Deep learning framework for Artificial General Intelligence - Part I

Shimon Komarovsky

***Abstract*— Here we discuss suggestions for a framework of artificial general intelligence. We also present a dynamic evolving neural network imitating the evolving human brain.**

I. INTRODUCTION

A. Deep learning current limitations

Deep learning (DL) as the technical implementation of Artificial intelligence (AI) is not fully exploited as it could and should be. First, deep neural networks (DNNs) are passive models, since they have a fixed or static structure, while in reality there are dynamical processes, such as construction/destruction of neurons in the brain. Second, "Learning" in DL is simply a categorization process based on sophistical memorizing, without involving any thinking or imagination.

Third, a successful deep learning model (DLM) requires, from its designers, to know the system, i.e. embedding either an implicit or explicit prior knowledge in the DLM.
Moreover, it is not fair to compare performances of DL to a regular control-based method designed by a human [1] or to the use of statistical methods [2], after their designer have studied the system. It is due to the fact, that DLM know the problem only from a limited set of examples, that frequently do not represent all situations, and it lacks the comprehensive vision as a human has. It implies the necessity of a conclusion-generating AI.
Fourth, DL is a very task-specific strategy. Even multi-tasking in DL is not enough since it is also pre-defined. Real AGI can generalize not only on unseen data, but also on unseen tasks. E.g. as it done nowadays via transfer learning, meta-learning, continual or lifelong learning, etc.

AI is a very wide term with multiple applications and refers to the simulation of a human brain function by machines. AI is classified into two parts, general AI and Narrow AI. General AI refers to making machines intelligent in a wide array of activities that involve thinking and reasoning. Also referred as high-level AI, true AI, or AGI.

Narrow AI, on the other hand, involves the use of AI for a very specific task. Machine learning (ML) [3] is a subset of the Narrow AI that focuses on a narrow range of activities, and is the only one currently implemented and used. ML is the ability of computer system to improve itself from experience without the need for any explicit programming. It can do so by three types of learning: supervised, unsupervised and reinforcement learning (RL).

Data science practical application of ML is to analyze data and make predictions. However it is expressed by a human engineering effort, via extracting relevant insights from data, which involves mostly statistics. Though there are studies try to automatize this effort, e.g. see [4].

Here we suggest a more general type of solution referred as AGI (Artificial General Intelligence) and implemented via DNNs. It should be evolving system, i.e. dynamic and flexible in its architecture. We do not implement it via other statistical AI methods, such as logistic regression, support vector machines (SVM), decision-trees, decision forests, kernel machines, Bayesian classifiers and more [5]. This is because these are all human-derived mathematical inventions, while DNNs, though being also human-derived, are the closest thing we have to an intelligence - since they are inspired by a human brain.

Though there are exist purely symbolic systems, and there are also a combination of them with DNNs (hybrid systems), such as [6], [7], still, this is only the halfway to AGI. In order to make a genuine flexible AGI, a full DNN system should be designed, without any limiting structures within it.

B. Motivation

If we regard human mind as DNNs, then people's consciousness is at the higher levels of this DNN, hence they are good at generalizing and simplifying things. Thus we use it for planning such as modeling and control.

However, we have generalized assumptions upon what we model and how to solve problems. We do not account for the fine details of a specific data set we deal with. Hence, the optimal algorithms we derive are efficient only for very limited cases. Moreover, these general methods search "blindly" for optimal solution, i.e. without considering the specific data we solve. On the other hand we can always learn ourselves the data we deal with, but it will take plenty of time and efforts. Eventually we will get much better tailored solution for this specific data.

Example: you can plan a model and control strategy on a general dataset, based on your general assumptions about it. Alternatively, you can concentrate on a specific dataset, starting with observing and learning it, and only then plan a much appropriate solution.

In conclusion, our motivation is to replace human-engineered specific or general solutions, with a one solution that can be both general and specific. I.e., a solution that can on the one hand generalize and abstract from multiple and different datasets, then plan general algorithms. On the other hand, this solution can narrow itself to a specific dataset at a request, then plan a specific algorithm.

II. Proposed Architectures

In this section we will present an AGI architecture and a DLM proposal, which can function as a module in this AGI architecture, specifically in the perception or actuation parts of it.

A. Proposed AGI model

A general sketch of the first draft describing the AGI model/structure is shown in Fig. 1.

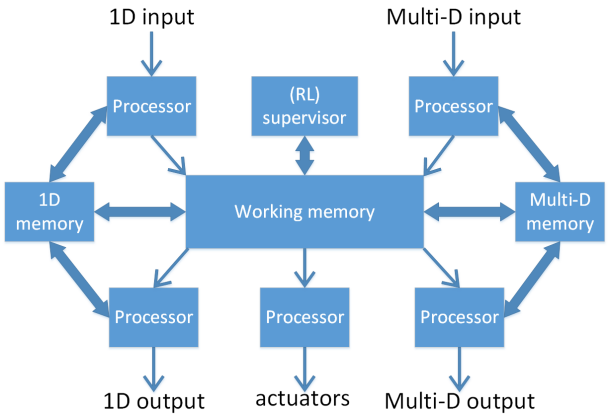


Fig. 1. AGI possible architecture.

This AGI is based on global workspace theory (GWT) [8]–[10]. It has no competition among its different and independent modules, i.e. processors and memories. Instead it has centralized control with different elements, where each element has a specific functioning.

1) Cognitive Architecture:

The diagram in Fig. 1 is also referred as cognitive architecture, which represents an AI agent structure and functioning. For example, in the traffic control field we have two studies [9], [11] implementing AI based on distributed *cognitive* architectures [12].

The first study [9] is based on Global Workspace Theory (GWT), which contains agents imitating four types of brain functions: (i) sensory type, which holds positions and velocities of vehicles; (ii) behavioral type, which determines the light of the traffic signal in a particular intersection; (iii) consciousness type, which represents the working memory and interacts with other brain functions; and (iv) motor type, which executes the chosen signal phase for each intersection.

In the second study [11], an AI is implemented on a single intersection using a Multipurpose Enhanced Cognitive Architecture (MECA) [6], which was adapted for traffic signal control problem. The design consists of Cognitive manager, a special kind of agent managing a set of physical objects made available at Internet, providing information about themselves and receive commands. E.g. car or an intersection agents. MECA is composed out of two independent systems that communicate with each other: (i) a fast reactive system that holds the input sensors and output actuators, which is suited for normal situations, and (ii) a motivational system which is goal-oriented and suited for unexpected situations. It is thanks to automatic reactive and the conscious elements, which together imply an intelligent behavior.

The papers use simple cases of small scale networks and they are rule-based designs. We, on the other hand, strive for a more adaptive and flexible designs, hence the use of DL instead of rule-based design. Consequently, we hope that this approach will achieve better learning by the agent, though certainly it will be guided and supported by humans, as it is occurs in infant-parents and students-teachers interactions.

We should note that any cognitive architecture is limited and constrained by its structure. Therefore, we should consider a more open view on this structure, thus considering the boundaries between the components to be not so well defined and perhaps changing.

2) AGI functioning:

If we mimic human intelligence, then similarly as we have 1D (audio) input and 3D (visual) input, we will use those as out inputs also. Although we do not limit ourselves with 3 dimensions for spatial type of input. It can be generally 1 or more dimensions, dependent on the environment our agent is deployed in.

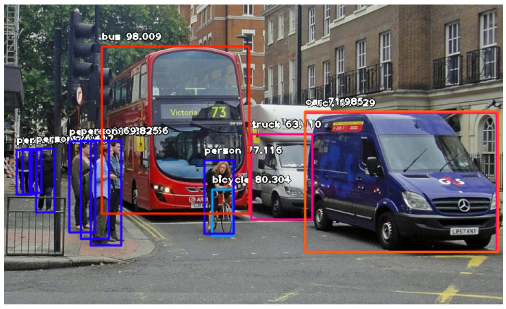
There is sequential processing of 1D and multi-D data separately, for feature extraction and categorization of objects (static entities) or events (dynamic entities). Eventually these objects propagate into the working memory. In case a request or a need to produce some output by the agent occurs, such as in case of some idea or thought emerging, then the output can either go through the 1D channel, as humans describe their inner thoughts to the outer world verbally, or through a multi-D channel, which is an extension of human vision (if it is possible at all), and can be regarded as "screening imagination", which is like projecting the current thought into a screen. Because if we have two types of inputs, a legitimate question arise: why should not there be outputs of these types also? Additionally, 1D information have some common memory for both input and output and the working memory (such as language), which is here presented as 1D memory. Likewise we have for the multi-D information. The bidirectional arrows represent the acquisition (reading) and the update (writing) operations with the storage module.

The output communication of 1D and multi-D information is also an issue. Whether it should be monitoring thoughts, or to wait for a meaningful output, or perhaps the AGI might have some sort of free-will to choose when to interact and via which of the two channels.

Notice, that this particular AGI is based upon Stimulus-Response behavioral theory [13], that claims that we cannot observe the mind itself, directly, but rather only communicate with it. This assumption is of course the worst-case scenario, similar to the Chinese Room Argument, since it means that we have no direct access to the operations within the agent, but only to its outputs. I.e. considering the intelligence as a black-box, as it is in DL nowadays. Or in other words, we have no explainability over the inner operations of intelligence. We have only output, which we can analyze. It is also called "intelligent behavior", and it is expressed via

human productivity over the history in numerous scientific fields, such as in: engineering, math, spiritualism, computer sciences, education, psychology and more.

Notice that AGI is more than merely static objects identification, as it is in DL. It extends to temporal dimension, where it is not only about objects, but also about events, objects' meaning and predicted behavior, associations from past experiences and more. It is illustrated in comparison between current AI (especially DL) and AGI, in Fig. 2.



(a) AI identifying.

A room in the morning, with a dining consists of 2 chairs and a table. A closed window with a building view outside. The cat sit relaxed, at some point will jump from the table.

(b) AGI interpretating.

Fig. 2. Comparison between AI and AGI comprehending the environment.

3) Two types of information:

In our presented AGI model, we thought that just as we have two channels of input: vision and language, similarly we can extend a design of AGI to have these two channels also as outputs. So that we can see its inner thoughts.

And then we wondered, why do not we humans have imagery tool as output? One can argue it would hurt our basic desire for privacy, but then just as we choose weather to talk or not to talk, we can similarly choose when to turn this tool on and when not.

Our current opinion is that the world we see with our eyes is what we all agree upon. Other then that our inner models of the world are totally different.

And why do not we have symbolic or linguistic channel to be objective as vision? Why we end up with inner individual and unique symbolic representation? We think it is because language is highly context-dependent, and since each person has different contexts along his life, or different experiences, then he/she develops a different meaning/feeling/understanding of the objective concepts we are all agree upon. Concepts of a language we use to communicate one with the other. Hence, the concepts we use in external communication are objective and common to all people, but their interpretation is different for each one of us, depending on the specific sequence of experiences each one goes through in his/her lifetime.

Therefore physical reality function is only for the objective agreement for an effective communication between us, which is realized via language. However vision is not a communicative channel for us.

Consequently, the purpose of having two types of channels is for distinguishing the outer and inner world that the agent interacts with. The outer world is objective, and the details in it can be agreed upon all AGIs, through the visual multi-dimensional channel. But each AGI is individual in the sense that it builds its own special and unique inner representation. But furthermore, humans (as should be followed by AGI) base their inner representation on spatio-temporal events, or operationable language (i.e. objects and actions/attributes), which can expressed by words or any symbolic language. This is why spatio-temporal information can be transfered to humans not by the objective world (static visual information), but rather by language, which in our case is 1D sequential type of data. Agents denoted as green circles, communicating via 1D and individually perceive multi-D input are illustrated in Fig. 3.

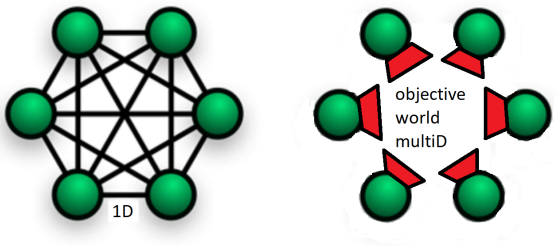


Fig. 3. Objective (right) verse Subjective inner representation (left).

Notice, that supervised (labeled) DL is actually a bypass of the actual intelligence development. Since an infant starts from unsupervised learning, and only after a long time period can make connection between the objects he have seen so far and their meaning (either vocal or symbolic). We continue this line of thought in the next subsection, where we present our DLM.

B. Proposed DLM

Until now we presented a general model for AGI. Now we turn to discuss which DLM can implement such AGI, or implement each or some of its different modules.

Before we dive into our proposed DLM, we introduce some necessary background, starting from general prior

knowledge in DL, then due to difficulties with matching the most proper prior knowledge to each specific problem we encounter - we continue with Network Architecture Search (NAS) that attempt to deal with these difficulties. Then finally we present our DLM as another possible NAS strategy.

1) Prior knowledge in DNNs: As shown from the above studies in the filed of traffic signal control [14]–[18], CNN and RNN present better prediction results in accuracy and stability compared to other methods, such as support-vector-machines and random forest. However, this might be due to being tailored to the specific problem, which makes them best due to specificity and particular planning. Apparently, CNN and RNN are examples of simplification of the general NN, since they are more intuitive and understandable, with the price of being task-specific [14].

However, these special structures have less variables and contain more prior knowledge, compared to fully-connected (FC) layers of regular (vanilla) NN. Hence we can see the transform from FC to CNN or to RNN as localization, where the network structure designed specifically to the data it handles. Another example we is graphical input, which requires an appropriate graph NN model to handle it.

Prior knowledge can appear in many forms: in the general type of the structure (e.g. different DLMs), in the hyper-parameters, in the regularization method (e.g. dropout, constraints, etc), in the decision about sharing or grouping [19] or separating features/variables and more. E.g. when we set apart traffic data from weather data [20], or roads from stations [21] and fuse them only later (how later is also prior knowledge to be decided upon). We can also separate tasks to groups [21]. [22] suggests sparsifying the NN (e.g. removing connections in CNN, or grouping/sharing parameters) - results with less parameters and more prior knowledge and efficiency, similar to the effect of hand feature engineering. It also removes redundancy.

However all these structures can be too restricted or best perform for a narrow data variations. Hence many studies try different hyper-parameters to get better performance. A more comprehensive topic where such more flexible models are studied called Network Architecture Search, e.g. AutoML [23], Neuroevolution, hypernetwork, Meta-learning (learning over learning) [24] and more, which are all searching for an architecture or adapting a given architecture, to better fit the data.

2) Network Architecture Search (NAS): In Network Architecture Search (NAS), such as AutoML and others, we first define/restrict the search space for models. Search techniques that are used are usually RL or Evolutionary Algorithms (EA) and more. E.g. in hyper-parameters tuning often used what is called grid search and random search.
A more efficient (faster) approach use weight sharing between proposed networks, instead of putting them in the search and training them from scratch (such as in EA). They do it by sampling sub-networks from a large parent network where they force all sub-networks to share weights. Only the best final model is trained from scratch. Alternatively, we can modify only some specific part in NN, while leaving all rest unchanged.
Unlike previous EA and RL, in Differentiable Architecture Search (DARTS) [25], the search is defined as differentiable and continuous problem. DARTS have some unknowns so they use multiple categories, where the final architecture discretized after the search is over.
Hypernetwork [26], is where the weights are learned not by training but via other network. It means that for every input you test, there will be different weights to predict the output.
In one paper [27] there is hypernetwork for both CNN and RNN as two ends of a spectrum, which allow it to be relaxed weight-sharing approach and allows us to control the trade off between the number of model parameters and model expressiveness.
In our opinion, Hypernetworks can be extended by constructing also second network to learn weights of the first one that learn weights for the main one. And so forth, strengthening the evolution in different time resolutions/scales (from very slow to fast).

When we use EAs for NAS it is referred as Neuroevolution [28]. Beyond being used to generate DNNs, it is also used in generating hyperparameters, NN building blocks, activation functions and even the algorithms for learning (rules). E.g. in NEAT (NeuroEvolution of Augmenting Topologies), it is used to replace back-propagation with Genetic Algorithm (GA) or combine them both, for searching of weights.

In SMASH method [29], they train a hypernetwork to predict the weights of a network in one shot instead of training all those candidate networks from scratch. They first train a hypernetwork to predict the weights of some given arbitrary network, then they randomly generate many architectures, and rather then training those models, they use a hypernetwork to obtain the weights. Finally they pick the architecture that performs best with those weights and train it from scratch.

In the following section we will present evolving DLM as additional NAS method.

3) detailed proposed DLM: Since we live in spatial-temporal reality, the AGI should be processing spatial-temporal information. Therefore, we present a demo of the DLM, for a time sequential visual input, and describe its evolution through the eyes of a new born infant. Similar models can be found in [30].

We presume, that infants do not have any supervised learning at the beginning of their lives, but an unsupervised one. First thing she/he does is segmenting the time period into simple events. But he starts with a single event (e.g. he's/she's total awaking period) detection through some initial DNN, see Fig. 4(a).

After a while, when enough counts detected the single event, a split performed of this event into two (or more) classes of events, e.g. day and night, see Fig. 4(b). By counts we mean the number of times the output class was

triggered. Now it can differentiate two events, sharing the same deep features. Later he/she can extend the number of events, and recognize as many events as necessary (adaptive NN structure, adaptive by necessity).

At some point of evolution, when connections (weights in DNN) and event identification (DNN's output layer counts) is strengthened and established, the model can change its attention (or free its resources, since the given level became more automatic, similar to the idea in [31]). It can build a new layer or level, if a simultaneous re-occurrence of several events is detected. E.g. the re-occurrence of seeing mom appearing and prepares herself to give milk suggest the infant that it is a composite event on its own, see Fig. 4(c).

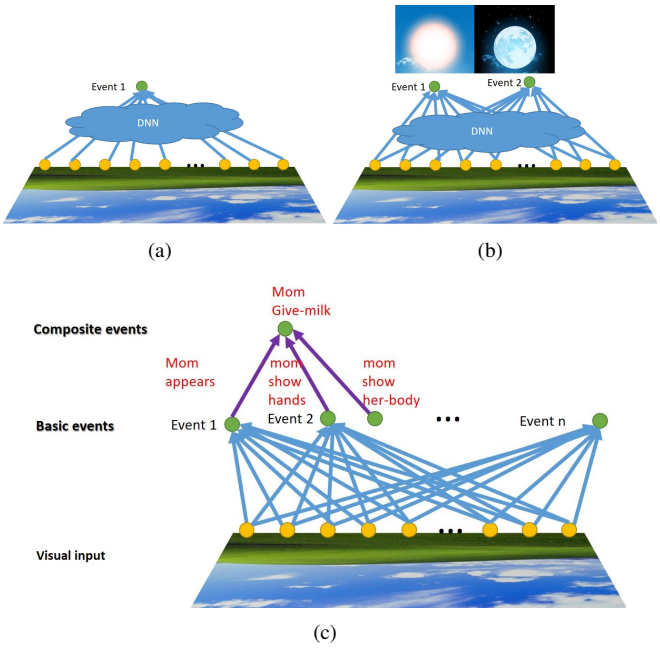


Fig. 4. Separation and composition of neurons in proposed approach.

An opposite structure-changing operations could be (i) deleting extremely rare nodes/edges in the DNN, abit similar to dropout regularization in DL; and (ii) decomposing an event, if it appears to be more complex than it was supposed to, i.e. turn it to be fine-grained. In other words, if previously it was treated as a specific-level event, now it is decomposed into a more simpler events (refinement). Either decomposed into existing events or new ones. If new ones, then they have to be attached to lower level events/features. E.g. see Fig. 5, where we decompose the "ball in the air" event into its basics: throwing, moving in the air, and being caught.

It does not have to be risen up to the higher layer, but it can be stay in the same level and be decomposed into lower levels. These both options may be available for reconstruction operators, because sometimes it is better to stay in the same level, and sometimes it is better to level-up.

Suggestion: In decomposition we create new events anyway, because we do not know which of them are actually new and which are existing already. Therefore, if two or more events from the same layer, occur simultaneously very

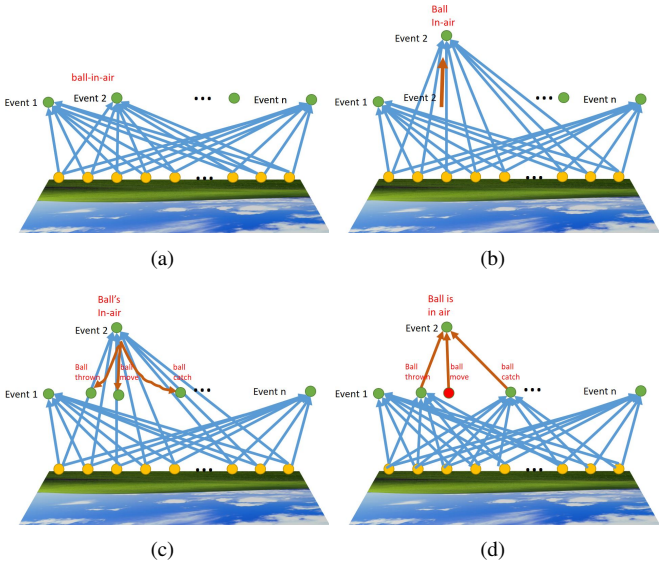


Fig. 5. Decomposition of neurons in proposed approach.

often, then it might imply that they are all the same event, hence should be eliminated, all except/leaving one.

This approach is self-supervised and not unsupervised, since we do not perform clustering into pre-defined number of categories. Number of categories, connections, and etc - are all dynamic, similar to NAS, and changed by the decision of some supervising algorithm.

Another reason for this dynamic algorithm is that real intelligence do not end up with categories like cat, dog, etc. Therefore it has to be evolving.

For this dynamic algorithm to work, we have to store number of visits of each weight (edge in DNN) and of each node in DNN. If scalability is an issue for large DNNs, we can reduce the visits memorization from storing for each neuron to a clusters of neurons in a large enough DNN.

Furthermore, the visits can occur during "waking" periods (when the DNN is fixed), and the structure update can be done during "sleeping" periods, when there is no stimulus from the sensors, while the frequency of trajectories within the DNN is stored in the neurons themselves, as mentioned above (perhaps even might be in the actual human brain). The rate of changing structure can also be modeled with a learning rate as in RL, where at first is mostly exploration (i.e. fast NN growth), and than lesser exploration and more exploitation. Finally, we can assume a finite number of nodes and connections, i.e. limited resources (so that it will not grow infinitely), and adjust the learning rate according to these constraints.

Another issue can be of recoverability: what if in the future we would regret some of the elements that were deleted from the DNN, can they be restored? One way to deal with it, is by the assumption that the infant first should grow up, and refine its categorization, i.e. we should start with only enlarging the network. Only in a big enough network, we can start deleting nodes/connections, cause then the agent accumulated enough confidence/experience.

It is important to note that in the common DNN the structure suppose to contain features as neurons, but these features depends on the data. Hence the necessity of checking several structures, as in NAS, since we probably wrong about guessing the number of features at each layer. Dynamic NN deals with this issue, by keeping only the relevant and true features.

One issue in dynamic structure NN could be splitting or deleting features that are supposed to be frequent or rare, and we should leave them as they are. One way to deal with it is via relative frequency, i.e. to have some min-max range of relative frequency among neurons (all of them or of each level or something else), which will not be splitted/deleted. Only those extremely frequent/rare outside of this range would be splitted/deleted.

This dynamic NN structuring algorithm (composing and decomposing) is like the right brain or the supervisor over the learning block, the DNN itself (which is like the left brain). This dynamic algorithm enables efficient and evolving knowledge and ordering. It has two main advantages over the FC NN: (i) It is less computational since it has less connections, similar to sparse NN; and (ii) It solves the problem of how to define in advance the NN structure for a given problem (number of neurons, number of layers). It is adaptive in nature and efficient in the sense of having only the necessary neurons in its structure.

Some optional additions to the model:

- A task/output in this demo model is an event, but it is a more general concept, hence it could be any task, such as an object or an action.
- We can probably update also the inner neurons, and not only the outer/expending ones.
- Also there is an option to include reallocation of activated-together neurons in the model, such that they could be in a similar neighborhood, which can probably ease on computation, and anyway this is a good reordering.
- We should constrain the model in all its adaptive parameters, such as number of total layers, number of neurons per layer, and so on. First of all due to our computation limitations (in time and storage). Secondly, as a regularization over the evolution, make the model "feel" that its resources are limited, hence it have to keep only highly necessitous decisions. I.e. eliminate redundancy and encourage competition/trade-offs. E.g., learning rate and constructing new neurons and layers will be scaled accordingly to the given limits.

In addition, the idea of brain plasticity and distribution of tasks over many regions in the brain, may imply that neuronal information transmission (chemical and electrical) is not the only operation type that occur in an active human brain. I.e., similarly to cell division and reproduction then assignment to different roles stated by the genome, it may give neurons different and perhaps adapting roles. That is, besides the adaptation and reconstruction of NN architecture, the neurons themselves can change their function, e.g. become convolutional, recurrent, recursive, attentive, etc.

4) Hierarchy of tasks in the DLM: Note that the extending of classes convert this DLM into hierarchical multi-class DNN. Such DNNs exists in literature. In [32] we have layers for tasks and then layers for tasks. Sometimes these layers can be mixed. There are those who find labels structure separately from the model [33], [34], and there are those who do it as a part of the model [35]–[37]. There are those that put all the tasks to be learned over one classifier, i.e. globally or in the last layer of the NN [34], [38], and there are those who insert intermediate tasks inside the NN, i.e. locally [32], [35], [39], [40]. There are those that learn the structure from the data [36], e.g. by unsupervised clustering of the labels via some similarity measure [33], and there are those that import the structure from external knowledge base [41], or use it to adapt this structure [34], [37].

Important note: unlike features that are a distributing representatives of data, which hold only some piece of the actual information, tasks are end-point independent representatives of data, thus ruining in a way, the distributive nature of NN. However, this is not their main drawback - the fact that they are informational points - enforce a huge memory, since we need lots of them to represent huge amount of terms and concepts. As opposed to a small group of inter-related features, which can characterize enormous amount of input data. Therefore, at some point we need to consider to replace or turn specific-task type of learning into NN of features.

Generally there may be different hierarchies besides compositional ones (as compositional events). E.g. family tree, parts of speech, table of contents, topics and sub-topics, etc. We propose the evolution into different hierarchies, just like tree expansion: in different locations of a given NN and in different structures. An illustration of multiple hierarchies formed in a given NN is in Fig. 3.

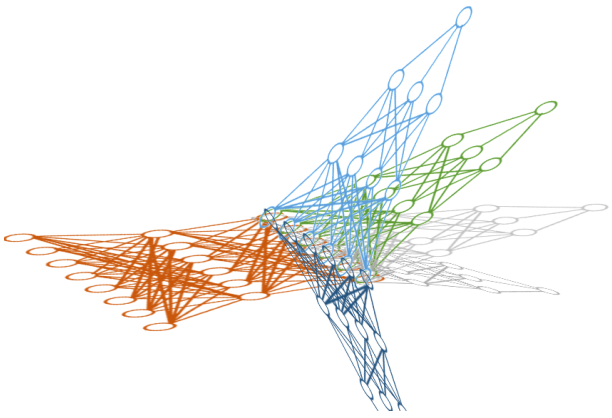


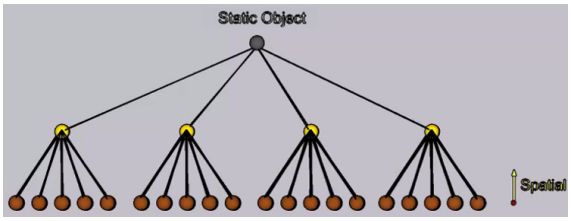
Fig. 6. Branching due to multiple hierarchies.

5) temporal dimension of the DLM: Until now the DNNs we have seen in the proposed DLM use regular static structure, i.e. single simultaneous set of inputs produce single simultaneous set of outputs, or in other words we have not seen any recurrent connections to include temporal sequence of inputs.

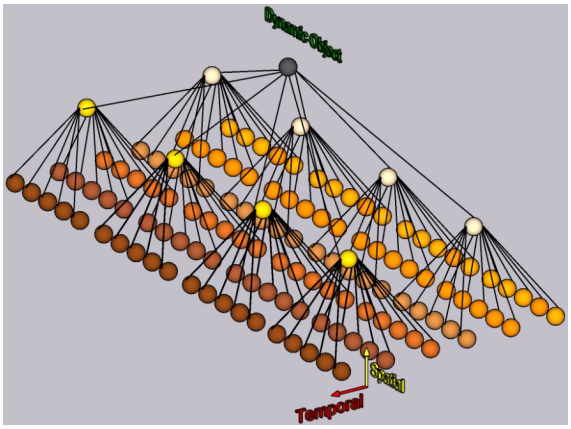
Known spatial-temporal models combine CNN with RNN in a different ways. Either

separately: CNN→RNN or interchangeably: CNN→RNN→CNN→RNN→CNN→RNN→CNN... Other way: seperate them (CNN and RNN) for separate inputs, e.g. textual for RNN and visual for CNN, with fusion module at the end. Hence event tasks, such as feature extraction or classification/clustering, can be done using these methods above.

On the other hand, we know that regular FC DNNs, are used for spatial object tasks, e.g. clustering or classification. But If we want to detect and extract features along also the temporal dimension, we can simply add recurrent connections to these DNNs. Alternatively, we can extend the DNN to include this dimension, without changing the spatial dimension, i.e. orthogonal to it. See Fig. 7.



(a) (Front view) Objects tasks, e.g. classification



(b) (Side view) Events tasks, e.g. classification

Fig. 7. Spatio-temporal DNN model.

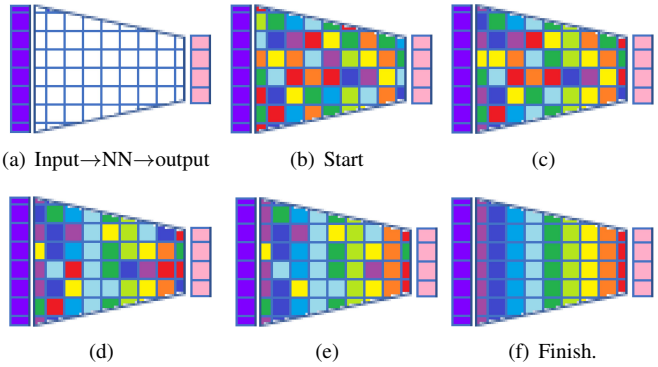
In Fig. 7 is a FC NN. However, if we want - we can specialize it in different ways, e.g. shared connections/parameters or convolutions, etc. And we can do it for either of the spatial or temporal dimensions or both.

III. Additional guidelines for an efficient AGI framework

A. Order

1) Organization of Data: There is an issue to define the problem of AGI. Unlike regular classification problem well defined in DNNs, it is difficult to know what is the overall purpose of AGI. We believe for now, that it is to better predict or/and organize data. However prediction by itself is only the test for how organized data is. So perhaps prediction is not the goal of the supervisor, but rather only its tool (inner or secondary objective) to estimate how well the organization is (which strive to low entropy).

See for example in Fig. 8, a sketch diagram, illustrating how supervised learning in NN, where the data given is input and output. The NN is structured hierarchely, i.e. it is features of features, etc. So starting from random dis-ordered weights (represented as rectangles inside the DNN), we gradually use inputs and outputs as magnets, for diffusion or rearranging the weights in hierarchical way, where the most input-related features will be closest to the input and the same for the output. This idea is inspired by the visual feature maps in CNN.



(a) Input→NN→output (b) Start (c)

(d) (e) (f) Finish.

Fig. 8. Evolution of the parameters in supervised learning in NN (from right to left).

This means that in our opinion, AGI is not an open-ended or an objective-less system, but it is actually have inner objective, such as organizing data in a utilizable way, learning react to the environment, and more.

DNNs are associative structures, which makes them very efficient tools to search for elements. On the other hand, when we represent data in a tree-shape structure, then for different problems we must apply breadth-first-search (BFS), depth-first search (DFS), heuristic types of search and so on. However these methods are only rarely efficient, since we do not know which of these methods best fit for a given data. I.e., these methods work blindly on unorganized data. While the brain do the opposite, it solves problems directly, after organizing data. That is why it is important for a good organization of data, prior to implementing any search method. DNNs in our opinion is the best tool, among other ML methods, for organizing the data to be retrieved efficiently later. E.g. association in NNs characterized via the multiple features an object has. Even if only small amount of these features are triggered, it will track this object. Hence, a solution is not optimal generally, but it is the best for the current data organization. Another example, is when a person thinks of a several solutions for a problem, then he/she is not do it via blind search, but via associative direct processing (no permutations involved at all). We have a survival type of processing; it cannot depend on time-consuming search in some space for solutions.

This is part of our more general vision in AGI, where data should be always reorganized, in order for the AGI agent to

be most efficient in solving multiple tasks, considering past tasks and perhaps future tasks too. E.g. in [42] there is self-supervision which allows predicting any masked data from an observed data. This demonstrate the idea of data being organized such that it can be tackled in multiple forms and scenarios.

2) Order in teaching: Order is important not only in the data itself, but also in the sequence we provide it to the AGI agent. Similarly in humans, it has to be from the simple to complex for example.

In DL the learning can be either incremental [32], [43] or simultaneous [39]. We should be aware of the catastrophic forgetting phenomenon in incremental learning [44], where the model abruptly forgets part of the knowledge related to a previously learned task as a new task is learned. However, [43] used successive regularization strategy to avoid this phenomena.

3) Growth Potential: Growth is important property of AGI, that distinguish it from other AI methods such as rule-based AI or static structures such as NN. It is the very important task of AGI designer to plan the AGI agent such that it can evolve and develop further independently and autonomously. Piaget theory [45] suggest that there are stages in child's development. Moreover, every stage is necessary as the basic level from which next level can be reached. Similarly, we anticipate the AGI agent to reach some stages of growth.

We see that current DL methods are working backwards: they start from highly complex data, such as language and/or highly complex visionary scenery, and try to process it, with the intention that this data (input and output) would be approximated by some mapping function, but neither understood nor follow the simple-to-complex rule as it should be.

B. Efficiency verse Effectiveness

We should not construct AGI based on cumbersome or complicated models, since it may hurt the model's further development, as it was understood decades ago in rule-based modeling. And we have to build an AGI such that is not expected to be stable or perfect right away, immediately at first execution. This is a wrong attitude towards actual intelligence.

On the contrary, we have to give it the time to develop, just as an infant baby or a child do, and not demand from it to give always "correct" answers. On the countrary, just as a human does, we should allow the AGI agent to make mistakes, based on partial understanding, and learn from it not as a new input in DNN, but as another step in a more general and mature step of development. I.e., as a more general point of view upon the data it encountered during its lifetime.

We can see resemblance to this idea in the comparison between serial thinking verse parallel one [46], [47]. In serial thinking we prefer a conclusive judgment and fast results with emphasis on certainty. In parallel, however, it is about being fluent and non-judgmental in favor of fluency and multiple solutions/options together with their probabilities. I.e. allowing and embracing uncertainty instead of fighting it. The same is here, model-based methods and control are mostly designed for fast good results, to prove effectiveness. But human intelligence shows, that it takes years for an infant to gather linguistic capabilities and fine motor sensing. It takes many months, in which the baby is mumbles or pronounce poorly and have a gross motor skills (i.e. in movement and drawing). This observation support the idea that the more AGI is general, in dealing with as variant knowledge as possible, the more efforts it requires to adapt and learn this knowledge. And the opposite is true also - the more the AGI is specific, like current control methods, than the more it fits to be effective in narrow sets of data and it is faster in results.

This is actually the difference between efficiency and effectiveness [48], where efficiency concentrates on the best exploitation of available resources, while effectiveness is about performance measure of how well the goal is achieved. Consequently, the AGI agent must be efficient more than effective, since we are less interested in some specific desired outcomes, but rather a good thinking machine which can be validated only on the long run.

C. Final words

Our simple belief is that humans are the only ones that have consciousness and experiencing the world via both feelings and mind. We do not think AGI can be built as a living agent just like humans, and even if do - we do not think it is an issue. Our view of an AGI is simply as a processing unit. We do not mind if it do not understand as stated in the Chinese room argument or if it does not have consciousness or feelings. All we care about is that it can solve problems and be creative. Its purpose, in our eyes, is to act as an engineer or a researcher. Hence, it does not matter if it process data, while humans perceive it as knowledge. What matters is that we know the way humans know. When the AGI agent is instructed to do something, it process our request, and finally its output return to humans.

In our opinion, solving separately intelligent aspects cannot be eventually assembled into a full AGI agent. I.e. we cannot assemble it from pieces, e.g. from modules originated in narrow AI. Instead, we should account for all aspects right from the beginning, because it is a holistic system. This approach has been advocated by many neuro-science studies, which showed that they could not locate separable operating regions for different tasks, in the brain.

Additionally, it is difficult for us to know how human intelligence actually works and particularly how it evolves, because we acknowledge it only in its final state, at adulthood. We have no access to early stages of its development, certainly not directly, e.g. how a baby or a child sees the world internally. We can only analyze it implicitly, via external measures, such as experiments.

IV. CONCLUSION

Most important takeaway from this paper, is the importance for holistic or complete attitude towards AGI. You should not construct it from parts.

REFERENCES

[1] B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," *Journal of Transportation Engineering*, vol. 129, no. 3, pp. 278–285, 2003.

[2] M. G. Karlaftis and E. I. Vlahogianni, "Statistical methods versus neural networks in transportation research: Differences, similarities and some insights," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 3, pp. 387–399, 2011.

[3] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[4] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, vol. 521, no. 7553, p. 452, 2015.

[5] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[6] R. Gudwin, A. Paraense, S. M. de Paula, E. Fróes, W. Gibaut, E. Castro, V. Figueiredo, and K. Raizer, "The multipurpose enhanced cognitive architecture (meca)," *Biologically Inspired Cognitive Architectures*, vol. 22, pp. 20–34, 2017.

[7] G. Granlund, "A cognitive vision architecture integrating neural networks with symbolic processing," *Künstliche Intelligenz*, vol. 2, pp. 18–24, 2005.

[8] R. C. M. da Silva and R. R. Gudwin, "An introductory experiment with a conscious-based autonomous vehicle," in *4th Workshop in Applied Robotics and Automation*, 2010.

[9] A. L. O. Paraense, K. Raizer, and R. R. Gudwin, "A machine consciousness approach to urban traffic control," *Biologically Inspired Cognitive Architectures*, vol. 15, pp. 61–73, 2016.

[10] J. A. Reggia, "The rise of machine consciousness: Studying consciousness with computational models," *Neural Networks*, vol. 44, pp. 112–131, 2013.

[11] R. Gudwin, A. Paraense, S. M. de Paula, E. Fróes, W. Gibaut, E. Castro, V. Figueiredo, and K. Raizer, "An urban traffic controller using the meca cognitive architecture," *Biologically inspired cognitive architectures*, vol. 26, pp. 41–54, 2018.

[12] A. Lieto, M. Bhatt, A. Oltramari, and D. Vernon, "The role of cognitive architectures in general artificial intelligence," 2018.

[13] J. B. Watson and P. Meazzini, *John B. Watson*. Il mulino, 1977.

[14] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transportation Research Part C: Emerging Technologies*, vol. 90, pp. 166–180, 2018.

[15] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, 2017.

[16] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[17] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, "Lstm-based traffic flow prediction with missing data," *Neurocomputing*, vol. 318, pp. 297–305, 2018.

[18] M. Aqib, R. Mehmood, A. Alzahrani, I. Katib, A. Albeshri, and S. M. Altowaijri, "Smarter traffic prediction using big data, in-memory computing, deep learning and gpus," *Sensors*, vol. 19, no. 9, p. 2206, 2019.

[19] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference*. Springer, 2018, pp. 593–607.

[20] A. Koesdwiady, R. Soua, and F. Karray, "Improving traffic flow prediction with weather information in connected cars: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9508–9517, 2016.

[21] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: deep belief networks with multitask learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, 2014.

[22] Y. A. Ioannou, "Structural priors in deep neural networks," Ph.D. dissertation, University of Cambridge, 2018.

[23] AutoML. Automl. [Online]. Available: https://towardsdatascience.com/how-to-apply-continual-learning-to-your-machine-learning-models-4754adcd7f7f

[24] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-sgd: Learning to learn quickly for few-shot learning," *arXiv preprint arXiv:1707.09835*, 2017.

[25] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," *arXiv preprint arXiv:1806.09055*, 2018.

[26] T. Galanti and L. Wolf, "On the modularity of hypernetworks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 10 409–10 419, 2020.

[27] D. Ha, A. M. Dai, and Q. V. Le, "Hypernetworks," *CoRR*, vol. abs/1609.09106, 2016. [Online]. Available: http://arxiv.org/abs/1609.09106

[28] E. Galván and P. Mooney, "Neuroevolution in deep neural networks: Current trends and future challenges," *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 6, pp. 476–493, 2021.

[29] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Smash: one-shot model architecture search through hypernetworks," *arXiv preprint arXiv:1708.05344*, 2017.

[30] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[31] J. Hawkins and S. Blakeslee, *On intelligence: How a new understanding of the brain will lead to the creation of truly intelligent machines*. Macmillan, 2007.

[32] R. Cerri, R. C. Barros, and A. C. De Carvalho, "Hierarchical multi-label classification using local neural networks," *Journal of Computer and System Sciences*, vol. 80, no. 1, pp. 39–56, 2014.

[33] Y. Papanikolaou, G. Tsoumakas, and I. Katakis, "Hierarchical partitioning of the output space in multi-label data," *Data & Knowledge Engineering*, vol. 116, pp. 42–60, 2018.

[34] H. Chen, S. Miao, D. Xu, G. D. Hager, and A. P. Harrison, "Deep hierarchical multi-label classification of chest x-ray images," in *International Conference on Medical Imaging with Deep Learning*. PMLR, 2019, pp. 109–120.

[35] D.-K. Nguyen and T. Okatani, "Multi-task learning of hierarchical vision-language representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 492–10 501.

[36] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang, "Scene graph generation from objects, phrases and region captions," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1261–1270.

[37] J. Gu, H. Zhao, Z. Lin, S. Li, J. Cai, and M. Ling, "Scene graph generation with external knowledge and image reconstruction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1969–1978.

[38] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2285–2294.

[39] V. Sanh, T. Wolf, and S. Ruder, "A hierarchical multi-task approach for learning embeddings from semantic tasks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6949–6956.

[40] J. Wehrmann, R. Cerri, and R. Barros, "Hierarchical multi-label classification networks," in *International Conference on Machine Learning*, 2018, pp. 5075–5084.

[41] S. Feng, C. Zhao, and P. Fu, "A deep neural network based hierarchical multi-label classification method," *Review of Scientific Instruments*, vol. 91, no. 2, p. 024103, 2020.

[42] Y. Bengio, Y. Lecun, and G. Hinton, "Deep learning for ai," *Communications of the ACM*, vol. 64, no. 7, pp. 58–65, 2021.

[43] K. Hashimoto, C. Xiong, Y. Tsuruoka, and R. Socher, "A joint many-task model: Growing a neural network for multiple nlp tasks," *arXiv preprint arXiv:1611.01587*, 2016.

[44] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[45] J. M. Gallagher and D. K. Reid, *The learning theory of Piaget and Inhelder*. iUniverse, 2002.

[46] E. De Bono, *Parallel thinking*. Random House, 2016.

[47] E. De, *Parallel thinking: From Socratic thinking to de Bono thinking*. Penguin Books, 1995.

[48] [Online]. Available: https://simania.co.il/bookdetails.php?item_id=145306