) ;
- Deviation- and Neighborhood-Based Clustering (Augmented DBSCAN) Algorithm;
- Cluster to Abstract Feature Algorithm;
- Stimulus Assignation Algorithm;
- Feature Attributes Indexing Algorithm.

In the modified DBSCAN algorithm implemented, distance and value deviations were statically configured. No minimum cluster size constraint was implemented, contrary to DBSCAN. The smaller the distance and deviation values configured, the larger the number of distinct features that will be identified in the data. Inversely, the larger the distance and deviation values the fewer distinct features will be identified. In the trial run, we used a radial pixel distance equal to 3 (square neighborhood size of 49) and a value deviation of $+/-3$ (average RGB value) from the base/center value of the neighborhood. For example, if the center value of the neighborhood had an average value of 176, the 48 other cluster member candidates within the neighborhood must be between 173 and 179 to be included in the cluster. It is important to note that a radial distance of 3 and a value deviation of 3 were used in this use case where feature resolution did not have to be optimized. In a real production system, an AGI system driving this preprocessing algorithm could dynamically modify these two parameters to change the resolution of feature detection within its sensor arrays.

An example of the feature output from the abstraction algorithm can be seen in Figure 5. The algorithm's ability to identify distinct objects within a scene in an unsupervised and untrained manner is demonstrated. The algorithm output shown identifies a human face, the sky, wooden boards, and a tree. The identified objects are shown separately for visual effect, though all features within the image are captured by the system.

Each cluster holds the indexes and values of all pixels that are part of a distinct feature. Before assigning a stimulus, the indexes and values are transformed into an abstract representation.

The representative statistics are configurable and extendable; in this implementation, the abstract attributes include:

- size (number of pixels);
- min, mean, max value;
- variance, standard deviation;
- geometric symmetry score.

The geometric symmetry score was obtained by scoring the histogram correlation (cosine similarity) of the X and Y index projections. The inspiration to evaluate symmetry and other visual characteristics comes from work concluding bilateral symmetry is a factor affecting visual stimulus [26,27]. The innate stimulus lists (EACs) configured in the trial run were limited to four distinct attribute range conditions and activation stimuli. These vectors were chosen so the activation conditions do not overlap with each other, and to demonstrate how agents can be stimulated differently depending on their innate EAC configuration.

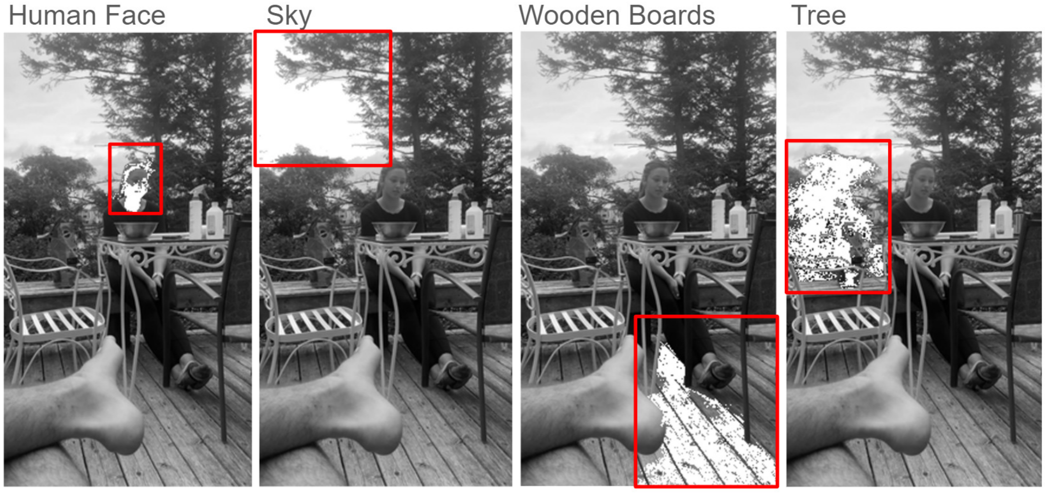

Figure 5. Unsupervised and untrained object identification

Agent 1 EAC Configuration:

- Audio EACs : ("mean": [0, 100], "activation": 0.05), ("variance": [0, 10], "activation": −0.005);
- Image EACs : ("size": [150, 250], "activation": 0.15), ("mean": [150, 230], "activation": −0.2).

Agent 2 EAC Configuration:

- Audio EACs : ("mean": [100, 200], "activation": 0.05), ("variance": [10, 20], "activation": −0.005);
- Image EACs : ("size": [50, 150], "activation": 0.15), ("mean": [80, 150], "activation": −0.2).

After all features in a moment are assigned stimulus, each feature the clustering algorithm extracts is indexed depending on its abstracted attributes. The implementation of the indexing scheme assigns a unique tag corresponding to the attribute range (color of a feature, size of the feature, etc.). The ID tags are based on the following attribute ranges:

- ID1: 5 tags corresponding to the greyscale range. 1: (0, 51), 2: (51, 102), 3: (102, 153), 4: (153, 204), 5: (204, 255);
- ID2: 3 tags corresponding to geometric symmetry score. 1: (0, 0.33), 2: (0.33, 0.66), 3: (0.66, 1);
- ID3: 3 tags corresponding to the geometric size of the feature. 1: (0, 50), 2: (50, 100), 3: (100, Max).

Given the human face feature example in Figure 5 with the attributes: greyscale color: 127, symmetry: 0.72, size: 73, the corresponding feature index ID would be "432". Based on the indexing scheme a total of 45 (5 × 3 × 3) indexes existed in the implemented trial.

The output moments and associated stimulus for each agent are depicted in Figure 6. The moments represent the system's experience of image and audio features and their related stimulus for three-second intervals—each interval contributing more data and stimuli to the index/data store.

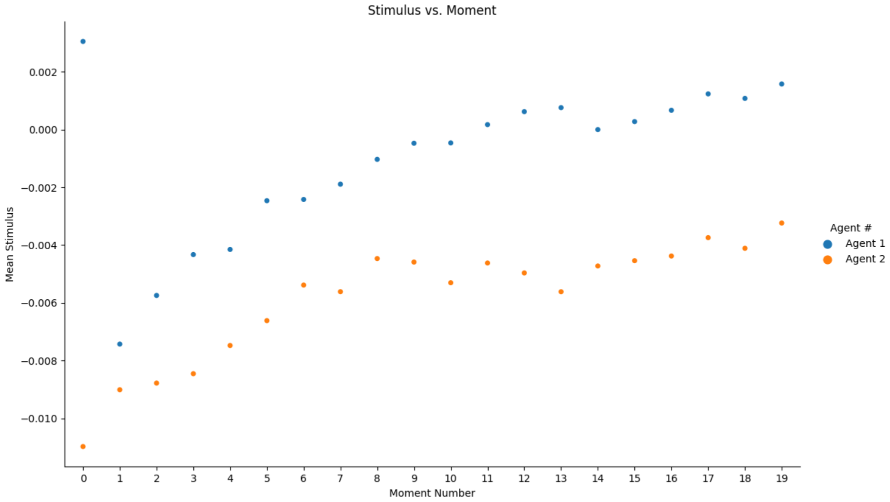

Figure 6. Stimulus vs. moment.

The output of the preprocessor is an indexed memory tree containing features with varying stimuli—enabling high-order processes to plug in and use the structured, impactful data for thought processes. The stimulus values in the indexed feature tree may be altered by higher-order processes if they are determined to be necessary when in pursuit of a multi-action goal.

The stimulus vs. moment plot displayed in Figure 6 shows the relationship between the mean stimulus and the mean number of features in each index of the memory tree for each entire moment. The main finding for both agents is when the system starts out the tangent of stimulus from one moment to the next is volatile. As the system experiences more moments, the stimulus levels out and changes become less drastic; the main contributing factor is the increased number of features present in each index of the memory tree—as more features are present in an index their stimulus interferes destructively and partially cancel out in each direction, reaching more of a steady state. From moment to moment, Agent 1 generally experiences higher stimulus than Agent 2. This is not the case when the stimulus is evaluated per memory index rather than averaged out at the moment level.

Figures 7–11 reveal stimulus and attribute relationships at a memory index level. The plots compare the relationships between the system's mean stimulus values and feature attributes in the memory tree for the two different agents. When comparing the number of features in memory indexes versus the mean stimulus of those features (Figure 7), Agent 1 generally experiences positive stimulus in memory areas with larger feature numbers, while Agent 2 experiences positive stimulus in memory areas with smaller numbers of features. In a full implementation, the cognitive processes of Agent 1 may favor environments with fewer discrete features, while Agent 2's cognitive functions would be stimulated more in environments with many discrete features. In the case of average feature size per memory index versus mean stimulus (Figure 8), Agent 1 experiences on average more negative stimulus associated with larger feature sizes, while Agent 2 experiences on average more positive stimulus in memory areas with large feature sizes. The same reverse correlation between agent stimulus exists when analyzing the average mean power value and symmetry relationships with mean stimulus (Figures 9 and 10). Overall, after both agents experience the same data, Agent 1's stimulus is slightly negatively biased relative to Agent 2's stimulus, which is more positively biased by its emulated environment (Figure 11).

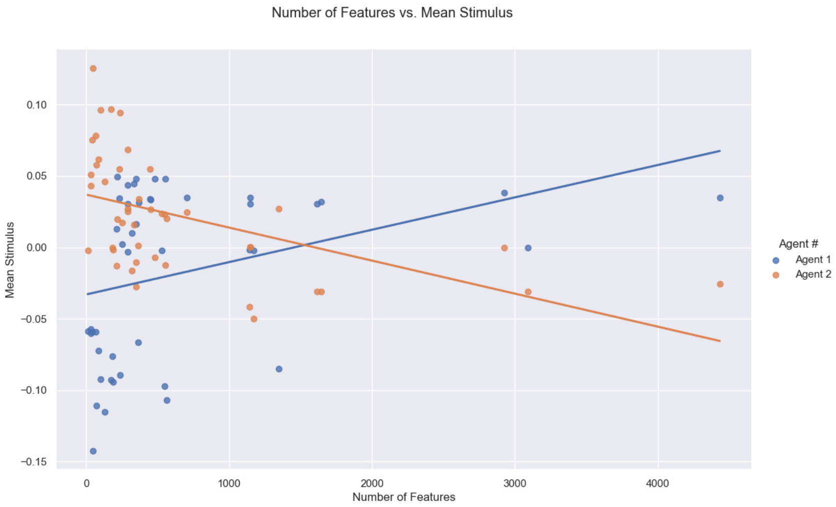

Figure 7. Number of features vs. mean stimulus.

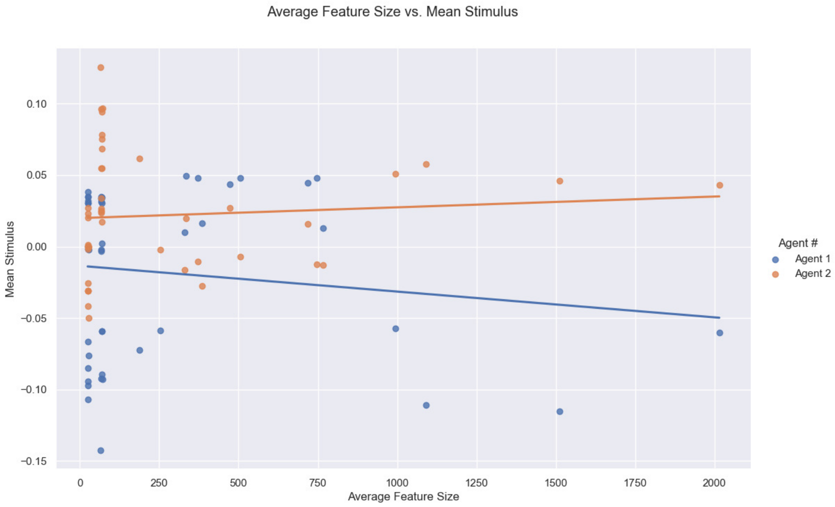

Figure 8. Average feature size vs. mean stimulus.

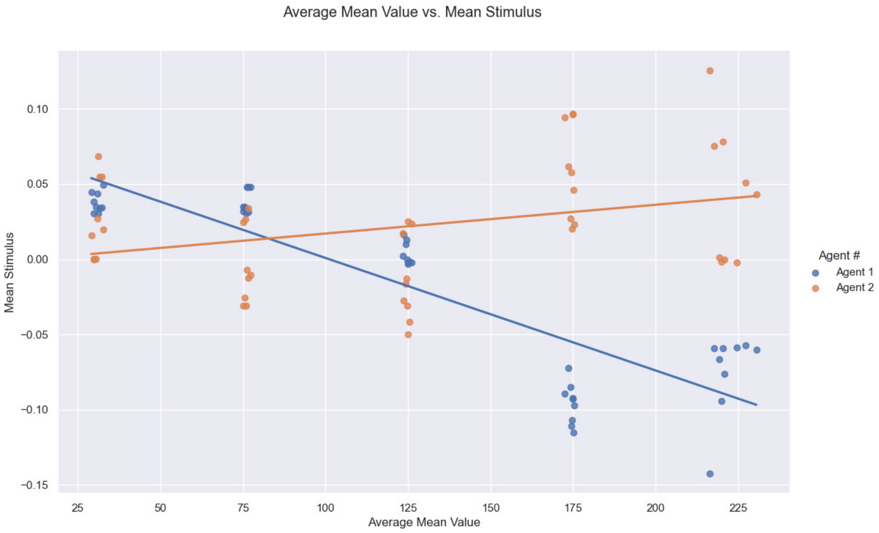

Figure 9. Average mean value vs. stimulus.

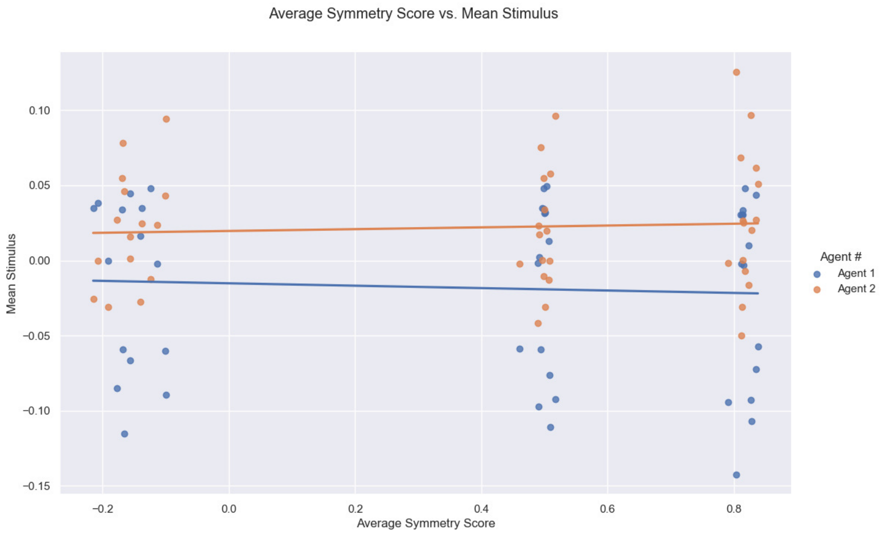

Figure 10. Average symmetry score vs. mean stimulus.

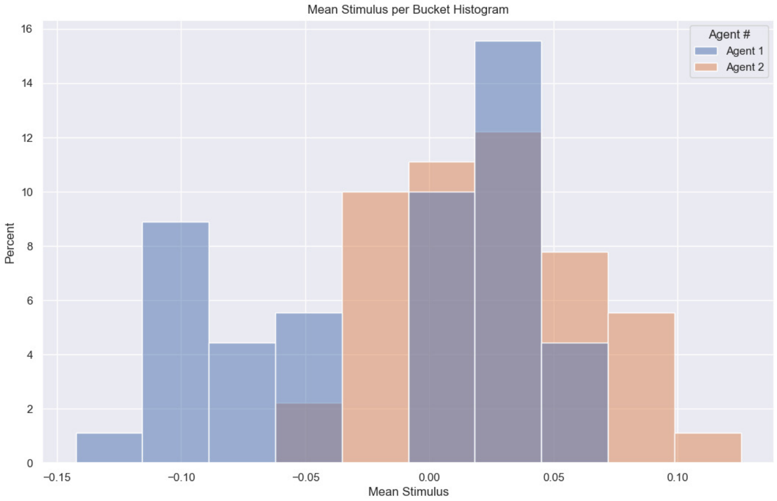

Figure 11. Stimulus distribution.

The next section discusses how the stimulus biases towards different feature attributes demonstrate the system's innate preferences—though in an integrated system (with higher-order cognitive processes) stimulus evaluation would grow past innate function as the system experienced more data and a formal cognitive loop-back was achieved.

4.3. Discussion

The results demonstrate how raw sensory information can be organized into meaningful abstractions for downstream memory and thought processes. The motivation behind the design of the preprocessor is to give data meaning. Traditional AI/ML applications are not impacted by data and there is no sense of self past reward or punishment. The problems presented by deep reinforcement learning's attempt at making data meaningful are addressed by the preprocessing system. Reward and punishment systems have narrow use cases. Input is too strongly correlated with reward/punishment as DRL's pure focus is on the optimization of weights and biases for the highest reward score and lowest punishment score. Although input data in DRL has an impact on the system, its inability to adapt to new problem sets arises because of the constraint created during optimization. The set of data attributes that cause stimulus is unbound in the proposed preprocessing method, in contrast to DRL where the system only has focus specific data attributes that impact reward/punishment. A different approach is required to spawn artificial general intelligence. This method of preprocessing may also enable easier regulation of stimuli as positive and negative stimuli decimate each other. The conjecture is not meant to compare the preprocessing method to DRL, as the preprocessing method is a foundation to facilitate intelligent behavior—only one functional block of an AGI.

The design incorporates two facilities conducive to explainable AI: the method in which the system perceives its input data and is impacted by it is visible and traceable, and the method in which the system abstracts discrete chunks of information is structured in a human-readable form rather than in the configuration states of a neural network. Another benefit is the separation of stimulus from adaptive function. It is not always beneficial for a learning system to be reactive and change its decision-making processes based on negatively impacting input data. In some scenarios, the solution to the problem presented may require actions that continue to cause negative inputs. The preprocessor presented enables the partial separation of thought and decision-making processes from a stimulus, presenting an opportunity for abstract cognitive regulation from higher-order processes rather than autonomous reactive regulation when presented with complex problem scenarios where the solution is dependent on many transitory actions.

Ultimately, the preprocessor will facilitate downstream processes to understand how data are affecting self—does the input data hurt them? Does the input data annoy them? Which series of features causes pleasure and happiness? These self-evaluation constructs are not possessed in current-day state-of-the-art AI/ML systems.

The research questions posed in Section 1 are answered and summarized.

RQ1: Can an unsupervised, untrained preprocessing module identify meaningful features in a scene?
The preprocessing module implemented was proven to identify features in a scene and turn them into an abstract representation holding descriptive attributes of the raw data.
RQ2: Can sensory data from multiple domains be abstracted by the same algorithm?
Sensory data were abstracted in the audio and visual domain by the same generic feature detection and extraction algorithm. This generic architecture can help manifest AGI as many different IoT sensor types can easily plug into the system with minimum configuration.
RQ3: Is feature abstraction from input data a requirement for AGI?
Feature abstraction is found to be a requirement for intelligence as it diminishes the size of input data, converts large data sets into small discrete abstract representations, and enables the system to be inherently stimulated by features as it experiences complex scenes over time.

4.4. Applications

The preprocessing system presented functions to aid an intelligent system's perception ability by continuously modeling its environment in an abstract representation and evaluating the impact of data on self. Higher-order AGI processes can plugin to this modular preprocessing component to provide full cognitive ability. The system's interfaces are generic and completely data-driven, allowing flexibility for edge/cloud integration. The IoT applications this design can be used for are generic but include cognitive virtual assistants and autonomous robotics.

5. Conclusions

In conclusion, an unsupervised preprocessing method was demonstrated with the capability of abstracting generic IoT sensor data into structured feature sets, each causing complex stimuli. The stimulus directives applied enable a system to not only be stimulated by attributes meeting preconfigured conditions, but stimulated by previously experienced features that did not originally cause stimulus. With this preprocessing method, data are meaningful to a system. The output of the preprocessing algorithm carries structured metadata and complex stimulus scores formed by its past experiences and may serve as a foundation for further intelligent processes to manifest. The system supports the ideology of explainable AI in that its experiences are its training data—much like a human. The visibility of how the system interprets its experiences over time is heightened by the preprocessing algorithm—as it enables external bodies to understand how the intelligence is manifesting and why an AGI system may make a certain decision. Future work will include the implementation of higher-level processes (memory and thought) that will use this preprocessing system as a foundation to manifest general intelligence.

Author Contributions: Conceptualization, N.D., A.L.; methodology, N.D., A.L.; software, N.D.; validation, N.D., A.L.; formal analysis, N.D.; investigation, N.D.; resources, A.L.; data curation, N.D.; writing—original draft preparation, N.D.; writing-review and editing, A.L.; visualization N.D.; supervision, A.L.; project administration, A.L.; funding acquisition, A.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), Grant no. RGPIN-2019-05250.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mukhopadhyay, S.C.; Tyagi, S.K.S.; Suryadevara, N.K.; Piuri, V.; Scotti, F.; Zeadally, S. Artificial Intelligence-based Sensors for Next Generation IoT Applications: A Review. *IEEE Sens. J.* **2021**, *21*, 24920–24932. [CrossRef]
2. Lin, Z.; Lin, M.; de Cola, T.; Wang, J.B.; Zhu, W.P.; Cheng, J. Supporting IoT With Rate-Splitting Multiple Access in Satellite and Aerial-Integrated Networks. *IEEE Internet Things J.* **2021**, *8*, 11123–11134. [CrossRef]
3. Lin, Z.; An, K.; Niu, H.; Hu, Y.; Chatzinotas, S.; Zheng, G.; Wang, J. SLNR-based secure energy efficient beamforming in Multibeam Satellite Systems. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, 1–4. [CrossRef]
4. Lin, Z.; Niu, H.; An, K.; Wang, Y.; Zheng, G.; Chatzinotas, S.; Hu, Y. Refracting ris-aided hybrid satellite-terrestrial relay networks: Joint Beamforming Design and optimization. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, *58*, 3717–3724. [CrossRef]
5. Goertzel, B.; Ke, S.; Lian, R.; O'Neill, J.; Sadeghi, K.; Wang, D.; Watkins, O.; Yu, G. The cogprime architecture for embodied Artificial General Intelligence. In Proceedings of the 2013 IEEE Symposium on Computational Intelligence for Human-like Intelligence (CIHLI), Singapore, 16–19 April 2013; pp. 60–67. [CrossRef]
6. Górriz, J.M.; Ramírez, J.; Ortíz, A.; Martínez-Murcia, F.J.; Segovia, F.; Suckling, J.; Leming, M.; Zhang, Y.-D.; Álvarez-Sánchez, J.R.; Bologna, G.; et al. Artificial intelligence within the interplay between natural and artificial computation: Advances in data science, trends and applications. *Neurocomputing* **2020**, *410*, 237–270. [CrossRef]
7. Chollet, F. On the Measure of Intelligence. *arXiv* **2019**, arXiv:1911.01547.
8. Rocha, F.M.; Costa, V.S.; Reis, L.P. From Reinforcement Learning Towards Artificial General Intelligence. In *Proceedings of the Trends and Innovations in Information Systems and Technologies*; Rocha, Á., Adeli, H., Reis, L.P., Costanzo, S., Orovic, I., Moreira, F., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 401–413.
9. Elton, D.C. Applying Deutsch's concept of good explanations to artificial intelligence and neuroscience—An initial exploration. *Cogn. Syst. Res.* **2021**, *67*, 9–17. [CrossRef]
10. Zhang, J.; Su, Q.; Tang, B.; Wang, C.; Li, Y. DPSNet: Multitask Learning Using Geometry Reasoning for Scene Depth and Semantics. *IEEE Trans. Neural Net. Learn. Syst.* **2021**, 1–12. [CrossRef]
11. Zhai, W.; Gao, M.; Souri, A.; Li, Q.; Guo, X.; Shang, J.; Zou, G. An attentive hierarchy ConvNet for crowd counting in Smart City. *Clust. Comput.* **2022**. [CrossRef]
12. Zhang, P. Multi-source information fusion based on rough set theory: A review. *Inf. Fusion* **2021**, *68*, 85–117. [CrossRef]
13. Vakil, A.; Liu, J.; Zulch, P.; Blasch, E.; Ewing, R.; Li, J. A Survey of Multimodal Sensor Fusion for Passive RF and EO Information Integration. *IEEE Aerosp. Electron. Syst. Mag.* **2021**, *36*, 44–61. [CrossRef]
14. Chung, S.; Lim, J.; Noh, K.J.; Kim, G.; Jeong, H. Sensor Data Acquisition and Multimodal Sensor Fusion for Human Activity Recognition Using Deep Learning. *Sensors* **2019**, *19*, 1716. [CrossRef]
15. Zhang, C.; Lu, Y. Study on artificial intelligence: The state of the art and future prospects. *J. Ind. Inf. Integr.* **2021**, *23*, 100224. [CrossRef]
16. Rababah, B.; Alam, T.; Eskicioglu, R. The Next Generation Internet of Things Architecture Towards Distributed Intelligence: Reviews, Applications, and Research Challenges. *J. Telecommun. Electron. Comput. Eng.* **2020**, *12*, 9. [CrossRef]
17. Pol, M.; Dessalles, J.L.; Diaconescu, A. Explanatory AI for Pertinent Communication in Autonomic Systems. In *Intelligent Systems and Applications*; Bi, Y., Bhatia, R., Kapoor, S., Eds.; Series Title: Advances in Intelligent Systems and Computing; Springer International Publishing: Cham, Switzerland, 2020; Volume 1037, pp. 212–227. [CrossRef]

18. Kelley, D.; Twyman, M. Biasing in an Independent Core Observer Model Artificial General Intelligence Cognitive Architecture. *Procedia Comput. Sci.* **2020**, *169*, 535–541. [CrossRef]
19. Carlson, K.W. Safe Artificial General Intelligence via Distributed Ledger Technology. *Big Data Cogn. Comput.* **2019**, *3*, 40. [CrossRef]
20. Kim, W.; Kanezaki, A.; Tanaka, M. Unsupervised Learning of Image Segmentation Based on Differentiable Feature Clustering. *IEEE Trans. Image Process.* **2020**, *29*, 8055–8068. [CrossRef]
21. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, Portland, OR, USA, 2–4 August 1996; pp. 226–231.
22. Grover, A.; Al-Shedivat, M.; Gupta, J.K.; Burda, Y.; Edwards, H. Learning Policy Representations in Multiagent Systems. *arXiv* **2018**, arXiv:1806.06464.
23. Lo, Y.L.; Woo, C.Y.; Ng, K.L. The necessary roadblock to artificial general intelligence: Corrigibility. *AI Matters* **2019**, *5*, 77–84. [CrossRef]
24. Dmytryk, N.; Leivadeas, A. A Data-Driven Learning System Based on Natural Intelligence for an IoT Virtual Assistant. In Proceedings of the 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Pisa, Italy, 14–16 September 2020; pp. 1–7. [CrossRef]
25. BBC. Nature Makes You Happy | BBC Earth. 2017. Available online: https://www.youtube.com/watch?v=1wkPMUZ9vX4 (accessed on 31 October 2022).
26. Rentschler, I.; Jüttner, M.; Unzicker, A.; Landis, T. Innate and learned components of human visual preference. *Curr. Biol.* **1999**, *9*, 665–671. doi: doi: 10.1016/S0960-9822(99)80306-6. [CrossRef]
27. Makin, A.D.J.; Poliakoff, E.; Rampone, G.; Bertamini, M. Spontaneous Ocular Scanning of Visual Symmetry Is Similar During Classification and Evaluation Tasks. *i-Percept.* **2020**, *11*, 1–12. [CrossRef] [PubMed]