The Immersion of Directed Multi-graphs in Embedding Fields. Generalizations.

Bogdan Bocșe[1 2] **and Radu Jinga**[1 3]

[1]Knosis.AI, Bucharest, Romania
[2]Envisage.AI, Bucharest, Romania
[3]Jiratech, Bucharest, Romania

The purpose of this paper is to outline a generalized model for representing hybrids of relational, symbolic and perceptual-sensory and perceptual-latent data, so as to embody, in the same architectural data layer, representations for the input, output and latent tensors (including from embedings space). This variety of representation is currently used by various machine-learning models in computer vision, NLP/NLU, reinforcement learning which allows for direct application of cross-domain queries and functions. This is achieved by endowing a directed *Tensor-Typed Multi-Graph* with at least some edge attributes which represent the embeddings from various latent spaces, so as to define, construct and compute new similarity and distance relationships between and across tensorial forms, including visual, linguistic, auditory latent representations, thus stitching the logical-categorical view of the observed universe to the Bayesian/statistical view.

latent space embedding | multi-graph | agi data store | ai memory | perceptual associative memory | machine learning database | norm database | normdb

Correspondence: *bogdan@envisage.ai, radu@jiratech.ro*

Introduction. The convergence of descentralized applications, deep learning and semi-autonomous machines is making it mandatory for the Future Enterprise of 2020 to prepare their architectures for the incoming waves of business demand and opportunity. Specifically, the design priorities and hardware restriction which shaped the last 30 years of relational (i.e. often called by the misnomer "SQL") and non-relational database (i.e. often called by the misnomer "NoSQL") history bare little relevance for the data life cycle and access pattern required in the decade of Artificially Augmented Intelligence (AAI*) and "in silico" direct and latent perception.
For this purpose, we explore a set of new concepts for data store and data warehouse architectures, starting from the low-level analytical, functional and canonical considerations of the many embodiment of data (i.e.*entropy*) and data structures (i.e. relationships between various types of entropy). As the general formal model may seem counter-intuitive for conventional business use cases, we shall also provides a few examples and embodiments of the Tensor/Typed-Multi-graph in real-world machine learning scenarios. We shall thus outline a generalized model for representing hybrids of relational, symbolic, perceptual-sensorial and perceptual-latent data, so as to embody, in the same architectural data layer, the symbolic and sensory input, output and latent space representations (i.e. embeddings spaces and/or fields) used by various machine-learning models in computer vision, in NLP/NLU, and in reinforcement learning. For this reason, we are going relevant work from disparate fields, including topology, graph and multi-graph theory, probability distributions and computation theory. This purpose-agnostic design allows for direct application of cross-domain queries and functions, *without requiring human intervention in the application of change requests*. This is achieved by endowing a directed multi-graph with *tensorial or typed* values at least for some edge attributes, which thus become delegate with representing the typed-embeddings from various latent spaces, so as to compute similarity and distance relationships between and across tensorial forms, including visual, linguistic and auditory latent space embeddings.

Definition of Entities. The generic multi-graph $G = G\{V;E\}$ is defined by the typed vertices

$$V = \left\{v | v_{id} = id_v(...); \textit{type(v)} \in T_V; [K_{T_v} \in D_{T_v} \rightarrow O \in D_{K_v(T_v)}]\right\}$$

and the typed edges

$$E = \left\{e | e_{id} = id_e(...); \textit{type(e)} \in T_E; [K_{T_e} \in D_{T_e} \rightarrow O \in D_{K_e(T_e)}]\right\}$$

with the following assumptions of notation:

- V is the set of vertexes, with individual items v
- E is the set of edges, with individual items e
- the function $type(...)$ returns the type of individual edge or vertex
- All $id_e(...)$ and $id_v(...)$ are embodiment-dependent, time-dependant, possibly-stateful, functions that generate pseudo-unique pseudo-random identifiers, with or without causally-dependant validation of absolute unicity. The idenfiers bit arrays generated to be pseudo-unique across both sets V and E, generated by juxtaposing a random number to the nanosecond clock of the machine. To avoid any confusion and to only allow for strictly-type comparisons (i.e. not allowing implicit comparison between edge IDs and vertex IDs), a specific bit-sequence will be prefixed to vertex IDs (ID_v) and a different bit-sequence to edge IDs (ID_e)
- The attribute values associated to the keys K_{T_v} and K_{T_e} are represented as the observable O, which is restricted to its type-specific dictionary

- the set K_{T_v} refers to all keys allowed for the vertex type T_v which is **<defined by>** and **<restricted to>** the vertex-type-specific dictionary D_{T_v}, of possible values.
- the set K_{T_e} refers to all keys allowed for the edge type T_e which is **<defined by>** and **<restricted to>** the edge-type-specific dictionary D_{T_e}, of possible values.
- The dictionary-restriction $D_{\{\ldots\}}$ for each key of each vertex/edge should be seen as **<a set of types>**, including enumerations, primitive types (int32, float64, ...), composite types (object hierarchies), *array*, *matrix* and *tensor types* (including strings, images, animations and higher-dimensional tensors, across a variety of partially ordered and partitioned dictionaries quantization). Let us assume, for the sake of simplicity, that we only allow those types which have **at least one** metric function *well-defined* and *computable in a tractable way*. Example of such metric functions include without limitation: a norm, a distance, a geodesic, a calculable path-finding cost function (including *eg. the Dijkstra algorithm or the A-Star* heuristic*), a definition of difference or variation (finite difference), possibly warped across space, time, or frequency dimensions (*eg. Dynamic Time Warping - DTW (1), Correlation Optimized Warping - COW (2), Levenshtein edit-distance (3)*).

The purpose of this *tensor-typed multi-graph* is to dismount limiting assumptions on the mostly algebraic and logic functions computable by conventional-relational data stores. Concretely, by means referenced and described in this paper, the controller(s) of such a multi-modal tensor-typed graph will be able not only to operate and compute with existing node types and edge types, but will also be able to define new types (classes) of objects and new types of (relationships) between objects, in the pursuit of some meta-objective functions, heuristics or estimators (predictability, safety, performance, robustness), in good awareness of cost and capacity constraints, and with offering the opportunity for human non-expert users to either **vote**, **bet** or **bid** on the changes of the quality-cost trade-off.

Examples of multi-graph instances. The following example represents a multi-graph with two vertexes, v_1 and v_2 :

$$T(v_1) \in T_V; T(e_{1,2,3}) \in T_e; T(e_1) \neq T(e_2);$$

Visualizing the probability-space-time-energy tensor space. The tensor space describing the electromagnetic field of perception can be described as five-dimensional tensor space, with three spatial dimensions, one frequency band (wavelength) dimension and one temporal (causal) dimension.

Riveting the Multi-Graph in Latent Spaces. In order to allow a smooth, continuous trade-off between exact matching

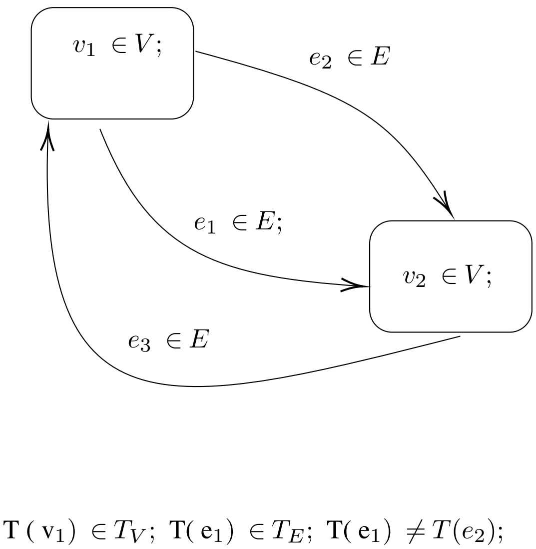

Fig. 1. Several edges of different types between the same nodes, in the same direction

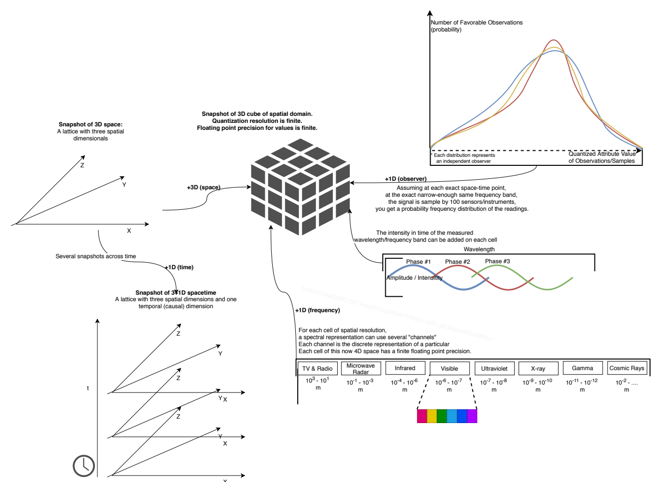

Fig. 2. Representation of a tensorial volume A generalized visualization of the input field for latent space embedding methods. This may include encoding audio, recognizing shapes, signs or faces or other types of more-or-less semantically-constrained <dimensional reduction>

(eg. the exact identity of two integers or the exact identity of all values in an array/string) and approximate, stochastic, probabilistic matching (eg. matching object/scene based on a nearest-neighbour algorithms) we propose a hybrid model, where each node can have one or several attributes.

Therefore, let us imagine that a multi-graph where each node has at least one attribute which represents the latent-space embedding of an input image (represented as a multi-dimensional tensor, such as the pSTEM example provided above). The latent-space embedding may be obtained through whatever computation means, digital, analog or quantum, using some deep neural network architecture or a similar dimensional-reduction estimator and can be assume to be a vector of given, fixed dimension or, more generally, a tensor with a given dimensional shape and local topology.

In this multi-graph, the nodes represent a means of disposing perceived, output, processed or acquired entropy (i.e. the raw data) in some topology that ensures the desired quality

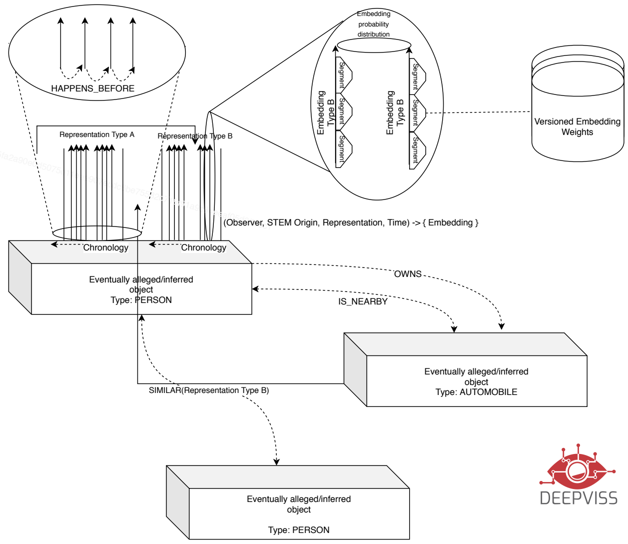

Fig. 3. Example utilization of multi-graph with vectorially- or tensorially-typed attributes for representing the result of video analytics, face recognition and surveillance: structures like the ones are used for internal representation of live face recognition and surveillance solutions, especially of the type-hierarchy and inclusion-structure of. Representation multi-graph building blocks. Each node contains several sets of extracted features, ordered chronologically and grouped by type. Latent space representation may have its own internal type/attribute/latent/residual representation and Coxeter-Dynkin (4) structure.

attributes, such a timely availability of certain type of lookup queries, efficiency of frequent traversals or transversals of the multi-graph, at least alongside specific sequences of node type and edge type.

In the figure below, you can notice a minimalistic non-trivial example of such multi-graph: between any two nodes/vertexes v_1, v_2 there can be several separate instance of relationships (edge, link) e_1, e_2, e_3, each with their direction. It is worth also noting that each of nodes has a type $T(v_1) \in T_V$

The measured Bhattacharyya distance between two histograms corresponding to two samples is looked up in a *big ante/post table* describing the tolerances to be usually expected from that particular sort of sampling (instrument, instrument configuration, measurements of other instruments).

Common types of edges T_e. To make imagining practical and technical applications of the tensor/typed multi-graph more readily-available, we have compiled a short list of edge types (i.e. types of relationships). Please note that some of these edge types are abstract and generic in either one or two executable variable functions. For example, the expression "they are probably brother, they have the same <Shape of face> in <black and white photos>" is one instance of the "IS_SIMILAR_AS_<METRIC>_ON_<FIELD>" relationship/edge-types, equipped with some metric of similarity shared by several observers and observables alike ("shape of face") and with some mention/restriction/indication of the spectral field in which the perception originated ("black and white photos"). Surely, the shape of their faces might not look similar at all given a color photo, a live video or a live video with depth projection. The inexhaustive list of generic edge types:

- **HAPPENS_BEFORE** : chronological, performed by comparison of value of a central, accepted clock or by analysis of vector clocks (5).
- IS_SIMILAR_AS_<METRIC>_ON_<FIELD>
- CONTAINS
- IS_<DIRECTION>_PART_OF
- IS_PART_OF_ / IS_ENCLOSED_WITHIN
- IS_LARGER_THAN_BY_<.ATTRIBUTE>
- IS_SEQUENCED_AFTER_BY_<COMPARATOR( )>
- IS_SEQUENCED_AFTER_BY_<ORDONATOR( )>

The Erdos-Kirchhoff multi-graphs. Typed flow and cargo.. In order to redefine the much needed notion of flow, we enrich a conventional multi-graph with a function of typed-flow of a specific transportable-observable (i.e. cargo) from a node A in the graph to a node B in the multi-graph. As a homage, we label such multi-graph equipped/endowed with a typed-flow function $Flux(A, B, cargo)$, as the *Erdos-Kirchhoff* typed multi-graph The Erdos-Kirchhoff multi-graph can be of two kinds:

- dense, with compact and complete structural topology: all possible vertices exist between neighboring nodes. This can also be thought of as a dense tesselation or a multi-dimensional, non-orthogonal lattice.
- sparse or rare, with no incomplete or inexisting topology, possibly with type constraints

The attachment of the typed-flow function implies the ability to enforce inbound and outbound constraints on throughput using high-level stochastic computational method, in a similar way in which the flow of many electrons is modeled by Kirchhoff's equations. Such duality between dense regions of the space, where full-mesh connectivity is more efficiently processed by a GPU or another similar shared-memory processing-core array, connected by loose/sparse edges/connections leads the way into presenting unified models, which can endure both particle constraints and fields constraints. The fitness of such models for both natural dynamics (fluids, electromagnetic waves, gravity waves) and sentient dynamics (adaptive perception, associative memory, cognition and choice) makes them a more suitable choice of implementation in the agent of machine learning, automated intelligence and generated latent representations. Moreover, the dual nature of the system, which periodically reconciles, mediates and/or any differences between the field and the particle (node) models, thus resulting in higher multi-modal robustness for real-world applications.

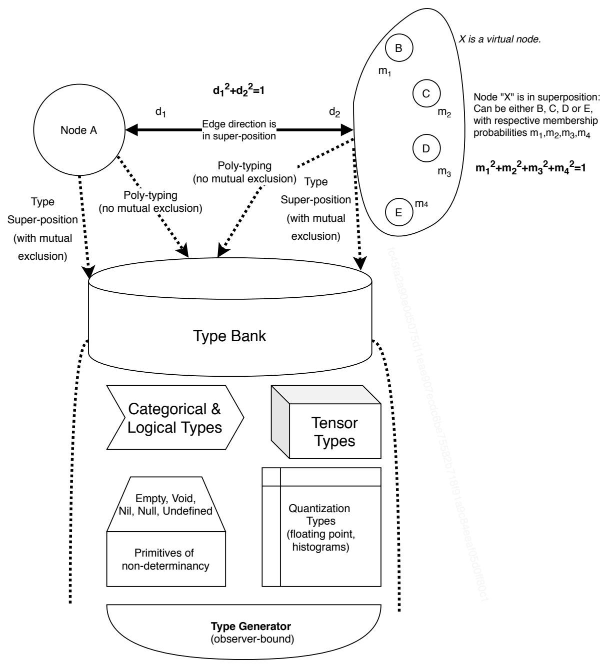

Fig. 4. generalizing the concept of "edge" in a graph from a deterministic object to a richer object, defined by direction, type and identity superposition.

A. Visualizing Edges and Nodes in Superposition. While Node A, on the lefthand side, is deterministic, the virtual node X (which is a super-position of B,C,D,E), with bounded probabilities.

The same principle of superposition can further be continuated to representing the "direction" of the relationship. While in deterministic graphs, the direction is either A->B, B->A or A<->B. In the case of an edge direction in superposition, only a normalizing quadratic constraint (similar to softmax) is placed on "directions" manifested.

A Generalized Model: The Hyper-Histogram . The Hyper-histogram (or "hypergram", for brevity) is a heterogenous, versatile, eventually-partially orthogonal data structured which seeks to use out efficient representations of observable information in a computationally-tractable manner that is also measurable. In plainer words, the Hyper-histogram is a data structure that attempts to maximize ante-post observability (in pursuit of the principle of eventual-completeness of the graph) while minimizing the number of bits required for its representation on the Host Machine. *as a Computationally-tractable Representation of Networked Observations*

Generalized, data structure, with *a posteriori* versioned vary-encoding (i.e. *varible encoding*, also "variencoding", "varycoding") for **typed** cells, potentially varied encoding between cells) and with cells connected either in a structured, partially-structured or unstructured fashion.

First, we need to consider the lattice cell (i.e. cellular structure of lattice), which defines a node or a vertex in the general multi-graph (multiple types of nodes and edges, only second order edges) or hyper-graph (multiple types of nodes and edge, with second-or-higher order edges)

- scalars (as a particular case)
- probability distribution functions (histograms)
- tensors
- and other vector or abstract spaces, with at least a ring structure

Another effect we need to concern ourselves with is the update latency and with the potentially-partial/non-atomic visibility across the changes. Cell update may be "eventually-causally consistent", in a way similar to which the *AtomicLong* and *AtomicInteger* are implemented in the Java specifcation (6): using several causally unmediated (independent) accumulators, which are only *eventually* reconciled into the causally observable global value.
Another matter that concerns us about a computationally-traversable space is to have it endowed at least with some of the attribute that will follow, as properties of topology, measure and norm which ultimately define the structure of the space. One more intuitive way to think about the ramifications of the topological structure of a multi-graph is to

- **metric dimensionality** or quantization dimensionality (addressable attributes, quantized with a certain epsilon-machine function (function which makes known minimum error of the machine))
- **connectional dimensionality** - also defines cell connectivity (connectivity), by enumerating the neighbours with which there is at least one connection
- **fractal dimensionality**, which may be introduced so as to represent variable, non-local levels of connectivity between points. This is a feature of common interest in the proximities (vecinities) of fractal boundaries (frontiers), which seems to pathologically violate our assumptions about the smoothness and uniformity of the unimaginatively "smooth" spaces, such as Hausdorff spaces.
- **curvatures** - defined by variation in angle between edges, per each pair of edge types
- **density** - increasing or decreasing density of cells, corresponding to an decrease or increase in node size and an increase or decrease in edge length, respectively (what is the relationship between density, node size and edge length?)
- cross-field connectivity (typed connections/edges/links between nodes/cells of *different* types) is useful when studying interaction between several fields (temperature, pressure) which have the same underlying topological-structure/support-lattice
- **intra-field (*intra-type*) curvature**, junction tension and differential curvature, as measured between all combinations of 2, 3, ... (up to the degree of the hypergraph, i.e. the maximum number of nodes that can be members of the same edge) of node-type of each edge-type

- link/edge type and edge category; relationship to other categories of edges. This can be useful to model statements, assertions and assumption such as *"most people who <are married> have more sex between themselves"*
- edge symmetry and assumptions of edge-traversal symmetry, so as to be able to model and to represent *non-conservative systems* and intuitions such as *"a two-mile road uphill is longer than a two-mile road downhill"*
- edge normalcity (edge normality). A normal (hyper-, multi-)graph is a graph the structure of which uniformly follows a predefined tile or tessellation. Therefore, in such graph, the topology of which would how such assumption
- node observational spin/whirl, per each dimension or per each edge type, allows us to model and represent the order in which neighboring nodes will observe changes about others. In a simplified way, you can imagine this as local topological parsing order.
- source of entropy for node, which represent the generalized equivalent for the cryptographic keys required for "reading" the node. This can be used for public-private key pairs, random number generators or external vaults of trust (such as a Hardware Security Module), any of them identified by a uniform resource locator, by a point or by some regular-expression or regular-expression-generator, set of materialized functions used by expression and ordered list of seeds (the seed vector, with quantized elements).

While the global tesselation and the local topology of spaces can be, to very wide extent, by using Coxeter-Dynkin diagrams or Schläfli matrices (as defined in (4), it is worth reviewing the intuitionistic attributes of connectivity, dimensionality, density before moving on to pursuing the formal definitions.

Intuition on Equality: the Bhattacharyya distance between probability distributions. Let us consider the Bhattacharyya distance between two probability distribution, defined as:

$$D_B(p,q) = -\ln(BC(p,q))$$

$$BC(p,q) = \sum_{x \in X} \sqrt{p(x)q(x)} \tag{1}$$

Using this distance definition, we can generalize the equality operator ("$\equiv^1$") between value (with scalar or tensor elements), time series, sampled set or other objects or events.

$$v_1 \equiv^1 v_2 <=> D_B(v_1, v_1) \preceq D_{B,threshold}$$

Here $D_{B,threshold}$ defines the threshold of the Bhattacharrya distance which is to be considered for the purpose of postulating the equality ($\equiv^1$) of the sampled values, v_1 and v_2

Let P be the cumulative distribution function corresponding to p over X (this makes the assumption that X is orderer)

$$P(u) = \mathrm{p}(x \preceq u) \tag{2}$$

We have used the "$\preceq$" to denote "less than or equal to", but also "precedes or is equal to".
We compute the $D_{B,threshold}$ using a criteria of overlap (convolution) between P and Q relative to an extra-categorical proof of equality

$$P(D_{B,threshold}) \tag{3}$$

Using this constructive principle defined above, imagine how new relationships, inferable or computable from other sources of data or with less expensive means, can be noticed, defined, validated, so as to structurally-enrich the graph not just with new data, but with new types of data, in a manner which is weakly-supervised at most.

Intuitions of equality: The Dirac-Gauss operator for transforming exact equalities into convoulutions. The Dirac delta distribution is an edge case for a Gaussian (normal) distribution with zero standard deviation. Thus, assuming that $\tilde{A}$ and $\tilde{A}$ are those object $\tilde{A} == \tilde{A}$ as $[x]_{T_1}(A)$
In the general definition of the Dirac delta function

$$\int_{\mathbf{R}^n} A(\mathbf{x})\delta\{d\mathbf{x}\} = A(\mathbf{0})$$

we can notice that reduces the behavior of any function that it is convolved over to the behavior of that function around the origin. This corresponds to the case where equality of objects is deterministically defined by two possible value: true and false.

Hyper-generalizations: Spinor Graphs, Memory-bus Mapping and Path-finding, Chromatic Numbers in Higher Dimensions, Hyper-graphs . Other concepts that help us design data and representation models which are truly useful to sentient we develop as partially autonomous include:

- Spinors in Erdos-Kirchhoff Graph and the particle-observer model are useful for defining the propagation-potential of a change to a node or an edge, considering the fastest propagation time of information in the network (eg. the speed of light c), the smallest "traversal-length" of a graph edge (eg. Planck length L_p), and also considering that each node/particle/element in the network will only observes its neighbourhood in a specific order-hierarchy of iterating through rotational observation/manifestation direction per each pair of dimensions, one time-tick at a time. The spin is this order-hierarchy of rotating the direction for the consideration (receiving, observation) and expulsion (transmitting, manifestation) of energy is the **"spin vector"** (for isotropic behavior in several dimensions) or /textbf"string" (for isotropic behavior in several dimensions) of either scalars (for deterministic behaviors) or histograms (for stochastic behavior). It is observed that, depending on how spin vector are or are not alligned between neighboring nodes, the flow function may exhibit lower values (disalignment) or higher values (alignment).

- The Coxeter-Dynkin diagram (Coxeter graphs, as defined and used in (7)) of memory and data-buses (i.e. "magistrals") as represented in spacetime. By representing the physical manifestation of logical operations (read, write) as a problem of pathfinding in a higher-dimensional space, we can then define the objective functions by which the Host Machine can "choose" between several functionally-equivalent paths of different computational costs (one path might be faster but block more resources and consume less energy, while another path might be slower, but more environmentally-friendly)

- The mapping of two Coxter-Dynkin diagram, one representing *memory layout over time* and one representing bus layout (allocation) over time. For computational efficiency on GPU architectures, it is worth noting that every Coxeter diagram has a corresponding Schläfli matrix. Study the multi-graph correspondence to compact simplex hyperbolic groups (Lannér simplices), to paracompact simplex groups (Koszul simplices), Vinberg polytopes or other hypercompact groups have been explored but not been fully determined.

- The computation of heuristics on chromatic polynomial and chromatic number $\chi(G)$ on higher dimensional tesselations and on multi-graphs. Specifically, it is of interest how does graph coloring expand to tensor-products or lexicographic-products of graphs or other multi-graphs /hyper-graphs? This is of especial interest as Hedetniemi's conjecture, which stated that $\chi(G \times H) = \min\{\chi(G), \chi(H)\}$, has recently been proven to be false, with a counter-example provided by (8). This opens the quest for lower chromatic numbers in structures obtained by joining two collection.

- Generalization to multi-hyper-graphs, which involves the relationship defined by an edge no longer encloses a pair of two vertices, but a tuple involving several edge. Furthermore, here we can distinguish two cases: hyper-graphs with tuple-edges (i.e. ordered lists), which have entropy-potential $O(2^n)$; and hyper-graphs with set-edges (i.e. sets) complexity $O(n!)$. This is useful for representing visual or narrative scenes with multiple subjects and/or multiple/objects, connected by the same co-occurring relationship or action. Use cases include representing relationships known or inferred between words in a sentence or representing relationships known or inferred between objects in a scene (eg. a scene from a comic strip).

Availability, Durability, Replication and Multi-functional materializations. While this work has focused primarily on the logical aspects of storing such multi-graphs, we want to bring to attention some considerations regarding the physical storage, durability, availability, partitioning and replication, as listed below:

- Availability (colloquially "hotness") of primitive node entropy

- Durability is expressed as the probability that the information for inspection after $p(\delta t)t$. Durability increases with replication. A durability that exceeds the horizon of utility by to much runs the risks of producing capacity contention or costs that are too high. For this reason, the process of collection (disposal, destruction, deresolution) assures that objects past their intended durability are deallocated and the "holes" they leave behind are eventually compacted (reduced, defragmented).

- Replication is expressed as the cardinal number of physical location where the information is stored. However, diversity of storage media, geographical location and network topology dependency are also characteristics that can increase durability, especially in the event of catastrophic or *force majeure* events. Excessive replication increases cost, may increase latency for both retrieval and updates (for those operation requiring the quorum or consensus of replica) or it may decrease the likelihood of consistency (for optimistic locks and best-effort consistency schemas).

- Multi-functional Materialization refers to pre-computation of several versions of the primitive entropy, so the latency of the data in its most-frequently used forms (projections) is reduce. In conventional databases, this is often called a materialized view. This end-to-end latency can be considered as latency to functionally-relevant-form, including latency of provisioning and executing any decoding tower/stack.

Discussion and Future Perspectives. In this paper, we have explored methods of expanding the existing node and edge types in a multi-graph using the constraints of a feature extraction (i.e. dimensional reduction) function which takes in as parameter an sensory representation (such as an image or an audio clip) and which is trained/solved numerically using a convolutional estimator and a tripartite loss loss function (i.e. objective function). This method can be used to "stitch together" in typed-relationships (typed edge) the various latent representations output by each of the estimators, thus creating a "middle-ground" between the true/false world of logic and category theory and the smooth, probabilistic view of Bayesian behavior. This "stitching" allows for then creating hybrid ensambles algorithms, some of which are stochastic in natural (eg. deep neural nets) and other are logical/deterministic (recursive enumerable algorithms).

Code and Open Source Initiatives. DeepVISS is a non-profit initiative for standardizing inter-disciplinary data pipelines and integration for machine learning, computer vision and differential programming. You can find instructions on the corresponding websites:

- https://github.com/deepviss-org
- https://deepviss.org/

COMPETING FINANCIAL INTERESTS
The authors are managing partners in several technology companies, as follows:

- Both authors (Bogdan Bocșe, Radu Jinga) are co-founders of Knosis.AI
- Bogdan Bocșe is the managing partner of Envisage.AI
- Radu Jinga is the managing partner of Jiratech

Bibliography

1. Philip Chan Stan Salvador. Fastdtw: Toward accurate dynamic time warping in linear time and space. *Florida Institute of Technology of Melbourne*, 2005.
2. Claus Andersson Giorgio Tomasi, Frans van den Berg. Correlation optimized warping and dynamic time warping as preprocessing methods for chromatographic data. *Journal of Chemometrics*, 2004.
3. Levenshtein Vladimir. Binary codes capable of correcting deletions, insertions and reversals. *Journal of Chemometrics*, 1966.
4. Coxeter. *Regular Polytopes.* Dover, 1973. ISBN 0-486-61480-8.
5. Lamport. Time, clocks, and the ordering of events in a distributed system. *Operating Systems*, 1978.
6. Atomiclong implementation reference. *Java Platform Standard*, 2020.
7. Pavel Tumarkin. Compact hyperbolic coxeter n-polytopes with n+3 facets. *Arxiv*, 2004.
8. Yaroslav Shitov. Counterexamples to hedetniemi's conjecture. *Arxiv*, 2019.