Deep learning framework for Artificial General Intelligence - Part I

Shimon Komarovsky and Jack Haddad*

***Abstract*—**

I. INTRODUCTION

DL as the technical implementation of AI is not fully exploited as it could and should be. First, 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 actually an identification process based on sophistical memorizing, without involving any thinking or imagination.

Third, a successful 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. Hence, this is another reason for the necessity with 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.

Artificial intelligence (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 artificial intelligence 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 to analyze data and make predictions. However it is expressed by a human engineering effort, extracting relevant insights from data, which involves mostly statistics. Though there are studies try to automatize this effort [4].

AI as a proven method to deal with big-data, is a good method to use to tackle the large-scale traffic control problem. Also in the AI field the use of NNs (specially DNNs [5]) is also great to deal with non-linearity.

The author suggests a more general type of solution referred as AGI (Artificial *(*General) Intelligence), using NNs and not other statistical AI methods, such as logistic regression, support vector machines (SVM), decision-trees, decision forests, kernel machines, and Bayesian classifiers and more [6]. This is because these are all human-made mathematical inventions, while NNs are the closest thing we have to an intelligence - an imitation of a human brain.

[4]

Though there were purely symbolic systems, and there are also a combination of them with NNs, such as [7], [8], still this is only the middle of the way to AGI. In order to make flexible AGI, a full NNs system should be designed, without any limiting structures within it.

A. MOTIVATION

There are two types of motivation for this paper:

1) General: If we regard human mind as deep neural networks (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. So the optimal algorithms we use are good only for very limited cases, otherwise they are inefficient. So we build general solution that search "blindly" for solution, i.e. without considering the specific data we solve. On the other hand we can always "sit" and learn ourselves the data we deal with, but it will take plenty of time and efforts, but eventually we will get much better tailored solution for this specific data. Either way, we are limited in our capacity.

Example: you can plan model and control strategy on traffic network, based on your general assumptions about them. But you can also plan it for a given network, observe it and learn it, and then plan.

Hence we have two choices here (see Fig. 1). We can either plan specifically for some task, or we can think more generally - how some AI can understand the data/system for multiple use/tasks. AI that actually learn the data and plan for us.

Shimon Komarovsky and Jack Haddad are with the Technion Sustainable Mobility and Robust Transportation (T-SMART) Laboratory, Technion–Israel Institute of Technology. * Corresponding author: jh@technion.ac.il

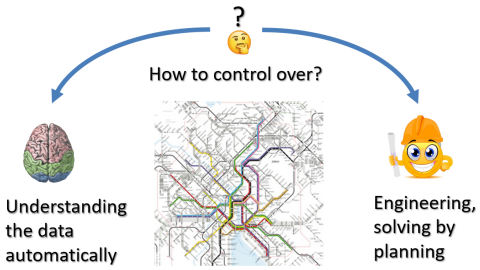


Fig. 1. How to control over some system?

2) Specific: This sort of things are the essence of DNNs or more generally of Machine Learning (ML), where we learn the data. But the aim of this research is multi-tasking that is not defined in advance, by some particular tasks. I.e. the goal is to end up with a system that learn the data and can operate on it in a variety of ways, efficiently. Today most ML models designed for single task, or multi-tasking with specific tasks. Any new task or type of data require retrain the model. The novelty here is about the ability to generalize not only for unseen data, but also over unseen tasks, as it is done in meta-learning field. More about this in IV-B.1.

Most important message of this research is that the goal is not to search best performance model, and compare it to other models. The goal is to try to understand what mechanism organize data in a usable way, no matter its primary quality. To build a slow machine but more intelligent - worth it.

Full intelligence comprises of modeling the world (understanding the data), then after it operate in it, this is the control part. This paper investigates the modeling, as the encoder of a AE system, where the decoder is the controller which executes what was decided in the model part of the system fused with the user's will.

Note that there is a small contradiction in designing AGI. In one hand we claim that human intelligence simplify and abstract too much, hence less data-driven, while current DL is mostly data-driven. On the other hand, the AGI we design is based on the only model of actual intelligence we know of - human's. So constructing AGI probably will lead to the same limitations that humans have. It is not suppose to deal with fine details, similarly to humans.

II. LITERATURE REVIEW

A. Cognitive Architecture

Finally, two studies [9], [10] tried to implement AI based on distributed *cognitive* architectures [11]. The first study [9] is based on Global Workspace Theory (GWT), where the local controllers usually perform a purely reactive behavior, and compete each other in order to define which of them is experiencing the most critical traffic situation. This architecture contains agents imitating brain functions, such as different types of memory and sensor, and implemented using codelets. The codelet is a sequence of machine instructions that can be represented by: NNs, fuzzy systems, evolutionary computation, rule-based systems, Bayesian networks and more. There arewhich contains agents imitating four types of brain functions, which implements four types of memories and sensorscodelet: (i) sensory codelettype, which holds positions and velocities of vehicles; (ii) behavioral codelettype, which determines the light of the traffic signal in a particular intersection; (iii) consciousness codelettype, which represents the working memory and interacts with other codeletsmemories/sensorsbrain functions; and (iv) motor codelettype, which executes the chosen signal phase for each intersection. Two regimes were tested in the study: (i) Parallel Reactive - where each intersection behavior codelet decides its phase based solely on the information it receives from its respective sensory codelet, and (ii) Artificial Consciousness - where intersections receive broadcast about the situation of the critical intersection in the network, and then decide based on this information its own next phase.

In the second study [10], an AI was implemented on a single intersection using a Multipurpose Enhanced Cognitive Architecture (MECA) [7], which was adapted for TSCP. 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 and an intersection agents. MECA is composed out of two independent systems that communicate with each other: (i) a fast reactive system whichthat holds the input sensors and output actuators, which is suited for automatic normal situations, and (ii) a motivational system which is goal-oriented and suited for unexpected situations.

Both papers show improvements in the overall mean travel time of vehicles in the network, compared to fixed timing plan controller, under different conditions. 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, e.g. an isolated signalized intersection and a corridor of 4 intersections. Hence, such results that concentrate on specific performance index and study simple casestherefore they provide only a proof-of-concept. Moreover, real AI should be tested on various conditions, and with different objectives, for a more comprehensive analyzes. The presented architectures in [9], [10] are rule-based design, while future research should strive to develop architecture based on DL or biological models.

B. Prior knowledge in DNNs

As shown from the above studies [12]–[16], 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 [12].

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 will see later when we will introduce graph input, which will require 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 [17] or separating features/variables and more. E.g. CNN uses sharing parameters. Most studies perform 3D convolutional kernels on an RGB input, thus treating this data as grouped information of each pixel. Other studies may treat them separately, where each of the RGB channels do not merge with others throughout the whole DLM. Similar is when we set apart the depth channel from RGB in a RGB-D input images. Or when we set apart traffic data from weather data [18], or roads from stations [19] and fuse them only later (how later is also prior knowledge to be decided upon). We can also separate tasks to groups [19].

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 [20], Neuroevolution, hypernetwork, Meta-learning (learning over learning) [21] and more, which are all searching or adapting a given architecture, to better fit the data.

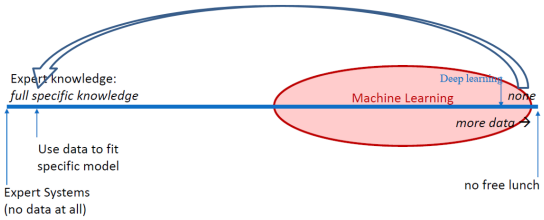
Nowadays, ML theory assumes best performance is in DL, see Fig. 2(a). Especially for large data set [22], see Fig. 2(b), where the main effect is for big data, since then there is a difference between the models. But in small data the different sizes of NNs are all behave similarlyapproximately the same.

However, in our humble opinion, unlike the place where AI supposed to be to the most right part of the scale, where we almost totally dependent on data (minimal prior knowledge), it is actually the opposite - we need full knowledge for designing AI (lots of assumptions), which do not depend on data at all, but is flexible and adaptive, and able to tune gradually and gently according to it. Whose more important here? It is not the data, but rather the assumptions and lots of planning of AI. Similarly to planning any system, e.g. control system, here we make it general as possible that account for as most cases as possible.

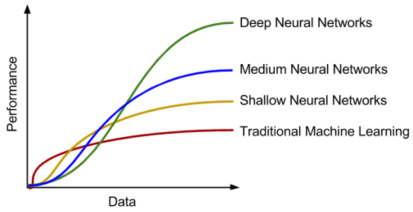
Same idea in [23]: *"learning ability does not come for free, and is far from automatic. It relies on very specific assumptions that are mostly encoded in the design of the NN architecture itself - here denoted as structural priors."*.

[23] also mention the Occam's razor approach, that a more complex model with more parameters, will be able to explain a wider range of data, however a simpler model will necessarily assign higher probability to the narrow range in which the data of interest lies. I.e. we should plan the AI to fit the data (all its variants) and not beyond it, since it will ruin its performance. Same effect is in robust control verse nominal [24].

[23] 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.



(a) No free Lunch for highly-designed AI.



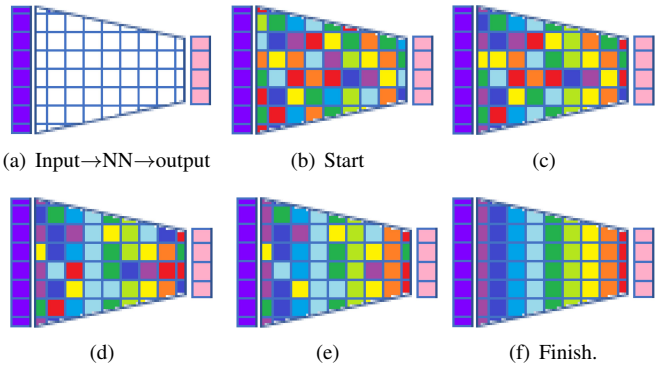
(b) Performance verse the amountdimensionality/scale of data, using different methods, taken from [22].

Fig. 2. Comparison between AI methods to comprehend the environment.

There is an issue to define the problem of AI. Unlike regular classification problem well defined in DNNs, it is difficult to know what is the overall purpose of AI. 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 to its tool (inner or secondary objective) to estimate how well the organization is (strive to low entropy).

See for example in Fig. 3, 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, 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.

Note that when training on random labels, then the task is actually merely memorization [25], since there is no consistency in the output data. Hence it also has no generalization.



(a) Input→NN→output (b) Start (c)

(d) (e) (f) Finish.

Fig. 3. 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.

C. Network Architecture Search

Network Architecture Search (NAS), such as AutoML and others, are first define/restrict the search space for models. Search techniques that are used are usually RL or EA and more.
A more efficient (faster) approach use weight sharing between proposed networks, instead of trashing 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 sctrach. Same idea in RL searching paper where try to transfer as much as possible from previous trained models. They modify architectures in a way that uses different parametrization to represent the same function. Aspired from "Network morphism" paper, where we modify only some specific part in NN leaving all rest unchanged. Lamarckian Evolution also uses this idea for EA search.
Differentiable Architecture Search (DARTS): Unlike previous EA, RL the search defined as differentiable and continuous problem. While I have combination of those: wake the network is static (also for the purpose to evaluate how good it is) and sleep where it's change. DARTS have some unknowns so they use multiple categories, where the final architecture discretized after the search is over.
Hypernetwork, where the weights are learned not by training but via other network. It means that for every input you test, there'll be different weights to predict the output. So it's dynamic structure at its essence, unlike my dynamic structure NN. Perhaps its enlarged expressability as I hope also in my DNN, is due to the nature of adaptative information in our reality. As I stated before, the human brain is like the best or very efficient NN designed specifically to represent the set of hypotheses that represent the data in this world. We have to seek for this specific type of NN, with a lot of prior knowledge, rather then simply trial-and-error methods.
In one paper there's hypernetwork for both CNN and RNN as two ends of a spectrum. Input vector is z. in RNN weights can vary across many timesteps. Hypernetwork's relaxed weight-sharing approach allows us to control the trade off between the number of model parameters and model expressiveness.
My note: Hypernetworks can build also 2nd network to learn weights of the 1st one that learn weights for the main one. And so forth, strengthening the evolution in different time resolutions/scales (very slow to fast).
Neuroevolution: is having evolving hyperparameters, e.g. NN building blocks (for example activation functions), hyperparameters, architectures and even the algorithms for learning themselves. E.g. replace BP with GA for searching for weights. Here we can also combine this GA idea in the evolution of NN.
NEAT (NeuroEvolution of Augmenting Topologies): instead of fixed NN, evolving through GA. Combining GA with BP.
HyperNEAT:
SMASH: train a hypernetwork to predict the weights of a network in 1 shot instead of training all those candidate networks from scratch. They 1st train a hypernet to predict the weights of some given arbitrary network, then they randomly generate many architectures, and rather then training those models, they use a hypernet to obtain the weights. Finally they pick the architecture that performs best with those weights and train it from scratch.
Optimization problems of: model architectures; training schedule, e.g the optimizers themselves; and the data, e.g it's order, size, diversity and so on.

D. Unsupervised Learning

Boltzmann Machines (BMs) BM is a two-layered NN, where the input layer called also visible layer, and the latent layer called also hidden layer. Besides having connections/weights between the layers, we also have within the layers: hidden-hidden and visible-visible interactions. However, in Restricted BM (RBM), we have connections only between layers. The objective of RBMs is to find the weights that reconstruct the input data given the hidden data. Deep Belief Network (DBN) is a stack of RBMs, where the training is gradual, starting from 1st and 2nd layers, then 2nd layer is act as the visible and 3rd layer as the hidden one, and so on.

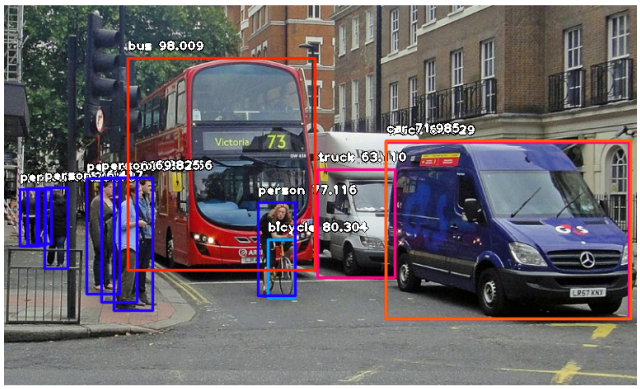
Auto-Encoders (AEs)
Sparse-Coding
Self Organized Map (SOM) NN
Deep Clustering

III. METHODOLOGY

A. Biology-inspired Deep Learning Models

Several AI biology-based systems should be explored. As a starting point, the auto-associative hierarchical memory model [26] will be chosen investigated. This model differs from regular DNN in several aspects: (i) DNN learning is based on abstract labels learned by humans after prolonged maturity, however, real learning is gradual and probably based upon abstraction of simpler concepts, by higher pattern

recognition, as it proposed in the subject model, see Fig. 5(a); (ii) DNN's perception is based on visual features only, but when human perceives what he/she sees, he/she does not only recognize objects in space, but also recognizes their meaning, their behavior, their associations from past experiences and more, see Fig. 9. Therefore, the developed model in this research should involve higher patterns, mainly sequential pattering. Fig. 5(b) also demonstrates the closed-loop control, since sensory and textual inputs ascend up the hierarchy, then after comprehensive perception, the proceeded data descends down to the output traffic signals, which eventually. The effect of this operation is fed back through input channelsand can be evaluated accordingly.



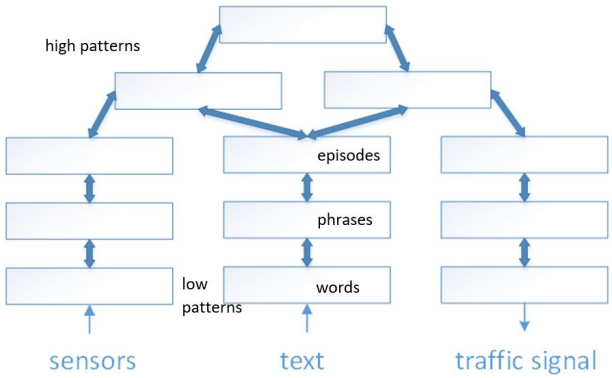
(a) AI identifying.

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.

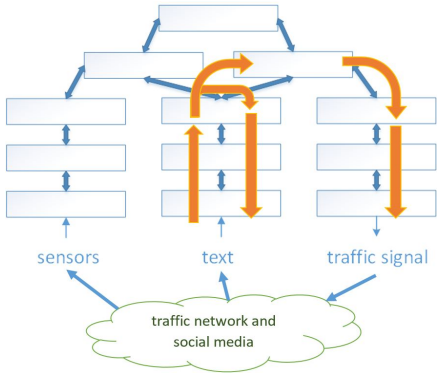
(b) AI interpretating.

Fig. 4. Comparison between AI methods to comprehend the environment.

Note that the auto-associative model [26] includes a memory-prediction propertyis a bi-directional model, see Fig. 5, where the feed-forward is utilized for memorization, while the backward propagation is used for prediction. The process of memorizing and representing sequence patterns in these deep networks is executed via hierarchical chain of



(a) AI pattern recognition.



(b) Hierarchy with pPrediction and closed-loop control.

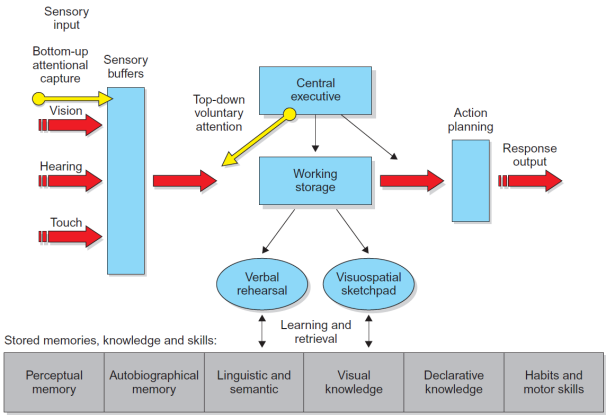
Fig. 5. Proposed hierarchical structure enabling prediction and memorization, based on [26].

commands: data moves simultaneously up (from sensors) and down (from high patterns) the layers. When data flow from above and below intersect in this hierarchical network, evaluation is performed, i.e. comparing perceived data coming from the bottom with the predicted data coming from above based on current recognized pattern. If the compared data is identical, then it is a known data which continues to move to the bottom. However, if the compared data is different, then this data represents an unfamiliar information and the assumed pattern is incorrect, thus it moves up in the chain of commands, to be assimilated and handled by higher layers.

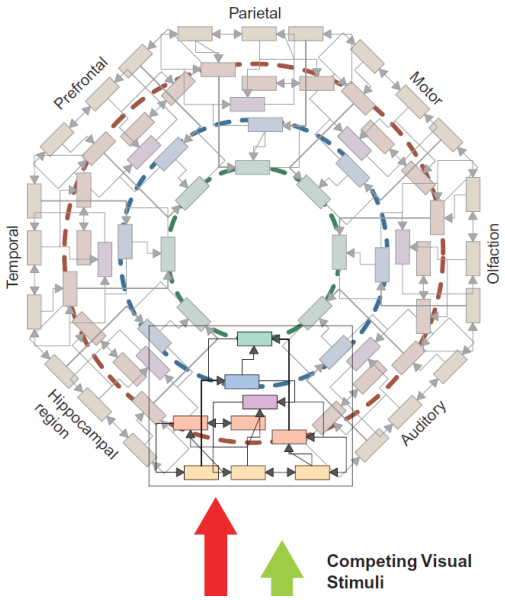
The lower hierarchy is characterized with specific and local features, instability, and constant change; while the higher hierarchy is characterized with general features, stability, and preservation. Thus, this hierarchical structure would be suitable for implementation in the traffic system, where fast reactions are handled by actuators at the lower level, while smooth reactions at the higher level, in which a more comprehensive point-of-view exists taking all different factors into account. This hierarchical temporal structure can be implemented not only in the execution component, but also in the perceptual component.

Another source for biological model is in [27], which is unique since it has 2 attention systems, (i) a bottom-up via competing stimulus, and (ii) top-bottom, which initi-

ates/starts from the central executive, which is the supervisor of the whole system. In it we have a detailed neural structure, see Fig. 6(b), similar to the one in Fig. 5. But here we introduce the concept of cognitive architecture, as shown in Fig. 6(a). It is a functional framework.



(a) Cognitive functional architecture.



(b) Detailed neural structure.

Fig. 6. Sensory hierarchical structure, taken from [27].

IV. Proposed AGI model

A general sketch of the first draft describing the AGI model/structure is shown in Fig. 7.

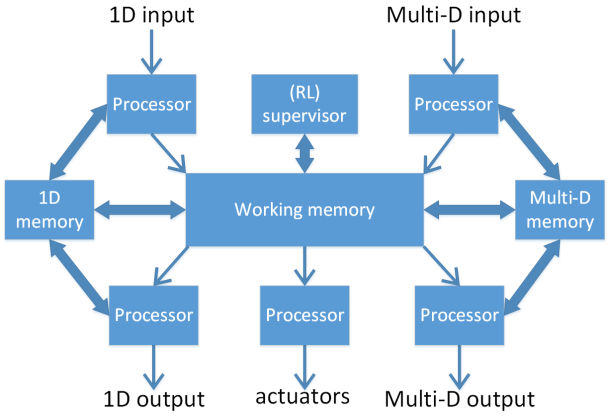


Fig. 7. AGI possible architecture.

is based on global workspace theory (GWT) [9], [28], [29], only here there is no competition among independent agents in the system, but rather centralized control with different elements with specific functioning.

But I don't believe in dichotomous structures such as in Fig. 5 and in Fig. 7, since they are limited in nature (as explained in Section **??**). I rather believe in continuum that appears in fuzzy logic and in Section IV-C.5. Nevertheless, we need to take into account different function components in these architectures and interactive-ties between them, but also remember that the boundaries between them are not so well defined and are changing, see Fig. 8.

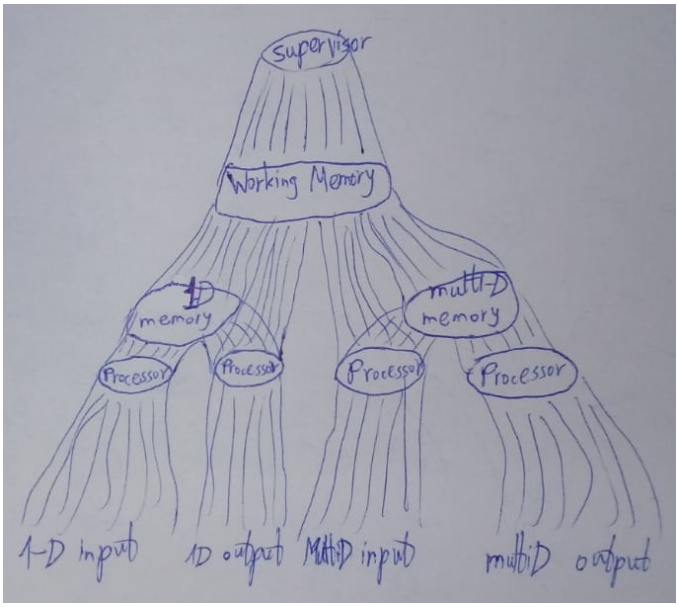


Fig. 8. AGI possible architecture in 3d hierarchical pyramid form.

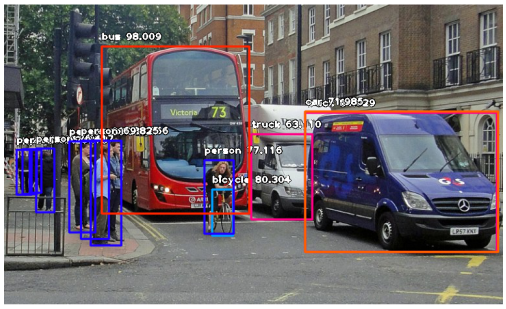
There is sequential processing of 1D and multi-D data separately, for data identification. Eventually using it in the working memory. And when event for output occurs, such as in case some idea or thought emerged, then the output can either go through the 1D channel, as humans describe their inner thoughts to the outer world verbally, or through a

multi-D channel, which is an extension of human structure (if it is possible at all), and can be regarded as "screening imagination". Because if we have 2 types of inputs, why shouldn't there be outputs of these types either? And of-course 1D information have some common memory for both input and output and the working memory (such as language), and likewise for the multi-D. The double arrows mean the acquisition and update of this storage.

Output communication is also an issue. Should it be monitoring thoughts, or wait for meaningful output, or maybe even AGI will have some sort of free-will to choose when to interact and with what.

Notice, that this particular AGI based upon Stimulus-Response behavioral theory [30], that claims that we cannot observe the mind itself, directly, but rather communicate with it. This assumption is of-course the worst-case scenario. It is also called intelligent behavior, and expressed by humans in many scientific fields, such as engineering, math, spiritualism, computer sciences, education, psychology and more.

Two types of data are needed to perform inference, since we use meaningful words (1-dimensional input) to construct world representation, seen by the multi-dimensional input. Also, it is not only identification of objects, but also their meaning and predicted behavior, as illustrated in comparison in Fig. 9



(a) AI identifying.

(b) AGI interpretating.

Fig. 9. Comparison between AI and AGI comprehending the environment.

Notice that we have an asymmetrical component of control in the AGI: the actuators (in our case traffic signals). Humans have touch senses and motor control very interconnected and interdependent [27], as can be perceived from third Newton's law, i.e. when one push/pull something (control action) he immediately sense it, as a feedback to confirm its prediction. However, in our specific case there is no sense for traffic signals, they are controlled intrinsically (ultimate control).

The purpose of having two types of channels are for distinguishing the outer and inner world that the intelligence 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 meaning, which expressed by words or any symbolic language. This is why meaning can be transfered to humans not by the objective world, but rather only by other meaning-based beings using the meaning-based language, which in our case is 1D sequential type of data. Illustrated in Fig. 10.

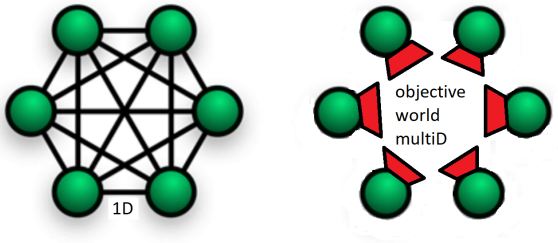


Fig. 10. Objective (right) verse Subjective inner representation (left).

This is the reason why there are no labeled outputs in DNN in reality, other than the one that represent meaning, which is given by the 1D communication with other similar beings.

Also, the equivalence of generating conclusions is actually finding some order in the data or pattern recognition, but in a more general sense rather than specific narrow-type pattern. And all this is by using 1D symbolic pattern/model construction.

Notice, that labeled DNNs are 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 seen and their meaning (either vocal or symbolic).

In DNNs it is known that network's layers are order from 1st layer as most specific to the last layer as most general. This means that one way to deal with fix structure and the problem of generalization of hypothesizes (where small data requires simpler model and bigger data require a more complex model), is by enlarging the network by adding more neurons in the layers or adding more layers. Which makes the structure dynamic, and apt for development as in humans. Also we'll use artificial cognition papers [7], [9], [11], [29],

[31]–[34], for suggestion of different cognition architectures and models.

More about AGI in literature: [11], [35]–[40].

Note: If we want to develop not only AGI model, but also control, meaning add to the system actuators as additional output, then Fig. 7 has to be updated by taking them into account.

A. *Specific part of the approach*

Similar to [41].

We present now a demo of the DLM, for a time sequential visual input, and describe its evolution through the eyes of a new born infant.

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 detection through some initial DNN, see Fig. 11(a).

After a while, when enough counts detected the single event, a split performed of this event into two (or more) classes of events, see Fig. 11(b). Now it can differentiate two events, sharing the same deep features. He can recognize as many events as necessary (adaptive NN structure, adaptive by necessity).

At some point of evolution, when connections and event identification is strengthened and established, the model can move its attention (or free its resources, since the given level became more automatic, similar to the idea in [26]). It can build a new layer or "floor", if a re-occurrence of 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. 11(c).

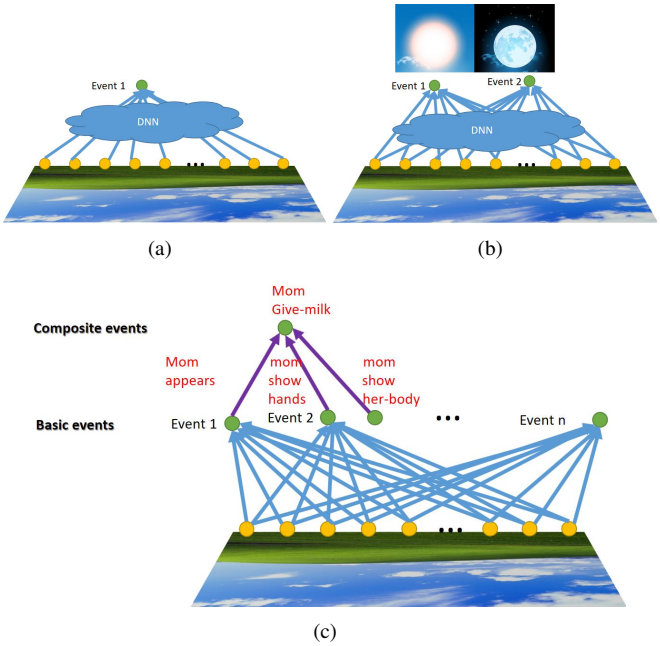


Fig. 11. Separation and composition of neurons in proposed approach.

An opposite structure-changing operation can be decomposing an event, if it appears to be more complex than it was supposed to. Previously thought of as a specific-level event, now is decomposed into a more simpler events (refinement). Either existing events or new ones. New ones than have to be attached to lower level events/features. E.g. see Fig. 12, where we decompose the "ball in the air" event into its basics: throwing, moving in the air, and been catched.

It does not have to be risen up to the higher layer, and decompose to a 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 often, then it might imply that they are all the same event, hence should be eliminated, all except/leaving one.

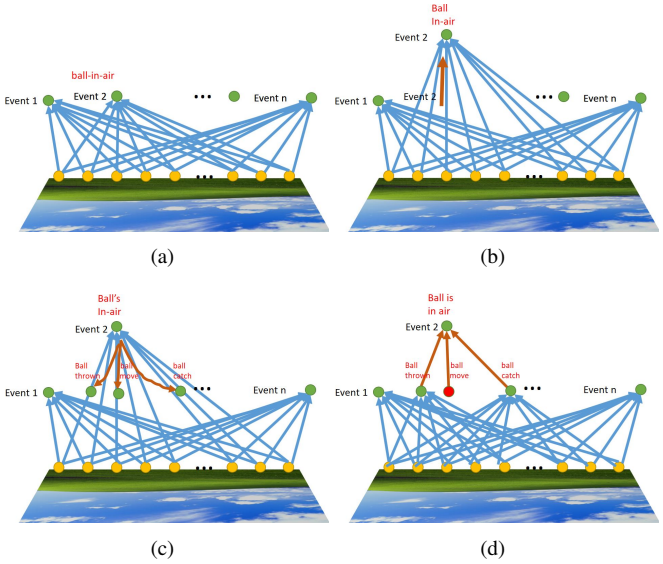


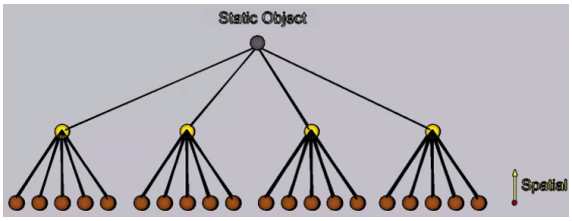
Fig. 12. Decomposition of neurons in proposed approach.

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, and changed by the decision of some supervisor.

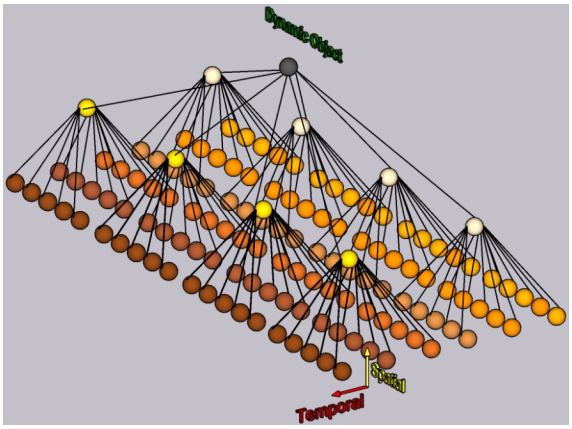
This dynamic NN structuring algorithm (composing and decomposing) is like the right brain or the supervisor of the learning block (left brain), enables efficient and evolving knowledge and ordering. It has two main advantages over the full-connected NN: (i) It is less computational since it has less connections, and (ii) It solves the problem of how to define in advance the NN structure for a given problem (#neurons, #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 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 he have to keep only highly necessitous decisions. E.g., learning rate and constructing new neurons and layers will be scaled accordingly to the given limits.



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



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

Fig. 13. Spatio-temporal DNN model.

Also, the idea of brain plasticity and distribution of tasks over many regions in the brain, may imply that neuronal information transmission is not the only operations occur in an active brain. I.e., similarly to cell division and reproduction then assignment to different roles stated by the genome, 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.

We now the regular DNNs, used for spatial object tasks, e.g. clustering or classification. But If we want to detect and extract features along also the temporal dimension? Than we simply extend the DNN to include this dimension, without changing the spatial dimension, i.e. orthogonal to it. Of course this is FC NN, we can hence specialize it in different ways, e.g. shared connections/parameters or convolutions, etc. Other ST models combine CNN with RNN, either separately: CNN→RNN or interchangeably: CNN→RNN→CNN→RNN→CNN→RNN→CNN.. Or having them as separate inputs, with fusion module. So event tasks, such as feature extraction or classification/clustering, can be done using this extension above.

See Fig. 13.

B. General part of the approach

The DNN is originally spatio-temporal, it have static mode which is the spatial one and dynamic which is spatial and temporal. There are two ways to learn object's features that are not related to time or events: as an infant when you treat everything as static and not as an infant when the situation you are looking at is stationary. Perhaps it is all mixed up together: the dynamic DNN changing both dynamic and static features. So might have no such differentiation for events.

Another thing distinguishing AGI from AI, is that real intelligence do not end up with categories, like cat, dog, etc as in DNNs. I.e. it is evolving.
Regular NNs have specific outputs. But as human learn more and more, he add more layers/abstraction or a new NNs branching from the old ones. So we need to think how to expend a given network. One suggestion is *branching of NNs*. Abstraction results in some hierarchical relationships, but there can be lots of different and unrelated hierarchies. E.g. family tree, parts of speech, table of contents, etc. It means that we cannot use one single 2D NN. We propose that it can expand as a tree: in different locations of a given NN, and in a different structures, depths and so on. An illustration of multiple hierarchies formed in a given NN is in Fig. 10.

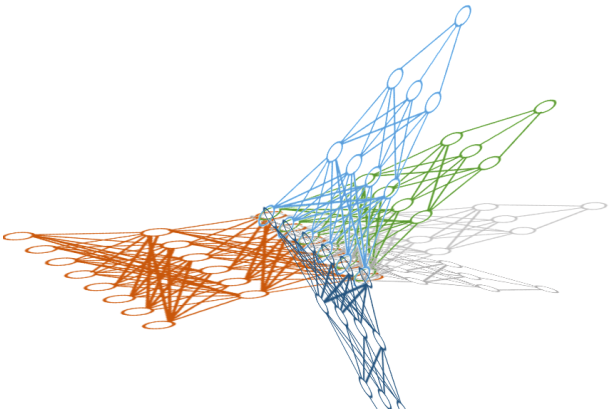


Fig. 14. Branching due to multiple hierarchies.

Also the structure update can be done during ”sleeping” periods/mode, when the frequency of neuron excitement trajectories are stored in the neurons themselves (perhaps it is so even in the actual brain). The rate of changing structure can be also modeled like in RL, where mostly at first is exploration (high randomization), than lesser and more exploitation (low randomization).

[42] demonstrates that NN is best for performance if its structure is appropriate to the data it is trained upon. Not too much neurons nor too little. This optimal number of neurons and layers might be because there are the most appropriate features to describe the trained data. Any other structure may result in some kind of a spreading over some features that are not really representative, of the data.
Hence, since usual DNN only guess the number of features at each layer, then it is required to check several structures. Dynamic NN dealing with this issue, by keeping only the relevant and true features.

An issue exists in dynamic structure NN: what about features that supposed to be frequent or rare and we should not split or delete them? However, perhaps this splitting, of an actual feature into several ones, may not worsen anything beside introducing new features.

Other principles can be applied to the proposed approach, and presented in the following.

Try to implement hierarchical principles IV-B.1. Especially an idea I was developing very early in my study. The idea is generally come from the fact that ... Also if we combine this with the idea of hierarchical learning, where we teach the DLM a specific task, then move on to a more complex task based on previous task(s), thus adding more layers for that reason. Then I suggest why don’t we do this automatically, as I assume our brain does? I.e. we can compare our fully-guided hierarchical learning to some automatic hierarchical learning, in which some supervisor program knows how to change the NN in dependence of the data it receives. Perhaps it is done by memorizing all the ”un-attended” data that couldn’t been handled in the awaken period, while we use only the proven-to-be-effectively-predictive NN in the awaken periods. Implementation of it can utilize hierarchical parameter tuning and some Network Architecture Search technique.

Guided teaching may be a problem, since we might be wrong in the correct order of teaching. E.g. we teach it very early to detect accidents in the network. We might realize later that for good detection it must be taught concepts as intersection and roads, also different modes of normal situations, and more. Hence the development of automatic learning is crucial in the long run.

This idea is also resembles the biological model of Jeff, where we pick the simplest data we can predict well, then we learn to predict more abstract data only after the more fundamental data has been established.

1) Hierarchical principles: An important notion repeating itself in this paper is hierarchy. It appears in different forms:

Fitting of model with the data it represents. No-free-lunch theorem [43] says that no learner can succeed on all learnable tasks: every learner has tasks on which it fails while other learners succeed. Therefore it must have some prior knowledge: either on distribution of the data or some finite hypotheses set.

It means that the information a person learns so successfully have to be pre-adapted to him in advance, to his/her learning system. If there were no fit between the data and the learner, then most chances that effective learning would not occur.

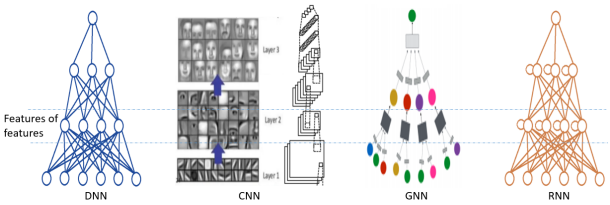
We can use the model-fit-data principle in the decision of how to expend the NN in our approach. I.e. just as CNN best fit for visual feature extraction, and RNN use recurrent connection since the sequenced data is recurrent in nature, and GNN use adjacency of neighbors since the data is behaves this way, see II-B, similarly we can check different NN types such as these mentioned, to best fit the extension of the present NN.

DNN is a structure that fit the data it models, the data itself is hierarchical in essence, and what the training does, is organize the data in its natural structure, hierarchical features structure. Hence we can exploit this structure to solve problems (by search for solution) directly and not blindly as it performed in heuristically types of search breadth-first-search (BFS), depth-first search (DFS) and so on. They are blind due to the fact that all of these methods are random, logic-based and combinatorial (trial-and-error). Hence search and optimization tools that randomly try to optimize some goal are inefficient when working blindly on unorganized data. While the brain do the opposite, it solves problems directly, after organizing data. E.g. association in NNs works by the multiple features an object has. Even if only small amount of these features are triggered, it will track this object. So solution is not optimal generally, but it is the best for the current data organization. E.g., when a person thinks of a several solutions for a problem, it is not via blind search, but 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. Therefore, we need to use this type of thinking in AI.

Hence, if we combine the idea from II-B and here we come to the following conclusion: just as we optimized blindly not

considering data in the past, nowadays we highly-dependent on data but blind to the hidden assumptions about the models that learn it. Hence the role now is to combine these faults together: we plan the AI considering as many scenarios as possible as a function of all possible input data.

In Fig. IV-B.1(a) we can see visual comparison of the different NNs we encountered until now. The common to all structures that they are all hierarchical. And in Fig. IV-B.1(b) we see some basic comparison via some features, from [44].



(a) Visual comparison.

Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

(b) Charachteristics comparison from [44].

Fig. 15. Comparison of different hierarchical structures in DL

Hierarchical parameter tuning. Just as we claim that NN structure is the best fit for the data it receives [42], and this is the prior knowledge, the same goes for the fact that our reality is ever-changing, always dynamic, only on different scales. There are stuff that change fast, and there are stuff that change slow. Same idea we propose to be in the hyper-parameters we tune in DNNs. Some should change frequently (e.g. weights) and some should change very slow (e.g. hyper-parameters). Many examples encourage this idea: Spall's SPSA [45], Network Architecture Search strategies (see MyDNN presentation).

Hierarchical learning. We learn by blind imitation via supervised learning (input↚desired output), but we propose to change it to gradual type of learning, where things are built one on top of the other.

Here [46] for example a gradual learning is proposed from a simple level to a complex level, either manually (expert guided) or automatically (scoring each sample by its training loss). However, this loss is highly dependent on the models and their hyper-parameters. Hence a different learning take place: from fewer categories or output tasks (local) to more categories (global).

"hierarchical learning", meaning instructions or commands are learn in a pyramid way, as a child, we first teach basic instructions, then composite instructions and so on. So that it will enable us in the future create highly supervising system, that can have multiple object functions at once, regulating the changes in it and react accordingly. For instance instead of an operator tells the machine "clear road 34 for presidential platoon", it can teach it to supervise the system, like automatic pilot, and clear roads when presidential tweet occur in the text input.

This hierarchical learning composed of two types of "stuff" to learn: objects and actions. So 1st teach basic elements/objects such as road, intersection, roundabout, pedestrian. Then continue to composite objects and when all objects learned - actions can be learned also.

It can learn using attention mechanism as guidance: We input text="intersection" and in visual input a graph/image with attention on some intersection, then we train it again on intersection with attention on other intersections. Similarly we can teach it instances of objects, such as "23th intersection" with attention on this specific intersection.

Similarly we teach actions, e.g. "from-to", we show it a path from x to y, then we train it with different paths from same x to same y. And can do it for different x's and y's. Also we can start from learning specific objects with specific NN components, then learning new stuff by adding more components on-top of them and freezing the 1st components if needed. E.g. learn about objects such as roads and intersections and other elements of transportation network including text and other resources, and then add new NN to learn now operations over these objects.
Also, this system is reversible, i.e. we do not need to learn from component to new component only, we can train actions after objects they operate upon and recall that we forgot to teach some object. So we can retrain only the relevant component, and then return where we were previously. Similarly, we can decide when to teach only the perceptive components, and when to add the executing component.

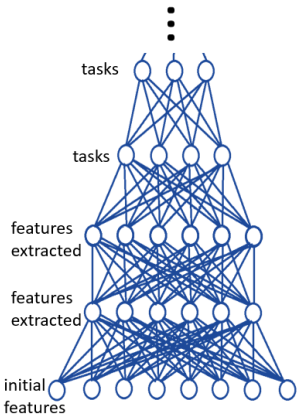
All these flexibility that can be embedded in the model, introduces the importance of different types of AGI's mental experiences or modes, such as learning, learning and acting/exploring, communicating or querying from the human operator (to perform specific tasks, such as controlling a network or solving a problem) and more.
All these flexibility options and modes are probably will be constructed manually, but in the future we can possibly make it automatic, thus closer to human's capability.

2) Sleep/wake periods: These modes can be learned from baby's development: in first months he is gaining a lot information and do not use motoric system to explore. He sleeps a-lot since the organizing or optimizing (such as backtracking) are vastly needed to start make sense. Meaning that in waking periods there is no processing involved, but rather some type of storing in memory for future arrangement. The more time passes, the less sleep he actually needs. And at some point he gained the insight that he can use its motorics, and at some point also for exploring.

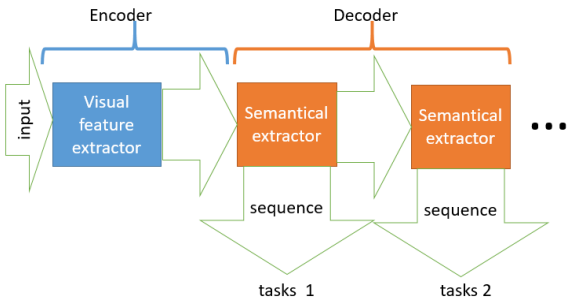
This can also derive a distinction between the left hemisphere which responsible for solving problems not analytically, similar to switching to other activities or meditation or sleeping, as mentioned, and the right hemisphere which responsible for conscious thinking. From the observation above, it seems that thinking do not mix with organizing data, but rather manipulating existing data, in its current order, from memory. So there are two processes: thinking

- which uses the current structure of knowledge, and self-organizing which is a background and unconscious process (since its internal restructuring cannot be viewed/interpreted logically as serial organized sequence of thoughts).

3) Demo of hierarchical learning: If we consider the hierarchical structure to be built as layers of features and tasks, see for example Fig. 16(a), then hierarchical learning can be done in the encoder as described in [47], e.g. in DNN or CNN. Also features can be between tasks, as in [47].
Also in my opinion, tasks are like pinpoints in the NN, or the intervention of external information.
But if our intention by "teaching" is matching words or phrases for the visual input, as in image captioning, a stationary type of learning, using the encoder-decoder structure, then we can separate tasks in different level in the decoder, as illustrated in Fig. 16(b). First the model consists of visionary and a single semantic feature extractors, and we train on all tasks of group 1. Then we add another unit of semantic feature extractor to train on the composite tasks of group 1, which is group 2. And thus go on with the same way. Adding units can be for example by expanding the RNN (assuming the decoder is RNN).



(a) The output of each layer may be features or tasks



(b) Hierarchical learning in visual captioning.

Fig. 16. Hierarchical learning Demo

Important notice: unlike features that are distributing representatives of data, that 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. Since a small group of inter-related features 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.

How to handle multiple outputs In this situation, where we have multiple outputs, we can either leave them as is, to be entered to the next component in some larger model, or we can choose the output with the highest confidence level (e.g. by leaving the outputs as probabilities, without applying some selective function).
For example we can further extract the features of these multiple outputs by a DNN whose input is a 2D NN as a graph of nodes, holding the amount of triggering of the different tasks (or features if we want to include them). This DNN is actually perpendicular to the original DNN. See Fig. 17.

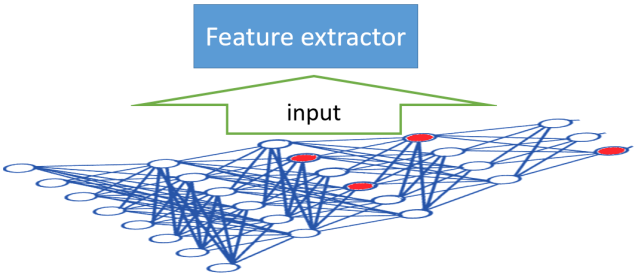


Fig. 17. Learn features of features

It does not have to end here, we can continue extract this new DNN's feature perpendicularly likewise, ending with a multidimensional NN structure.

4) Literature review: [48] note that MTL generalize better than via single-task learning, since it apply inductive bias (just as regularization do), thus preferring more suitable hypotheses for the learning model. MTL implemented either via hard or soft parameter sharing of hidden layers. Hard sharing means set of layers are shared totally between all tasks, leaving other layers for each task alone. However, in soft sharing we do not impose same parameters, allowing separate layers for each task, only constraining the distance between the parameters, to encourage the parameters to be similar.
There are also more adaptive approaches for sharing parameters. E.g. [49] propose to tackle the issues of hard sharing: under-transfer in the classifier layer is when knowledge cannot be adaptively propagated across different classifiers, and negative-transfer in the feature shared layers, when sharing information with an unrelated task might actually hurt performance. It propose FC layers to fit task-specific structures, with parameters that are modeled by tensor normal priors for learning multi-linear relationships of features, classes and tasks. This idea similar to ideas of attention mechanism that search which input features most influential and the idea of automatic feature extraction in DNNs, where neurons can behave both as excitatory and as prohibitive, to

choose only the most relevant features and deny the irrelevant ones. Other adaptive idea [50] works in reverse: a bottom-up approach that starts with a thin network and dynamically widens it during training promoting the grouping of similar tasks.

Besides sharing issue, we have to attend the tasks themselves. Most known is structure of multi-tasking in NNs is flat one, but there is also a graph structure of these tasks, e.g. tree-like hierarchical relationships between labels/tasks/classes. There are many different approaches for implementing these structured tasks, which eventually can be combined in various ways.

There are those who find labels structure separately from the model [51], [52], and there those who do it as a part of the model [53]–[55]. 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 [52], [56], and there are those who insert intermediate tasks inside the NN, i.e. locally [47], [53], [57], [58]. There are those that learn the structure from the data [54], e.g. by unsupervised clustering of the labels via some similarity measure [51], and there are those that import the structure from external knowledge base [59], or use it to adapt this structure [52], [55]. The learning can be either incremental [47], [60] or simultaneous [57]. We should be aware of the catastrophic forgetting phenomenon in incremental learning [61], where the model abruptly forgets part of the knowledge related to a previously learned task as a new task is learned. However, [60] used successive regularization strategy to avoid this phenomena.

[62]

Many principles will be evolved in the development of this AGI. Most principles are come from philosophical and brain sciences (top to bottom effect). And from the practical side - principles that come from engineering, computer science, in particular data science and ML (bottom to top effect).

C. Principles from Humanities (high order)

It is important to note, that most of the psychological and educational research in the field of intelligence, is actually about the external end of the inner and real intelligence. Intelligence test (e.g. IQ) and logical test are all about testing and analyzing the final edge of intelligence. Therefore, one has to consider this as the most external tool to conclude about how the inner intelligence operate. It's problematic in couple of things: 1st it might be very far to conclude, 2nd it's not only external but also highly developed in advance stage of it, and we have to extrapolate backward in time to figure how it has to start and develop.

1) Human plan effect: Human is the one that plans the AGI. Also, the desire is to have a logical output from AGI, and a human-like AGI, in-order to have a good communication and interpretation. All these creates naturally a limited AGI in advance [63]. Meaning we aim to tune the AGI to be fit to humans, so it must act as planned (so it is biased for humans). This limitation must be considered in the development.

2) Order verse disorder: Although we assume nothing about the data, we do assume that the data is patterned, meaning ordered or organizable. In other words [26], nested and hierarchical just as the brain.

Hence the data human learn is such, and the learning is actually organizing it. The architecture of the brain is designed to find order, and cannot function in chaotic data. May be because we need meaning and purpose from the data we encounter, which requires order.

This idea expressed also in the important aspect of the order of teaching the AGI. As in humans, it has to be from the simple to complex. Order also appears in the relation forms between data units: causality (cause-effect), hierarchy and more.

3) Convergent verse Divergent thinking: Convergent thinking is the process of gathering many ideas and process them into some implementation, while divergent thinking is the opposite - we have a problem, and now we start creating as many ideas as possible.

This idea can be related metaphorically also to the PID controller, as illustrated in Fig. 18.

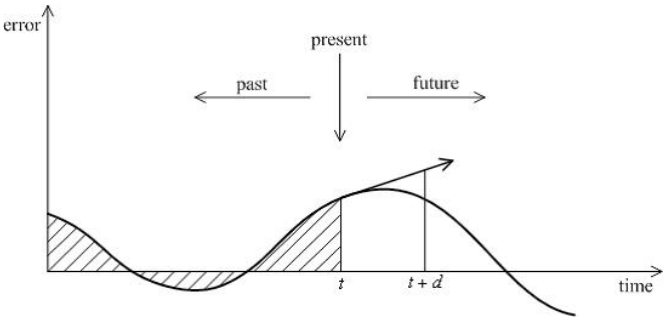


Fig. 18. PID error as a function of time, taken from [64].

Where integrated operation expressed using integral is converging using the past, and differential operation expressed using derivative predicts the future. Similarly, analysis verse synthesis, or feed-forward NN vs backtracking of feedback.

But PID is a good analogy for the full AGI system, which includes control also. Meaning having both sensors (input) and actuators (output), where the control is state-feedback control, similarly to MPC and IC, in the sense that it depends on the present and the future. The present expressed by the low-level fast/immediate control, while the future expressed in the full perception of reality not only in that instance but for some period/sequence, thus enabling also planning and prediction in the higher levels of hierarchy.

But additionally, as in AI methods, such as NNs or RL, it is dependent also on the past (experience).

Similarly to state-feedback controllers, such as MPC, IC, LQR, etc, here the state describes the traffic network that the AGI observe. The model is AGI's neural network, which decide upon the control. Also illustrated in Fig. 19.

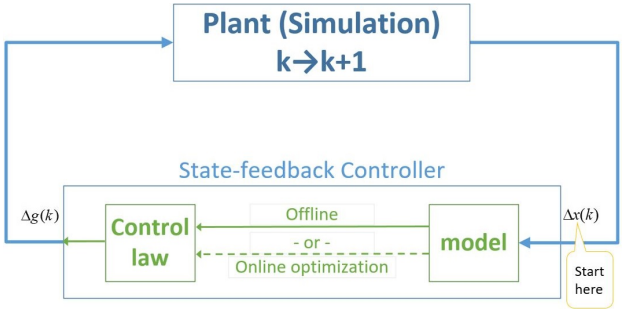


Fig. 19. State-feedback Controllers implementation on a plant.

4) NN black-box issue: Every part of an algorithm is interpretable, and we can easily understand what the algorithm does, but NN, which has also inputs and outputs like the algorithm – is not interpretable.

It is important to separate the inner operation of the brain with its external manifestation by logic, language and thus explanation. Explanation/logic is the output of the NN, hence NNs today are used merely as a tool for control or function approximation, while missing their full potential or their original function as AGI (whose logic reached only via communication and not via its internal structure).

The NN function is a black box to the designer. You may have an intuitive sense of how this function operates and the hidden features that this network has identified, but you do not know the reason behind the value of any given weight or bias. You also do not know what these features represent. So if the function does not meet a specification or if the operating environment changes, you will not know how to adjust the policy to address that problem.

Unlike NN, control system is explainable therefore manageable. You can also isolate a specific controller or loop and focus your analysis there, if an issue occur. You also cannot be sure it generalizes correctly unless you test it in different conditions, unlike control system which were planned after human generalization for as many as possible scenarios. In control system unlike NN, there are many types of formal verification/analysis: do not have to test to make sure a signal will always be nonnegative if the absolute value operation of that signal is performed in the software. Also: calculating robustness and stability factors like gain and phase margins.

Causality: understanding the effects of actions (sometimes called interventions, treatments). See regression for example: Understanding correlations is not enough to understand the effects of actions. That's why NN isn't enough, sequential pattern recognition are necessary. Most applications of data science are predictive, not causal. They map inputs to outputs, but they do not consider how the world would look like under different courses of action (whether the diagnosis would change if we operated on the retina) which is called Counterfactual inference. Causality can be seen in algorithms and classical control planning (see 1 above). All of which eventually necessary for decision-making process, since causality allow test different scenarios of a given problem (using imagination), to choose the best set of actions. The purpose of NNs is not the logic of some process, but a representation of data, see (Deep Learning For Sequential Pattern Recognition/Pooyan Safari): "The deep learning philosophy is fed by the assumption that the truth is not a solid monolithic thing, but rather a distributed concept flows along and among variety of things, phenomena, categories or here units and layers, which might all be integrated to superficially form a solid entity, event, being. One (human or machine) might not realize the whole, or even sometimes a part of it, without trying to perceive the overall structure and interactions between these single and monadic elements. Sometimes these individual elements have no meaning , each within itself, to us, however these are the individuals that form a complex, a whole." It's even more: "The job of a physicist or an applied mathematician is often to come up with a functional description of a phenomenon from first principles so that it's possible to estimate the unknown parameters from measurements and get an accurate model of the world. Deep neural networks, at the other end, are families of functions that can approximate a wide range of input/output relationships without necessarily requiring one to come up with an explanatory model of a phenomenon. In a way, you're renouncing an explanation in exchange for the possibility of tackling increasingly complicated problems. In another way, you sometimes lack the ability, information, or computational resources to build an explicit model of what you're presented with, so data-driven methods are your only way forward." (Deep Learning with PyTorch Essential Excerpts Eli Stevens and Luca Antiga). I come up with the understanding that logics and therefore explainability are both the output of thinking, while the thinking itself is unexplainable in essence, it only have a base, NN, where data can be self-organized, for logic to be founded on top of it. Since it first has to have organize model of the world. Also, using DNNs for tasks is not explainable since it's not built for that. The explainable part will be if we combine textual NN. The NN is actually a system of nested conditional operators, so it is not so unexplainable.

In Fig. 20 there is a scheme representing how current action is decided. On what it depends.

PID is based on closed-loop error correction control, where I=integral correcting based on accumulating error from the past, P=proportional is correcting proportionally to the current error, and D=differential correcting by the expected change depending on the slope direction.

Similarly position→velocity→acceleration are the changes in the previous measure within this chain. I.e., the differential future expectation, while the opposite direction is the source and calculated via integration (of the past).

MPC, IC [24] are control methods based on a known model, and they calculate control for future steps, finally implementing the first move/control. Each time depending on the current state.

MECA [7] depends on past till present, and it also introduces randomality. It has two units, 1st one is reactive, depending on the past learning, while the 2nd one is goal-oriented, non dependent on the past, and performs according to some higher objective(s).

MECA is actually represent the right hemisphere of the brain, which is not dependent on previous learning, and search for new insight for a given situation.
DNN, DBN and RL are learn either from past examples or from past experimentation (RL), but they all act at each moment as state-feedback, meaning DNN/DBN are give output specifically by the given input, similarly RL has specific action by specific current state.
RNN however, introduce the ability to account for not only past learning but also past dependence for current reaction. Which is important for evaluation of the forming situation before our eyes.

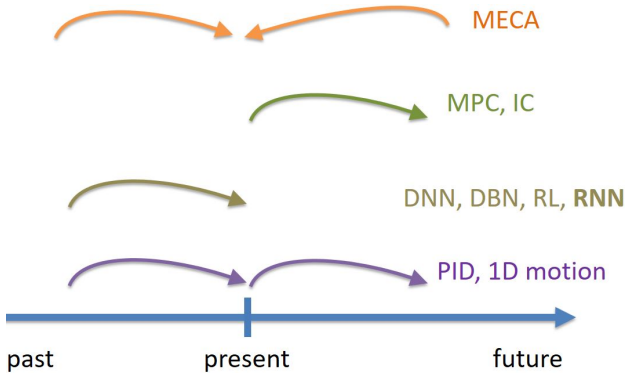


Fig. 20. How current action is decided, in different models.

5) Stimulated AGI (high objective): In humans, the brain is operate non-stop. When we sleep, close our eyes or meditate, or even by unaware physiological processes. But even cognitive operations work at all times. Meaning we have to excite the AGI, in times when its development from the input channels or from the self-organizing processes is terminated. It is due to the fact that any energy-dependent process in nature eventually dissipates, as it does here also. This effect occurs in adulthood when a person feels he has no significant cognitive development as he sensed in the earlier years. In relation to stability verse balance, the meaning is not to fixate in the middle of some range (e.g. between convergence and divergence), but to stimulate going to some edge, but keep balancing it using the contrary edge.

Other examples for running about the edges is in [65], where planning occurs in the middle, between fixation or concurrent data, and the dynamic/possible/random. Same idea goes for free-will as the mid of determinism and stochastic.

It also means that the system has another input beside the usual input channels. It has also some external control from a human user, either a direct control or a programmed one, that stimulate the AGI to keep the development process going.

It reminds the performance index or objective function from optimal control field. In reference to that term, the AGI also has to have such general objective, that manage the AGI. Just as anything in the world have a purpose and therefore a design, same is here. It can be either implicit or explicit. Philosophically it can be expressed also as some permanent deficiency, drive humans to fulfill and propagate.

As in NNs, where learning defined as the problem of improving some measure of performance when executing some task, through some type of training experience, similarly in AGI will be some internal performance index that strive to better the knowledge assimilation and organizing it in an efficient way. Additionally, an external goal to perform some human dictated tasks will be inserted directly into AGI or indirectly by commanding using communication.

As a part of this higher goal supervisor job, will be also handling attention [66].

6) Context generalization: Just as in vision, where we receive sensory information, but save it as a sequence in-order to identify some known pattern, similarly it is done with textual information, where we put each word in a context of a sentence, and than in the context of the last set of sentences, and finally in the context of the current speech and mood. Same can be done with any type of time-evolving information, such as the transportation network state [67], finally generalizing the situation in a hierarchical pyramid fashion [26].

The idea [26] is that at each instant, the AGI recognize the general state it is, after the gain of time-serial information, which can than as a feedback go down the hierarchy to control also how to collect the information more efficiently.

This idea can be implemented also on predictive behavior, for different time scales (from immediate response until long-range). But more important, hierarchical model can be also for operators on data that creates and updates our knowledge base. Starting from low operators to high operators, that can function as the organizers and the managers of AGI's structure or as called in psychology as meta-cognition.

7) Efficiency: AGI have to be efficient in the meaning that it has to stand in many tests. It must not be optimized only for some specific task, but rather have a multi-objective openness [68]. Optimization do not organize the data (but rather act upon some unknown data), therefore changing objectives to optimize have to perform new calculations from the beginning. But information organization take this case into account, to be efficient for many possible goals, such as efficient memory for fast retrieval, searching for best solution, searching for fast solution and more.

8) Efficiency verse Effectiveness: We should not construct AGI based on cumbersome or complicated models, since it hurt model's further development, as understood decades ago in rule-based modeling. And we have to build an AGI such that it 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 and a child do, and not demand from it to give always "correct" answers, but instead - as a human does - allow it to make a 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. As a more general point of view on the data he encountered during its lifetime.

In other words, as the comparison between serial thinking

verse parallel one [65], [69] show, that in serial one we prefer judgment and quick selection or fast results with emphasis on certainty, in parallel however it is fluent and non-judgmental in favor of fluency and multiple solutions and options together with their probabilities, i.e. allowing and welcoming 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, to deal with as variant knowledge as possible, the more efforts it take 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 special data and faster in results.

This is actually the difference between efficiency and effectiveness [68], 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 AI 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.

9) Growth Potential: This 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 AGI such that it can evolve and develop further independently, with the anticipation of it to reach some stages of growth.

The problem with MECA for instance, as an example for a complex cognitive structure or a rule-based structure, is that if it is designed with lot of components than it is prone to further growth in complexity if new situations occur to deal with [45]. At some point it might become so complicated that it may loose its reliability/credibility.

10) Reasonable AGI: There problem of using CNN with LSTM with attention over an image to generate text caption describing an image [70], is that it is not intuitive. Meaning, this is simplified method to bypass the actual complexity needed for this task. Even if it works, this is not the way it can scale the AGI to develop further. Since text description is actually high-cognitive operation, that requires the knowledge of words, linguistic grammar, and more-over it has nothing to do with attention over an image beside the single attention on the object(s) described.

Similarly, supervised NN can be described as sticking a pin in a middle of a big NN delivering a label, that by back-propagating influence the hierarchy of properties needed to be distributed between this label and the inputs. Not only that, but number of outputs represent classifications or clusters, which represent some concepts. But in reality there are huge amount of concepts, so real intelligence supposingly using NN should have huge number of outputs. I claim that most of the concepts suppose to be represented in the features spread in the network, rather being at the edge of the network.

Also in reality humans have no labeling at all. Same goes for language processing using documents.

These methods are working backwards: they start from highly complex data, such as language (even worse - an abstract terminology) and highly complex visionary scenery, and try to process it, with the deliberate intention that this data only need to be fitted, not understood.

It suppose to spread naturally from sensors to conscious and then to memory and so on..

In summary, in order for any AGI would work, it have to make sense.

ADD THIS: It seems we don't use supervised learning at all and reinforcement learning is just a primitive method of a more general one. At some point the baby start to use its motoric actuators (speech, muscles) but only after organizing the memory to grasp the skill for these in order to let it out. And when the actuators start to work, it can be added somehow as the input influencing the result we see.

The problem with most ai research either visual or textual that they work backwards. They always start from high-complexity data and try to reach it. Instead of simple to complex method as it should be.

Babies cry often if awaken (meaning interrupted sleep) perhaps because sleep mechanisms are more critical for them comparing to adults. Or perhaps just as any deficit such as food or any other need have to be satisfied fully and immediately.

ADD THIS:

כמו שכל ספירה ימנית משפיעה על השמאלית, החל מהראש (חכמה על בינה) כך גם חסד שמסמל את האור והיום לעומת דין המסמל את הלילה והדין שאדם נותן בשנתו. ביום לאדם יש פעילות ספיגה וקליטה של כל מה שארע ובלילה יש פעילות אחרת כשאזורי הקליטה מכובים. אז נכנס לפעולה פעילות הפוכה (שעם קליטה לא יכולה לעבוד בו זמנית כנראה כי אולי הן סותרות). פעילות בעיקר של סדר וניקיון של היום שעבר. כעת במקום כיוון ספיגה זה עובד לכיוון ההפוך. אולי פריקה לעומת טעינה. כנראה יש זיכרון אפיזודי של אירועים שמתמלא במהלך היום וזקוק לפירוק. לכן אסור לאדם לא לישון. כי זה לא מאפשר לא לנתח ולארגן אירועים שכבר הצטברו אצלו. הוא חייב תהליך שינה כדי לפנות מקום לזיכרון היומי הזה. כלומר מחזור יום-לילה בעצם סוג של מאתחל את המח ליום חדש, אלא הפעם פעילות הסידור של הלילה יצרה איכשהו סדר טוב יותר של האירועים. סדר שמשפיע על כל סוגי הזיכרון שיש לטווח הארוך. גם אירועים גם הצהרתי וגם חושי. לכן כל יום יש התקדמות.

Fig. 21.

Just like back-propagating and feed-forward can not act simultaneously, similarly wakening absorbing data and process it cannot occur together with reordering the system itself. Otherwise it makes the brain inconsistent. During the day you act as you are. And you upgrade the system when its non working.

Should explain the difference between DNN and CNN for example object image recognition verse actual high-complexity abstract object recognition, where we look at a scene and more than simple recognize bodies in it but also recognize their behavior and functionality. A meaningful interpretation and understanding (show the picture and add it source). Than i explain the stick in CNN OR DNN unlike DBN that allows the wakening propagation move forward to all other parts of the brain such as memories and etc.

D. Principles from Exact sciences (low order)

1) Evolutionary Algorithms: Ant colony, GA and even Reinforcement Learning (RL) are examples for applying

randomization together with keeping/storing the best results gained so far [71]. Ant colony use random paths to find optimal solution, but preferring the most walked paths so far. GA is a bit less random in the sense that for a chosen population the new members in it are only a random mutation of the best members so far, but not a totally random members. RL on the other side starting as most stochastic trial-and-error exploration, but as the time passes there is less exploration and more adapting best rewarded actions.
All these methods represent a "calculated risk". Meaning a mid point between total randomness and total accuracy, e.g algorithmic.
One suggestion may be yielded from this to DNN back-tracking optimization algorithm: not to choose the steepest descent, but rather some fraction of it, in-order to leave some room for future learning and reduce over-fit to the data. Or to use regularization method that constraining the optimization problem (as if pushing the brakes on optimization) and reduces over-fit this way.

In any case, we want a fast updating weights method, very fast for huge NNs. Unlike precise GD/SGD, slow convergence but low number of examples N, we will use fast convergence and high N.

My simple belief is that humans are the only one have consciousness and experiencing the world via both feelings and mind. I do not think AGI can be built as a living agent just like humans, and even if do - I do not think it is an issue. My view of an AGI is simply as a processing unit. I do not mind if it do not understand as stated in the Chinese room argument or if does not have consciousness or feelings. All I care about is that it can solve problems and be creative. Its purpose, in my eyes, is to act as an engineer or a researcher. So it does not matter if it process data, while humans precieve it as knowledge, what matters is that I know the way human know, I instruct the AGI to do something, it process my request, do it as I wish, and finally its output come back to me, back to a human mind.

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

For example, VQA or QA are a dead-end in DL, since they cannot be evolved into AGI. It is because they are built upon supervised learning from a set of inputs and outputs, while in general QA the response can be any one of many potential responses, some of which are actually unknown (e.g. discoveries). I.e. QA isn't merely finding some function mapping inputs to outputs, because it involve non-deterministic (chaotic) functions, like rationality and imagination.

Actually, this approach is the opponent approach against rule-based methodology, such as solving a problem, while implicitly or explicitly accounting for a narrow range of scenarios, and then try to evolve the basic solution to build a more general solution, to take into account new scenarios, and so on.

We should notice the recurrent motif of dealing always with data and operations on it or its arguments. Hence the embodiment argument in AGI community and the well-known use of encoder-decoder systems implementing perception-action tasks.

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

2) Order: In my opinion it is all about order. We can see that we actually counteract nature. Since 2nd law of thermodynamics tells us that everything tends to disorder, or in scientific form: entropy is always increasing. But human brain is doing the opposite. It searches for patterns, and not strictly low-level patterns, but actually more comprehensive and general as possible, with the purpose to find the order in everything. Maxwell's demon thought experiment only support this claim. In this experiment imaginary demon can open a door to allow fast gas molecules to go right and slow ones go left, thus making the entropy higher, or create more order, thus contradicting the 2nd law. But notice that the demon has intelligence, to classify and measure the particles. So it's no different than a person cleaning his messy room. He simple apply his intelligence to create order out of disorder, thus contrary to nature, he impose order into nature. In another article, they talk how important is to find the places in life where we waste lots of energy, and then plan how to reverse these losses. In another words, it's simply intervene in the nature which always goes to disorder, and apply human mind to insert order in it, thus rising against it the entropy. In contrast to nature's "laziness", searching for minimum energy, humans and living things are inserting energy all the time, otherwise the brain would converge to some minima and stop.

At first, prediction might have been the main thing in AI, but then overfitting was discovered, hence tried to reduce its importance in training by using some perturbation and noise in the system, via regularization, dropout and more. Assuming the bigger goal is lower entropy in data, then prediction is only a tool, or the secondary goal. The supervising goal (such as order) evaluate itself by prediction. In my opinion, efficient order is the main goal, because we want fast reaction to circumstances. Hence the need for efficient organization of data. Prediction is only a tool to get this done. E.g. using simulations of the data itself or simulations of possible scenarios or questions, to get a correct answers. One possibility is some kind of rearrangement of the NN

itself, to be consistent with previous experiences stored in the episodic memory. - Order importance expressed also in the different representations and embeddings in DL. Where as it is well known, a good representation for a given problem is the most part of the solution.

Since we concluded that order and making sense is more important then prediction of tasks, we can apply this idea on multi-tasking. Instead of having model to learn weights for specific task, it should learn or relearn for multiple tasks in advance or as they introduced over some time. Similar idea is implemented in MTL, where we have shared parameters of different tasks. For example for different questions about some subject, there will be a different focus or different association with specific features, such as it is done nowadays with attention mechanism. It would be great if there were some way to accumulate the learning of different tasks instead of relearning everything when a new task appear. In my opinion, during sleep or relaxation, when the brain is free of abundant external stimulations, it should perform predictions tasks which are should be versatile and even self-supervised, to produce better generalization and usability of given data. Self supervised means the use of short-term memory to be assimilated into long-term memory.

LeCun in [72] support this claim, where self-supervision 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, exactly as association connections connect all possible relevant concepts.

I think that the major issue in AI, which prevent it to reach AGI is the issue of will (or intention/meaning when it is sent to recipients). Humans always have some will, and it is like system's objective function tell it that there is a process with a destination. It is hidden in the need for big data and attention in DL models, as an implicit will that determine what should be the correct output. It is also hidden in in the curiosity-driven RL models, where curiosity is only one specific type of will, and in humans it is purposeful, i.e. specific. Similarly in exploration of RL. Instead, will should be addressed directly and explicitly.

One intuitive way to insert will, is simply externally in a specific input channel only design for that. This is the explicit way. Complementary or as an alternative, we can use communication and recent context.

Reminder about intuition behind attention: it is about how to calculate context. Instead of summarizing information flowing from input in a fixed manner, as in RNNs or CNNs, we perform selective summary of elements in the input (which is image patches in vision or a sequence of tokens in text), which determined via similarity to some query, hence a dynamic context. The queries are different in different NN architectures. E.g. in transformer's self-attention we query the elements themselves; in multi-modal the queries are intra-modal (self) and inter-modal (cross). This idea is implemented also in memory NNs, such as NTM [73] and DNC [74]. It is very similar to associations, since it based on similarity, and when several layers of attention exist, then it is like skipping from association to the other. This effect implemented also in multi-hop attention model. DNC include two important factors we use in our associative idea also: time of access/write and usage level, which may in our case be combined into some importance measure of the association elements. ME-LSTM model [75] also introduce the need of fusing LTM with STM.

I think that current meta-learning is wrong at the core, considering having shared parameters learned between tasks. Because most of the skills we learn not related to previous tasks at all. Hence it should be progressive.

Logic is rigid. Therefore logic-based AGI such as cognitive architectures can act merely as inspiration, but not as means to construct AGI. AGI most probably should be based on neuro-sciences, since in order to handle a large variety of scenarios and properties that it should have - it must be flexible, fluid, adaptive, evolving and alike.

V. 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] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[6] 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.

[7] 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.

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

[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] 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.

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

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

[21] 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.

[22] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

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

[24] S. Komarovsky and J. Haddad, "Robust interpolating traffic signal control for uncertain road networks," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3656–3661.

[25] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.

[26] 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.

[27] B. J. Baars and N. M. Gage, *Cognition, brain, and consciousness: Introduction to cognitive neuroscience*. Academic Press, 2010.

[28] 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.

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

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

[31] K. Raizer, A. L. Paraense, and R. R. Gudwin, "A cognitive architecture with incremental levels of machine consciousness inspired by cognitive neuroscience," *International Journal of Machine Consciousness*, vol. 4, no. 02, pp. 335–352, 2012.

[32] D. Hassabis, D. Kumaran, C. Summerfield, and M. Botvinick, "Neuroscience-inspired artificial intelligence," *Neuron*, vol. 95, no. 2, pp. 245–258, 2017.

[33] R. R. Gudwin, "Evaluating intelligence: A computational semiotics perspective," in *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*, vol. 3. IEEE, 2000, pp. 2080–2085.

[34] S. Ribeiro, A. Loula, I. de Araújo, R. Gudwin, and J. Queiroz, "Symbols are not uniquely human," *Biosystems*, vol. 90, no. 1, pp. 263–272, 2007.

[35] B. Khayut, L. Fabri, and M. Avikhana, "Modeling of intelligent system thinking in complex adaptive systems," *Procedia Computer Science*, vol. 36, pp. 93–100, 2014.

[36] A. Cortese, B. De Martino, and M. Kawato, "The neural and cognitive architecture for learning from a small sample," *Current Opinion in Neurobiology*, vol. 55, pp. 133–141, 2019.

[37] P. R. Lewis, A. Chandra, S. Parsons, E. Robinson, K. Glette, R. Bahsoon, J. Torresen, and X. Yao, "A survey of self-awareness and its application in computing systems," in *2011 Fifth IEEE Conference on Self-Adaptive and Self-Organizing Systems Workshops*. IEEE, 2011, pp. 102–107.

[38] J. M. Chein and W. Schneider, "The brain's learning and control architecture," *Current Directions in Psychological Science*, vol. 21, no. 2, pp. 78–84, 2012.

[39] D. S. Levine, *Introduction to neural and cognitive modeling*. Routledge, 2018.

[40] W. Duch, R. J. Oentaryo, and M. Pasquier, "Cognitive architectures: Where do we go from here?" in *Agi*, vol. 171, 2008, pp. 122–136.

[41] 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.

[42] P. Gora and M. Bardoński, "Training neural networks to approximate traffic simulation outcomes," in *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2017, pp. 889–894.

[43] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[44] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.*, "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[45] J. C. Spall and D. C. Chin, "Traffic-responsive signal timing for system-wide traffic control," *Transportation Research Part C: Emerging Technologies*, vol. 5, no. 3-4, pp. 153–163, 1997.

[46] H. Cheng, D. Lian, B. Deng, S. Gao, T. Tan, and Y. Geng, "Local to global learning: Gradually adding classes for training deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4748–4756.

[47] 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.

[48] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017.

[49] M. Long, Z. Cao, J. Wang, and S. Y. Philip, "Learning multiple tasks with multilinear relationship networks," in *Advances in neural information processing systems*, 2017, pp. 1594–1603.

[50] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. Feris, "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5334–5343.

[51] 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.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

[56] 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.

[57] 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.

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

[59] 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.

[60] 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.

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

[62] L. Zhang, S. K. Shah, and I. A. Kakadiaris, "Hierarchical multi-label classification using fully associative ensemble learning," *Pattern Recognition*, vol. 70, pp. 89–103, 2017.
[63] F.-Y. Wang, X. Wang, L. Li, and L. Li, "Steps toward parallel intelligence," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 4, pp. 345–348, 2016.
[64] [Online]. Available: http://sysbook.sztaki.hu/sysbook6.php
[65] E. De Bono, *Parallel thinking*. Random House, 2016.
[66] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1480–1489.
[67] M. Bielli, G. Ambrosino, M. Boero, and M. Mastretta, "Artificial intelligence techniques for urban traffic control," *Transportation Research Part A: General*, vol. 25, no. 5, pp. 319–325, 1991.
[68] [Online]. Available: https://simania.co.il/bookdetails.php?item_id=145306
[69] E. De, *Parallel thinking: From Socratic thinking to de Bono thinking*. Penguin Books, 1995.
[70] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.
[71] P. Lucic and D. Teodorovic, "Transportation modeling: an artificial life approach," in *14th IEEE International Conference on Tools with Artificial Intelligence, 2002.(ICTAI 2002). Proceedings.* IEEE, 2002, pp. 216–223.
[72] Y. Bengio, Y. Lecun, and G. Hinton, "Deep learning for ai," *Communications of the ACM*, vol. 64, no. 7, pp. 58–65, 2021.
[73] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *arXiv preprint arXiv:1410.5401*, 2014.
[74] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou *et al.*, "Hybrid computing using a neural network with dynamic external memory," *Nature*, vol. 538, no. 7626, pp. 471–476, 2016.
[75] P. Liu, X. Qiu, and X. Huang, "Deep multi-task learning with shared memory," *arXiv preprint arXiv:1609.07222*, 2016.