Dynamic and Evolving Neural Network as a Basis for AGI

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

Technion - Israel Institute of Technology, shiman@campus.technion.ac.il

Abstract. Artificial general intelligence (AGI) should be founded on a suitable framework. Existing rule-based design is problematic, since it has to be manually updated if new and unaccounted for data is encountered. Current Deep Learning (DL) is also insufficient to become AGI, but it has the potential to be extended into one. Therefore an appropriate AGI has to be 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. 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 · Dynamic · Evolving.

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 must be pre-defined. However, real AGI can generalize not only to unseen data but also to unseen tasks (as in transfer/continual learning). Nevertheless, we propose a dynamic and flexible DLM that can be extended to AGI. We use DNN and not another machine-learning (ML) tool, since it is based on genuine intelligence.

2 Proposed architectures

In this section, 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.

2.1 Proposed AGI model

A general AGI model sketch is shown in Fig. 1. This AGI is based on GWT [26, 29], which describes a multi-agent system. In GWT, the agents are local controllers behaving reactively, and compete with each other over access to the working memory (WM). 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.

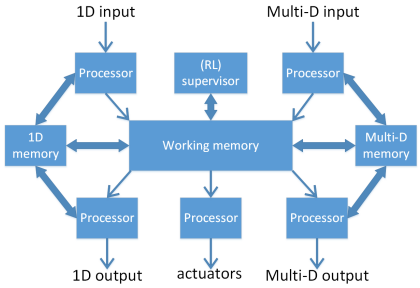


Fig. 1. AGI proposed architecture

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 [12, 24] implementing AI are based on distributed *cognitive* architectures [19].

The first study [24] 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 [12], an AI is implemented on a single intersection using a Multipurpose Enhanced Cognitive Architecture (MECA) [11], 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 receiving 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 an intelligent behavior.

These papers are rule-based designs using small-scale networks. 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.

Additionally, as in infant-parent and student-teacher interactions, the DL agent should be guided and supported.

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 be 1 or more dimensions, depending on the environment our agent is deployed in.

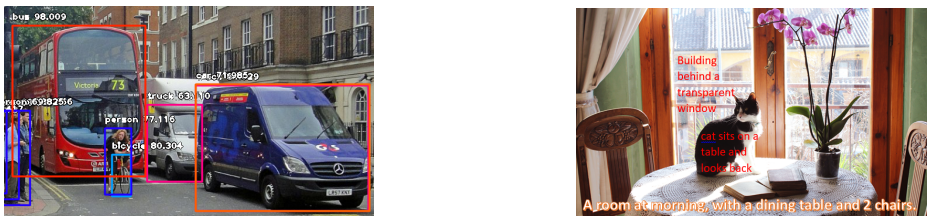
There is separate sequential processing of 1D and multi-D data, for feature extraction and categorization of objects (static entities) or events (dynamic entities). Next, these objects/events propagate into the WM. Finally, an output is produced either through the 1D or the multi-D channel. If the output is an emerging idea/thought, it can be expressed via 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 [31], 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", also expressed by a human productivity over time, in fields such as science, psychology, and technology.

This AGI does more than static/dynamic object identification or scene understanding [18], 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.



(a) AI identifying. (b) AGI interpretating.

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

AGI characteristics Firstly, If AGI's main purpose is to organize information to be utilized optimally in a variety of tasks, prediction might be only a tool to estimate this main goal. Additionally, DNN is an efficient algorithm and memory structure, which can achieve this goal, since it organizes the data with the intention of recovering it later.

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 correct teaching order (simple to complex) and the ability to grow/evolve in compulsory stages.

Two information types Here we discuss the rational behind the unique functioning of the visual and auditory channels.

Firstly, we examine why humans do not posses 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. 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 with 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, physical reality's purpose is nothing but the objective agreement for effective communication between humans, realized via language. However, vision is not a communicative channel for us.

Consequently, the purpose of having two channel types is distinguishing 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 by the static/objective world, but rather by language. Agents denoted as green circles, communicating via 1D and individually perceiving multi-D input are illustrated in Fig. 3.

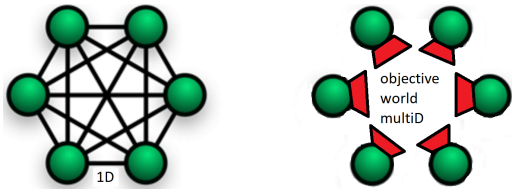


Fig. 3. Objective (right) verse Subjective inner representation (left).

2.2 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.

Some necessary background is needed. We start by reviewing prior knowledge in DL, also known as inductive bias. Then due to difficulties with matching the most proper prior knowledge to each specific problem we encounter - we continue with reviewing NAS dealing with these difficulties. Finally, we present our DLM as a possible NAS strategy.

Prior knowledge in DNNs Prior knowledge can appear in many forms. One form is structure's general type, such as CNN and RNN. These NNs present better prediction results in accuracy and stability compared to other ML methods [21, 33, 34]. 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.

Another forms are the hyper-parameters and the regularization method, e.g. dropout or constraints. Other form is the decision about sharing or grouping [28] or separating features/variables. For example, when traffic data is set apart from weather data [17], or when roads being separated from stations [15] and then being fused later. Tasks can also be separated into groups [15]. Finally, [16] suggests sparsifying the NN, e.g. removing connections in CNN, or grouping/sharing parameters, results with 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.

Network Architecture Search (NAS) Firstly, in NAS, e.g. AutoML [2], 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.

EA [9] 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 [6], it is used to replace back-propagation with Genetic Algorithm (GA) or combine them both, in searching for weights.

A more efficient/faster approach use 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.

Unlike EA and RL, the search in Differentiable Architecture Search (DARTS) [20], is defined as a differentiable and continuous problem. DARTS use multiple categories, where the final architecture is discretized after the search is over.

In hypernetwork approach [8], the weights are learned not by training 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 [13] 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.

Another example is the SMASH method [3], 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.

Next, our evolving DLM is presented as an additional NAS method, and implemented in a continual learning strategy.

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 an evolving DLM, starting from small NN, extending successively to a bigger one while encountering new data. Similar models to our DLM can be found in [25]. 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, rather an unsupervised one. 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 awaking period) through some initial DNN. See Fig. 4(a).

After a while, when enough counts detected the single event, a split of this event 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 [14]. It can now build a new layer/level 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).

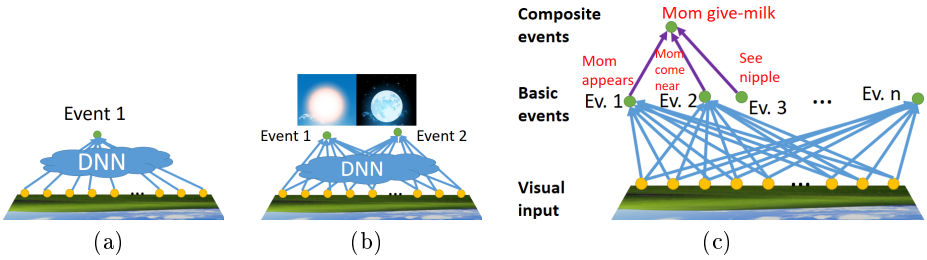


Fig. 4. Neuron separation and composition in the proposed approach (Ev.=Event).

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.

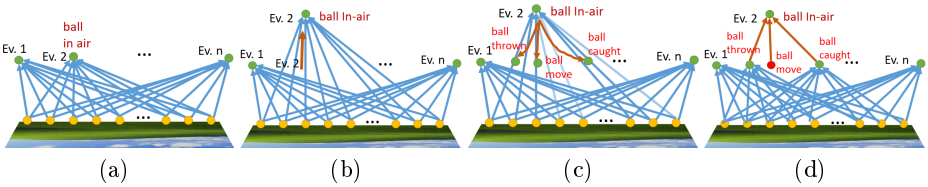


Fig. 5. Neuron decomposition in the proposed approach (Ev.=Event).

Decomposition is highly uncertain operation, since it is unknown weather 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, but it is out of the scope of this paper.

For this dynamic algorithm to work, the number of visits have to be stored for each weight (edge in DNN) and for each node in the DNN. 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 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. Perhaps it may even be so in the actual human brain. 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 result with adjusting the learning rate accordingly.

An issue can be in the recoverability, i.e. the ability to restore once deleted elements. One possible solution assume 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. A 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 could benefit from other optional additions, such as:

* 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 on 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 have different functions, such as convolutional, recurrent, recursive, or attentional.

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 a cat or a dog. Moreover, most AI research works backwards. It always starts from high-complexity data and try 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 computational 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.

Task hierarchy in the proposed DLM The extending of classes converts this DLM into hierarchical multi-class DNN. Such DNNs exist in the literature. In [4] we have feature layers and then task layers. Sometimes these layers can be mixed up. Labels structure can be found separately from the model [5,23], or as a part of the model [10,18,22]. All the tasks can be learned over one classifier, i.e. globally or in the last layer of the NN [5,30], or alternatively, intermediate tasks can be inserted inside the NN, i.e. locally [4,22,27,32]. The structure can be learned from the data [18], e.g. by unsupervised clustering of the labels via some similarity measure [23], or it can be imported from external knowledge base [7], or used to change this structure [5,10].

Similarly 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 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 - enforces a huge memory, since we need lots of them to represent a huge amount of terms and concepts. As opposed to a small group of inter-related features, which can characterize an enormous amount of input data. Therefore, at some point, a replacing/converting tasks with/into features should be considered.

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.

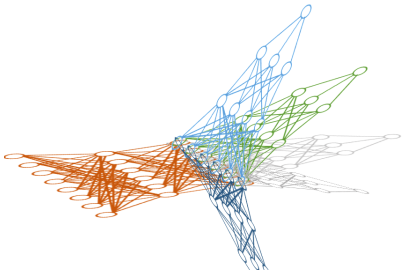
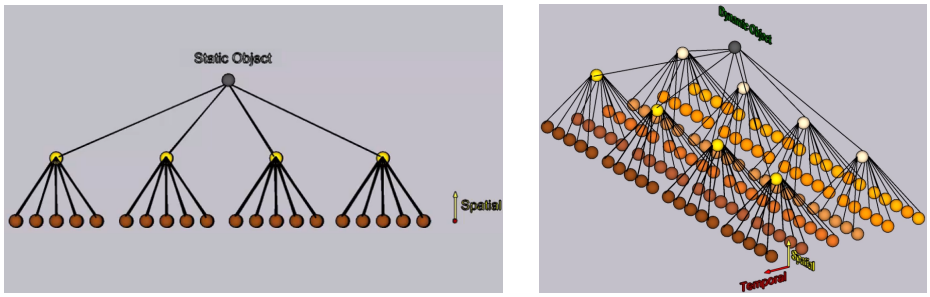


Fig. 6. Branching due to multiple hierarchies.

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.

On the other hand, 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 the temporal dimension, without changing the spatial dimension, i.e. orthogonal to it. See Fig. 7 illustrating 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.

References

1. https://simania.co.il/bookdetails.php?item_id=145306
2. AutoML: Automl, https://towardsdatascience.com/how-to-apply-continual-learning-to-your-machine-learning-models-4754adcd7f7f
3. Brock, A., Lim, T., Ritchie, J.M., Weston, N.: Smash: one-shot model architecture search through hypernetworks. arXiv preprint arXiv:1708.05344 (2017)
4. 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)

5. 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)
6. 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)
7. Feng, S., Zhao, C., Fu, P.: A deep neural network based hierarchical multi-label classification method. Review of Scientific Instruments **91**(2), 024103 (2020)
8. Galanti, T., Wolf, L.: On the modularity of hypernetworks. Advances in Neural Information Processing Systems **33**, 10409–10419 (2020)
9. Galván, E., Mooney, P.: Neuroevolution in deep neural networks: Current trends and future challenges. IEEE Transactions on Artificial Intelligence **2**(6), 476–493 (2021)
10. 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)
11. 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)
12. 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)
13. Ha, D., Dai, A.M., Le, Q.V.: Hypernetworks. CoRR **abs/1609.09106** (2016), http://arxiv.org/abs/1609.09106
14. Hawkins, J., Blakeslee, S.: On intelligence: How a new understanding of the brain will lead to the creation of truly intelligent machines. Macmillan (2007)
15. 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)
16. Ioannou, Y.A.: Structural priors in deep neural networks. Ph.D. thesis, University of Cambridge (2018)
17. 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)
18. 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)
19. Lieto, A., Bhatt, M., Oltramari, A., Vernon, D.: The role of cognitive architectures in general artificial intelligence (2018)
20. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055 (2018)
21. 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)
22. 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)
23. Papanikolaou, Y., Tsoumakas, G., Katakis, I.: Hierarchical partitioning of the output space in multi-label data. Data & Knowledge Engineering **116**, 42–60 (2018)

24. 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)
25. 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)
26. Reggia, J.A.: The rise of machine consciousness: Studying consciousness with computational models. Neural Networks **44**, 112–131 (2013)
27. 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)
28. 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)
29. da Silva, R.C.M., Gudwin, R.R.: An introductory experiment with a consciousbased autonomous vehicle. In: 4th Workshop in Applied Robotics and Automation (2010)
30. 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)
31. Watson, J.B., Meazzini, P.: John B. Watson. Il mulino (1977)
32. Wehrmann, J., Cerri, R., Barros, R.: Hierarchical multi-label classification networks. In: International Conference on Machine Learning. pp. 5075–5084 (2018)
33. 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)
34. 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)