Article

Not peer-reviewed version

Introduction to the E-Sense Artificial Intelligence System

Kieran Greer *

Posted Date: 28 April 2025

doi: 10.20944/preprints202309.0370.v5

Keywords: brain model; memory model; neural model; cortex; statistical clustering

 doi:10.20944/preprints202309.0370.v5

Article

Introduction to the E-Sense Artificial Intelligence System

Kieran Greer

Distributed Computing Systems, Belfast, UK; kgreer@distributedcomputingsystems.co.uk

Abstract: This paper describes the E-Sense Artificial Intelligence system. It comprises of a memory model with 2 levels of information and then a more neural layer above that. The lower memory level stores source data in a Markov (n-gram) structure that is unweighted. Then a middle ontology level is created from a further 3 phases of aggregating source information. Each phase re-structures from an ensemble to a tree, where the information transposition may be from horizontal set-based sequences into more vertical, typed-based clusters. The base memory is essentially neutral, but bias can be added to any of the levels through associative networks. The success of the ontology typing is open to question, but results suggested related associations more than direct ones. The third level is more functional, where each function can represent a subset of the base data and learn how to transpose across it. The functional structures are shown to be quite orthogonal, or separate and are made from nodes with a progressive type of capability, including unordered to ordered. Comparisons with the columnar structure of the neural cortex can be made and the idea of ordinal learning, or just learning relative positions, is introduced. While this is still a work in progress, it offers a different architecture to the current frontier models and may be able to give different views of the data from what they can provide.

Keywords: brain model; memory model; neural model; cortex; statistical clustering

1. Introduction

Artificial Intelligence is now at the apex of Computer Science. With advancements in pattern recognition and learning [26,31,34,37,44] and recently in prediction [42,43,48,60], the systems can perform many specific tasks as well as humans. Improvements in computing power and automated learning (for example, [14,33]) have also contributed. If the final bastions of reasoning and understanding can be mastered, then AI systems may well challenge humans in a general sense. However, the proponents are quick to point out that the systems are still mostly statistical, even though a new property of emergence has been realised in the very large distributed systems (Large Language Models [43]) that is not statistically predictable. While the path to success seems clear, there are still some hurdles and the problems that autonomous vehicles [11] have could be an example of this. Researchers are always inventing new ways to do things and this paper offers a different architecture to the established theory that may be able to complement the existing systems.

The new system is called E-Sense (Electronic Sense, or Essence). It comprises of a memory model with 2 levels of information and then a more neural layer above that. See Figure 2, section 4.1, later. The lower memory level stores source data in a Markov [12] (n-gram) structure that is unweighted. In a spatial sense, this would still mean that similar patterns would be clustered together. Then a middle ontology level is created from a further 3 phases of aggregating source information. Each phase re-structures from an ensemble to a tree, where the information transposition may be from horizontal set-based sequences into more vertical, typed-based clusters [16]. The ontology is not critical to the results of this paper, but it may be useful in future, for suggesting alternative concepts during search processes. The success of the typing is open to question, but results produced answers, based more on use and context. The potential of it is described in section 4 and the Appendix A examples. The design and implementation of the model is an extension of the author's previous

 doi:10.20944/preprints202309.0370.v5

publications. For example, exactly how the information transposition for the ontology should be done is an open question, but the philosophy behind it is consistent with earlier work. The base memory is essentially neutral, where any weighted constraints or preferences should be stored in the calling module. This would allow different weight sets to be imposed on the same linked structures, for example, or bias could be added using associative networks (see section 4.5). The third level is more functional, where each function can represent a subset of the base data and learn how to transpose across it. The functional structures are shown to be quite orthogonal, or separate and are made from nodes with a progressive type of capability, from unordered to ordered. This was a surprising result that is described further in section 5. Comparisons with the columnar structure of the neural cortex can even be made. This is only a first implementation of the model and in fact, a use for its' functionality is still not clear, when compared to what existing systems can achieve. But it offers a different architecture to the current frontier models and has some interesting biological comparisons that may be able to give different and new views of the data. Direct comparisons with the real human brain are made throughout the paper, where previous work by the author includes [16,17,23].

The rest of the paper is organised as follows: Section 2 briefly introduces the original model again, while section 3 gives some related work. Section 4 describes the new memory model that is the lower 2 levels, with some test results. Section 5 then introduces the new upper neural level and section 6 introduces the idea of ordinal learning, again with some test results. Section 7 makes comparisons with purely biological concepts, while section 8 gives some conclusions on the work.

2. The Original Cognitive Architecture

The original architecture [23], shown in Figure 1, described 3 levels of increasing complexity. The lower-level optimised links locally, using stigmergy, for example. The middle-level aggregated the lower-level links and the upper-level aggregated those into more complex concepts. There was also an idea that one complex concept would trigger another one.

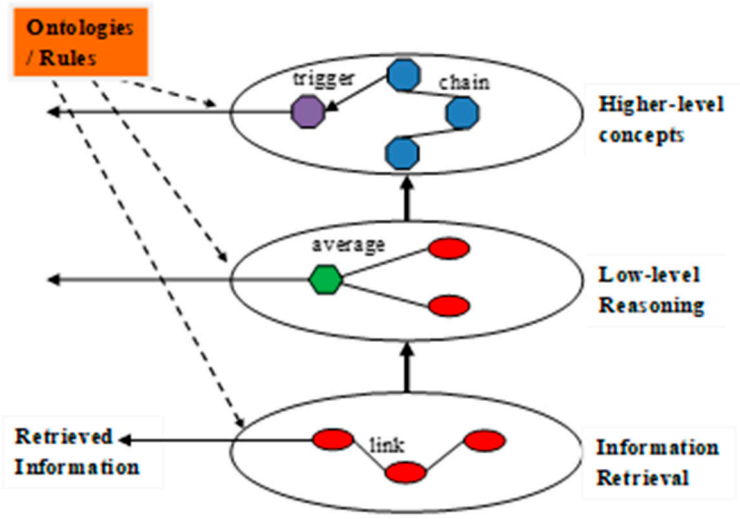

Figure 1. The 3-Level Cognitive Model [23] with a related ontology.

Section 4 describes that these levels now form the basis for the new memory and neural models. An external ontology [25] was also part of the original architecture and that is now integrated as the new middle level. The memory model uses a statistical clustering process, rather than semantics and rules, however. The author supposes that this effect is covered in modern NLP programs by using Word Vector models [42], for example. Fully-connected neuron structures are central to some themes in the system, where this idea is quite common (for example, [1,28] and some of the author's earlier papers).

 doi:10.20944/preprints202309.0370.v5

3. Related Work

There are a few notable AI systems that already produce human-like results, where most systems would claim to represent one or more parts of the human brain. In-line with the author's theory, the review paper [49] describes that cognition calls for a mechanistic explanation, where intelligence may depend on specific structural and functional features in the brain. It gives an overview on what types of neural network are used to model which parts of the brain. For example, auto-associative networks have been used to model the cortical regions [57], while feedforward networks have modelled memory or vision. No network type has modelled the whole brain however, probably because of their fixed structures. The paper [40] also describes modular networks for modelling the human brain. For this task, 'heavy-tailed' connectivity becomes important and several papers have discovered this phenomenon when mathematically modelling the neural connectivity (for example, [39,51]). With heavy-tailed connectivity, synaptic connectivity is concentrated among a select few pairs of neurons that are attractors. This results in a sparse network of strong connections dominating over other interactions. The paper [39] has shown that they can occur simply through a mixture of Hebbian and random dynamics, or the preferential attachment model [2]. The paper [51] studied how different rewiring functions would affect the resulting structure that was generated. They observed that random structural connectivity was reshaped by 'ordered' functional connectivity towards a modular topology, which also indicates synchronous firing patterns. One example was that rewiring peripheral nodes inside a module was homogeneous, with high synchrony and low-dimensional chaotic dynamics. On the other hand, central hub nodes connected with other modules, exhibited unsynchronized, high-dimensional stochastic dynamics. To reduce chaotic activity and energy therefore, it makes sense that between-module interaction would occur through a select number of key nodes only.

The paper [54] describes a model of the Hippocampus that uses auto-associative networks with generative learning. It describes how episodic memory is constructive, rather than the retrieval of a copy. But it needs the resource of semantic memory, from the neocortex, which is factual knowledge. They used a modern Hopfield network, where feature units activated by an event were bound together by a memory unit. The two layers here, binding features, is considered in section 5, for example. The generative networks were implemented as variational autoencoders [33], which are autoencoders with special properties, so that the most compressed layer represents a set of latent variables. These variables can be thought of as hidden factors behind the observed data and can be used to re-create the data again. The paper [57] considered if an auto-associative network can accurately model the cortical region. It considered the different levels in the cortex [27,45] and the different types of connection and function in those levels. The conclusion was that an auto-associative network has sufficient capacity to be used as a memory model, but may require the addition of these other factors. Since then, the creation of modern Hopfield networks [35] has shown that this type of network can have sufficient capacity in general, but needs multi-dimensional functions. Gestalt theory has been mentioned before [22]. With Gestalt psychology, objects are seen independently of their separate pieces. The booklet [4] gives a formal description of the theory and a mathematical proof that links the psychology theories of memory span [5,41] and duality [50]. The relation to Gestalt theory is discussed in section 7.1.

3.1. Current State-of-the-Art

Deep Neural Networks [31,34,37] probably kicked the current revolution off, but other notable successes would include Decision Trees [26], for example. Category Trees [19] might be an interesting alternative. Deep Learning [44] then combined Deep Neural Networks with Reinforcement Learning. It can automatically learn features from the data, which makes it well-suited for tasks like object classification and speech recognition. DeepMind (the company behind deep learning) introduced neural Turing machines (neural networks that can access external memory like a conventional Turing machine), resulting in a computer that loosely resembles short-term memory in the human brain. The

 doi:10.20944/preprints202309.0370.v5

model [44] used a convolutional neural network, which is organized similarly to the human visual cortex. The advantage of this kind of network is that the system can pick out particular features from the data, automatically. It is then able to comb through massive amounts of data and identify repeated patterns that can be used to create rules and processes. The general architecture means that DeepMind's algorithms have taught themselves to play Atari games and beat the best humans in Go or Chess. DeepMind has since moved on to tackling more and more real-world problems, such as unravelling the likely structures of Proteins. Then recently, Large Language Models [43], such as OpenAI's ChatGPT and Chat-GPT4 [48] have advanced the state-of-the-art again. Some argue that GPT4 already exhibits a level of Artificial General Intelligence, maybe because of the emergence property. These systems can make use of Word Vector models [42] and Transformers [60], for example, to predict what comes next in a sequence, rather like an n-gram [3,12] for text. They can be trained on a large corpus of data but are then able to create answers for any type of question, even ones not known about. They can use the same process to manage images, mathematical equations and even computer code. The discovery of transformers allowed them to predict to a level not encountered before and together with deep learning, huge distributed models with billions of nodes can be built. But even with all the recent advances, some papers show that there can still be problems, even with benchmark datasets [9,32,47]. New solutions would also want to make the models more economic and reduce their reliability on data.

3.2. Alternative Models

Other designs are described in [38], where one option, used in SPAUN [10], was to transform an ensemble mass into a vector-style structure, with weighted sets of features. SPAUN is one of the most realistic brain model designs, but context is still a problem. This is also clear in one of the original designs called SOAR [36]. That system adhered strictly to Newell and Simon's physical symbol system hypothesis [46], which states that symbolic processing is a necessary and sufficient condition for intelligent behaviour. SOAR exploited symbolic representations of knowledge (called chunks) and used pattern matching to select relevant knowledge elements. Basically, where a production matched the contents of declarative (working) memory the rule fired and then the content from the declarative memory was retrieved. SOAR suffers from problems of memory size and heterogeneity. There is also the problem that production rules are not general knowledge but are specific and so there is still not a sufficient understanding at the symbolic level. IBM's Watson [29] is also declarative, using NLP and relies on the cross-referencing of many heuristic results (hyperheuristics) to obtain intelligent results. Context is a key feature of the Watson system however. A recent paper [61] describes a process for recognising 'relation patterns' between objects. Humans acquire abstract concepts from limited knowledge, not the massive databases that current systems use. Their relational bottleneck principle suggests that by restricting information processing to focus only on relations, it will encourage abstract symbol-like mechanisms to emerge in neural networks and they suggest a neuro-symbolic [52] approach. They argue that an inductive process can be used to learn a relation like 'ABA', where A or B can then be anything. The symbolic and connectionist approaches can be reconciled to focus on relations between objects rather than the attributes of individual objects. They propose to use inner products, which naturally capture a notion of relations. A 'small changes' theory is part of this paper's model, described later in section 5.2. Also to be noted for the idea of massive overlap is the 'Thousand Brains Theory' [27].

A good organisation ability may be an inherent property of humans, or even the animal kingdom and would be something that can be improved in the current systems. This is discussed further in section 6. The paper [30] suggests a theoretical framework that would try to convert the fixed neural network architecture into one that can represent images in more abstract part-whole hierarchies. The structure would not be so fixed, but neurons would be allocated to clusters dynamically. It is something that humans do, but neural networks currently cannot do. The paper is quoted in [13], which is interested in the mechanistic processes of human cognition. A section there on abstract encoding of sensory input used a vector format to encode data and create columns of vector sets.

 doi:10.20944/preprints202309.0370.v5

Similar columns can then be used to find parts of an object in a visual scene and the framework does not suffer from overfitting. The paper [55] used statistical mechanics to try to explain some of the mechanisms that occur in the biological brain. They then showed how their results suggest a thermodynamic limit to the neural activity, but have no definite explanation of why, and this limit suggests a boundary. They also noted that the brain is a nonequilibrium system and asked the question of how it then obtains equilibrium. Most of these papers consider the brain activity to be more entropic than local. The paper [8] is very mathematical, but it might give a solution to the problem of these looser constructs. It proposes to use sheaves and writes about unary and binary typing's. Rather than global, they argue that time can be constructed, like local events, when it might also be thought of as an ordering and is entropic.

4. The Memory Model

The original cognitive model was based on a 3-level architecture of increasing complexity, which included an ontology that would be available to all the levels. Ontologies [25] describe the relations between concepts in a very structured and formal way. They are themselves high-level structures and it is not clear how they could be built simply from statistical processes. The ontology of this model is therefore not at that level, with sets of specific relations between concepts. Instead, it provides a loose clustering of concepts, but also a transition from context to type.

4.1. Memory Model Levels

Aligned with the cognitive model, the memory part is implemented in two lower levels, with some referencing in the upper neural level. This is not surprising, as it is thought that memory is stored in all parts of the human brain. The 3 memory levels are therefore as follows, also shown in Figure 2:

(1) The lowest level is an n-gram structure that is sets of links only, between every source concept that has been stored. The links describe any possible routes through the source concept sequences, but are unweighted.

(2) The middle level is an ontology that aggregates the source data through 3 phases and this converts it from set-based sequences into type-based clusters.

(3) The upper level is a combination of the functional properties of the brain, with whatever input and resulting conversions they produce, being stored in the same memory substrate.

It may be that the first 2 levels can be made from simpler structures, in the sense that it does not have to be functional. For example, the paper [16] describes that more recently, the perineuronal network [58] has received a lot of attention and may be a sort of simpler memory substrate. An earlier paper [17] described the types of knowledge transitions that may occur. If using a classification of experience-based or knowledge-based information, then experience-based information is dynamic and derived from the use of the system. Knowledge-based information is static and built from the experiences. The paper described that the transitions in the cognitive model may be:

1. Experience to knowledge.
2. Knowledge to knowledge.
3. Knowledge to experience.

If looking at the new Figure 2 design, then from the top level to the bottom level we get these transitions again. The two outer-most layers would be the sensory input or the cortical columns and would be experience-based. Then between them are transitions into and out of knowledge. At least 1 time-based layer needs to be added to this region, which will be considered in future work. The paper [49] does not note auto-associative networks as being whole-brain models, but the new cognitive model could be seen as an auto-associative one, where the middle knowledge-based level would represent compressed knowledge variables. The knowledge-to-knowledge transition would be between the middle and upper levels, but currently, that information flow is not implemented. It is

 doi:10.20944/preprints202309.0370.v5

also shown that knowledge is stored in tree-like structures, while experience is stored in ensemble-like structures. The upper-level 'ensemble to tree' is reversed in the middle level.

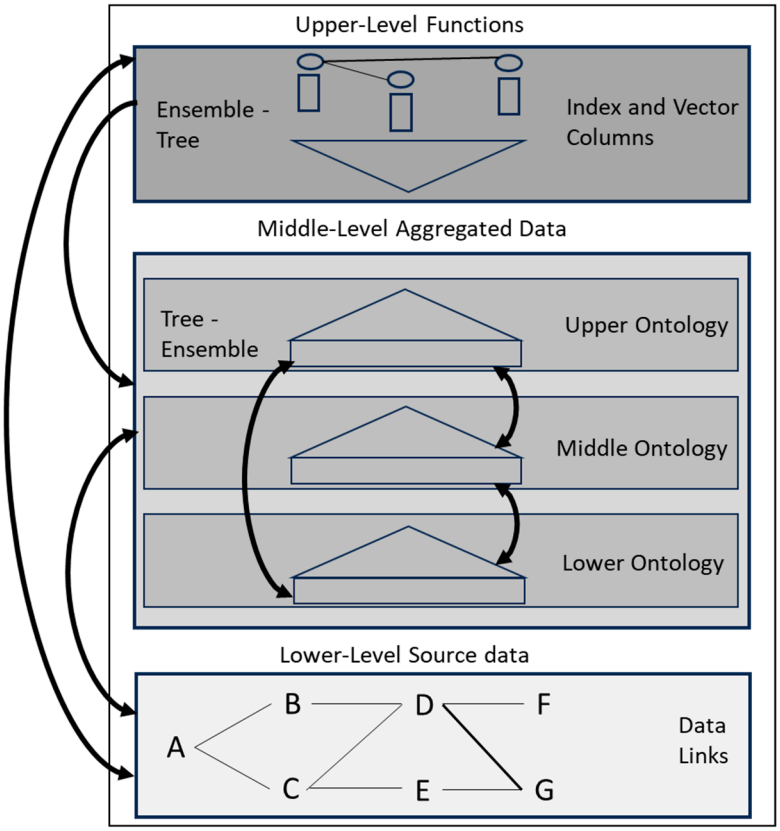

Figure 2. The new 3-Level Cognitive Model.

4.2. Lower Memory Level

In the model, the lowest level is an n-gram, where each node contains links to each possible next node. Thus, tracing through these links can return different sequences, but to provide some direction, possibly a 5-gram needs to be used. It is also possible to note start or end nodes in a sequence, to help with defining the sequences better. The database structure is appealing because it is very compact, but while it works well for most of the time, it may not always return every sequence, exactly as it was entered. Unlike neural representations however, this structure does not have to be weighted. A link between nodes in a region is noted only once, no matter how often it occurs during the input. The structure therefore only stores equal possibilities, where preferences or weighted choices would have to be transferred over to a calling module. To make it more accurate however, it can be combined with an associative network that can recognise specific sequences, where that does then introduce a certain amount of bias. If the n-gram is sufficient, then the theory would state that a Markov process may be sufficient to describe a Gestalt process. This is discussed further in section 7.

4.3. Middle Ontology Level

The middle ontology level uses transitions from ensemble to tree, where the final trees look more like single vector lists. While the ontology is still a low-level structure therefore, it may contain one important property in that it is able to convert set-based sequences into type-based clusters. This would introduce a small amount of knowledge into the ontology that a search process can make use of. While the lower-level database can be described by an n-gram, the middle ontology level is more complicated. It makes use of the Frequency Grid [22] to generate word clusters, but there would be more than 1 way to implement the aggregation process – from ensemble to tree, for example. The author has chosen a version that prunes the most nodes from the result, to save on time and resources.

 doi:10.20944/preprints202309.0370.v5

This means that a text document the size of a book may return at the top ontology tree, only a few words clustered together (see Appendix A), but as the search could move back down the structure to the lower levels, it will be able to discover most of the text from matching there as well.

4.4. Ontology Tests

A computer program has been written in Java that implements the two memory-model levels for basic testing. The success of a test is measured rather arbitrarily, by judging if the words in a cluster have some relation and preferably, are not simply part of the same sentence. The author has judged that this is often the case. Each result is from clustering only on a single book however. It is even more difficult to judge how accurate the clusters are when texts are combined, which will be future work. One problem that has occurred with the frequency grid before is when 2 or more smaller clusters are joined together. This can result in a single cluster with apparently 2 or more meanings. This also occurs in some of the final upper ontology clusters, described in Appendix A. Rather than the program recognising associated antonyms, or something like that, it may have incorrectly combined 2 lower-level clusters somewhere. While the algorithms in this paper are different to what was used in earlier work, this becomes an interesting feature when constructing the neural level, described in section 5, however.

Appendix A therefore lists some well-known texts [56], together with the final upper ontology sets that the program produced. The resulting structures were very narrow, where each value in each list was a child node of the one before. This would be consistent with a conversion from a horizontal set-based description to a vertical type-based one. However, the row ordering can change and so an ordered sequence might be an illusion, or might be relevant only sometimes. The results are very subjective, but the author hopes that it is possible to see how some level of real meaning in the words has been derived from the statistical process.

4.5. Unit of Work

A ‘unit of work’ structure has been suggested in earlier papers, for example [17] (section 3), and also the related ensemble-hierarchy [22]. In the new model it can take different forms, depending on the functional level. In the middle ontology level, it is an ensemble-tree, for filtering purposes only. In the bottom memory level, the sequences can be combined with an auto-associative network, which would be able to recognise when a sequence represents something. For this, a recent design by the author [15] would be an ideal candidate. A set of signals from the base would pass up the associative network to end points, where if a common terminal point is found, it would reverberate back to the base sequence, to enhance the signal. In this design, it is interesting that the base ensemble stores the ordering for the recognised object, while the associative network only recognises it as a whole concept. With a myriad of possibilities, this could still be very useful and is probably what Transformers [60] over the LLMs do in modern systems. The base level associative networks are not yet implemented, but they make good architectural sense. A similar type of structure has also been suggested for the upper neural level, as part of other research. Kolmogorov and Shannon were written about in [18], with regards to trying to measure intelligence. Shannon based his Information Theory [53] on Entropy and a Markov model. Kolmogorov Complexity theory ([6], chapter 7) states that the shortest sequence is the most likely, and also the best. Combining these gave rise to a related design in [18], where in this case, the hierarchy would be Shannon network loops that represent repeating functional acts, and extend from the base Kolmogorov sequences, intended to get an entity from A to B in the most economical way.

5. The Neural Level

The neural level is the top level of the design and contains more cognitive or functional units. There are to be 2 different types of neural level in the final model – one is interested in cognitive processes that may be described by a Cognitive Process Language [20], while the other is more

 doi:10.20944/preprints202309.0370.v5

interested in logical processes (for example, [17] and the related papers). This paper deals only with the logical neural model, which comprises of functions that operate on the source data of the lower level. It does not follow-on from the middle ontology, but is separate from it, although in theory, it would query the ontology if needed. A typical definition of a function is something that maps a set of input values to a set of output values. It may do this by generating the output, or if the output already exists, then it is a selection process. If it can make use of existing output, then the function reduces to something more like a constraint, with a description like - a function in some cases, may be seen as something that reduces the possibilities for the next step. As the following sections describe, the neural level now looks quite a lot like the cortical regions in the human brain.

5.1. Function Identity

If there are lots of functions in the neural level, then they want to be recognised and made distinct. One option is to consider every node in the function and comparing this between functions would probably produce differences. Another option may be to consider only a set of base 'marker' nodes, when the rest of the function can be built on-top of these. In fact, these key nodes can become an index set that should be found first in the source data and then a larger set of vector values, representing the whole function, can be searched for. This is in fact what modern search engines do [7]. The index set could also help with maintenance. It could be checked for first, to see if some process returns the same index key. If not, then it is likely that something has changed in the base data, linked to by the function, where further processes would need to determine what to do.

5.2. Function Structure

A function is therefore based on an ensemble of these index nodes, each with a set of index values. Each node then links to a larger set of vector values that represent a feature in the function. Each index node matches with 1 or 2 n-gram sequences from the lower-level database and it also stores the relative position(s) of the sequence(s), so that the correct ordering can be re-constructed from values that may be arbitrarily ordered as input. The operation might be like the relation patterns of [61], but there may also be practical differences. Each function part, in fact, resembles quite closely, the Symbolic Neural Network that was written about in [24]. The index values are quite orthogonal, or do not overlap very much. The features, as whole sets, are mostly unique as well, but some index nodes can share the same feature set. There could thus be closures at both the top and bottom of this structure. This type of structure was shown in [24] to be able to filter-out noise quite well, for example. Both the most commonly occurring terms and the least commonly occurring terms are stored in a feature for a sequence. The most common allow potential matches with the sequence to be found in a larger database. Then the least common allow this potential set to be filtered further. A group of the SNNs are stored in an 'Ordinator,' which is really the whole function for a particular operation. Because there is a lot of overlap in the results returned from each SNN, this produces only small changes in the statistical result, but usually, any changes then need to be included. When building the structure, some weighted components might be used, but when using the structure afterwards, it is mostly an un-weighted process. There might be some frequency counts, but not much else. It is probably the case that the orthogonal nature of the structure reduces the need for weights.

5.3. Index Types

The process of creating the index nodes and related vectors seems to have produced 3 different types. While it was not the case every time, a clear pattern of 3 distinct types emerged. One type may have a longer list of index terms but no related feature set. The other 2 had both index terms and related feature sets, but differ as explained next. These types map quite closely to known neuron types, as follows:

- Unipolar Type: this has a list of index terms that is generally a bit longer and is unordered. It can be matched with any sequence in the input set, but to only 1 sequence.

 doi:10.20944/preprints202309.0370.v5

- Bipolar Type: this has a list of index terms and a related feature set. The index terms should be matched to only 1 sequence and some of the feature values should also match with that sequence. This matching should be in order however, where the order in the feature should be repeated in the sequence. Then the rest of the feature values can match with any other sequence and in any order.
- Pyramidal Type: this has a list of index terms and a related feature set. The index terms however are split over 2 specific sequences. Both the index terms and the related feature set should match with 2 specific sequences and the matching should be ordered in both.

There is thus a very interesting progression through the 3 types and suggests a progression in functionality as well. While the index structure maps to these neuron types, it could also be used to create more columnar structures. It would make quite a good basis for the neocortex columns [27,45], for example, with a columnar unit comprising an index node and feature set, and the index nodes would also have horizontal connections. This would occur in Figure 2, in the top neural level.

6. Ordinal Learning

The author would like to introduce the idea of Ordinal Learning. It is being given a specific name, because current methods do not appear to do it. It is maybe more algorithmic than functional. Ordinal learning is concerned with re-creating the order of sequences it was trained with. But in this case, it can also interpret for previously unknown input that is statistically close to what it was trained with. Having a sense of order may be deeply inherent in animals, even insects. The papers [39,40], for example, map the neural connectome for some animals and the heavy tails that separate the network into modules are described. The brain is also thought to have a scheduling functionality at the top of the cortex, probably to perform such tasks. Neural networks are able to interpret what a pattern is with some missing information, but do not typically re-order the information that is presented. Although, a second network or module might learn pattern ordering, for example. Large language models also predict across a known sequence but would not intuitively know how to change a faulty sequence order. A change in the sequence order would lead to a change in the question and thus prediction. But this is still a common algorithmic problem with many solutions already. It could probably be solved in a few lines of code in many cases, and so it remains to be seen if the much more complicated method of this paper is more useful.

The basis for the Ordinator that learns the ordering, is to have something like a heavy-tailed neuron at each ordinal position. Thus, while heavy-tailed neurons are due to a preferential attachment or rich-get-richer mechanism [39], they would now have a particular function as well, that is to order the surrounding neurons. The ordinator would use heavy-tailed neurons to place the sequence of another neuron into that ordinal position. There would then be a vote from each ordinator node, to claim their position for the neuron, where the most connections would dominate. But like a computer program, it is still a statistical process, where if the train example is missing, the sequence with the closest statistical match will be selected instead. A schematic of the ordinator is given in Figure 3. The system would firstly create and store the feature network for the learned positions. This is the node types described in section 5.3. Then, new input sets can match with it. There is a hierarchical path leading to each of the ordinator position nodes. It is not clear what exactly should be in the path, but the 2 positions from the neuron types and maybe the context of the query would be possible. Then, query answers that result in sequences being selected for either position 1 or 2 can be added, or in a simpler model, they could be simply linked with the position itself. A majority vote can then be done and would resemble the neurons competing for their position.

 doi:10.20944/preprints202309.0370.v5

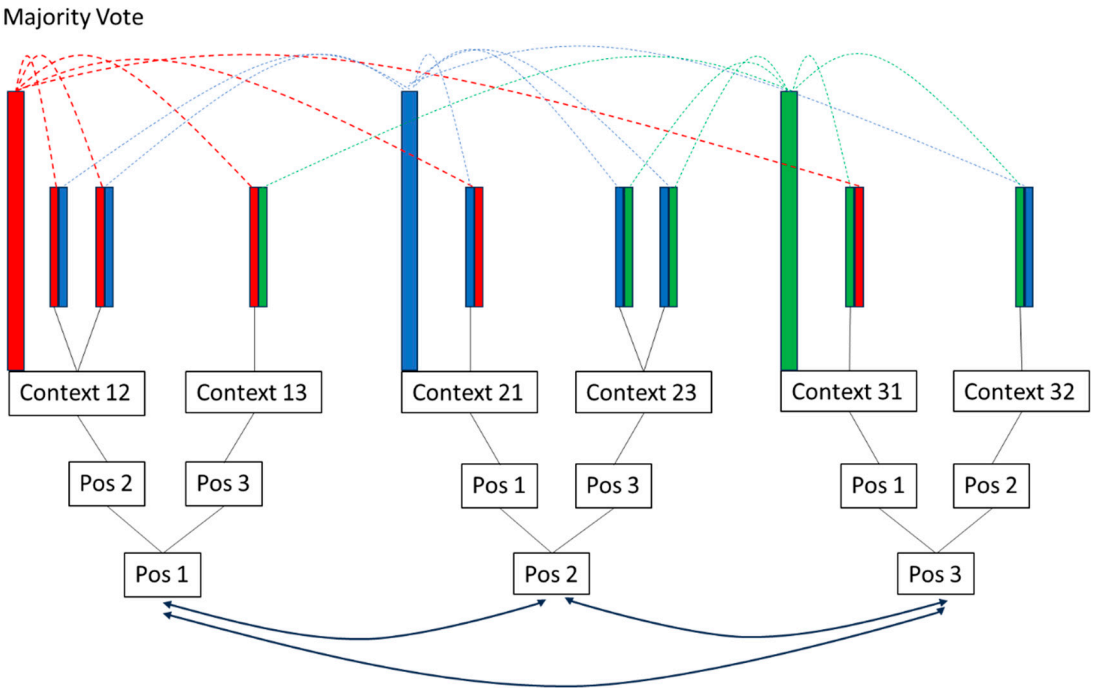

Figure 3. Schematic of the Ordinator function with 3 positions (red, blue, green).

6.1. Ordinal Tests

The ordinal learning process has also been implemented in Java code, for basic testing purposes. The code is in no way optimised and it could take minutes to learn the ordering for a 3K document (40 sentences) on a standard laptop. The learned structures were also much larger in size, than the raw data, but the intention is that they now contain knowledge and can be re-used. Also, the amount of train data is quite small and so it appears to be able to use the data quite efficiently. The process comprises lots of smaller separate algorithms and so it is likely that it could be parallelised. A train text document can be read and stored in a source (lower level) database. It can then be queried for information about the main concepts in it. In fact, an ensemble approach is required, but the process is mostly automatic. A bag-of-words, for example, can determine the main terms in the document and so queries can be run to retrieve the related sequences and build the logic structures. But the process is also recursively repeated for each query, with results from that query. Thus, lots of the results are mostly the same. The index values are also part of the feature vectors and so a structure representing this would probably be quite self-contained. The paper [21] made use of the frequency grid to produce a self-organising system, but the problem of the order of the input data rows became clear and an ensemble solution was required to at least partly, overcome this problem. Some neural networks also have a problem when the input rows are presented in a different order, but this recursive method seems to solve that issue. After the ordinator has been created, a different test document can be loaded, or the same train document can be used to test it as well. There is no specific ordering in the base database for a whole document, and so the retrieval of sequences from the database will not be in the original document order. The purpose of the function is to recognise itself in the source data again. It therefore tries to match with the source data and then sort that into the ordering that it has learnt. The ordering is only relative, where the number of sequences in the train and test documents can be different. For example, ordinal positions of 1 and 2 in the Ordinator simply mean that the sequence in relative position 1 should be before the sequence in relative position 2, not that the positions are exactly 1 and 2.

Appendix B gives the results for some basic tests. Two cooking instruction documents were selected. The first train document described how to cook a hard-boiled egg and the second described how to make Panna Cotta. If the test document was the same, then the data would be returned correctly. For a second test, an ordinator was generated for either train document and then the

 doi:10.20944/preprints202309.0370.v5

database was changed to one which contained both - a different description of how to cook a hard-boiled egg and the Panna Cotta description. The egg ordinator was able to select the sequences relating to the second description and also place them in order, as shown. The Panna Cotta ordinator also performed, as shown. So, while it did perform to 100% accuracy for these two small documents, using slightly different words might change the result. The process is more entropic than local connections, however and so there would be a balancing act to adding specific rules about something. It is also slightly stochastic, but the ensemble training method helps to keep the results mostly the same. This is still a statistical process, where if the test document has other sequences that match better with the feature concepts, then they will get selected instead. The distinct concepts in a feature are thus very important. Results also showed however, that for larger documents, of even 40 lines or more, some sequences would typically be missed, or not even retrieved from the source database. It is therefore unlikely that the process can be used to simply rote learn a large corpus of information. But it could be a useful guide and along with search processes, sort through information and make some sense of it. There are still options to be tried, to make it more accurate.

7. Some Biological Comparisons

This section makes comparisons with some purely biological ideas.

7.1. Gestalt Psychology

With Gestalt psychology, objects have an 'other' interpretation of the sub-features and not just a summed whole of them. Gestalt theory makes use of ideas like similarity and proximity (and good continuation) to group objects and it believes that the brain has an internal order and structure that it places external stimuli into. With the theory, the brain may try to fit the outside world into its internal structure, rather than copy the external world exactly. Gestalt theory could be realised in the lower model level. Because the links are unbiassed, one interesting aspect of the structure is that it may not return exactly what was input, thus satisfying the theory that the whole may be different to the parts. Consider this example where the following 2 sentences are added to the lowest-level memory:

The cat sat on the mat and drank some milk.

The dog barked at the moon and chased its tail.

Start words would now include 'the' and end words would include 'milk' and 'tail.' Each word in the sequence also has a link to any words that immediately follow it. If, for example, the memory system is asked to retrieve a sequence that results in the 'and' word being considered, then there are two possibilities after that – 'drank' or 'chased.' Therefore, a question about a dog could retrieve either of the following two sentences:

The dog barked at the moon and chased its tail, or

The dog barked at the moon and drank some milk.

If the second sentence was returned, then it would not be violating the memory system and the person would probably not have a reason to disbelieve it. It could therefore be a legitimate answer, even though it is different to what was originally entered. If an associative network is to be included however, then that will add some stability and bias, but then the second sentence could be seen as a new option that might be considered if the learned sentence is not suitable. It can also be argued that changing the information in this way is not a creative process, but simply taking a different route through the linked structure.

7.1.1. Numbers in the Gestalt Process

As part of the computer model, an n-gram depth in the lower level can add accuracy, requiring that several previous concepts are present to establish a sequence, where in fact a 5-gram is currently used. The decision to copy 5 tokens in order was a guess, again to give some stability. However, the number 5 is also important in Gestalt theory. Buffart [4] developed a mathematical method to

 doi:10.20944/preprints202309.0370.v5

formulate Gestalt theory. The outside or environment should contain at least 4 elements. Then, the process requires regularity when comparing elements and one of the theorems states that the relations between the elements in one event can be interpreted at most in two ways if, and only if, the number of elements in that event is at most seven. Moreover, for events with five, six or seven elements the possible interpretations are the same. Two interpretations is also well known in perception as the duality principle and seven minus two is well known in memory and attention research. In his 'seven minus two and structure' section, Buffart presents a proof that with up to seven materials in focus, each complete structure can be described by the combination of at most two complete interpretations. For spans larger than seven, at least three interpretations can be required to represent a complete structure. Thus, the numbers 2 and 5 (to 7) become very important to brain theory. One might consider two to be a stable state, allowing a single comparison. Five to seven then relates to structural stability, where less or more than this can change the dynamics of what elements are linked together. The n-gram length is 5, but could be 7 and the new model process did result in node types, described in section 5.3, that are either unary or binary. Thus, if this is the stable state, the number 3 has also occurred in the paper for more dynamic states. A value of 3, for example, would allow something to be rooted (1), while two parts are being compared (2 and 3). If the brain likes to synchronise to balanced states, for example, then this asynchrony might encourage the system to explore a step further.

7.2. Animal Brain Function

This section follows-on from ideas in [16] that describe how cells may have evolved from invertebrates to the human brain. Organisation was a key principle there, and also the conversion to types. The brain is typed, where even insects like ants can recognise types. Thus, if traversing through the new cognitive model of Figure 2, the middle ontology layer converts from the lower-level ensembles of set-based values to more type-based vectors. Then, between the middle and upper levels the types can be clustered back again into sequences, based on time and use. What if these new clusters are missing some of the sensory information? If the cortex is mostly about actions, or how to do something, then it does not require additional sensory feedback, which might also help to protect mental states. It would only want to know what the objects are and how to use them. So possibly sensory information can remain deeper in the brain, where it would also be closer to the senses themselves. Again, if thinking about a conscious experience, then it would require the sensory feedback from a whole-brain activation, not just local circuits in the cortex.

7.2.1. Theory of Small Changes

Thus, a constructive process can be seen in the building of brain structures. However, any new structure should not be so radical that it disturbs the mind. Thus, new structure should be added in small amounts each time. If we do not know about a subject for example, then the first structures for it should probably contain the basics or fundamentals for the subject. What this means is not clear, but possibly, what other concepts link with. If we already have some memory about a subject, then we can add to it instead, enriching the information. But the most important blocks and a basic structure is added first. Turing [59] wrote about sub-critical and super-critical processes that a human brain would recognise more than an animal. But he suggested that single events were key and that maybe even the mind had some influence on what gets stored. A mechanical process could simply make small changes until enough of them became critical and forced a larger change (the single event). But then the mechanical process should not try to store every piece of information. Therefore, some type of reinforcement from a more intelligent region that helps to select important input could indeed happen. But again, critical selection could result not only from more intelligence, but also less emotion, as described in the last section. To follow-on from Turing's idea and considering the 'unit of work' structures suggested in section 4.5, it is indeed possible to put them in terms of the proposed evolutionary processes. For example, the base memory uses smaller auto-associative networks to recognise something. At this level it is only recognition, not understanding and this level is also closer

 doi:10.20944/preprints202309.0370.v5

to the senses. In the middle level, some typing is achieved and then in the upper level, the objects can be manipulated through the typing. It is only in the upper level that a better understanding is achieved. More basic animals make use of memory, recognition and sensory feedback, while more intelligent ones can also manipulate and understand better, which corresponds with evolution of the neocortex.

8. Conclusions and Future Work

This paper describes a first implementation stage for the 3-level cognitive model, now called E-Sense. While it is essentially a computer model, it is based strongly on our understanding of the human brain. Some direct comparisons with biological processes have been made and the fact that the model is mostly un-weighted should be an advantage. There are 2 memory levels that are economic in space and can transpose the information from set-based to types. This introduces new knowledge and the transition to types may in fact be helpful to an upper level that wants to know more about objects and the technical 'how'. The upper level is more neural. Generating it produced a type of progressive functionality that could be loosely mapped to neuron types, or maybe just standard unary and binary operators. It might be convenient to look at the whole brain model as auto-associative. The bottom memory substrate / sensory level and the cortical levels map to each other through the middle-level transpositions. The two views are not the same however and so there is still an economy of storage, but the related parts in each map together. Each level then repeats this, but at a smaller scale, with tree units combining with base ensembles, to generate knowledge.

There is still a lot of work to be done in all the levels. The results are not too bad and as the name suggests, they may be more about some type of general understanding, than the more direct results that the current systems provide. Some specifics probably need to be added, to increase the intelligence level. The model does push a lot of biological buttons however, including Gestalt, small changes and what heavy-tailed neurons might be for. The functional structure that maps to cortical columns is interesting and also the fact that a natural kind of ordering can be integrated into the structure. Considering future work, the source database does not always return exactly what was entered, so it would need to be determined if this is critical. Should it return everything exactly, or would that require storing the original source as it is? It is a very compact structure. Then, there is not a clear progression yet from the middle to the top level. The top level is currently created from accessing the source database directly, but middle to top could be included in a more dynamic system. Then also, making the system more accurate and useful.

Appendix A – Upper Ontology Trees for Book Texts

This appendix lists the upper-level ontology trees that were created for some well-known books. The clustering relates to the use of the word. Each row is a child node of the row immediately before it, but in fact the row ordering can change.

Romeo and Juilet, William Shakespeare [56].

Clusters
thou
love, o, thy
romeo, shall
death, eye, hath
day, give, lady, make, one, out, up, well
go, good, here, ill, night, now
come, thee
man, more, tybalt

 doi:10.20944/preprints202309.0370.v5

The Wonderful Wizard of Oz, L. Frank Baum [56].

Clusters
dorothy
asked, came, see
city, emerald
great, oz
again, answered, away, before, down, made, now, shall, toto, up
scarecrow
lion, woodman
back, come, girl, go, green, head, heart, man, one, over, upon, very, witch
little, out, tin

The Adventures of Sherlock Holmes, Arthur Conan Doyle [56].

Clusters
back, before, came
down, know
more, room, think, well
day, eye, face, found, matter, tell
upon
holmes, very
little, man, now
one
away, case, good, heard, house, much, nothing, quite, street, such, through, two, ye
go, here
come, hand, over, shall, time
asked, never
door, saw
mr, see
out, up
made, way

Computing Machinery and Intelligence, A.M. Turing [59].

Clusters
answer, computer, man, question, think
machine
one
such

Appendix B – Documents and Test Results for the Neural-Level Sorting

This appendix lists the train and test files for testing the ordinal learning. The results of applying the ordering to the test files is also shown.

 doi:10.20944/preprints202309.0370.v5

Train and Test Files

Train File – Hard-Boiled Egg
Place eggs at the bottom of a pot and cover them with cold water.
Bring the water to a boil, then remove the pot from the heat.
Let the eggs sit in the hot water until hard-boiled.
Remove the eggs from the pot and crack them against the counter and peel them with your fingers.

Train File – Panna Cotta
For the panna cotta, soak the gelatine leaves in a little cold water until soft.
Place the milk, cream, vanilla pod and seeds and sugar into a pan and bring to a simmer.
Remove the vanilla pod and discard.
Squeeze the water out of the gelatine leaves, then add to the pan and take off the heat.
Stir until the gelatine has dissolved.
Divide the mixture among four ramekins and leave to cool.
Place into the fridge for at least an hour, until set.
For the sauce, place the sugar, water and cherry liqueur into a pan and bring to the boil.
Reduce the heat and simmer until the sugar has dissolved.
Take the pan off the heat and add half the raspberries.
Using a hand blender, blend the sauce until smooth.
Pass the sauce through a sieve into a bowl and stir in the remaining fruit.
To serve, turn each panna cotta out onto a serving plate.
Spoon over the sauce and garnish with a sprig of mint.
Dust with icing sugar.

Test File – Hard-Boiled Egg and Panna Cotta
Remove the vanilla pod and discard.
For the panna cotta, soak the gelatine leaves in a little cold water until soft.
As soon as they are cooked drain off the hot water, then leave them in cold water until they are cool enough to handle.
Squeeze the water out of the gelatine leaves, then add to the pan and take off the heat.
Spoon over the sauce and garnish with a sprig of mint.
Stir until the gelatine has dissolved.
Place the eggs into a saucepan and add enough cold water to cover them by about 1cm.
Pass the sauce through a sieve into a bowl and stir in the remaining fruit.
Divide the mixture among four ramekins and leave to cool.
Place into the fridge for at least an hour, until set.
To peel them crack the shells all over on a hard surface, then peel the shell off starting at the wide end.
For the sauce, place the sugar, water and cherry liqueur into a pan and bring to the boil.
Place the milk, cream, vanilla pod and seeds and sugar into a pan and bring to a simmer.
Reduce the heat and simmer until the sugar has dissolved.
Take the pan off the heat and add half the raspberries.
Using a hand blender, blend the sauce until smooth.

 doi:10.20944/preprints202309.0370.v5

Bring the water up to boil then turn to a simmer.
To serve, turn each panna cotta out onto a serving plate.
Dust with icing sugar.

Test Results

Selected Sequences from the Hard-Boiled Egg Function
[place, the, eggs, into, a, saucepan, and, add, enough, cold, water, to, cover, them, by, about]
[bring, the, water, up, to, boil, then, turn, to, a, simmer]
[as, soon, as, they, are, cooked, drain, off, the, hot, water, then, leave, them, in, cold, water, until, they, are, cool, enough, to, handle]
[to, peel, them, crack, the, shells, all, over, on, a, hard, surface, then, peel, the, shell, off, starting, at, the, wide, end]

Selected Sequences from the Panna Cotta Function
[for, the, panna, cotta, soak, the, gelatine, leaves, in, a, little, cold, water, until, soft]
[place, the, milk, cream, vanilla, pod, and, seeds, and, sugar, into, a, pan, and, bring, to, the, boil]
[remove, the, vanilla, pod, and, discard]
[squeeze, the, water, out, of, the, gelatine, leaves, then, add, to, the, pan, and, take, off, the, heat]
[stir, until, the, gelatine, has, dissolved]
[divide, the, mixture, among, four, ramekins, and, leave, to, cool]
[place, into, the, fridge, for, at, least, an, hour, until, set]
[for, the, sauce, place, the, sugar, water, and, cherry, liqueur, into, a, pan, and, bring, to, the, boil]
[reduce, the, heat, and, simmer, until, the, sugar, has, dissolved]
[take, the, pan, off, the, heat, and, add, half, the, raspberries]
[using, a, hand, blender, blend, the, sauce, until, smooth]
[pass, the, sauce, through, a, sieve, into, a, bowl, and, stir, in, the, remaining, fruit]
[to, serve, turn, each, panna, cotta, out, onto, a, serving, plate]
[spoon, over, the, sauce, and, garnish, with, a, sprig, of, mint]
[dust, with, icing, sugar]

 doi:10.20944/preprints202309.0370.v5

References

1. Anderson, J.A., Silverstein, J.W., Ritz, S.A. and Jones, R.A. (1977) Distinctive Features, Categorical Perception, and Probability Learning: Some Applications of a Neural Model, Psychological Review, Vol. 84, No. 5.
2. Barabasi A.L. and Albert R. (1999). Emergence of scaling in random networks, Science, 286:509-12.
3. Brown, P.F., Della Pietra, V.J., Desouza, P.V., Lai, J.C. and Mercer, R.L.. (1992). Class-based n-gram models of natural language. Computational linguistics, 18(4), pp.467-480.
4. Buffart, H. (2017). A formal approach to Gestalt theory, Blurb, ISBN: 9781389505577.
5. Cavanagh, J. P. (1972). Relation between the immediate memory span and the memory search rate. Psychological Review, 79, pp. 525 - 530.
6. Cover, T.M. and Joy, A.T. (1991). Elements of Information Theory, John Wiley & Sons, Inc. Print ISBN 0-471-06259-6 Online ISBN 0-471-20061-1.
7. Dobrynin, V., Sherman, M., Abramovich, R., and Platonov, A. (2024). A Sparsifier Model for Efficient Information Retrieval, AICT'24.
8. Dobson, S. and Fields, C. (2023). Constructing condensed memories in functorial time. Journal of Experimental & Theoretical Artificial Intelligence, pp.1-25.
9. Dong, M., Yao, L., Wang, X., Benatallah, B. and Zhang, S. (2018). GrCAN: Gradient Boost Convolutional Autoencoder with Neural Decision Forest. arXiv preprint arXiv:1806.08079.
10. Eliasmith, C., Stewart, T.C., Choo, X., Bekolay, T. DeWolf, T., Tang, Y. and Rasmussen, D. (2012). A Large-Scale Model of the Functioning Brain, Science, 338(6111), pp. 1202 - 1205.
11. Feng, S., Sun, H., Yan, X., Zhu, H., Zou, Z., Shen, S. and Liu, H.X. (2023). Dense reinforcement learning for safety validation of autonomous vehicles. Nature, 615(7953), pp. 620 - 627.
12. Fink, G.A. (2014). Markov models for pattern recognition: from theory to applications. Springer Science & Business Media.
13. Friedman, R. (2021). Cognition as a Mechanical Process. NeuroSci, 2, 141–150. https://doi.org/10.3390/neurosci2020010.
14. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.
15. Greer, K. (2024). An Auto-Associative Unit Memory Network, Preprints, https://www.preprints.org/manuscript/202412.1209/v1.
16. Greer, K. (2022). Neural Assemblies as Precursors for Brain Function, NeuroSci, 3(4), pp. 645 - 655. https://doi.org/10.3390/neurosci3040046. Also published in Eds. Parnetti, L., Paoletti, F.P. and Gallart-Palau, X., Feature Papers in NeuroSci : From Consciousness to Clinical Neurology, July 2023, pages 256. ISBN 978-3-0365-7846-0 (hardback); ISBN 978-3-0365-7847-7 (PDF) https://doi.org/10.3390/books978-3-0365-7847-7.
17. Greer, K. (2021). New Ideas for Brain Modelling 7, International Journal of Computational and Applied Mathematics & Computer Science, Vol. 1, pp. 34-45.
18. Greer, K. (2021). Is Intelligence Artificial? Euroasia Summit, Congress on Scientific Researches and Recent Trends-8, August 2-4, The Philippine Merchant Marine Academy, Philippines, pp. 307 - 324. Also available on arXiv at https://arxiv.org/abs/1403.1076.
19. Greer, K. (2021). Category Trees - Classifiers that Branch on Category, International Journal of Artificial Intelligence & Applications (IJAIA), Vol. 12, No. 6, pp. 65 - 76.
20. Greer, K. (2020). New Ideas for Brain Modelling 6, AIMS Biophysics, Vol. 7, Issue 4, pp. 308-322. doi: 10.3934/biophy.2020022.
21. Greer, K. (2020). A Pattern-Hierarchy Classifier for Reduced Teaching, WSEAS Transactions on Computers, ISSN / E-ISSN: 1109-2750 / 2224-2872, Volume 19, Art. #23, pp. 183-193.
22. Greer, K. (2019). New Ideas for Brain Modelling 3, Cognitive Systems Research, 55, pp. 1-13, Elsevier. DOI: https://doi.org/10.1016/j.cogsys.2018.12.016.
23. Greer, K. (2012). Turing: Then, Now and Still Key, in: X-S. Yang (eds.), Artificial Intelligence, Evolutionary Computation and Metaheuristics (AIECM) - Turing 2012, Studies in Computational Intelligence, 2013, Vol. 427/2013, pp. 43-62, DOI: 10.1007/978-3-642-29694-9_3, Springer-Verlag Berlin Heidelberg.

 doi:10.20944/preprints202309.0370.v5

24. Greer, K. (2011). Symbolic Neural Networks for Clustering Higher-Level Concepts, *NAUN International Journal of Computers*, Issue 3, Vol. 5, pp. 378 – 386, extended version of the WSEAS/EUROPMENT International Conference on Computers and Computing (ICCC'11).
25. Gruber, T. (1993). A translation approach to portable ontology specifications. Knowledge Acquisition, 5, pp. 199 - 220.
26. Gupta, B., Rawat, A., Jain, A., Arora, A. and Dhami, N. (2017). Analysis of various decision tree algorithms for classification in data mining. International Journal of Computer Applications, Vol. 163, No. 8, pp. 15 - 19.
27. Hawkins, J., Lewis, M., Klukas, M., Purdy, S. and Ahmad, S. (2019). A Framework for Intelligence and Cortical Function Based on Grid Cells in the Neocortex, Frontiers in neural circuits, 12, p. 121.
28. Hawkins, J. and Blakeslee, S. On Intelligence. Times Books, 2004.
29. High, R., 2012. The era of cognitive systems: An inside look at IBM Watson and how it works. IBM Corporation, Redbooks, pp.1-16.
30. Hinton, G., 2023. How to represent part-whole hierarchies in a neural network. Neural Computation, 35(3), pp.413-452.
31. Hinton, G.E., Osindero, S. and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets, Neural computation, Vol. 18, No. 7, pp. 1527 - 1554.
32. Katuwal, R., Suganthan, P.N. (2019). Stacked Autoencoder Based Deep Random Vector Functional Link Neural Network for Classification, accepted: Applied Soft Computing (https: // doi. org/ 10. 1016/ j. asoc. 2019. 105854).
33. Kingma, D.P. and Welling, M., 2019. An introduction to variational autoencoders. Foundations and Trends in Machine Learning, 12(4), pp.307-392.
34. Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pp. 1097-1105.
35. Krotov, D. (2023). A new frontier for Hopfield networks. Nature Reviews Physics, 5(7), pp. 366 - 367.
36. Laird, J. (2012). The Soar cognitive architecture, MIT Press.
37. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015). https://doi.org/10.1038/nature14539.
38. Lieto, A., Lebiere, C. and Oltramari, A. (2017). The knowledge level in cognitive architectures: Current limitations and possible developments, Cognitive Systems Research.
39. Lynn, C.W., Holmes, C.M. and Palmer, S.E. (2024). Heavy-tailed neuronal connectivity arises from Hebbian self-organization, Nature Physics, 20(3), pp.484-491.
40. Meunier, D., Lambiotte, R. and Bullmore, E.T., 2010. Modular and hierarchically modular organization of brain networks. Frontiers in neuroscience, 4, p.200.
41. Miller, G. A. (1956). The magical number seven, plus or minus two: some limits on our capacity for processing information. Psychological Review, 63, pp. 81 - 97.
42. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 26.
43. Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X. and Gao, J., 2024. Large language models: A survey. arXiv preprint arXiv:2402.06196.
44. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D. (2015). Human-level control through deep reinforcement learning, Nature, Vol. 518, pp. 529-533.
45. Mountcastle, V.B. (1997). The columnar organization of the neocortex, Brain: J Neurol, Vol. 120, pp. 701 - 722.
46. Newell, A. and Simon, H.A. (1976). Computer science as empirical inquiry: Symbols and search, Communications of the ACM, Vol.19, No. 3, pp. 113 - 126.
47. Nguyen, T., Ye, N. and Bartlett, P.L. (2019). Learning Near-optimal Convex Combinations of Basis Models with Generalization Guarantees. arXiv preprint arXiv:1910.03742.
48. OpenAI. 2023. GPT-4 Technical Report. (2023). arXiv:cs.CL/2303.08774.

 doi:10.20944/preprints202309.0370.v5

49. Pulvermüller, F., Tomasello, R., Henningsen-Schomers, M.R. and Wennekers, T. (2021). Biological constraints on neural network models of cognitive function, Nature Reviews Neuroscience, 22(8), pp. 488 - 502.
50. Rock, I. (1977). In defence of unconscious inference. In W. Epstein (Ed.), Stability and constancy in visual perception: mechanisms and processes. New York, N. Y.: John Wiley & Sons.
51. Rubinov, M., Sporns, O., van Leeuwen, C., and Breakspear, M. (2009). Symbiotic relationship between brain structure and dynamics. BMC Neurosci. 10, 55.
52. Sarker, M.K., Zhou, L., Eberhart, A. and Hitzler, P. (2021). Neuro-symbolic artificial intelligence. AI Communications, 34(3), pp. 197 - 209.
53. Shannon, C.E. (1948). A Mathematical Theory of Communication, The Bell System Technical Journal, 27(3), pp. 379 - 423.
54. Spens, E. and Burgess, N. (2024). A generative model of memory construction and consolidation, Nature Human Behaviour, 8(3), pp. 526 - 543, https://doi.org/10.1038/ s41562-023-01799-z.
55. Tkačik, G., Mora, T., Marre, O., Amodei, D., Palmer, S.E., Berry, M.J. and Bialek, W. (2015). Thermodynamics and signatures of criticality in a network of neurons, Proceedings of the National Academy of Sciences, Vol. 112, No. 37, pp. 11508 - 11513.
56. The Gutenberg Project., https://www.gutenberg.org/browse/scores/top. (last downloaded 2/9/23).
57. Treves, A. and Rolls, E.T. (1991). What determines the capacity of autoassociative memories in the brain?, Network: Computation in Neural Systems, 2(4), p.371.
58. Tsien, R.Y. Very long-term memories may be stored in the pattern of holes in the perineuronal net. Proc. Natl. Acad. Sci. USA 2013, 110, 12456-12461.
59. Turing, A.M. (1950). Computing machinery and intelligence. Mind, 59, pp. 433 - 460.
60. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
61. Webb, T.W., Frankland, S.M., Altabaa, A., Segert, S., Krishnamurthy, K., Campbell, D., Russin, J., Giallanza, T., O'Reilly, R., Lafferty, J. and Cohen, J.D. (2024). The relational bottleneck as an inductive bias for efficient abstraction. Trends in Cognitive Sciences.