Dynamic and Evolving Neural Network for Event Discrimination

Shimon Komarovsky

August 1, 2022

Dynamic and Evolving Neural Network for event discrimination

Shimon Komarovsky[1][0000−0002−9036−0282]

Technion - Israel Institute of Technology, cm5099@yahoo.com

Abstract. Artificial general intelligence (AGI) should be founded on a suitable framework, e.g. a rule-based design or Deep Learning (DL). Here we choose the DL to be the basis for AGI. An appropriate AGI is defined, followed by its appropriate DL implementation. We introduce an AGI, in the form of cognitive architecture, which is based on Global Workspace Theory (GWT). It consists of a supervisor, a working memory, specialized memory units, and processing units. Additional discussion about the uniqueness of the visual and the auditory sensory channels is conducted. Next, we introduce our DL module, which is dynamic, flexible, and evolving or growing. It can be also considered as a Network Architecture Search (NAS) method. It is a spatial-temporal model, with a hierarchy of both features and tasks, tasks such as objects or events.

Keywords: Deep learning · General intelligence · Evolving · Growing.

1 Introduction

DL, as one of the Artificial intelligence (AI) approaches, is not as fully exploited as it could be. First, deep neural networks (DNNs) are passive models, since they have a fixed structure, while in reality there are dynamic processes, such as the neurons' construction/destruction in the brain. Second, Learning in DL is simply a categorization process without involving any thinking or imagination. Next, a successful DL model (DLM) requires its designers to know the system, i.e., apply implicit or explicit prior knowledge in the DLM. Moreover, a carefully designed rule-based system may outperform a DLM, due to its dataset limitation, while a rule-based system is designed for much broader and more diverse scenarios. Finally, DL is highly task-specific. Even multi-tasking in DL requires all tasks to be pre-defined. However, real AGI can generalize not only to unseen data but also to unseen tasks (as in transfer/continual/meta learning). Nevertheless, we propose a dynamic and flexible DLM that can be extended to AGI.

Next, we present an AGI architecture and a DLM, which can function as a module in this AGI architecture, e.g. in the perception/actuation module.

Please note that this paper presents a short version. The DLM and especially the AGI are preliminary ideas, and described roughly and generally, without mathematical details or implementation/results.

2 Proposed AGI model

A general AGI model sketch is shown in Fig. 1. This AGI is based on GWT [48,53,60] , describing a multi-agent system, where the agents are local controllers behaving reactively, and competing with each other over access to the working memory.

Our AGI, however, has no competition among its different and independent modules, i.e. processors and memories. Instead, it has centralized control with different elements, where each element has a specific function.

Examples of similar cognitive architectures are in Appendix 4.

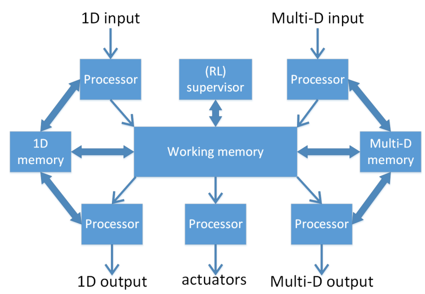


Fig. 1. AGI proposed architecture

2.1 AGI function

Here the function of the proposed AGI in Fig. 1 is described.

As in humans, our AGI uses 1D (audio) input and 3D (visual) input, however, it also uses them as outputs. Moreover, the visual channel can be extended to 1 or more dimensions, depending on the environment our agent is deployed in.

There is separate sequential processing of 1D and multi-D data, for feature extraction and categorization of objects (static entities) or events (dynamic entities). Next, these objects/events propagate into the WM. Finally, an output is produced either through the 1D or the multi-D channel. If the output is an emerging idea/thought, it can be expressed via a 1D channel, similarly to humans describing verbally their inner thoughts to the outer world. Alternatively, it can be expressed via the multi-D channel, thus can be regarded as *screening imagination*, which is like projecting the current thought into a screen.

Additionally, 1D information (such as language) has a shared memory for input, output, and WM, denoted as 1D memory. This is also true for the multi-D information. The bidirectional arrows in Fig. 1 represent the acquisition (reading) and the update (writing) operations with the storage module.

The output communication of 1D and multi-D information can have various modes, such as continuously monitoring thoughts or waiting for a meaningful output. In addition, the AGI may have a degree of independent choice of when to interact and through which of the two channels.

This particular AGI is based upon Stimulus-Response behavioral theory [70], which states that the mind can be communicated with, although unobservable. This assumption is similar to the Chinese Room Argument, since there is only direct access to the output of the agent and not to the operations within. In other words, there is no explainability over the AGI's inner operations (it is a black-box), and so only the output can be analyzed. It is referred to as *intelligent behavior*, which is also expressed by human productivity over time, in fields such as science, psychology, and technology.

(a) AI identifying.

(b) AGI interpretating.

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

This AGI does more than static/dynamic object identification or scene understanding [39], as in DL. It extends to the temporal dimension, by including: events, objects' behavior and function, associations from past experiences, etc. It is illustrated by comparing the current AI and the proposed AGI, in Fig. 2.

Just like Einstein's relativity theory, space and time are not separated, but treated as one whole concept. Similarly, our DLM is based on this hypothesis.

2.2 AGI characteristics

Firstly, we consider AGI's main purpose to be organizing information to be utilized optimally in a variety of tasks. Hence, the self-supervision approach is a suitable tool to estimate this main goal. Additionally, DNN is an efficient model and memory structure, which can achieve this goal, in the sense that it organizes the data with the intention of recovering it later, see more in Appendix 5.

Secondly, we advocate that efficiency is more important than effectiveness, in AGI, since it is about the exploitation of available resources, while effectiveness is about how well a goal is achieved [1], e.g. the common attitude in DL to compare performances.

Finally, other characteristics an AGI should have are those imitating humans, such as having human guidance and support as in infant-parent and student-teacher interactions, having a correct teaching order (simple to complex), and the ability to grow/evolve in compulsory stages.

2.3 Two information types in AGI

Here we discuss and propose a rationale behind the unique functioning of the visual and auditory channels.

Firstly, we examine why humans do not possess an imagery output tool like the multi-D output we permit in our AGI. One can argue it would hurt our basic desire for privacy, but then just as we choose whether to talk or not, we can similarly choose when to turn this tool on. Another argument could be due to evolutionary survival reasons. Our current opinion is that the world we see with our eyes is what we all agree upon. Other than that, our inner models of the world are totally different.

Secondly, we reflect upon the reasons for humans not having a symbolic or linguistic channel to be objective as vision, i.e. why we end up with inner and unique symbolic representation. We think it is because language is highly context-dependent, and since each person has different contexts along his life, or different experiences, then he develops a different meaning/feeling/understanding of the objective concepts we all agree upon. Hence, the concepts we use in external communication are objective and common to all people, but their interpretation is different for each one. Therefore, the visual perception purpose is nothing but the objective agreement for effective communication between humans, realized via language. In other words, vision is not the main communicative channel for us, though, deaf people can bypass it by using sign language and textual format.

Consequently, the purpose of having two channel types is to distinguish the outer and inner world that the agent interacts with. Furthermore, humans (as should be followed by AGI) base their inner representation on spatio-temporal events, or operational language. A language comprised of objects, actions, and attributes, and expressed by words/symbols. Therefore spatio-temporal information can be transferred to humans not only by the static/objective world, but even more broadly by language. Agents denoted as green circles, communicating via 1D and individually perceiving multi-D input are illustrated in Fig. 3.

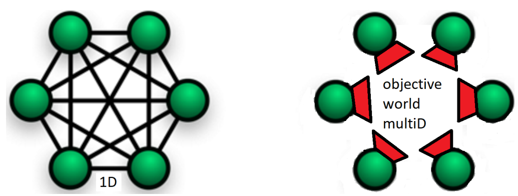


Fig. 3. Objective (right) verse Subjective inner representation (left).

3 Proposed DLM

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

Any DLM requires some prior knowledge, also known as inductive bias. Then due to difficulties with matching the most proper prior knowledge to each specific problem we encounter - many studies try different hyper-parameters or architectures, to get better performance, e.g. they use Network Architecture Search. See more about it in Appendix 6.

Therefore, the DLM we propose is adaptive for continuous learning and can serve also as a NAS method.

3.1 Proposed DLM function

Our proposed DLM is based on the inductive bias principle. It states that small data requires simpler model while bigger data requires a more complex one. Complexity in DLMs is expressed in the NN size. Hence, assuming gradual learning like in infants, we propose evolving DLM, starting from small NN, extending successively to a bigger one, following abstraction, while encountering new data.

In the following, we describe our DLM evolution with comparison to an infant, while perceiving a spatial-temporal type of data.

We presume, that an infant does not have any supervised learning at the beginning of his life, but rather an unsupervised one. Only later that he fuses multi-modal information about objects and their meaning.

The first thing he does is segment the time period into simple events. But he starts with a single event detection (e.g. his total waking period) through some initial DNN with several layers. See Fig. 4(a).

After a while, when enough counts detected the single event, a split of this event is performed into two (or more) classes of events, e.g. day and night, see Fig. 4(b). Counts are the number of times the output class was triggered. Now, the agent can differentiate two events, sharing the same features. Later it can extend the number of events, and recognize as many events as necessary. Consequently, it is an adaptive NN structure, adaptive by necessity.

At some point of evolution, when connections (weights in DNN) and event identification (output layer's counts) are strengthened and established, the model can change its attention or free its resources, since the given level had become more automatic, similar to the idea in [32]. It can now build a new layer/level on top of the previous ones, if a simultaneous re-occurrence of several events is detected. For example, the re-occurrence of seeing the mom appearing and preparing herself to give milk suggests to the infant that it is a composite event on its own, see Fig. 4(c), where yellow=visual sensors, and green=neurons.

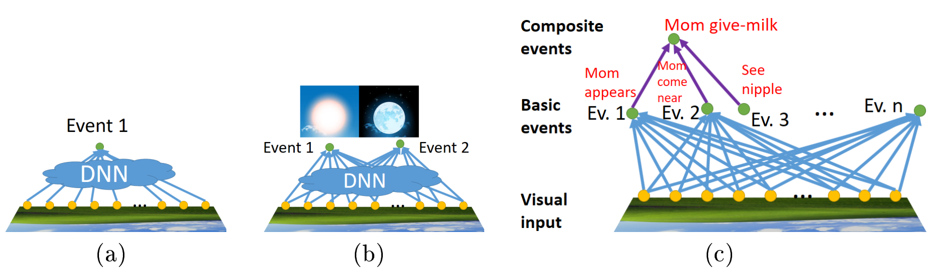


Fig. 4. Neuron separation and composition in the proposed approach (Ev.=Event).

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

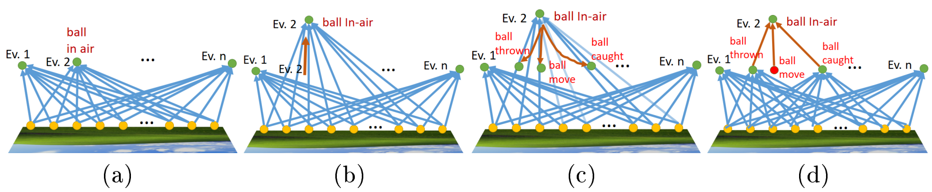


Fig. 5. Neuron decomposition in the proposed approach (Ev.=Event).

Decomposition is a highly uncertain operation, since it is unknown whether an event is compositional, and if it does - how many events it consists of, and which of these events are new and which are not. There are numerous ways to deal with it, e.g. see studies in 3.5, but it is out of the scope of this paper.

For this dynamic algorithm to work, the number of visits has to be stored for each weight (edge in DNN) and each neuron (node). If scalability is an issue for large DNNs, the visits memorization can be reduced from being stored for each neuron to being stored in each cluster of neurons in a large enough DNN.

Furthermore, the visits can be counted during *waking* periods (when the DNN is fixed), and the structure update can be done during *sleeping* periods, when there is no stimulus from the sensors, while the trajectory frequency within the DNN is stored in the neurons themselves, as mentioned above. The rate of structure changing can also be modeled with a learning rate as in RL, where at first it is mostly exploration (i.e. fast NN growth), and then lesser exploration and more exploitation. Finally, a finite number of nodes and connections is presumed, i.e. limited resources (so that it would not grow infinitely), thus resulting in adjusting the learning rate accordingly.

Finally, additional aspects for the DLM are presented in Appendix 7.

3.2 Advantages of the proposed DLM

This approach is self-supervised and not unsupervised, since it is not about clustering into a pre-defined number of categories. Here, similar to NAS, the number of categories and connections are all dynamic, and change according to the decision of some supervising algorithm.

Another reason for this dynamic algorithm is that real intelligence does not end up with categories like cat/dog (it evolves into more complex models). Moreover, most AI research works backward. It always starts from high-complexity data and tries to learn it from scratch, instead of simple to complex learning as it should be in an evolving AGI.

Additionally, this dynamic algorithm is less computationally expensive since it has fewer connections compared to FC NN, similar to sparse NN.

Finally, NAS is used to find some optimal hierarchical structure of features represented via neurons for a given dataset. Thus, it is probably wrong to guess the number of best-describing features at each layer. Dynamic NN deals with this issue, by keeping only the relevant and true features/events.

3.3 Task hierarchy in the proposed DLM

The extending of classes converts this DLM into a hierarchical multi-class DNN. Such DNNs exist in the literature. In [12] we have feature layers and then task layers. Sometimes these layers can be mixed up. Labels' structure can be found separately from the model [14, 47], or as a part of the model [27, 39, 46]. All the tasks can be learned over one classifier, i.e. globally or in the last layer of the NN [14, 68] , or alternatively, intermediate tasks can be inserted inside the NN, i.e. locally [12, 46, 57, 71] . The structure can be learned from the data [39], e.g. by unsupervised clustering of the labels via some similarity measure [47], or it can be imported from external knowledge base [20], or used to change this structure [14, 27].

Similar to these papers, additional features can be inserted between task layers in our proposed DLM, for example.

Additionally, unlike features that are distributive representatives of data, holding only some piece of the actual information, tasks are end-point independent data representatives, thus ruining in a way, NN's distributive nature.

However, this is not their main drawback. The fact that they are informational points - enforces a huge memory, since we need lots of them to represent a huge amount of terms/concepts. As opposed to a small group of inter-related features, which can characterize an enormous amount of input data. Thus, at some point, replacing/converting tasks with/into features should be considered.

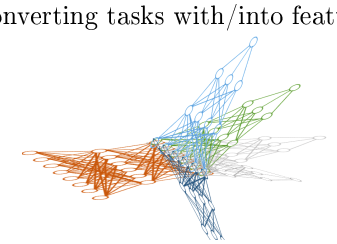


Fig. 6. Branching due to multiple hierarchies.

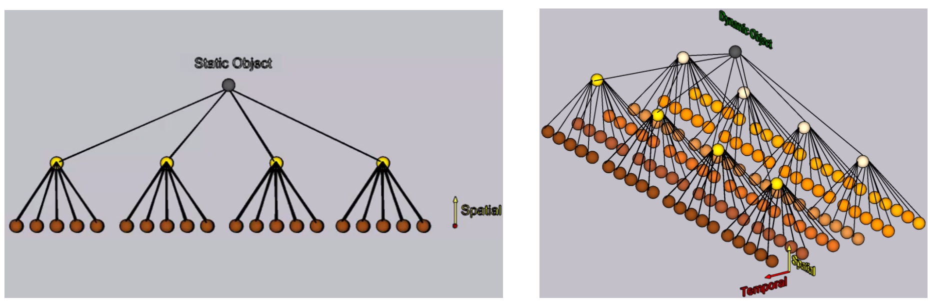
Generally, there may be different hierarchies besides compositional ones, e.g.: family tree, parts of speech, table of contents, topics, and sub-topics. One solution could be, is for the evolution to develop into different hierarchies, just like tree expansion: in different locations of a given NN and in different structures. An illustration of multiple hierarchies formed in a given NN is in Fig. 6.

3.4 Temporal dimension of the DLM

Until now the presented DNNs in the proposed DLM were represented via static structure, i.e. a single simultaneous set of inputs produced a single simultaneous set of outputs. In other words, there were no recurrent connections to include a temporal sequence of inputs.

Usually, spatial-temporal models combine CNN with RNN in different ways. Either separately: CNN→RNN or interchangeably: CNN→RNN→CNN→RNN... Another way is to separate CNN and RNN to separate inputs, e.g. textual for RNN and visual for CNN, with a fusion module at the end. In conclusion, event tasks, such as classification/clustering, can be done using the methods above.

Nonetheless, regular FC DNNs are used for spatial object tasks. But if our goal is to extract features along the temporal dimension also, a simple addition of recurrent connections could be made. Alternatively, an extension of the DNN could be done to include a temporal dimension, without changing the spatial dimension, i.e. orthogonal to it. See Fig. 7 for static and dynamic object tasks.



(a) (Front view) Objects tasks (b) (Side view) Events tasks

Fig. 7. Spatio-temporal DNN model.

In Fig. 7 it is shown a FC NN. However, if required, it could be specialized in different ways, e.g. by shared connections/parameters or convolutions. And it can be done for either the spatial or temporal dimensions, or both.

3.5 Related Work

Several topics are involved with our DLM: continual/lifelong learning; unsupervised learning, specifically deep and non-deep clustering; event detection; multi-label and multi-task learning; Network Architecture Search, and more.

From the aspect of our task, video recognition tasks such as event detection [2, 38, 54, 69, 72] is a large topic in computer vision, and the most relevant to our DLM, whose task is the continual refinement of events. However, these tasks mostly involve batch learning, not continual learning, and utilize fixed architectures.

One practical application of our task, is for navigating robots to recognize events, e.g. in [50]. However, they use time series of sensor and motor signals to recognize important events, i.e. they do not use fully visual spatial data.

From the aspect of our architecture, similar models are belong to the family of growing networks [6,33], e.g. Incremental Grid Growing (IGG) [9], Growing Cell Structure (GCS), Growing Self-Organizing-Map (GSOM) [3], Growing Neural-Gas (GNG) [22], and their variations [4, 62, 64–66]. They all are unsupervised methods, learning the data distribution. However, they are based on shallow NNs with designed features [19,62], while we are focused on DNNs that automatically extract features, to allow learning of more complex and diverse events.

Also, some of the methods above [6], utilize an age counter, similar to the number of visitations in our model, which in general can be extended to other counters, holding additional information for better clustering.

Nevertheless, there are variations of SOM that produce non-flat data structure, e.g. the growing hierarchical SOM (GHSOM) [6, 45, 52], which induces hierarchical bias over the data to be learned. However, it implements only a top-down generating hierarchy, which is equivalent to our decomposition operation and can act as a legitimate implementation of this operation. We also implement bottom-up operations such as splitting and merging. We actually construct the hierarchy bottom-up, and the top-down is just an additional option.

Hierarchical clustering is usually illustrated via dendrogram, and it exists also in other models, such as in Linkage based clustering, Tree-Structured SOM, and Hierarchical Feature Map [6, 52].

Nevertheless, [62] combines the two aspects, by using GSOM for anomaly detection in changing surveillance scenes, i.e. same task in similar online settings as we have. However since they use the shallow NNs described above, the features are engineered, in this case behavioral features of the scenes. Moreover, its growing feature is used only for adapting to changing events, i.e. to find anomalies in a changing environment. It is not made for gradual learning of events. Also, unlike the anomaly detection task, our task is to learn normal recurrent events.

All the methods above use different heuristics [4] to improve clustering in different tasks. Additionally, the search for the closest neuron to a given input (like in k-nearest neighbor clustering) is the most expensive task, a step that is absent in our approach. Finally, these methods, including ours, are of the clustering type, and all have in common the problem of how to choose the suitable measure/distance. Hence, a more adaptive approach is needed, e.g. a deep clustering topic that exists in DL.

Similarly, our DL approach, suggests the hierarchy will not include only the neurons representing events, but also feature neurons in-between, to enable more flexible learning and clustering of events.

Besides, there are growing networks for supervised learning [18, 44, 51] and semi-supervised learning [51, 64, 67], especially for continual learning to avoid catastrophic forgetting [49]. Networks that involve both growing and pruning, such as Progressive Neural Networks (PNN) [49], Dynamically Expandable NetworkS (DEN) [49, 74], and DeepDPM [56].

3.6 Contribution

Finally, the novelty/contribution of this paper is the notion that spatial-temporal dimension is inseparable, hence it should be learned as it is right from the start, contrary to the object detection tasks and alike. In addition, the learning must be gradual, continual, and unsupervised all the time, and as our DLM demonstrated, it must also practice gradual growth accordingly. Both of these principles are essential for an AGI agent. Consequently, some ideas were formed from the principles above, and should be refined further.

References

1. https://simania.co.il/bookdetails.php?item_id=145306
2. Aakur, S.N., Sarkar, S.: A perceptual prediction framework for self supervised event segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1197–1206 (2019)
3. Alahakoon, D., Halgamuge, S.K., Srinivasan, B.: Dynamic self-organizing maps with controlled growth for knowledge discovery. IEEE Transactions on neural networks **11**(3), 601–614 (2000)
4. Amarasiri, R., Alahakoon, D., Smith, K.A.: Hdgsom: a modified growing self-organizing map for high dimensional data clustering. In: Fourth International Conference on Hybrid Intelligent Systems (HIS'04). pp. 216–221. IEEE (2004)
5. Aqib, M., Mehmood, R., Alzahrani, A., Katib, I., Albeshri, A., Altowaijri, S.M.: Smarter traffic prediction using big data, in-memory computing, deep learning and gpus. Sensors **19**(9), 2206 (2019)
6. Astudillo, C.A., Oommen, B.J.: Topology-oriented self-organizing maps: a survey. Pattern analysis and applications **17**(2), 223–248 (2014)
7. AutoML: Automl, https://towardsdatascience.com/how-to-apply-continual-learning-to-your-machine-learning-models-4754adcd7f7f
8. Bengio, Y., Lecun, Y., Hinton, G.: Deep learning for ai. Communications of the ACM **64**(7), 58–65 (2021)
9. Blackmore, J., Miikkulainen, R.: Incremental grid growing: Encoding high-dimensional structure into a two-dimensional feature map. In: IEEE international conference on neural networks. pp. 450–455. IEEE (1993)
10. Blazek, P.J., Lin, M.M.: Explainable neural networks that simulate reasoning. Nature Computational Science **1**(9), 607–618 (2021)
11. Brock, A., Lim, T., Ritchie, J.M., Weston, N.: Smash: one-shot model architecture search through hypernetworks. arXiv preprint arXiv:1708.05344 (2017)
12. Cerri, R., Barros, R.C., De Carvalho, A.C.: Hierarchical multi-label classification using local neural networks. Journal of Computer and System Sciences **80**(1), 39–56 (2014)
13. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407 (2019)
14. Chen, H., Miao, S., Xu, D., Hager, G.D., Harrison, A.P.: Deep hierarchical multi-label classification of chest x-ray images. In: International Conference on Medical Imaging with Deep Learning. pp. 109–120. PMLR (2019)
15. Chen, L., Alahakoon, D.: Neuroevolution of augmenting topologies with learning for data classification. In: 2006 International Conference on Information and Automation. pp. 367–371. IEEE (2006)

16. De, E.: Parallel thinking: From Socratic thinking to de Bono thinking. Penguin Books (1995)
17. De Bono, E.: Parallel thinking. Random House (2016)
18. Evci, U., Vladymyrov, M., Unterthiner, T., van Merriënboer, B., Pedregosa, F.: Gradmax: Growing neural networks using gradient information. arXiv preprint arXiv:2201.05125 (2022)
19. Feng, J., Zhang, C., Hao, P.: Online learning with self-organizing maps for anomaly detection in crowd scenes. In: 2010 20th International Conference on Pattern Recognition. pp. 3599–3602. IEEE (2010)
20. Feng, S., Zhao, C., Fu, P.: A deep neural network based hierarchical multi-label classification method. Review of Scientific Instruments **91**(2), 024103 (2020)
21. French, R.M.: Catastrophic forgetting in connectionist networks. Trends in cognitive sciences **3**(4), 128–135 (1999)
22. Fritzke, B.: A growing neural gas network learns topologies. Advances in neural information processing systems **7** (1994)
23. Galanti, T., Wolf, L.: On the modularity of hypernetworks. Advances in Neural Information Processing Systems **33**, 10409–10419 (2020)
24. Gallagher, J.M., Reid, D.K.: The learning theory of Piaget and Inhelder. iUniverse (2002)
25. Galván, E., Mooney, P.: Neuroevolution in deep neural networks: Current trends and future challenges. IEEE Transactions on Artificial Intelligence **2**(6), 476–493 (2021)
26. Gora, P., Bardoński, M.: Training neural networks to approximate traffic simulation outcomes. In: 2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS). pp. 889–894. IEEE (2017)
27. Gu, J., Zhao, H., Lin, Z., Li, S., Cai, J., Ling, M.: Scene graph generation with external knowledge and image reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1969–1978 (2019)
28. Gudwin, R., Paraense, A., de Paula, S.M., Fróes, E., Gibaut, W., Castro, E., Figueiredo, V., Raizer, K.: The multipurpose enhanced cognitive architecture (meca). Biologically Inspired Cognitive Architectures **22**, 20–34 (2017)
29. Gudwin, R., Paraense, A., de Paula, S.M., Fróes, E., Gibaut, W., Castro, E., Figueiredo, V., Raizer, K.: An urban traffic controller using the meca cognitive architecture. Biologically inspired cognitive architectures **26**, 41–54 (2018)
30. Ha, D., Dai, A.M., Le, Q.V.: Hypernetworks. CoRR **abs/1609.09106** (2016), `http://arxiv.org/abs/1609.09106`
31. Hashimoto, K., Xiong, C., Tsuruoka, Y., Socher, R.: A joint many-task model: Growing a neural network for multiple nlp tasks. arXiv preprint arXiv:1611.01587 (2016)
32. Hawkins, J., Blakeslee, S.: On intelligence: How a new understanding of the brain will lead to the creation of truly intelligent machines. Macmillan (2007)
33. Heinke, D., Hamker, F.H.: Comparing neural networks: a benchmark on growing neural gas, growing cell structures, and fuzzy artmap. IEEE transactions on neural networks **9**(6), 1279–1291 (1998)
34. Huang, W., Song, G., Hong, H., Xie, K.: Deep architecture for traffic flow prediction: deep belief networks with multitask learning. IEEE Transactions on Intelligent Transportation Systems **15**(5), 2191–2201 (2014)
35. Ioannou, Y.A.: Structural priors in deep neural networks. Ph.D. thesis, University of Cambridge (2018)

36. Koesdwiady, A., Soua, R., Karray, F.: Improving traffic flow prediction with weather information in connected cars: A deep learning approach. IEEE Transactions on Vehicular Technology **65**(12), 9508–9517 (2016)
37. Komarovsky, S., Haddad, J.: Robust interpolating traffic signal control for uncertain road networks. In: 2019 18th European Control Conference (ECC). pp. 3656–3661. IEEE (2019)
38. Kuehne, H., Arslan, A., Serre, T.: The language of actions: Recovering the syntax and semantics of goal-directed human activities. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 780–787 (2014)
39. Li, Y., Ouyang, W., Zhou, B., Wang, K., Wang, X.: Scene graph generation from objects, phrases and region captions. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1261–1270 (2017)
40. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: Learning to learn quickly for few-shot learning. arXiv preprint arXiv:1707.09835 (2017)
41. Lieto, A., Bhatt, M., Oltramari, A., Vernon, D.: The role of cognitive architectures in general artificial intelligence (2018)
42. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055 (2018)
43. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies **54**, 187–197 (2015)
44. Mixter, J., Akoglu, A.: Growing artificial neural networks. In: Advances in Artificial Intelligence and Applied Cognitive Computing, pp. 409–423. Springer (2021)
45. Nascimento, Z., Sadok, D.: Modc: a pareto-optimal optimization approach for network traffic classification based on the divide and conquer strategy. Information **9**(9), 233 (2018)
46. Nguyen, D.K., Okatani, T.: Multi-task learning of hierarchical vision-language representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10492–10501 (2019)
47. Papanikolaou, Y., Tsoumakas, G., Katakis, I.: Hierarchical partitioning of the output space in multi-label data. Data & Knowledge Engineering **116**, 42–60 (2018)
48. Paraense, A.L.O., Raizer, K., Gudwin, R.R.: A machine consciousness approach to urban traffic control. Biologically Inspired Cognitive Architectures **15**, 61–73 (2016)
49. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: A review. Neural Networks **113**, 54–71 (2019)
50. Prem, E., Hortnagl, E., Dorffner, G.: Growing event memories for autonomous robots. In: Proceedings of the Workshop On Growing Artifacts That Live, Seventh Int. Conf. on Simulation of Adaptive Behavior, Edinburgh, Scotland. Citeseer (2002)
51. Qiang, X., Cheng, G., Wang, Z.: An overview of some classical growing neural networks and new developments. In: 2010 2nd International Conference on Education Technology and Computer. vol. 3, pp. V3–351. IEEE (2010)
52. Rauber, A., Merkl, D., Dittenbach, M.: The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. IEEE Transactions on Neural Networks **13**(6), 1331–1341 (2002)
53. Reggia, J.A.: The rise of machine consciousness: Studying consciousness with computational models. Neural Networks **44**, 112–131 (2013)
54. Richard, A., Kuehne, H., Gall, J.: Weakly supervised action learning with rnn based fine-to-coarse modeling. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 754–763 (2017)

55. Ridella, S., Rovetta, S., Zunino, R.: Plastic algorithm for adaptive vector quantisation. Neural Computing & Applications **7**(1), 37–51 (1998)
56. Ronen, M., Finder, S.E., Freifeld, O.: Deepdpm: Deep clustering with an unknown number of clusters. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9861–9870 (2022)
57. Sanh, V., Wolf, T., Ruder, S.: A hierarchical multi-task approach for learning embeddings from semantic tasks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 6949–6956 (2019)
58. Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: European Semantic Web Conference. pp. 593–607. Springer (2018)
59. Shwartz-Ziv, R., Tishby, N.: Opening the black box of deep neural networks via information. arXiv preprint arXiv:1703.00810 (2017)
60. Silva, R.C.M., Gudwin, R.R.: An introductory experiment with a conscious-based autonomous vehicle. In: 4th Workshop in Applied Robotics and Automation (2010)
61. Spall, J.C., Chin, D.C.: Traffic-responsive signal timing for system-wide traffic control. Transportation Research Part C: Emerging Technologies **5**(3-4), 153–163 (1997)
62. Sun, Q., Liu, H., Harada, T.: Online growing neural gas for anomaly detection in changing surveillance scenes. Pattern Recognition **64**, 187–201 (2017)
63. Tian, Y., Zhang, K., Li, J., Lin, X., Yang, B.: Lstm-based traffic flow prediction with missing data. Neurocomputing **318**, 297–305 (2018)
64. Vasighi, M., Abbasi, S.: Multiple growing self-organizing map for data classification. In: International Symposium on Artificial Intelligence and Signal Processing (2017)
65. Vasighi, M., Amini, H.: A directed batch growing approach to enhance the topology preservation of self-organizing map. Applied Soft Computing **55**, 424–435 (2017)
66. Vasighi, M., Talebi, M., Ballabio, D.: A dynamic molmap approach for pattern classification in three-way data
67. Wang, G., Xie, X., Lai, J., Zhuo, J.: Deep growing learning. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2812–2820 (2017)
68. Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., Xu, W.: Cnn-rnn: A unified framework for multi-label image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2285–2294 (2016)
69. Wang, L., Zang, J., Zhang, Q., Niu, Z., Hua, G., Zheng, N.: Action recognition by an attention-aware temporal weighted convolutional neural network. Sensors **18**(7), 1979 (2018)
70. Watson, J.B., Meazzini, P.: John B. Watson. Il mulino (1977)
71. Wehrmann, J., Cerri, R., Barros, R.: Hierarchical multi-label classification networks. In: International Conference on Machine Learning. pp. 5075–5084 (2018)
72. Wu, C., Ma, Y.F., Zhan, H.J., Zhong, Y.Z.: Events recognition by semantic inference for sports video. In: Proceedings. IEEE International Conference on Multimedia and Expo. vol. 1, pp. 805–808. IEEE (2002)
73. Wu, Y., Tan, H., Qin, L., Ran, B., Jiang, Z.: A hybrid deep learning based traffic flow prediction method and its understanding. Transportation Research Part C: Emerging Technologies **90**, 166–180 (2018)
74. Yoon, J., Yang, E., Lee, J., Hwang, S.J.: Lifelong learning with dynamically expandable networks. arXiv preprint arXiv:1708.01547 (2017)
75. Yu, H., Wu, Z., Wang, S., Wang, Y., Ma, X.: Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. Sensors **17**(7), 1501 (2017)

76. Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: A graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 1234–1241 (2020)

4 Appendix - Cognitive Architecture

The diagram in Fig. 1 is also referred to as cognitive architecture, which represents an AI agent structure and functioning. For example, in the traffic control field, two studies [29, 48] implementing AI are based on distributed *cognitive* architectures [41].

The first study [48] is based on GWT, containing agents imitating four brain function types: (i) sensory type, holding positions and velocities of vehicles; (ii) behavioral type, determining the light of the traffic signal in some intersection; (iii) consciousness type, representing the WM and interacts with other brain functions; and (iv) motor type, executing the chosen signal phase for each intersection.

In the second study [29], an AI is implemented on a single intersection using a Multipurpose Enhanced Cognitive Architecture (MECA) [28], which was adapted for the traffic signal control problem. The design consists of a Cognitive manager, i.e. a special kind of agent, like a car or an intersection agent, managing a set of physical objects available on the Internet. These objects provide information about themselves and receive commands. MECA is composed of two independent systems communicating with each other: (i) a fast reactive system holding the input sensors and output actuators, which is suited for normal situations, and (ii) a goal-oriented motivational system suited for unexpected situations. Overall, the automatic reactive and conscious elements of MECA produce intelligent behavior.

These papers are rule-based designs using small-scale traffic networks, i.e. small data. However, for more adaptive and flexible designs, the DL is preferable to the rule-based design. Moreover, any cognitive architecture is limited and constrained by its structure. Therefore, we should have a wider view of it, thus considering the boundaries between the components to be not so well defined and perhaps changing.

5 Appendix - Order importance for AGI

5.1 Data organization

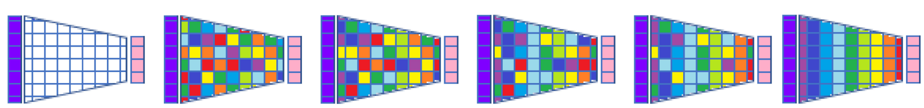
There is an issue to define the AGI problem. Unlike regular classification problems well defined in DNNs, it is difficult to know what is the overall AGI purpose. We believe for now, that it is to better predict or/and organize data. However, prediction by itself is only the test for how organized data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

See for example in Fig. 8, a sketch diagram, illustrating how supervised learning changes the parameters (weights and biases) during training in NN, where the given data is tuples of (input,output). The NN is structured hierarchically, i.e. it is features of features, etc.

Hence, starting from random dis-ordered weights (represented as rectangles inside the DNN), see Fig. 8(b), we gradually use inputs and outputs as magnets, for diffusion or rearranging the weights in a hierarchical way, where at the end of training (see Fig. 8(f)), the most input-related features will be closest to the input and most output-related features will be closest to the output. This idea is inspired by the visual feature maps in CNN, and by Information theory in [59].

[59] demonstrates diffusion of information in the encoder-decoder interpretation of a DNN. It shows that while propagating the DNN, the input is forgotten and the output becomes more dominant. It uses Bottleneck method where the learning compresses the input data and simultaneously captures relevant info (reduce noise).

Note that when training on random labels, then the task is actually merely memorization [76], 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. 8. Parameter evolution in supervised learning in NN (from right to left).

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

Beyond the demonstration of organization illustrated above, DNN is also an efficient model and memory structure, which not only organizes the data, but it does so with the intention of recovering it later, e.g. it enables recalling by associations, clues, imperfect or missing data. E.g. association in NNs characterized via the multiple features an object has, which are distributed all over the network. The DNN also compresses data, by storing a huge amount of patterns in a limited and much smaller set of weights and biases compared to the amount of all the samples it perceives.

For comparison, when we represent data in a tree-shape structure, then for different problems we must apply breadth-first-search (BFS), depth-first search (DFS), heuristic types of search, and so on. However, these methods are only rarely efficient, since we do not know which of these methods best fits for a given data. I.e., these methods work blindly on unorganized data. While the brain does the opposite - it solves problems quickly, after organizing data.

Another example, is when a person thinks of several solutions for a problem, then he/she does not do it via blind search, but via associative direct processing (no permutations involved at all). We have a survival type of processing; it cannot depend on a time-consuming search in some space of solutions.

Another example, is regular binary memory systems in computing devices, which are highly inefficient in the recovery phase. For example, it has to utilize a binary search for an exact match.

Hence, the encoding or the storing phase in a memory system is highly important. It affects how well and how fast the recorded data can be utilized. Moreover, data that was organized accordingly to previous but also presupposed/hypothesized future tasks, strengthen its efficient usability.

For example, [8] performed self-supervised training, which allows predicting any masked data from observed data. This demonstrates the idea of data being organized such that it can be tackled in multiple forms and scenarios.

5.2 Order in teaching

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

In DL the learning can be either incremental [12, 31] or simultaneous [57]. However, there is the catastrophic forgetting phenomenon in incremental learning [21], where the model abruptly forgets part of the knowledge related to a previously learned task as a new task is introduced. [31] for example, uses a successive regularization strategy to avoid this phenomenon.

5.3 Order through growth

Growth is an important AGI property, distinguishing it from non-growing AI methods such as some of the rule-based methods or the fixed structures in DL. It is a very important task of the AGI designer to plan the AGI agent such that it can evolve and develop further independently and autonomously.

Considering humans, Piaget's psychological theory [24] suggests that there are developmental stages in a child's development. Moreover, every stage is necessary as the basic/foundation level from which the next level can be reached. These stages accomplishment depends both on the environment (external) and on heredity (internal).

Similarly, we anticipate the AGI agent to reach some stages of growth.

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

5.4 Efficiency verse Effectiveness

We should not construct AGI based on cumbersome or complicated models, since it may hurt the model's further development, as it occasionally occurs in rule-based modeling. And we have to build an AGI without expecting it to be stable or perfect right away, immediately at first execution. This is a wrong attitude towards actual intelligence.

On the contrary, we have to give it the time to develop, just as an infant baby or a child does, and not demand from it to give always correct answers. In other words, just as a human does, we should allow the AGI agent to make mistakes, based on partial understanding, and learn from it not as a new input, but as another step in a more general and mature step of development, i.e., as a more general point of view over the data it encountered during its lifetime (like life-lessons after hardship and obstacles).

We can see a resemblance to this idea in the comparison between serial thinking verse parallel one [16,17]. In serial thinking we prefer a conclusive judgment and fast results with an emphasis on certainty. In parallel, however, it is about being fluent and non-judgmental in favor of fluency and multiple solutions/options together with their probabilities. I.e. allowing and embracing uncertainty instead of fighting it. The same is here, model-based methods and control are mostly designed for fast and 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 mumbles or pronounce poorly and has a gross motor skills (e.g. in movement and drawing). This observation supports the idea that the more the AGI is general, in dealing with diverse knowledge, the more effort it requires to adapt and learn this knowledge. And the opposite is true also - the more the AGI is specific, like current control/ML methods, the more it fits to be effective in narrow sets of data and it is faster in results.

Similarly, [35] mentions 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 a higher probability to the narrow range in which the data of interest lies. The same effect is in robust control verse nominal one [37].

All the above can be summed up to the difference between efficiency and effectiveness [1], where efficiency concentrates on the best exploitation of available resources, while effectiveness is about the performance measure of how well the goal is achieved. Consequently, the AGI agent must be efficient more than effective, since we are less interested in some specific desired outcomes, but rather a good thinking machine that can be validated only in the long run.

5.5 AGI design suggestions

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

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

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

6 Appendix - Prior knowledge and NAS

6.1 Prior knowledge in DNNs

Prior knowledge can appear in many forms. One form is structure's general type, such as CNN and RNN. As shown from the studies in the field of traffic signal control [5,43,63,73,75], these NNs present better prediction results in accuracy and stability compared to other ML methods, such as support-vector-machines (SVMs) and random forest. Nonetheless, it is due to being tailored to their particular problem, having fewer variables and containing more prior knowledge, compared to fully-connected (FC) layers of regular/vanilla NN. Hence, the transformation from FC to CNN or RNN can be viewed as localization, where the network structure is designed specifically for the data it handles. Another example of prior knowledge is when the input is in a graphical form, which requires an appropriate graph NN model to handle it.

Other forms are the hyper-parameters and the regularization method, e.g. dropout or constraints. Another form is the decision about sharing or grouping [58] or separating features/variables. For example, when traffic data is set apart from weather data [36], or when road features are separated from station features [34] and then fused later (how later is also a prior knowledge to be decided upon). Tasks can also be separated into groups [34]. Finally, [35] suggests sparsifying the NN, e.g. removing connections in CNN, or grouping/sharing parameters, results in fewer parameters, more prior knowledge and efficiency, and redundancy elimination.

However, these structures can be too restricted or best perform for narrow data variations. Hence, many studies try different hyper-parameters or architectures, to get better performance, e.g. they use Network Architecture Search. NAS consists of many approaches, e.g. AutoML [7], Neuroevolution, hypernetwork, Meta-learning (learning over learning) [40] and more, which are all searching for an architecture or adapting a given architecture, to better fit the data.

6.2 Network Architecture Search (NAS)

Motivation One motivation for NAS, can be found in [26], which demonstrates that NN is best for performance if its structure is appropriate to the data it is trained upon. Not too many neurons nor too few.

This optimal number of neurons and layers represents the most appropriate features to describe the trained data. Any other structure may result in some kind of spreading over some features that are not really representative of the data.

Hence, since the usual DNN only guesses the number of features at each layer, then it is required to check several structures. Dynamic NN deals with this issue, by keeping only the relevant and true features.

A similar approach is in [10], where instead of the usual differentiable DNNs they use concepts neurons, in what is called essence neural networks (ENNs), which allow symbolic reasoning and more. For example, it can learn rules to be extended and applied to other tasks or inputs, hence representing abstraction. It is as if differentiable (via GD) neurons are continuous features, while ENNs are the discretized version of it, to make the features represent meaningful and whole features.

This discretization idea may also explain why sometimes pre-trained unsupervised learning DNNs (e.g. SAEs or DBNs) work better than some weight initialization, before training on some supervised learning task(s). It is due to clustering, which perhaps may be into some concepts. Surprisingly, this is the exact method ENN uses to learn its (sub-)concepts.

A similar idea appears in dropout or sparse NNs (or attention), such as Google's PaLM, due to having the most desirable amount of neurons in a NN, i.e. that represent full features and not a mashup of them.

Research Firstly, in NAS, the search space for models is defined/restricted. For example, grid and random search are often used in hyper-parameter tuning.

NAS has various approaches, such as reinforcement learning (RL), Evolutionary Algorithms (EA), and Hypernetworks.

When EA is used for NAS it is referred to as Neuroevolution [25]. It is usually used in various search tasks, such as in the generation of DNNs, hyperparameters, NN building blocks, activation functions, and even the algorithms for learning (rules). For example, in NEAT [15], it is used to replace back-propagation with Genetic Algorithm (GA) or combine them both, in searching for weights.

A more efficient/faster approach uses weight sharing between proposed networks, instead of putting them in the search and training them from scratch, as in EA. It is done by sampling sub-networks from a large parent network where they force all sub-networks to share weights. Only the best final model is trained from scratch. Alternatively, only specific parts in NN can be modified, while leaving the rest unchanged.

Lamarckian Evolution also uses this idea for EA search.

Unlike EA and RL, the search in Differentiable Architecture Search (DARTS) [42], is defined as a differentiable and continuous problem. DARTS uses multiple categories, where the final architecture is discretized after the search is over.

In the hypernetwork approach [23], the weights are learned not by training the original NN but rather determined via other NN. That is, for every tested input, different weights are generated to predict the output. For example, in [30] there is a hypernetwork for both CNN and RNN as two ends of a spectrum, which allow it to be a relaxed weight-sharing approach and allows controlling the trade-off between the number of model parameters and model expressiveness.

In our opinion, Hypernetworks can be extended by constructing also a second network to learn the weights of the first one that learns weights for the main one. And so forth, strengthening the evolution in different time resolutions/scales (from slow to fast).

Another example is the SMASH method [11], where a hypernetwork is trained to predict network weights in one shot instead of training all candidate networks from scratch. First, a hypernetwork is trained to predict the weights of some given arbitrary network, then it randomly generates many architectures. Then, rather than training those models, a hypernetwork is used to obtain the weights. Finally, best performing architecture is selected and trained from scratch.

Another example is Spall's Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm [61] implementing two control loops. The lower loop is for real-time feedback control and an upper loop that updates the DNN weights in long-term periods. This demonstrates the idea of some parameters are changing frequently (e.g. weights) and some are changing very slowly (e.g. hyper-parameters).

Other optimization problems involve: model architectures; training schedule, e.g the optimizers themselves; and the data, e.g its order, size, diversity, and so on.

7 Appendix - Additional aspects for the DLM

7.1 Assumptions

The proposed DLM is based on the following assumptions:

* The task is to identify events.
* Gradualism assumption: the AGI generally, and the proposed DLM specifically, should be growing along with its experience with the world. It should grow internally, and following that - the complexity it encounters should also grow accordingly.

* Growing based on experiences: is embedded in the perception experiences of the DNN, i.e. in the visitations over neurons and weights, e.g. by counting these visitations/occurrences.
* Small data perform best in a small model, while big data perform best in a large model. See [13].
* The DLM perceives input and updates its weights accordingly to the DL methodology, i.e. feed-forward perception and back-propagating weight adaptation.
* In addition to the previous assumption - an external supervising algorithm, that controls the changes in the DNN architecture, is assumed.

7.2 Issues

Some of the issues that should be addressed are presented.

One issue can be in the recoverability, i.e. the ability to restore once deleted elements. One possible solution assumes that the infant first should grow up, and only later refine its categorization, i.e. first phase is only enlarging the network. Then, in a big enough network, a node/connection removal can begin, because then the agent accumulated enough confidence/experience.

Another issue in dynamic structure NN could be splitting or deleting features that are supposed to be frequent or rare, yet they should be left as they are. Similar issue is discussed in plastic neural gas method [55]. One possible solution is via relative frequency, i.e. to have some min-max range of relative frequency among neurons (e.g. for all of them or for each level), which will not be split/deleted. Only those extremely frequent/rare outside of this range would be split/deleted.

Finally, the proposed DLM has one obvious limitation. It has a single function, which is event discrimination, in different resolutions. Hence, its role in AGI is limited to perceptual or actuator modules, to differentiate procedural or conceptual events.

Moreover, the DLM tackles instant events, i.e. it does not generalize into classes of events. It only learns instances of basic and complex events, which becomes a combinatorial issue, due to the enormous possible combinations to define a composite event.

7.3 Optional additions to the DLM

The proposed DLM could also benefit from other optional additions:

* Besides being an event, the task/output could be also an object or an action.
* The inner neurons could also be updated.
* Allow reallocation of activated-together neurons to be in proximity to each other, to ease computation.
* The model should be constrained in all its adaptive parameters, such as the number of total layers and number of neurons per layer, to eliminate redundancy and encourage competition/trade-offs.

* The neurons can be adaptive in their function, thus having different functions, such as convolutional, recurrent, recursive, or attentional.