Note 4

A Note on Artificial Intelligence and the critical recursive implementation: The lagging problem of 'background knowledge'[1]

Humans tell themselves stories in order to get themselves to work on this or that. It is almost always the case that these high-level stories are relevant only as motivation and not really relevant to what eventually happens in terms of technical understanding.

Allen Newell

Opening Remarks

Most historians of the *Cognitive Revolution* consider the now historic 1956 MIT *IRE Conference* 'Transactions on Information Theory' to be the conceptual origin of the revolution. It was at this conference that three of the most important papers in the emerging field of AI would be read:

(i) George Miller's *Human memory and the storage of information* (coupled with an earlier 1955 paper *The magic number seven, plus or minus two: Some limits on our capacity for processing information*).
(ii) Allen Newell & Herbert Simon's paper *The logic Theory Machine: A complex Information processing system.*
(iii) Noam Chomsky's paper *Three models for the description of language.*

But it would not be long before splits would occur in the very defining of AI. For some, let's call them the **AI-soft** crowd, despite the ever-growing consensus that the brain really did not function like a computer after all, (as was earlier suggested by the naïve 'brain is computer' metaphor of the time), the AI-soft crowd, against the push-back, were content to go their own way and see just how far they could actually push their learning algorithms in solving 'real-world' problems (eventually using Bayesian networks). Most early cognitive scientists of this time—while now at least partially acknowledging and accepting the fact that what they were doing was indeed not real 'human-intelligence' modeling—would nonetheless remain

[1] This is a **Draft chapter** of 'Note-4' as part of my monograph entitled 'Recursive Syntax' (In prep, 2019) **LINCOM** Publications. Joseph Galasso, CSUN~Linguistics Dept. 2019.

undeterred from learning about how to improve upon these non-human-like networks. One AI-soft champion that stands out here would be Frank Rosenblatt and his *Perceptron* model for visual learning (1959-1962).

The other side of the split quickly emerged contesting that the brain is not just composed of neurons firing (viz., that the human brain is indeed much more than the sum of its parts). The rallying-cry would be that the brain is not at all data-driven, but that an inner *a prior* blue-print encodes how humans see and interpret the environmental data around them. For the hard-AI crowd, the human brain/mind cut (not necessarily promoting Descartes's dualism, but dualistic nonetheless), was said to 'boot-strap' a **Theory of Mind** (via a 'brain-to-mind' bootstrapping), and that such a theory was, *inter alia*, intuitive-based, (even superstitiously so), full of imaginary concepts (at times unreal)[2] and symbolic & categorical (rule-base)[3] in nature—all of which were viewed as being uniquely un-tethered to actual environmental stimuli & data[4]. This disconnect between (i) an inner computational algorithm and (ii) the environmental data would foster completely different predications and would run completely counter to any symmetrical algorithmic language of X=x, where weighted probabilistic outcomes supersede all else. This latter group which countered AI-soft might rather be called **AI-hard**, where any naïve attempt to model the brain by simple recurrent, Bayesian networks were pooh-poohed as being a simple product-calculation of patterns found in the data, and nothing more. This side of the debate was spawned by the likes of Marvin Minsky (see Minsky vs. Rosenblatt debates):

> *'Where Rosenblatt would argue that his neural networks could do almost anything, and Minsky would counter that they could do little…'*

One essential aspect of the hard-AI crowd was how they viewed human language as human reasoning. The most recent contribution here comes from Judea Pearl (a UCLA cognitive scientist who some espouse as the one of the founders of second-generation AI). Pearl argues that Minsky's skepticism can be completely understood now in the sense that all Bayesian networks can do is achieve a probabilistic outcome as based on **symmetrical** language, of the sort X=x, as understood within (B.F. Skinner's) Behavioral theory of **Association.** Pearl shows

[2] See S. Toulmin's (1961) great book on the topic of a 'model-based' vs 'model-blind' dichotomy (attested in the rival approaches between Babylonian and Greek science).
[3] See Pinker, (1999) 'Words & Rules' Theory.
[4] A 'Poverty of stimulus' argument originally proposed by Chomsky. (See Chomsky 2002 pp. 5-6, 8 for review).

how this is the symmetric language of algebra: If X tells us about Y, the Y tells us about X (a one-to-one relation). But the human brain/mind does not reason like this: e.g., 'rain may cause mud, but mud doesn't cause rain'. Pearl rather is animate that the only kind of complete AI that could deliver a real-like human thought-process (AI-hard, or AI-complete) must be based upon **asymmetrical** language of the sort: X tells us something about Y, but Y doesn't tell us anything about X (a one-to-many relation). Such a calculus for asymmetrical language is non-algebraic (perhaps rather geometrical) and has only been conceived as a written language quite recently (over that last two decades). In short, what is needed is recursive mathematics of the kind Pearl describes. Pearl has worked up such a model for how to achieve such asymmetrical processing.[5]

(i) Associative model: P(y|x) (typical activity = *seeing* on an iconic 1-1 relation).
Theory: Behaviorism, symmetrical language, probabilistic. **{α, β}**

(ii) Intervention model: P (y/do (x), z) (typical activity = *doing* of a 1-many relation).
Theory: categorical, symbolic, non-probabilistic **{α{α, β}}.**

The former *recurrent* model tells us: <u>What is this</u>? (as based on frequency of data).
The latter *recursive* model tells us: <u>What if</u>? (as based on causal reasoning).

The above dual distinction of '**recurrent** vs. **recursive** structure' is what will be more fully expanded upon in this note on AI—a distinction which draws out a critical contrast between:
(i) **associative**/means (with the flat structure of [x,y]), as compared to
(ii) intervention/**recursive** means (with the hierarchical structure of [x[x,y]]).

Pearl is one of today's most animate cognitive scientists who have come out strongly against any claim that mere association, as defined by the relation of naked data (which entirely relies on pure input-data of a brute statistical kind), can ever be an **Operating System** (OS) which delivers real human-like thought

[5] See J. Pearl (2018) *The Book of Why: The new science of cause and effect.*

processing. For Pearl, a **recursive** and **causal** reasoning algorithm, at the very least, is required of such an OS.

Finally, as things stand today regarding AI, most cognitive scientists still find themselves folded somewhere along the Rosenblatt/Minsky cline: for example, considering current events, such a spectrum might look like this— (a first group) those faithful to self-driving cars are the children of Rosenblatt, while (a second group) those who entertain a healthy skepticism of its human-like driving ability are children of Minsky… (See Gary Marcus and further discussions below).

Overview

More than 20 years ago, it was already being claimed in the emerging AI world that, following a kind of Gestalt psychology, any putative 'computing to human-behavior' process could not simply rely on analyzing things into their atomic parts or logical symbols, and then to expect 'symbol-manipulation' processings of those parts to deliver any logical meaning. It turned out that real human cognition was much more fluid than that, as human behavior produced (at the very least epiphenomal/peripheral) noise within its own signal (viz., human cognition is not just the sum of its material atomic parts, even if we were to understand what those parts actually are, which, a the moment, is very far indeed beyond our current reckoning). Thus, human behavior cannot survive any material reductionism as would be required via AI. Two very hard problems, 'flexiblity' and 'context-sensitive' procedures most surely be implemememented in any viable AI/deep-learning algorithm. As part of several ongoing experiments (chief among them today's experiments for autonoums self-driving vehicals), what the ensuing thesis seems to be—which continuously emerges and resettles within both AI-advocate and skeptic camps—is that what we want from a strong AI program is not just to ask *what* the input is (as in 'what's the source of the encylopedic entry?, etc.'), but also to ask *how* that source/input is delivered into the algorithm. While this latter source of '*how over what*' is a property often over-looked by those building AI code, at least to my mind, it is a quientessential property most critical in implementation if we wish AI-deep learning to come closest to simulating true human behavior. In order for AI to avoid AI winter[6], theory of mind & background-knowledge-based

[6] 'AI-winter' is a term used by AI skeptics (cognitive scientists who have long-held believed that AI will not (anytime soon) be able to simulate true human behavior. AI-winter also has

learning capacities must be implemented in the system: deep-learning computers must learn not just *what*, but also *how* its source-input is being delivered into its system.

Artificial intelligence (AI) implementations have exponentially grown over the second half of the last century. However, little can be said about a real 'qualitative shift' since first-generation computing. Today's AI scientists are seemingly still grappling with problems—some of which are quite basic, if not altogether primitive—problems stemming from the fact that AI doesn't know how to get 'from here to there' in simulating human-like cognition. Real advances have rather come in a quantitative manner, with speed, size of memory, multilayer connections (so-called 'perceptrons'), etc., but little if any headway has come in a substantially qualitative manner.

Today, as was the case back in the 1950s, most AI-coding operations assume a 'catastrophic-decision-based' platform where all its recurrent operating system (OS) can do is statistically *average* weighted input [A] with weighted input [B], whereby the weighed and statistically-average product is the mere sum of its parts [A, B], etc. In this linear and recurrent manner, [A], [B] have been simply combined with the two being averaged to produce [[A], [B]=[A, B]]. So-called 'simple recurrent networks' (SRNs) with an OS based on input-outputs with recurrent loops cannot remember past individual inputs since all information has been consequentially merged. The OS can't recall past inputs more than one single computational time-step (CTS) away: they hold no history. (This computational amnesia will become a point of interest to us if we wish to simulate human-like cognition, such as intuition and background- knowledge). But while they can average, they then can't properly adjust, it seems, given their tight restrictions of 'locality' (another point of interest for us). This is precisely how *recurrent* OS differs with *recursive* OS's. In human-cognitive processing—which rely on non-local variables, as well as recursive multichannel processing—inputs, even of the [A, B]-type, resend via feedback loops which may go against the grain of averaged distributional weights and frequency effects. Such loops have an uncanny ability to remember a CTS many steps away. In other words, the information is preserved throughout the computational history, thus allowing not just for averages to accumulate across different pathways (say, from point-α to point-β given parallel processes), but that such accumulations can be the result of a historical record, where previous input (lost in SRNs) can be recalled to add an adjustment across variable pathways. This in essence is how

an economic aspect to it since much of the current U.S. economy has priced-in the stock market billions of dollars of future speculation based on AI promises.

human learning works: where (de)learning can be established based on past errors and/or processing glitches.

Now, while this narrative runs for earlier SRNs (of the simple, linear and recurrent type found in G-1 (see below for history), the same processing essentially is intact for more up-to-date complex recurrent networks (CRNs). Whereas SRNs (with no more than one vertical stack of inputs) had notorious trouble generalizing to new and novel items, CRNs, now allowing for more than one hidden unit (having multiple hidden units), could multi-channel in the sense that feedback loops could maintain some history over many CTS. Given this added feature, however, the final actionable processing by the very nature of its OS, had to give priority to local constraints.

A case in point is where we currently find ourselves regarding autonomous (self)-driving vehicles (ADV). These operating platforms are still 'catastrophic-decision-based'—they essentially are supped-up SRNs which rely on averages of an input stored over time, but where the actionable final input is merely the best calculation that has been added over, say, thousands/millions of generations (CTS) (data-driven experiences). Each decision is allied to a specific point in the spectrum of a spread-out data field without so-called *gradual decay* of past information which might be preserved in any calculus. Irrespective of these multi hidden units, the final input has no memory of how those exchanges took place from point α to β between various generations. Why it should matter becomes a point of contention for us regarding how we believe human-cognitive learning takes place.

At the very earliest conception of AI, most scientists assumed a 'brain-to-computer' analogy with the rationale that since brains are essentially composed of neurons (along with a patch-work of neuro-nets) than what AI first needed to do was begin an OS that mimicked low-level neuron processing. Starting 'small' is always the best path forward whenever beginning a new scientific enterprise—whether it is with first studying the earthworm for genetic material before inquiring about the human genome, to analyzing young children speech to understand the properties and structure of adult language. So, given this logical strategy, the first AI OSs attempted to mimic lower-level neuron firing (with the Hebbian cliché of 'what fires together wires together'). SRNs with their local neuro nets seemed to logically fit the model. In the historical context of behaviorism, the model gained acceptance.

A very brief AI history

(G-1) The very first pioneers of Artificial Intelligence (AI) (Generation-1 (G-1)) were actually mathematicians and logicians (intellectual philosophers of the Rationalist persuasion). In 1955, Allen Newell and Herbert Simon created the **Logic Theory** approach to AI (at The Rand Corporation think-tank in Santa Monica, California). They held a rationalist philosophical view (form Descartes, Leibniz, to Kant, Husserl). This was traditional to their fields at the time—namely, that the 'body-brain' cut was indeed real (following from a Cartesian principle) which meant that there remained a 'disembodied-mind view' which saw that knowledge could only arise out of abstract symbol-manipulation: that knowledge begins in the mind, and that the world is then copied outside from the inner mind). (The neo-Cartesian philosophy of the 17th century saw its resurgence come from the Chomskyan paradigm (**nativism**) of the 1950s, based out of MIT). Abstraction was assumed since, in their rationalist framework, that was the only way the mind could possibly work.

In this view, inner-subjective symbols and rules (brain/mind) necessarily linked to the objective things in the outside world to give meaning. Hence, any sense of AI-meaning could only arise via symbols and/or categories. (Recall, the neo-Cartesian effort of Chomsky came out of studies of human language, syntactic analyses and child language acquisition). One could claim that G-1 logical-theorists really didn't take the word 'artificial' in the term 'artificial intelligence' seriously, since what they were in fact grappling with was 'real' intelligence of the human-logic form. In the end, while this 'cognitive-revolution' version of AI enhanced our understanding of symbol manipulation of logical forms (the 'brain as calculator' metaphor, the brain as 'rule-based computer', or the brain as a 'digital operator', etc.), it ultimately would fail since it avoided the *hardware* problem of how the brain is really structured. It would later be rightly claimed that the brain is not really structured in abstract symbols (nor is it a computer), but rather the brain/mind is comprised of billions of neuro-net connections (more in-line with features/properties of connectionism).

So, the hardware question became a critical one: Do we wish to simulate what the brain is actually doing at the lowest levels of processing (at the neuro-network connection), or do we wish to just stay within abstract theory (of symbols and rules) and see how far we could get? Well, the end result was that Newell and Simon really didn't get too far with symbols and rules (outside of basic tasks), and so the next generation would more seriously have to reconsider the hardware problem and develop neuro-net connections. This hardware problem was what led to the second generation (G-2) school called 'connectionism' of Rosenblatt (1957).

Another factor in G-1 was the (false) premise that 'digital-seriality' was necessary for human behavior/thought (so-called 'serial processing'). Otherwise, it was thought that under a human 'analog- parallel' mode of processing (a best-scenario) one part of your human cognition could simply cancel out the other, or, perhaps (a worst-scenario), one part could not stop what you are thinking in the other part. However, upon closer examination, this serial/digital approach to human thought also turned out to be wrong: (the human mind/brain is an analog machine (not digital), a parallel-processing machine (not serial) (as observed by Patricia Churchland).

But human parallel processing turns out to be a very large problem indeed—a problem still yet to be completely overcome by our fastest computers in current AI. (Problems such as C*ommon-sense, background-knowledge* and *intuition* are very unique human-specific *procedural* activities and seem to defy serial *declarative* processing—they are very likely to be the murky residual result of 'parallel' processes between two or more modes of input/output channeling at any given time). Hence, any attempt at AI could never simply be about 'data collection' (declarative facts), whether with G-1 linear symbol manipulation or G-2 neuro-net connectionism (the former dealing in abstract logic, the latter in simple recurrent networks (SRN). Rather, what future AI would have to do—well beyond the earlier prosaic modes of programming—would be to take 'data collectively' (procedurally) over a spread of multi-layer processing levels.

The trick, as we will discuss below, is that each time-step of the 'data collective' (*pace* 'data collection') must be self-preserving and remain distinct—that is, the data can never be compromised at any of the intermediate steps as a result of, say, when one datum {α} is combined with another datum {β} and then averaged together as a new single result {γ}, thus losing each datum's individual and unique value. This problem in fact was what plagued the two aforementioned earlier modes of AI, and, moving forward, is what continues to plague AI/deep learning today.

(G-2) But in the beginning of G-2, early networks were very limited in the sense that they were 'vertically processed' (stacked on top of one another) and had no more than two layers of stacking. These units were processed 'vertically' as linear units without so-called feedback loops. In other words, the strict associative (**behaviorism**) of a one-one input-output algorithm was all that was assumed. Frank Rosenblatt (1957) (at Cornell University, Cognitive Systems Research Program) was the first to attempt a **connectionist theory** (of binary classifiers) towards human behavior (vision) in this way. His *perceptron*—an attempt of a first computer program system that could learn new skills by associative trial and error—used a type of G-2 neural network that simulated human-thought processes. Of course, while G-2 was an advance from G-1 symbols, the problem was that the G-2 neuro-network was much too limited both in memory and in speed, and as a result ultimately would fail.

The classic Minsky and Papert paper (again, a product of MIT) quickly shot-down any assumption that human cognitive learning could be simply mapped in such a stark binary and linear manner (again the brain is analog, not digital). Also, by this time (1956) Chomsky's first important paper had already come out (seemingly in support of the doomed G-1 model) defending the skeptics of linear behaviorism in his paper *Three models for the description of language*. Chomsky, though seemingly in support of the 'backwards-looking' G-1 symbol manipulation theory, was correct and 'forward-looking' in calling out that such a simple associative AI theory would only ever be able, at best, to map top-down input already installed in the program to tags matching the output. AI, under such limitations, would never simulate human cognition.

But, advances would still be made on the system: e.g., such top-down encyclopedic tagging (with knowledge already inputted from the top) would ultimately pave the way for so-called 'micro-worlds' of expert AI-systems (closed-worlds where only specific, relevant material to the task could be drawn upon and where the so-called background/commonsense problem that humans innately enjoy (seemingly without step-processing) wouldn't be required). In the main, such simply associative learning was attempted by most connectionist programs of the day with little success. For about twenty years thereafter, AI was both frustrated and stagnant: many of the basic problems we thought we could solve turned out to be ridiculously out of reach: (such problems may be so-called 'hopeful monsters')[7], requiring a completely new way of bootstrapping our

[7] Richard Goldschmidt was the first scientist to use the term 'hopeful monster whereby it was thought that 'small gradual changes' could NOT cause a change between the (tiny) microevolution-level and (large) macro level. Hence, something quick and large would have to be the source, a so-called 'saltation theory' (or quick jump, as in Steven J. Gould's

understanding. A 'saltation theory' is required as in the notion that the evolution of a species might not come from incremental-gradual adaption over a long period of time, but rather might 'jump-up' ('salto', Latin for 'jump') all at once and rather quickly via an 'exaption' from micro-changes either in the environment (via *neo-Lamarckian inheritance*) or in the genome itself (via *cryptogenetics*).

For instance, consider one such aspect of the 'memory and storage' problem related to encyclopedic micro-worlds: a seemingly trivial task for human behavior, but a major problem for AI seems to be the case, even at a basic associative level, that if one were indeed capable of successfully down-loading all human-encyclopedic knowledge into a connectionism system, there would still exist the problem of 'how' one would be able to access the right material needed at any one time (at the right time). Humans get the 'accessing problem' right away. It's called procedural-access of declarative-knowledge. We don't even ask the question of 'how' to access 'what' 'when'. It's just part of our background, common-sense knowledge.

Well, it turns out that even **if** we get the 'memory & storage' problem right, we still have to deal with the other side of the equation, the proper 'access & retrieval' problem. Hubert Dreyfus (while at the Rand Corporation, contemporary with Newell and Simon) was once quoted (talking about encyclopedic machines at that time) as saying (my rephrasing in italics) 'The problem *may not be* how you can get the vast amount of knowledge into that super-big memory (he doubts that you can), but even if you did, how could you possibly access the part that you needed in any given situation'.

So, for G-2, some of the problems they were already grappling with was somehow related to this commonsense/background knowledge we all have, apparently inherent in our design of the brain/mind, with spin-off attributes leading to subjectivism and theory of mind. The next generation G-3 would either have to deal with this hopeful monster in new and creative ways, or else sweep it under the carpet as some imposter artifact, as some unique feature outside the peripheral realm of AI, and somehow not implicated in the grammar coding of future expert systems. In other words for G-3, the macro-world of human commonsense-

'punctuated equilibrium'). In his book *The Material Basis of Evolution* (1940), Goldschmidt wrote 'the change from species to species is not a change involving more and more additional (small) atomistic changes, but rather a complete (large) change of the primary pattern or reaction system into a new one, which afterwards may then produce intraspecific variation by micromutation' .

background knowledge will either be an impediment to AI (perhaps provoking leading AI-skeptics of the day), or it simply won't factor in to the AI equation.

G-3 Then a breakthrough came in the 1980s (largely from work happening at the University of California, at San Diego (UCSD)) by the likes of Geoffrey Hinton and James McClelland, David Zipser (all three who contributed to the seminal book *Parallel Distributional Processing (PDP)*), Paul and Patricia Churchland, then later Jeff Elman—all of whom in one way or another made major contributions to work in 'parallel distributive processors'. Most crucially, and different from earlier associative-connectionists models (of Rosenblatt), these new and immensely more powerful PDPs were of a very new kind of processor—whereby so-called 'hidden units' (hidden layers) and 'back propagation' (feedback loops) where inherently encoded into the architecture and design of these superfast computers.

Such advanced work at UCSD rekindled old and often heated debates in the 1990s between strong AI-advocates such as Daniel Dennett, James McClelland, and Jeff Elman (of the Peter Norvig persuasion)[8] versus AI-skeptics (of the Noam Chomsky persuasion) such as John Searle, Jerry Fodor, and Gary Marcus[9]. I personally can recall the debates between the latter-two counterparts (Elman v Marcus) in the 1990s (while I was a doctoral student with Harald Clahsen (University of Essex)). From the tenor and passion of their arguments at the time, one got a pretty good sense of the direction to be taken and all the problems to be had facing future AI as it would pave its way towards the 21century.[10]

The 'PDP-promise' included a framework which was closest to an artificial neural network found in human cognition. The model stressed the parallel nature of neural processing and distribution.

[8] Link-29.
[9] Link-30. See Marcus for a recent summary.
[10] Link-31.

The new PDP promised: [11]

(i) that the computer architecture would connect many more than the previous G-1 simple 'two-layer unit'. (In fact, PDPs could now use, at least theoretically, unlimited amounts of connective networks (perhaps limited only by the size of its hardware),

(ii) that rules/symbols would no longer be the engine of computation—rejecting the putative 'human-like' (and more analog) manipulation of rules over the favored (and more digital) connectionist computation (but recall Patricia Churchland's observation above that the human brain/mind is **analog** and **not digital**),

(iii) that PDPs would allow for real 'deep-learning' to take place via feed-back loops which fed into multiple hidden-units. These new 'post-hidden-structured' unit would in turn loop backward from the incoming data-stream and return it as a new forward signal.

G-4 Recursion.

We are about two thousand years from having a serious theory of how the mind works... Jerry Fodor.

I'd like to address our contemporary Generation-4 (G-4) in specific terms as it currently relates to the micro-world of **autonomous-driving vehicles (ADV)** (currently hot off the press). First of all, most cognitive scientists along with strong AI researchers today recognize the notion (now, a close truism) that the mind is not a digital computer. Sure, at one level, say at the lowest level of processing, specific neuron firing may have certain attributes analogous to binary operations (of off &

[11]PDPs allowed for: Processing units, represented by a set of integers, to be activated for each unit. Each output function for each unit is represented by a vector of time-dependent functions on the activations. An activation rule for combining inputs to a unit determines its new activation, as represented by a function on the current activation and propagation. A learning rule (algorithm) is implemented for modifying connections based on experience, which are represented by a 'change' in the weights based on any number of variables. An environment that provides the system with experience is represented by sets of activation vectors for some sunset of the units.

on). But surely, learning and knowledge is not the mere brute result of any single statistical product: in brief, human learning is not a kind of 'statistical inference' but rather a sort of formation derived by abstract theory. So, at one level, at least at the lower level, a kind of associative/connectionism may in fact be what is going here: it is true, for most micro-word expert systems (say, analogous to lower-level mind operations), statistics do seem to drive the computational result (and the best result is always on the upside of the 'statistically averaged') The bell-shape curve of a competency of a skill follows a statistical curve. But it may be that the skill-competency of activity {x} is not the same as knowledge of {x}. Statistics may grant us an understanding of how a hypothesis of {x} is tested against the body of data {y, z}, but statistics alone cannot grant us an *understanding* of how the hypothesis was generated.

It seems a more abstract, higher-level processing beyond mere statistics is required. But it is true, at lower levels, a kind of Hebbian[12] calculation seems to hold: Hebb's expression '*what fires* (statistically) *together wires* (statistically) *together*' is now a well-accepted biological truism. But it seems, still at G-4 PDP-connectionism, that all we have done is push the associative, lower-level 'binary statistic' up towards a pretend 'higher-level' processing. But it has not arrived there. Are we just faking learning? We do this faking whenever we claim that self-driving cars can 'learn' beyond statistics: but statistics have no means of learning: numbers can't 'learn', they can only 'average'! Sure, it may turn out that their averages improve and the bell-shape curve shifts upon more and more experience, but there is no learning in the sense of how humans learn. Considering the ADV method of learning, it may be that a hypothesis of the driving act can be measured and tested against a body of data, but even so, we still don't know how that hypothesis was generated in the first place.

For some AI researchers the question may not even come up: What does it matter to them if they don't understand how the hypothesis was generated when they have successfully tested their data? But, sooner or later, it will matter—as when ADV hypothesizes that a very large white scene just in the view ahead is statistically analyzed, averaged, and wrongly interpreted as free-open road (and the self-driving car runs into the lateral side of a tractor-trailer killing its unaware passenger), or, as a preliminary report from federal safety regulators have detailed, when an ADV's sensors had in fact detected a 'woman pedestrian' but its decision-making software discounted the sensor data, concluding it was likely a false positive

[12] Donald Hebb way back in 1949 showed that neuron clustering could alter synaptic coupling which in turn could change synaptic strength.

(and the 'woman' is struck and killed). Again, **Learning** is never a statistical gathering of atomic parts, but rather true learning happens at a 'higher-level' processing (not the sum of its parts), it is abstract, and it seems to tap into fundamentally different but parallel modules of the mind. I'd rather prefer to postulate that for future G-4 AI, the question should be 'Can we get some success, even minor success, out of a machine that goes beyond statistics?' At least we would then be telling the truth: with such a question, we realistically tether our promised AI to the very primitive associative methods we have at one's disposal, and nothing more beyond that. As Fodor says, it may take us two thousand years to understand how we get from low-level firing of neurons to higher-level consciousness and thought. When we finally do, only then can we claim we understand how the mind works.

At the moment, our best algorithms for ADV semantic imagining (the ability to recognize the environment) rely totally on trained end-to-end, pixels-to-pixels, semantic segmentation.[13] But, while the best AI-imaging software currently being deployed, as connected to deep learning, seems to be very good at very difficult imaging averages, the software has major catastrophic breakdowns when dealing with the simplest of things. Here's such an example (quoted from Gary Marcus *NY times*, 2017):

> *Even the trendy technique of 'deep learning', which uses artificial neural networks to discern complex statistical correlations in huge amounts of data, often comes up short. Some of the best image-recognition systems, for example, can successfully distinguish dog breeds, yet remain capable of major blunders, like mistaking a simple pattern of yellow and black stripes for a school bus. Such systems can neither comprehend what is going on in complex visual scenes ('Who is chasing whom and why?') nor follow simple instructions ('Read this story and summarize what it means').*

Such installment of 'semantic information processing' into the operating system reminds me of initial problems beset early SRNs of the 1980s, when problem-solving programs went in operation hoping to 'solve problems' but without any 'knowledge of relevance' of the problem itself. It was the improbability of such tasks that got AI cognitive scientist to rethink the importance of belief, intuition,

[13] Link-32. See Marcus for a review.

and commonsense/background information—what would become known as the 'commonsense problem' (See Dreyfus's book *Mind over Machine*).

What Gary talks about in the review is the need for future AI to have two modes of simultaneous processing: namely, slow **bottom-up** processing which is sensual in nature (environmental) (perhaps equating to so-called 'declarative' knowledge), plus rather faster **top-down** processing which is theory-formation based (equating to so-called 'procedural' knowledge). Such an AI dual processing would mimic what we find at two levels of human processing—with the lower-level and slower **serial processing** having a connectionist-network quality (say, at the neuron level in the brain) and **parallel processing**, which is bootstrapped by lower-level **connectionist** but simultaneously acting in an autonomous manner from the lower level having a (very human) symbolic and rule-based theory formation quality. The former serial processing occurs at a much slower in speed in humans and is conscious (as in 'step-by-step'/ingredient building), while the latter parallel processing is much faster (and perhaps subconscious in humans), and is rule-based and abstract. I'd extend the analogy by suggesting that the human **brain** is rather a slow serial processor (with neurons firing to make connections)—that's why computers so greatly outmatch human capacity, computers are extremely fast serial processors at every step of the way at the lower levels. Humans get muddled-up between steps as we attempt to surface up bottom-up results to our top-down understanding.

It is my attempt below to try to simulate a thought experiment of what it would mean for an AI algorithm to have such a dual mode of processing, whereby both (lower-level) recurrent processes overlap with (higher-level) recursive processing.

Neural Networks

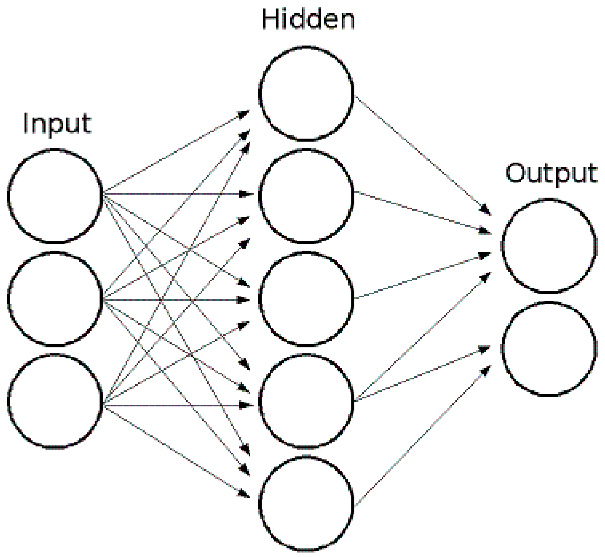
Artificial Neural Networks (NN), as the basis for most SRN operating systems, was the AI response to how brains were thought to compute. The analogy of the brain as a computer, on one hand, seemed natural enough since brains are indeed bundles of neuro-circuits (of a binary nature [0,1] off & on switches). NNs, in this way, were thought as sufficiently capable of handling most brain-related tasks since such modeling were essentially non-linear. NNs are extremely capable of modeling nonlinearity tasks such as classification, associative memory and clustering. Such a diverse means of processing allowed NNs to be modeled to solve many related tasks. The problems early NNs faced were mostly due to limitations of training:

large amounts of time had to be reserved in order for the NN OS to learn and cope with the task at hand. Also, as stated above, the appeal to NNs was that it, like the brain, consisted of a collection of neurons (or perceptrons/nodes), and each perceptron had its assigned input to output production. Training of such NNs (multilayer perceptrons) involved error-feedbacks: each time an error value is fed back, its weights of that connection is adjusted so that over time, learning can be achieved. Over time (training) the averaged and weights were adjusted so that better approximation could be produced (hence, learning). The factors behind NN amount to how many hidden units or layers were predetermined in the system for feedback. Also, another factor was how the units were labeled (either via mere associative chunk of item (for example 'cat' [cat]), semantic feature of item ([cat]: {+furry}, {+ four-legged}, {+ Purrs}, {+whiskers}, {+ animal}), or even based on its phonological units ([Cat]: onset /k/ nucleus /æ/ coda /t/). The most common use of NNs are called multi-layer perceptron which have multi inputs along with hidden units, where labeling and identification of input is determined by either label of item (chunk), semantic (features) or phonology. However, what's been overlooked so far is how to employ syntax into such NNs. Recalling that it is precisely syntax that allows to depart from strict one-one associative/recurrent formations and allows for recursive/hierarchical structure to be employed as part of its neuro-network—recalling that a neuro-node has only one output value. Neurons don't actually calculate. They simply pass on the input value to the next layer. The notion of 'learning' for such Networks is tantamount to associative-memorization systems which operate under 'rate of frequency' (occurrence, distribution, position, etc). As an exercise, try to imagine how such a network would treat {s} in the following two words:

[1]	'Fix'	IPA = /fIks/	processing =	[fIks]
	'Speaks'	/spiks/		[[spik]s]

(Note: For NNs, there would be no way to tease out the syntactic distinction that both final {s}'s, though both have exact position (final), have the exact distribution (_ks) (both verbal), and have the same sound /s/, nonetheless would have competently different underlying processing. Only an NN with recursive treatment and comparison of the two {s}'s would show that the {s} in the verb [fix] is part of the stem, while the {s} in the tensed verb [[speak]s] is an inflectional features of the syntactic features {3P, SG, Pres} (third person, singular, present tense)).

Fig. 1 Multilayer feed-forward neural network



(For back propagation, neuro-pathways would also go backwards, in so-called error-trained learning).

Back-propagation models

In back-propagation models (multi-layer perceptrons) an 'external teacher' (or supervisor) is required to steer learning. This supervisor either must be pre-installed in the system (innate architecture) or can come via some pieces of information found in the environment. In any case, learning is not the mere product of input and outputs but rather is steered via some parameterization. For AI enthusiasts who claim that the greatest feature of connectionism is that it assumes no innate structure would be hard pressed to explain-away why such supervision is required in the first place. But recall, that even the most ardent anti-innate people have to at least assume some small amount of innate structure in order to first calibrate an initial (neutral) weight for weighted learning. But, to a large degree, such an 'external teacher' could be made available in the environment itself—e.g, a small piece of info that could then bootstrap larger chunks of info, thus leading to wholesale learning. Once trained, such models could, for example, predict the next word in a sentence (in a sentence-learning program). For instance: what is the next word in the sentence 'The dog chased the__'? For a sentence-learning algorithm, 'the dog chased__' would have an weighted averaged for [dog= A] + [chase=B__] which would trigger [C=cat] if we were to have weighted nodes which push the probability connection of dog+chase+X as X is a Noun (some other animal, high average is [+N, cat], etc.). This could be learned from the environment if previous sentences with certain semantic features behaved accordingly (*dogs bark at cats, dogs fight cats*, etc). Of course, one problem might me for a sentence-prediction algorithm to ascertain embedded structures such as 'The dog$_j$ the girl chased__$_j$' (predicate what comes next: what is [__])? Such structures which involve object-raising may prove difficult for mere SRNs without necessary recursive networks. (We'll return to recursion in latter sections).

Claims 'For' and 'Against' Multilayer Perceptrons (ML-P).

'For' ML-P

Claim 1. As mentioned above, **multilayer-perceptrons** (ML-P) seemed to have a lot going for it since they did well to mimic what we (we thought) we knew about the brain: nodes are analogous to neurons and connections to synapses. To a large degree it is true: brains are composed of bundles of neuro nodes (synapses), they are binary based (+/-habituated), and are valued (=weighted learning) as determined on the averaged frequency of input. Hence, brains/neuro-nets were understood to be nothing more than little *Skinner schemes* (B. F Skinner) as understood in most behaviorism models of the day (viz., associative-learning schemes). It was taken for granted that connectionism of the day was the best fit for how the brain actually functions. This 'brain to computer' analogy gained much favor in the AI world, (at least until Minsky & Papert's 1969 paper came out to prove sever limitations on such meager associative models). Also, it was becoming well established at the time that the brain didn't work as a symbol manipulator (based on rules) since rules held no biological foundation (they are only abstract conventions). These two factors placed together (brain as neuro-net, not as symbol manipulator) paved the way for NNs over the next couple of decades despite Minsky & Papert's disclaimers[14] and others how question if the brain was really digital after all (Minsky, Fodor and others *pace* Chunchland and others) as discussed elsewhere in this text). Others preferred ML-Ps since they required very little in the way of innate architecture (although, that is not quite right since even with advanced ML-Ps, some amount of pre-installed (innate) values had to be assumed

[14] First Minsky & Papert demonstrated the severe limitations of earlier SRNs which lacked hidden units, and even went so far to dispel how systems with abundant hidden units would still remain insufficient in handling human-like thought-processing. (See Hadley 2000 for review).

in order to get the 'weighted'-values started). People such as Churchland were animate that brains could learn directly from the environment and no innate structure had to be presumed. This innate debate that carried over into the AI field was a direct result of debates being had in the field of language and child language acquisition (Noam Chomsky)—arguments such as the 'poverty of stimulus' debates, etc.

Claim 2. ML-Ps also seemed to gain much favor in the AI community since such networks didn't assume any innate structure (which was the Holy Grail for AI). But, as hinted above, as it turns out, this is not entirely right: some amount of innate structure is always required—it being either assumed in the architecture, or is assumed in the way a search is conducted of the environment.

Claim 3. True: the style of learning achieved by ML-Ps do seem to mimic human learning to the extent that e.g., **over-regularization** can be produced by the learning algorithm when it comes to the learning of language inflection (in child language development, as stage exists which sow examples such as *singed*, *breaked, taked, goed*, and even *wented*—all examples of over-generalization of the regular rule (past {ed}) onto irregulars. Such a thing we would want to see of a real learning algorithm. However, only later with back-propagation do they correct such irregular sound formations (e.g., _ing > _ang, for sing to sang) and so only when the input has been extremely modified (i.e., abrupt adjustment) to fit such sound pattern sequences. In other words, the whole vocabulary had to be reinstalled in order to cope with irregulars, and an entirely different vocab index was used for regulars. The problem here is that the networks was be sheltered by two different (artificially supervised) inputs since any ML-P can only work via a single mechanism (a single mechanism model). (See Marcus et al. 1992 for review. See Pinker 1999 for review of a dual mechanism model).

Even when trained with irregulars-vocab input, the ML-P was unable to handle so-called *denominals* such as 'ringed' (here, where an inherent (irregular) verb takes on regular qualities behind the derivational processes of Noun (*a ring*) to Verb (*to ring*) => [N-V {{ring} ed}]. It seems at the very least a dual mechanism model is needed to handle such diverging data regarding regular vs. irregular processing. (See 'Dual Mechanism Model' below).

'Against' ML-P:

The Sandwich problem

Claim 1. It remains clear that ML-P can't deliver the full range of human thought—viz. ML-Ps seem unable to capture a class of functions such as [+/-partitive, +/-individual] (a typical 'object [+/-specific] feature associated with determiners (*each, some*) as compared to pronouns and proper nouns—viz., a comparison of 'kinds/Noun' versus 'individuals/Pronoun', etc). For instance, in the dual proposition *John kissed Mary$_j$ & Fred kissed Mary$_j$ [kiss M* [J&F]] the ML-P can properly treat 'Mary' as one in the same persons [+individual] (as an individually-coded item)—viz., as the Pronoun (Mary/she) which properly triggers a [-partitive/+individual] function. But the harder problem rather may be with simple nouns. For instance, in the dual propositions *John ate a sandwich$_j$ and Mary ate a sandwich$_k$* an ML-P may not be able to recognize the referential distinction that the *sandwich* is not one in the same—namely, ML-P would rather treat both *sandwiches* as the same, subscripted with same index (sandwich$_j$).

In order to capture the partitive function in coding, the register/node which codes for generic 'sandwich' say e.g., [0110110] would also have to be doubly coded for *sandwich-1* [01101101] vs. *sandwhich-2* [011011011], etc. Such itemized weighted functions would place a heavy burden on the limit of coding (which may create problems associated with so-called 'cross-talk', e.g., when an overlap of coding creates ambiguous functions). (See *Blending* problems below). It rather seems what we want out of a well-functioning ML-P is the ability to represent both relations between **local nodes** (= sandwiches), as well as **distant variables** (= *sandwich$_j$ $_\&$ sandwich$_k$*)—and to be able to compute across the two representational systems. (Keep in mind that our upcoming proposal calling for a *dual mechanism mode (DMM)* (see below) will be in a unique position to do both: allowing for (i) local neuro-nets to represent +frequency-based registers (nodes), while (ii) non-local relations between variables of indexes/diacritics are maintained. It is here that we find that non-local representations between nodes and variables are unbound by frequency of the stimulus, but are rather recursive/symbolic in nature. Hence, by definition, a DMM is inherently a *parallel processor*, whereby both [+Freq] sensitivity of nodes and [-Freq] of variables can be simultaneously coded.

Recall, that the only source of information an ML-P has to 'sandwich' is a set of input nodes that are activated at the moment the item 'sandwich' is retrieved. There seems to be no way a *single mechanism model (and serial processor)*—which is completely reliant of local-node frequency—could go beyond this one-one associative coding. Of course, in human-thought reference (pragmatics), people assume that there must be two sandwiches, whereas the Proper noun Mary codes for an individual.

Another possible problem with ML-P is the matter of frequency (recall, that all neuro-networks rely heavily on one determining learning-factor—that of 'frequency'). For example, in work carried out by Harald Clahsen, the German plural {s} was found to function as the default setting for plural. So, for example, when a novel word was introduced to a German speaker, the default plural of the Noun [N] was add {s} : [[N]s]=Pl. However, the problem with this is that German plural {s} is not the most frequent in the data. In fact, when compared to alternative plural inflections {en}, {er}, the {s} only occupies less than 10% of productive plurals (Clahsen 1995, Marcus et al. 1995). In this case, if frequency is the generator behind (weighted) learning, than there would be no way to handle such anomalies as default rules untethered to frequency. The only way to guarantee that ML-Ps can generalize from limited data (in the ways that humans do)—given that the German plural data show highest frequency plural for {en}, {er}—is to incorporate a **Dual Mechanism Model** (DMM), a system for frequency (irregulars), and a system for symbols (rules, defaults). The question becomes: How can a non-frequency-based weighted distribution be incorporated in a frequency-based ML-P system?

Perhaps the biggest problem with ML-Ps is that they are simple recurrent networks, that is, they merely can **blend** (combine) input data.

Blending

The problem with blending is that in the simple combining of [A], [B], both individual values attributed to each item is lost as soon at the two items are calculated and merely averaged together—i.e., *blending* doesn't preserve the individuated information (information is lost over the spreading/blending of the two). Take for example, the representation of 'A' as [1010], 'B' as [1100], 'C' as [0011], and 'D' as [0101]: if we blend, say 'A' with 'D', we get [1111], but so too would be the result of 'B' & 'C'. Hence, there is no way to distinguish the individuated items. In this case, we have lost the unique code for the specific items once blended. In order to save the information over, say, several computational time-steps, we would need some form of recursive measures, such as [1010[0101]] , set = {'A', 'D'} with order unambiguous set. Here, info is preserved. Conversely, such loss of info as a result of blending matters when 'words' are assigned such binary codes (as with standard binary code ASCII)—where e.g., *A dog* could be coded as 'A' and *A cat* as 'D', hence losing the proposition 'cats and dogs'.

A second problem with blending (also particulate schemes) is that no word order is gained from out of the mere combination. As is seen below within the treelet structure, the phrase 'box inside pot' has to somehow insure that it is the 'box' that is inside the 'pot' (and not 'the pot that is inside the box').

A third problem with 'blending' here is known as *superpositional catastrophe*. Recall, that this blending problem is what we find of non-recursive sister relations found in our main discussions of syntax, e.g., of the [house, boat] vs. the [boat, house] variety. Recall, that as long as the two items merely blend as sisters []+[], there is no way to capture hierarchical structure—viz., is it a kind of *boat* or a kind of *house*? The same would be lost regarding [dog and cat]: is it a kind of dog or cat? It is only through recursive measures [[]] that the individuated information of the item is preserved: [house [~~house~~, boat]] is a kind of 'boat', with unambiguous word order and meaning. Another problem regarding 'blending' is if you are dealing with the combining of **semantic features** e.g., {cat: [+ furry], [+animal], [+ purrs]}. If one feature overlaps, superposition catastrophe of semantics could surface. Such a problem is cited in the literature regarding {penguin} whereby one of its semantic feature [-fly] bleeds into the semantics coding for 'fish'. (See Marcus 2001, p. 94, where he refers to this as *catastrophic interference*).

Blending > Particulate > Recursive schemes.

a. **Blending**: [A] + [B] => [C]. Info is lost.

b. **Particulate:** [A] + [B] => [A, B] Info is lost.

(semantic network connectionism)

c. **Recursive Network:** [A] + [B] + [C] => [A [_ B, C]] Info is preserved.

[B [A, _ C]]

[C [A, B, _]]

What we want out of a human-like cognitive process is to have a representational system that can operate over variables across two or more operating systems, run in parallel whereby such an overlap of **Dual** systems, along with their representations, creates a representational hierarchy. Only recursive systems can deliver these prerequisites: Only Recursion can allow its operations to spread across variables in this way. In recursive systems, items are not simply **nodes** based on frequency, but rather are such primitive (nodes) become **codes** whose computations themselves act as symbolic manipulations which can change their status as determined by their very arrangements (order). This is what we see below regarding the 'boat-house' example. If the scheme was, by innate definition, 'flat', then there could be no distinction between the two-word structure [boat-house] since in either 'blending' or 'particulate' schemes, no necessary word order would arise.

These three types of functions (*blending, particulate, recursive*) can be defined in terms of **Low vs High-level of processing**:

Low-level:

(i) **Blend** (combine): is seen in SRN, ML-P networks.

(ii) **Particulate**: is seen in semantic networks.

High-level:

(iii) **Recursive** (Treelet structure)

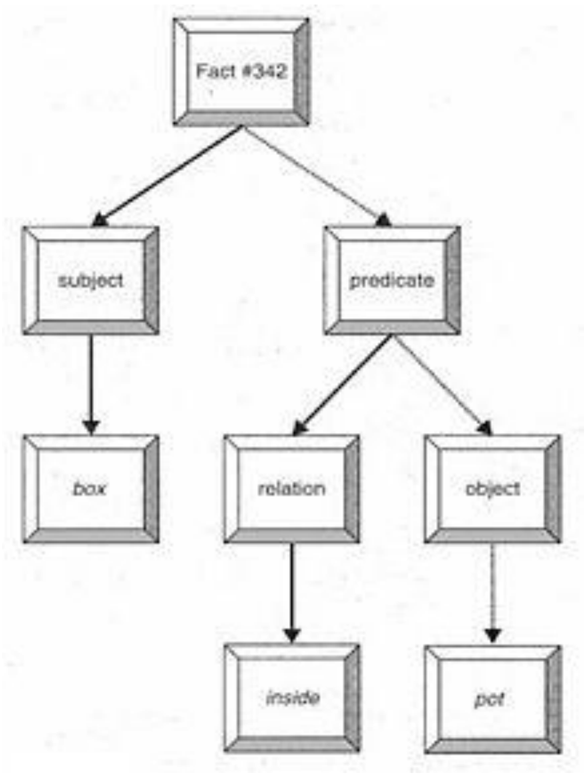
(Keep in mind that these low-to-high level distinctions on processing map onto neurological processes of the human brain/mind—what I have come to refer to as a **Dual Mechanism Model** of the brain/mind).

<u>**Low level**</u>

At these lower levels (*Blend, Particulate*), the analogy which works best here is that of 'Cell transportation' whereby neuro strength is activated and reinforced by neuro bundling found at local, adjacent levels. The region of the brain best suited for such functions is the **Temporal Lobe** (*insular cortex*) areas where 'frequency of blend/combine' works at the highest efficiency. (Note: In terms of language development, this is where we would find *lexical development* (a word-learning processes associated with Wernicke's area located in the temporal lobe). At such low-level processing, frequency has a very strong role to play—the Hebbian expression holds here: 'What fires together wires together'.

Low-level, in fact, can be assumed under connectionism/ML-Ps since local-firing of neurons trigger input-outputs as related to established **register-sets**.

Fig.2 Register-sets (Copied and reinserted here as found in Fig. 4 of Appendix #). (Marcus 2001, p. 111)

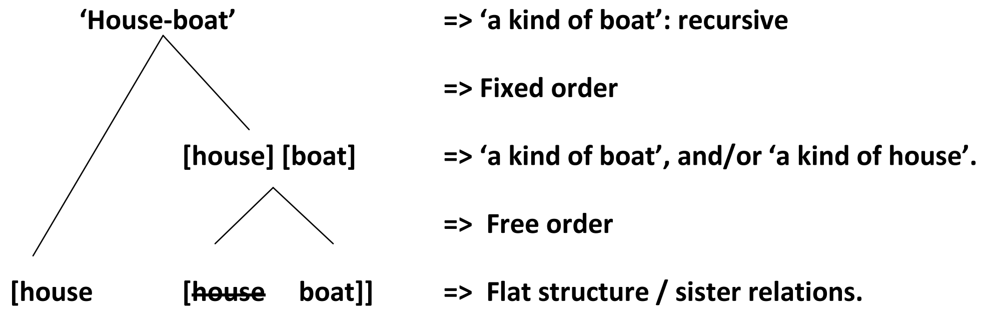


Within a treelet structure (above), consider the 'register-sets' [boxes marked *subject, predicate, relation*, etc) to be analogous to neurons (cells) as found in the brain. The Hebbian express 'What fires together wires together' accurately describes this local level of function as 'frequency-effects' plays a critical role at this level (just as with word learning)—any associative model that relies on strength of reinforcement must establish such local wiring & firing domains attributed to one-one associations (**behaviorism**). In this sense, keen connectionism is a new form of behaviorism. Notice how treelet structures allow nodes to be coded in ways which establish **categories**, such that (categorical) relations hold between e.g., **subject** and **predicate**. This is the main difference between a serial search which would be conducted exclusively via 'nodes only' versus searches which could be done in parallel over nodes and categories. Treelets resolve the problem that other low-level processing had (SRNs, ML-Ps) since treelets, with these new categorical relations between nodes can generalize to new concepts. Hence, **category, creativity,** and the ability to function with **variables** inters into the processing domain. Such ingredients would be critical for any true human-like language to be mimicked successfully in any AI machine. So, what AI needs is a program (a domain) that uses (abstract) *codes/symbols* and not just (associative-based) *nodes*. Recall, that in human language, much of our language intuition goes against the evidence supported by our data—we move beyond data. This fact can only be handled by an Operation System which can process both/dual nodes and symbols simultaneously (leading to hypothesis calling for a Dual Mechanism Model (Marcus 2001, Pinker 1999, Clahsen 1999, Marcus et al. 1995).

Note how *symbolic* codes are reliant on blends, or semantic features (Particulate), but are rather fully **syntactic** in nature—hence, the form of the syntactic tree. For instance, with regards to a simple blend of the lexical items (*box, inside, pot*)—and with there being no syntactic subject-predicate relation— there would be no way to determine the relational predicative arrangements of items: for example, is 'the box inside the pot', or is 'the pot inside the box'? Such <Subject <Predicate>> is crucial in order to determine meaning and simple *blend/merge sequences* do not provide such information. We'll look at this below.

As another example, consider our 'house-boat' example below in terms of **blend** (particulate) vs. **recursive** formation:

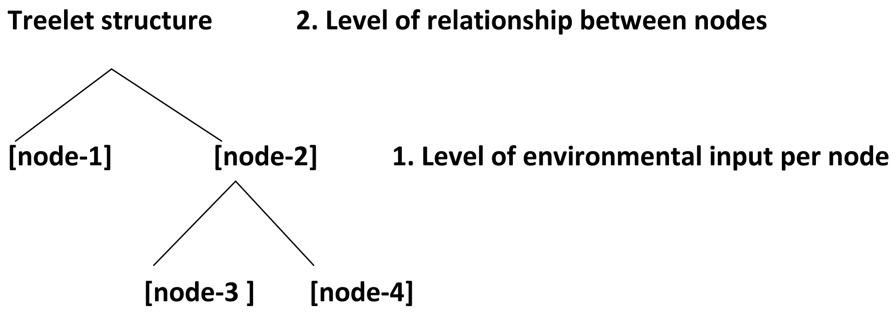
[2] **Boat-house example**


'House-boat'
=> 'a kind of boat': recursive
=> Fixed order
[house] [boat]
=> 'a kind of boat', and/or 'a kind of house'.
=> Free order
[house
[~~house~~ boat]]
=> Flat structure / sister relations.

The **particulate** (*semantic*) units would contain the encyclopedic information having to do with the entries of <boat> & <house>. A **Blend** of the two items could yield either <house, boat> or <boat, house> without word order. It is only at a recursive level would we be able to interpret 'house-boat' (a kind of boat) from 'boat-house' (a kind of house), etc.

The above 'Boat house' example can be projected onto a treelet scaffolding:

[3]


Treelet structure
2. Level of relationship between nodes
[node-1]
[node-2]
1. Level of environmental input per node
[node-3]
[node-4]

[Nodes 1-4 are 'register sets')

Each node plays a neuro analogy to a cell (low-level, frequency driven). This processes is analogous to what is referred to as **cell transport** and can easily be replicated by connectionist models such as SRN and ML-Ps. What the high-level processing allows is for relationships to form between such nodes, creating hierarchical arrangements. Let's assume Nodes 1-4 (= **register-sets**) to be analogous to neurons (cells) such that their formation (wiring) is determined by the strength of frequency (firing). The relation between input (environment) and the coding of the register-set has been referred to as a kind of **cell transport**. This cell transport takes place a low-level processing akin to associative mechanisms which drive any kind of behaviorist model. Of course, in order to capture true human thought, other higher-order processes must take place. These higher-order processes inter alia involve relationships between nodes in building a hierarchy (e.g., mother-daughter relations which break otherwise flat sister-sister relations). For example the relationship between nodes 3, 4 are sister relations such that two words may be constructed to form a phrase {boat-house//house-boat} (showing no order). Once movement takes place, say, between node-3 and node-1 such rising has moved a sister item from out a flat structure into a recursive hierarchical structure (showing necessary syntax) e.g., [boat [~~boat~~-house]].

So with **blending** processing, the specific information of the item [A, B] is lost (as the info is averaged and/or combined): e.g., Blend [A] + [B] = [C]. Again, this is exactly what we find with simple recurrent networks. Regarding **particulate/semantic** networks, what we come up with is [A]+[B] = [A, B]. Recall, what we want out of a human-like operating system is a processing network which breaks symmetry in order to represent unambiguous orders. As presented herein, what we are claiming is that only a true recursive processing procedure can achieve this: e.g., [A] + [B] = [A [B, C]]. Returning to the 'box-pot' (#) example above, what we need is a formation that delivers a recursive [box [inside pot]] and not a serial flat structure of [box inside pot] (the former being recursive, the latter blend/particulate). Marcus has a very clever way to describe the drawbacks found in **particulate/blending processing vs recursion**—he goes on to suggest that, e.g., the two items (colors) [*black*] & [*white*] are not somehow halfway between black and white (like some shade of gray) (Marcus 2001, p. 87). Rather, while it may be true that particulate features must be self-preserving (information must not be lost [A] + [B] = [A, B], etc.) there is still no way to represent unambiguous distinct relations between elements or semantic features.

Problem: 'A box inside a pot'.

It would not be enough to simply activate the individual elements [+box], [+ put], [+inside] since the same coding would trigger 'a pot inside a box'. Hence, semantic features via particulate function cannot generate order. If *box* = A [1010], *put* = B [1100] and *inside* = C [0011], the coding of [A, B, C] even if info is preserved doesn't render a hierarchical order. Again, what is needed is a recursive structure which yields [A [B, C]] or [box [inside pot]] (as seen in English SVO word order).

<u>High-level</u>

The relation between register-sets doesn't seem reliant on local/frequency-based computations, but rather a relation is formed via rule-based/symbolic manipulation. It is in fact the codes with symbol manipulation that leads to recursive treelet-structures—a treelet structure allows *search* to be performed in *parallel* (**parallel search**). The human **brain** (at low level processing) is enriched with the capacity to bootstrap itself up to higher-level processing thus allowing for the creation of the uniquely specific human **mind.** In the same way as the brain bootstraps neurons and creates symbol manipulation, so too do treelet-structures allow for nodes to be abstracted away in order to make room for abstract and symbolic *codes*—codes which can establish **categories**.

Note. Symbolic codes are not reliant on:

(i) **Blends** [A] + [B]= [C] or,

(ii) **Particulates** (semantic features) [A] + [B] = [AB],

…but rather are syntactic in nature, yielding

(iii)**Recursive** structures [A [B, C]].

This is the main difference between the search of 'nodes' (a serial, low-level manipulation) versus the parallel search over codes and categories. Treelet-structures resolve the problem that other low-level processing had regarding SRNs and ML-Ps—treelets can generalize to new concepts. SRNs and ML-Ps have a very difficult time generalizing novel items or extending into realms of new information. The most celebrating problems cited in the AI literature we find when we attempt to use common set of nodes (in either a semantic capacity or in a combined capacity) is that there is very often a **catastrophic interferences** which follows.

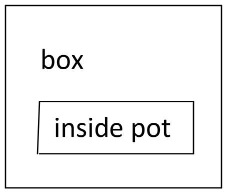
Catastrophic Interferences. One such example (cf. the 'Rumelhart-Todd' network) showed how if a network was taught that <birds have wings and can fly>, and then taught that <a penguin could not fly>, it falsely assumed/learned that <the penguin must be a fish> (and as a learned subsequent of cascading features, assumed it also to have gills, scales, etc.). This is just one example of how semantics and blending may bleed into other unwanted interpretations. Of course, it is always possible to code such additional info into the network, but I think you can see how overwhelmed a model can get by what humans take for granted in terms of irregular exceptions ('to the rule'), intuition about how the world words, common-sense, etc. The multitude dynamics that enter into any meager attempt to encode all such features into a SRN or ML-P (without recursive parallel search) very quickly outpaces the capacity for such rudimentary models.

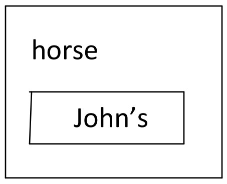
The Two-Horse Problem

Consider the '*Two-horses*' problem (Norman 1986 cited in Marcus 2001, p. 123), or another example referred to above as the '*Sandwich problem*'. The problem is to be able to handle different instances of the same concept, at the same time. For instance, let's code <sandwich> as [01101], then if Helene eats a *sandwich* [01101] and John eats a *sandwich* [01101], the system has to somehow go beyond the simple one-one coding in order to treat these *sandwiches* as different. Specifically, the problem here is how can we get an operating system (OS) to handle both (i) general properties of *categories* ('sandwich') as well as (ii) properties of *individuals* (Helene's sandwich). If coding denotes only general properties of objects/items, then we need some overlapping code to distinguish between *kind* vs. *token*.

The 'two-horses' problem is the same: in order to capture the ASCII code for HORSE as e.g., [100011], how do we distinguish e.g., '*John's horse*' from '*Mary's horse*'—where the one-one code for HORSE has no way to handle the individual distinction (there are two individual horses involved). Formal semanticists using predicate calculus often denote representations of 'individuals' with lower case (<horse> = John's horse) and representations of KIND as upper case <HORSE> (general property of HORSE). Returning to 'the box inside the pot' example above, there is something very similar to how an operating system (OS) might deal with this distinction—namely, we'll have to devise a single processing domain to handle two fundamentally different procedures at the same time: one processing which handles the items and one which handles individuals. Embedded structures seem to be the best bet to handle such a duality or processing. The expression 'The box inside the pot' has to distinguish somehow that 'the box is inside the pot' and that 'the pot is NOT inside the box'. If simple one-one nodes only codes for items, then there is no way to tease out the distinction. But keep in mind if we can somehow show a coding system which delivers two distinct processes, then we might be in luck. Consider just how a recursive code (using diacritics) might be used:

[4] a. [box [inside pot]] b. John's horse [horse [of John]]





As represented by the 'two-horse' problem (as well as the 'sandwich' problem above, cited in Marcs 2001), both SRNs as well as ML-Ps—in fact much of what is behind current platforms which support AI today—have a very difficult time in handling both *individual* tracking (over time) as compared to *kind* representations (static). The ability to represent general, associative properties of an object/entity/event is one thing, but to represent individual over kind seems to place an overwhelming burden on the SO. Cognitive scientists and linguists use diacritic indexes (which are by their very nature recursive) in order to handle such dynamic information. Only via an implementation of a DMM where nodes and features (blending & particulate structures) can merge with recursive co-indexing would we have an OS worthy of capturing this full dynamic.

So, what might a DMM look like in capturing this twin processing? Let's consider again the difference between the 'kind' [Horse] vs. the 'individual' [John's horse]. The OS (operating system) of a dymanic DMM would look as follows (using indexes):

[5] Item Kind

(i) John's horse = [horse$_j$] => [HORSE$_j$]

(ii) Mary's horse = [horse$_m$] => [HORSE$_m$]

(j = John, m= Mary).

Formal semanticists using predicate calculus often denote representation of *individual* with lower case, and representation of *kind* with upper case. Diacritic usage as a recursive means would looks as follows:

(iii) $\boxed{\text{HORSE } \boxed{\text{John}}}$ => 'John's horse'

(iv) $\boxed{\text{HORSE } \boxed{\text{MARY}}}$ => 'Mary's horse'

Clearly, this is very similar to what we saw above showing how the 'flat recurrent' structures, typical of SRNs and/or ML-Ps, have a difficult time handling such embedded dual processing: for instance, just as there is no way to capture 'boat-house' vs. 'house-boat' as discussed above by simply coding for <boat> and <house>, so too would there be no way to handle 'box inside pot' from 'pot inside box', or the kind HORSE vs the individual 'John's horse', etc. A flat structure would ambiguously code both instances as [box/pot inside box/pot] as well as not being able to capture the dynamics that 'John's horse' and 'Mary's horse' (or 'Mary's sandwich' for that matter), is not one in the same item.

The above discussion provides some details into why mere *particulate* argument structure (semantics) and/or simple statistical averages of *blends* cannot suffice to produce human-like learning model (as fully expressed within human language).

Intermediate Summary on the Dual Mechanism Model (DMM).

The DMM proposal calls for a single domain which encodes for a dual processing whereby a single representational format operates over variables across two or more operating systems run in parallel within a single domain. Such an overlap of operating systems (within one domain) along with their (co-indexed) representations creates a unique ability to recognize and process representational hierarchy.

The DMM: Human-brain processing // AI Platform, processing

(i) Low-level (local): Cell transport[15] Temporal Lobe, Insular Cortex, [+Frequency]
// SRN, ML-P, 1. Blending: [A] + [B] = [C].
2. Particulate: [A], [B] = [A, B].

(ii) High-level (distant): Broca's region, displacement
[-Frequency] (symbolic: syntax, music)
// Rule-based, Hierarchical (diacritic, index)
3. Recursive [A [B, C]], or [A [~~A~~, B]].

[15] (Local) 'cells which fire together, wire together' (an adjacency condition) (See Hebb).

With a DMM, both systems can coexist in a single domain (i.e., the human brain/mind). This duality is what we find as the unique operating system behind human language.

Intermediate Conclusion to remarks on 'Two-Horses' Problem as it has to do with a Single Mechanism Model (SMM) and/or Multi-layer Perceptions (ML-Ps) versus a Dual Mechanism Mode (DMM).

The above problems point to processing of representation tasks which have to be performed in a multi-fold manner. If codings (as sequenced in a *single mechanism model* (SMM) have no way to code for token over kind, or item (individual) over kind, then the operating system essentially becomes very impoverished in the kind of knowledge humans entertain. Of course, there may be ways to override this problem by coding for item and subsequent coding for type (two interdependent codes), but as it turns out, such overlapping coding places huge amounts of burden on processing: a problem that is not easily overcome by ML-Ps. But such processing is easily secured by children, even effortlessly done, by the age of three-years. Young children innately know how to identify individuals over kinds. (See Sorrentino 1998 for experiments such as the so-called 'Zavy' experiment presented in Marcus, 2001, p. 125). Experiments such as Zavy—where stuffed animals are shown to children first as 'kinds' and then as 'individuals', and then where they are tested on tracking one over the other—demonstrate that what poses a significant problem for SMM, ML-Ps, is easily performed by children as young as three years of age. (Such knowledge is often termed **Object Permanence**).

We'll take a look on how the *'kind versus individual'* distinction can easily be coded in recursive structures via diacritics. The claim for recursive/syntax (=DMM) over recurrent/semantic (=SMM) will be extended in the following sections where it will be argued that the distinction (along with its tracking, as demonstrated by the Zavy experiment) requires **two types of search capacities:** one search for the item itself (associative/recurrent), and another parallel search for how to determine how the items are manipulated as a category (intervention/recursive). So, what we want out of a well-functioning ML-P (if that is what is called for our AI operating system) is the ability to represent relations between local nodes and distant variables, and to be able to compute across the two representations.

ML-P intermediate summary

In sum, ML-Ps have several difficulties, as listed below:

The model has a hard time—

(i) presenting novel items, since the model can only blend/combine pre-existing representations previously encoded.
(ii) computing across two different representations, such as nodes plus variables.
(iii) using semantic features in a syntactic manner.

The limits of a SMM/ML-P vastly reduce how representations can be computed over time and across space. ML-Ps are prewired and can only be altered via so-called strength condition (*à la* behaviorism) of frequency. This is not 'learning' (learning is not of a sole, brute-force memorization). What we want of a true AI operating system that comes closest in mimicking human learning/thought is a system of input/units that are not prewired as determined by strength alone, but rather that their unit-values are computed freely over competing pathways—pathways, possibly cascading in parallel over a dynamic and hierarchical domain. This uncanny ability to track similar and/or competing inputs across two parallel operating systems is what is called upon for learning. Only a DMM run in parallel could perform this feat. A SMM would not only be too slow, but would also lack the architecture required for such true learning.

Poverty of stimulus.

In addition to arguments that have been laid out in our 4-sentences section, consider that claims made by the linguists Peter Gordon (1985) that young children know not the keep plurals embedded within compounds—what do you call a person who eats rats? Children respond 'rat-eater' (the delete the {s}) and they never respond *rats-eater. Gordon suggests that children innately know that inflectional morphology {s} can't be kept embedded within a compound, even though they have never been explicitly shown that such data is in violation of some English grammar. The mere fact that they never hear it (because it is, in fact, ungrammatical) doesn't explain why children never entertain the prospect: children say loads of erroneous things that they have never heard before. Hence, even though children have no empirical evidence (negative stimulus) that such constructs are wrong, they still shy away from compound-embedded plurals. This is what is referred to as the 'poverty of stimulus'—namely, when children's inferences go beyond the data they receive. Gordon suggest in this sense that there must be some innate built-in machinery constraining child learning of language. So, to put this into our discussion of ML-Ps, if input-to-outputs models are the square product of environmental learning, one question that will come up is: How does such learning deliver a result such as found with the poverty of stimulus case? Perhaps symbol manipulation of rules will be required in some fashion after-all. But, if so, perhaps we need to rethink the brain as a mere neuro/digital net. In fact, the analogy of the brain as a digital computer had been under attack for some time—as Gary Marcus (2001) claims (we still know very little about how the brain works at the higher level. Perhaps at the lower level the brain-to-computer analogy holds (where local firing of neurons takes place, etc.) but with higher functions, when we talk of a 'mind over brain' of how a brain bootstraps a mind) there may need to be a fundamentally different processing with an entirely different underwriting.

Michael Tomasello vs. Ken Wexler (See 'Lenneberg's Dream', Wexler 2003)

Perhaps one of the more passionate debates which have arisen regarding how this distinction of 'recurrent/association' vs. 'recursive/rule-based symbolism' can be played-out in developmental child syntax is that (naïve) notion that the errors found in early child syntax is simply the product of a lack of memory, or a 'bottle-neck' of cognitive-processing sorts. Let's just take a last moment to flesh this hypothesis out.

There is a case to be made that if, for example, a child says 'He open it (where agreement {s} is missing (e.g., He opens it)), one could claim that such an utterance is in fact available in the input via the utterance '[Should [he open it]]', etc. Likewise, 'She eat grapes' could come from available positive evidence of 'Does she eat grapes? etc. where if the initial finite verb is dropped (forgotten) of the matrix main clause, the remaining structure would be consistent of what young children say. This theory relies on a partial **mimicking theory** of an X=x type, where the initial mimic has been lost. Tomasello argues for such a theory of child language which is completely based on associative mimicking (and/or the lack thereof of certain parts of the base mimic). Well and good! But problems quickly emerge from such a naïve theory. For one thing, if a partial mimicking theory is correct, young children (two years of age) would also say things like 'Did you want'? as derived from the partial mimic of [what [did you want]]? However, they don't say such utterances. Rather, a more typical stage-1 expression (years and younger) might be

'What _ you want?' where the second word (auxiliary verb) is missing from the string. Children at stage-1 don't produce such Aux-first words with Wh-words missing. They rather deleted Aux and maintain the fronted Wh-word. The fact that positional errors can no longer be part of the theory suggests that the child is operating under a structure-dependent hypothesis of language, and not a positional-dependent (or structure independent) hypothesis.

Other even more interesting examples include why a child might say 'Him do it'. Tomasello might argue that they hear [I saw [him do it]], and so on. But still, even if children process in this way, we would have to account for why the first part of the sentence is ignored given that in frequency, first-parts are most marked. This would seem to strike at the heart of Tomasellos theory. Note that such a theory of mimicking is completely reliant on Frequency. (But the highest frequency word in English is the word 'The'. But the word 'the' is notoriously the last word acquired by a young child (e.g, 'The car is broken' => 'Car broken')). 'Him do it' as shown above are so-called **Small Clauses.** While it may be true that such small clauses are abundant in the data, such a theory would also predict the following: [Mary knows [I like candy]], and so the prediction would be that the child might drop the initial 'Mary knows'-clause and say [I like candy] with nominative case 'I' (the adult utterance and the small clause are homogeneous)[16]. However, the child does nothing of the sort. The child strictly says 'Me like candy', etc. Such utterances pose a problem for any naïve theory based on mimicking or forgetfulness of mimic. In this case, children could be said to go beyond their data, they go beyond frequency of input. Their stage-1 utterances are rather the result of a lack of structure, and not the lack of positional memory. Stage-1 child grammars are systematic, rule-based (or the lack thereof), are constituency-sensitive and based on syntactic properties (provided by UG).

In sum, returning to our AI-discussion, any ML-P system which hopes to learn human language would have to be sensitive to such a processing of learning. The system must be symbolic and rule-based and must be able to go beyond mere +Frequency/mimicking of structure.

ML-Ps, as prewired, can alter settings via strength—but this is not learning. Again, what we want of a true-learning algorithm is that units/inputs are NOT prewired as determined by strength alone, but that their values can be accessed over competing pathways, cascading in parallel over a dynamic hierarchy. Language

[16] There is a clear relationship (syntactic) between [+Nom] nominative case and [+Fin] finite verbs.

learning/acquisition is the ability to track similar and/or competing inputs across two or more operating systems. Such 'checking-off' of a retrieval of input between the two or more pathways (operating systems (OS)) must work in parallel since a serial OS between the two would be too slow. Recall, this is the minimum of what we want…to be able to handle human-like processing between __ks as found in [1] above.

Brain-mind bootstrapping

The idea that a *brain can bootstrap a mind* may have its origins in theoretical linguistics, particularly looking at child language development of syntax. One very promising model which has implications to AI and the cognitive sciences is the notion that these low-level brain processes which map local configurations on a frequency-based threshold may be only one part of the brain's processing of language (a more primitive part which is responsible for lexical look-up, retrieval mechanisms dealing with objects, items, etc.) while a second more abstract mode of processing takes such general properties of items and spreads them over categories whereby recursive operations may allow diacritics and indexes to work as variables. Steven Pinker's 1999 book entitled *Words & Rules* captures this dual mechanism model distinctly. By extension, mush of what I am on about here in this section speaks to the notion that within a single domain of processing, a dual operating system may be in use which allows for this *individual* versus *kind* distinction. Minds can nicely grapple with categories and recursive structures which can handle the tracking of *individuals* (where indexes, diacritic variables are in operation) while the brute-force calculating brain serves one-one properties of *kinds.* (Recall in 'Against ML-P' Claim #1 above the use of diacritics to help resolve the so-called 'two-horses (sandwich) problem').

The 'Brain-to-Computer' Analogy: 'Low vs High' levels as a linguistic function.

Low-levels. At the lowest levels of the hierarchical brain spectrum are found neuro connectionist systems which rely on approximation to 'local-dependency'– these are so-called finite-state grammars of the SRN type discussed below (G-1). (Included in such OS's would be so-called 'multilayer perceptron'). At these low levels, statistical regularities work in local configurations so that they bundle together. In linguistics terms, these low-level grammars create so-called **Merge** properties: {X, Y} = {XP {X, Y}} (where X is Head and Y is complement) as when two word/items merge in becoming a phrase. Also linguistic compounding can be said to be a result of such merge. For a slightly more sophisticated example, consider ***linguistics Root-compounds:*** e.g*., 'chain-smoker' where only the two items merge {chain} {smoker}* and where no movement is required. (Notice a move-based product becomes ungrammatical, one can't derive the compound as **A smoker of chains*). It's only at a higher-level of processing where two merged items become more than the sum total of their parts. Consider what happens to the seemingly similar linguistic compound '*cigarette-smoker'* where one can say *'A smoker of cigarettes'.* This is such an example of a higher-level processing (albeit linguistic) which shows how the two merged items retain their specific past memories over two or more computational time-steps (CTS).

Consider the two distinct processes below (starting with low-level/Merge to high-level/Move:

[6] (1) Local: [X, Y] => {XP}: [Chain] + [smoker] = [chain-smoker]

(2) Distant [X, [X, Y]] => {XP}$_t$ (where t is trace of prior memory of structure)

[Cigarette] + [smoker] = [cigarette-[smoker ~~of~~ ~~cigarettes~~]]

In [6,1], a local neuro function, say within one CTS (CTS-1), shows the memory limit in how two adjacent inputs (and *adjacency* does seem to be a prerequisite, common to how neuro firings work in adjacent bundling) combine to achieve an averaged-weighed product. However, if we were to apply the same 'local neuro-firing' to the syntactic compound found in [6,2] above, the interpretive distinctions between root vs synthetic compounds would be lost. Consider (1-2) restated in (3-4) below showing CTS numerations:

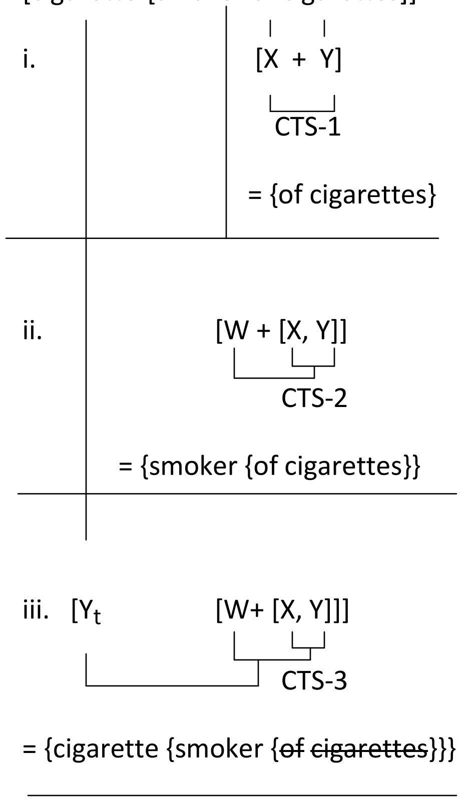
Root Compounds (RC) vs. Syntactic Compounds (SC) as an analogy to computational time-steps (CTS)

(3) Local: [X, Y] => {XP}: [Chain] + [smoker] = [chain-smoker]

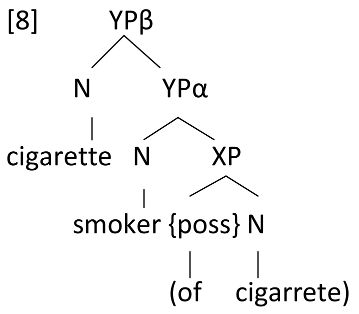
$\underbrace{[X] \quad + \quad [Y]}_{\text{CTS-1}}$ => (memory between one CTS)

CTS-1 [X= chain], [Y= smoker] => local firing of two units as found in a multilayer recurrent system.

[7] [Cigarette] + [smoker] = [cigarette-[smoker ~~of cigarettes~~]]



Showing a linguistics syntactic tree, the CST-1 found in [7i] would be represented in [8] below:

[8]


The above notion of 'locality vs. distance' as bound by computational time-steps (CTS) has antecedence to levels of computational processing found in the brain—with 'Low-level' computations being assigned to exact one-to-one neuro firing (say, having to do with the triggering of a specific node within a connectionists model), while 'High-Level' processing establishes distant relationships, say, between nodes. The expression that the brain bootstraps itself in creating a mind can be played out in such a dualist scenario of local vs. distant neuron triggering (with the brain being pegged to locality conditions (the 'associative brain'/Temporal lobe region) and the mind to non-local freedom (the symbolic 'rule-based' brain/Broca's region). (See 'treelet' structure below for further discussion).

High-levels. At the higher levels, statistical regularities seem not to be dependent on local constraints, as shown in [7] above. Hence, the syntactic/semantic interpretational distinctions found between the above *'root vs syntactic' compounds* could be drawn as analogous to 'local vs distant' neuro/unit firing (where unit would be labeled here as **word** (*cigarette*) and **grammatical feature** *(possessive)*. This same dual distinction is also very nicely seen within linguistic rules (so-called regular-rules/distant versus irregular-rules/local).

For instance, notice how sound patterns of **irregulars** work on a low-level where frequency-effects of 'bundling of feature' can impact either neuro or linguistic processing: e.g. [_ [ing]] > [_ [ang]] > [_ [ung]] which generates *sing>sang>sung* may over-generate (over-trigger) based on a sound-frequency effect to *bring>brang>brung*, (but not **bling> blang> blung*). Notice how such novel words (so-called made-up 'nonce' words used for experiments) generate the distant rule—e.g., *Today I bling it, yesterday I blinged it*.

Such distant true rules never become dependent on local frequency-effects (or local neuro-firings), with true rules projecting over a variable/category such as instruction: <do x to category Noun>, <do y to category Verb>: add {s} to N when plural *one wug* > *two wugs* (see Berko), or, as just demonstrated above, but with an altered final consonant from /g/ to /t/ e.g., *today I blink*, and *yesterday I blinked*. True rule-formations of [N+s], [V+ed] work independently of frequency-bundling and their productivity allows new and novel items to be freely expressed as categorical variables across data spreads. Such a multilayer-perceptron model would have difficulty showing such a dual-level processing since perceptron models (SRN's, CRN's) would only code for a single mechanism model (SMM)—viz., the same mechanisms would have to be involved between RC's and SC's, thus losing the distinction.

In other words, regarding CTS's, 'cigarette-smoker' would be forced into a local neuro-firing between two local units as expressed in [9] below:

[9] [cigarette] [smoker]

$\underbrace{[X] \quad + \quad [Y]}_{\text{CTS-1}}$ => (memory only between one CTS)

For example, multilayer-perceptrons can only average (approximate) a broad range of functions, based on local distributions of a single mechanism model (SMM). It has been found (Marcus, Brinkmann, Clahsen, Wiese & Pinker, 1995; (Hadley, 2000) that these SMM's cannot capture such a class of operations which spread of two or more CTS's and/or that follow from a recursive, embedded coding: (noticing how the progressive structure in [7i,ii, iii] require embedded clusters (i.e., **recursive nesting**)). (See Pinker 1984 for initial reports of grammar modeling, and Pinker 1999 for review of a dual mechanism model, Galasso 2016 for 'First Merge, then Move' model in early child syntax).

The conclusions reached here are that multilayer-perceptrons cannot generalize from a limited data the same way as humans do.

Thought-experiment on recurrent vs. recursive implementation in AI

<Insert>

[0] **Code snippets** (taken from Fitch 2010. p. 77).

```
(1) define function [A^nB^n] (n) :
    if n is 1, then return "AB"
    else  return  ("A" + [A^nB^n] (n-1) + "B" ; //recursive call
```

⇨ This grammar generates a recursive [A [AB] B], structure-preserving embeddedness.

```
(2) define function [A^nB^n] (n):
    interger counter i
    A_ section = "A";
    B_ section = "B";
    If n >1 then {
            for (i=2) to (i=n)
            A_section = A_section + "A";
            B-section = B_section + "B";
    end
    }
    Return A_section + B_section
```

⇨ This grammar generates a flat-recurrent/non-recursive [AB], a simple recurrent network.

AI-Statements *(code snippet functions):*

[1] True AI-Recursive (AI-R) (strong generating) grammars project at least a dual tracking of constituencies:

a). AI-R codes AB-strings in the middle of other AB-strings (center-embedded): [A [AB]B]. Such a recursive rule has the unique property of self-embedding.

b). AI-R is structure-preserving and can span memory across two or more phrases. (A circuit which remembers past structures only for a single computational time-step is not capable of representing complex structures. Memory must be able to span across multiple time-steps, while, at the same time, being able to look into multiple center-embedded structures).

[2] Center-embedded structure can affect left/right peripheral strings and vice versa.

e.g, $$\begin{tikzcd}{[}A & {[}AB\ {[}\cancel{A}B{]}{]}{]} \arrow[l, bend left]\end{tikzcd}$$:

Recursive = structure-preserving loops, and not just 1-way feedback loops.

[3] ‘The ability to retrace a stack of function calls [*from peripheral to embedded stings*] must be specifically designed into the programming language and hardware’ (Fitch 2010, p. 77, *italics* belong to JG). Any recursive grammar which cannot maintain statements 1 & 2 are not full, true AI-R grammars and may be more **recurrent** in nature. This ability to retrace (track) also includes not only strings but variables associated with strings—some additional memory mechanism must be designed into the system to keep track of differences between variables$^{(nm)}$ –e.g., [A^nB^m]-grammars where the variable n= “repeat number of string” and the variable m= “alternate string within embedded structure”. The fact that a single function must be able to count and compare requires feedback loops which may arise from one structure and fall into another, whereby an embedded unit/circuit^1 may affect a string unit/circuit^2 found in the left/right periphery.

[4] The code itself must contain representation of structure—codes, and tracking of codes must be structure-dependent (like natural language where syntactic tree diagrams contain representation of structure). A nice analogy to the tree would be that words/strings Nouns, Verbs make-up a beads-on-

a-string grammar (a *weak-generation* found e.g., in non-recursive flat grammars [AB]), while the phrase NP/(DP), VP/(TP) would make-up representation of structure (a *strong-generation*). The embedded nature between the two (weak vs strong) is both implicit and explicit in the design.

[5] So, while we read through this brief note on recursive AI-grammars, let's keep in mind that what we want from a well-advanced AI-operating system—a system which could come closest in dealing with what we know of human-behavioral processing—is a grammar-system which not only looks into its 'left/right-peripheral' adjacent AB-strings [AB] (so-called *weak* generation), but also has the ability to look into its 'center-embedded' [A [AB] B]-structure (so-called *strong* generation). In the latter case, the grammar calls itself. This is found in the defined function of the snippet code found in [0] above.

[6] **Introduction**

Much of current AI, as I understand it, either uses as an operating system (OS) (i) a simple recurrent network (SRN) (as much discussed in Jeff Elman's early work (cf), or (ii) a complex recurrent network (CRN)—the former which is more or less an approximate calculation of the combined net-value reached of two inputs, (with or without architectural hidden levels), and the latter which uses not only feedback loops at the 'on-time' computational time-step (CTS)[17]— (so-called time-1 'hidden structures'), but also 'backward-looking' (time-2) interactions across at least one previous CTS. In either case, both (non-recursive) SRN & CRN AI-platforms are of a 'brute-force' nature which delivers an on-time catastrophic decision based upon probabilistic on-time calculi. Let's consider how the two aforementioned operating systems might differ from a truly recursive operating system in regards to what we might hope to build of an AI machine worthy of simulating true human behaviors (e.g., autonomous vehicles, machine language learning and language translation, decision-making tasks based upon CRN-encyclopedic knowledge, etc.).

Let's first consider an [A^nB^n]-grammar which derives e.g., A^nB^n (4) as AAAA, BBBB (as discussed in Fitch 2010). This OS is what is referred to as a

[17] The time-step is the incremental change in time for which the governing equations are being solved.

recurrent grammar whereby the *stacking* of two items (which could go on indefinitely) works in a flat non-recursive manner. This is a non-recursive structure and would be similar to any SRN: the crucial note here is that the variables within AB $\{^{n,n}\}$ would not need to be tracked since they are identical. Now consider a more complex system (CRN) with memory of backward feedback-loop connections (between stem-nodes and their respective variables) which may compare inputs and outputs over the span of maximally one CTS. This could potentially give us an [A^nB^m]-grammar where variables:

(n= "*repeat" form a list of what you know*),

(m= "*alternate" from a list of what you know*)

yields A^nB^m(4) as AAAA, <u>AB</u> A <u>BA</u> BAB*.

*But the problem here is that from this point in the memory, a backwards look-up device would have to store in memory two simultaneous non-congruent inputs (and both inputs would be structure dependent)—namely,

[7] (i) that [<u>AB</u>…] is the product of the sequence, and

(ii) that [<u>BA</u>…] is not the product of the sequence (since 'BA' is the result of [B [A~~B~~]].

In other words, there would have to be some additional memory mechanism (of variables) embedded within memory of nodes, viz., [memory^2 [memory^1]] or, say, a [declarative [procedural]] interchange held across at least two or more phases.

It is my current understanding that what we are usually implementing in all AI/OSs across the broad these days is mostly based on either a upgraded SRN or a CRN OS, but that there are few if any currently implemented AI-OS's that can:

[8] (i) maintain a sufficiently dual-memory embeddedness (DME^2),

(ii) hold and compare memory inputs from prior CSTs across at least two or more phrases, and perhaps most crucially, and

(iii) whose code itself contains a representation of structure[18].

Why are these three properties found in [8] so important? Well, it may have something to do with how the unique human brain/minds works, how *intuition* can ultimately be gained, and how humans over our evolutionary-time span have bootstrapped a simple neuro-network *brain* into a 'much-more-than-the-some-of-its-parts' complexity of theory of *mind*.

Let's flesh out what (DME^2) might look like via the following thought-experiment (having to do with, say, an AI-encyclopedic OS): The task is: <u>Who is the first president of the United States?</u>

***A 'look-inside-two-boxes'* scenario:**

[9] AI-OS: so, here's the (very tentatively) imagined system:

(i) Let's say *the left/right periphery* [x, y] of [x [a,b,c] y] codes for overt *declarative processing* of the kind associated with questions and responses of a certain task. Only this overt/declarative operation has an interface with the outside world and is reinforced by the input received (self-learning). So, for example, when the (Q)uestion is asked, it processes the language of the question and maps the Q onto a relevant (R)esponse. The periphery [x, y] however is able to look into the covert center-embedded [a,b,c]-grammar as its *procedural-knowledge* source, with both peripheral and center-embedded structures allowing for 'feedback loops' for reinforcement of net-values, adjustments, etc.

(ii) Let's say the *center-embedded* [[a,b,c]] codes for covert *procedural processing* of the kind associated with the mapping of an internal look-up ("a list") which then allows retrieval of the item to surface to the declarative mode of processing (with the interface):

[declarative [procedural] declarative]

[18] Note that our hypothesized DME^2 may lead to the two types of human knowledge systems, *declarative* and *procedural* knowledge.

[10] (Q)uestion put to our AI-OS : Who is the first president of the United States?

So, there are two boxes:

box-1 = three (P)eople (P1,2,3) (box-1 = time-step-1 of input for box-2)[19]

box-2 = AI (with DME$^{\text{2-recursion}}$)

Box-1: P1 and P2 think they know the answer to Q, but P3 is unsure. As each of the three people in box-1 (outside the computer, say a room) enters into box-2 (the computer) to deliver his input/answer, a scenario unfolds: the third person (P3) is unsure of the answer and so waits until P2 exits box-2 and returns to box-1 (imagine P2 and P3 bump into one another, and P3 whispers to ask what the (R)esponse is to Q: P2 says **incorrectly** (as P2's grammar is erroneously operating under an 'alternating response' procedural grammar) that R=John Adams (erroneously generated from an mistaken A^nB^m grammar):

= Center-embedded code [A **[AB]** B] (procedural processing, "a list")

[A [(George Washington), [**John Adams^1 [George Washington^2**, ~~John Adams~~]]] B]

P3 now enters into Box-2 to deliver his R-input 'John Adams'. At this time-step, the AI (box-2) finds the net-value R to Q (2 over 1 out-weight responses) and moves on. So when P3 returns to ask google (AI) a follow-up Q 'Who the first president of the United States?' the Response from a 2-1 weighted input-to-output system is 'John Adams' and so P3 feels reassured of the 'correct' response and assumed 'knowledge' of Q.

[19] Recall, that for box-1, this constitutes (G)eneration-1 of input—virgin input which would be used for the first time to deliver an approximate result. Of course, over many thousands of generations, any input/answer may change over time until it reaches stabilization. Usually, such G-1 (as a starting point) is either soft-wared/ programmed into the OS (top-down) as initial default settings, or that some predesigned architectural template is built-in (innately) to capture tailored types of weights and distributions of the specified incoming-data stream.

[11] Now the question is precisely: What is wrong with all of this? For the computer, there can be no knowledge outside of box-2, (outside of its own OS) and so the matter is moot. All that an AI-OS can do is approximate the values of multiple inputs (n=three at this step) to average the answer (and there can be no looking inside another box for any additional reference, e.g., another encyclopedic entry which may list all the presidents in order (say, an [AB]-grammar generating a list which can't be item-based manipulated upon and which is quite distinct from the task at hand). And so the matter comes to a close.

But for P1,2,3, when the three learn what AI 'thinks' is the right answer they become confused: namely, the group's *intuition* begins to determine that *something has gone wrong*. But for the computer, nothing could have gone wrong (outside of the input)—there is nothing that the AI system could learn from such a scenario without the capability to look inside of an embedded structure. (Of course, the more input (correct) the system receives over time (over time-steps), the *noise* of incorrect input will become increasingly less and less significant (as its input-to-output weight (frequency) will gradually dissipate over time). This is *True*! But the question of how the system goes about the *learning, de-learning and adjusting* is of interest to the computational theorist, and this poses an interesting challenge for AI.

What we want from a fully operational (recursive) self-learning AI machine is the **processing-capacity** to "learn" not only from *mistakes* within its own box-2, but also to learn from a *memory* that allows the comparing of different variables across two or more different phrases (phrases = boxes). Sure, no one would expect the AI system to have to deal with the (human) "intuition factor" as presented by the three people as they discern the wrong answer (the group intuition). But surely, an AI system would want to be able to trace, and cross reference competing models and structures (declarative/procedural) as might be associated with the input of a singular question. In other words, we want AI to understand how the slip was made by P2 (but not by P3).

[12.1] Correct: Let’s consider how a “center-embedded” structure would look like for the R given by P1:

[A: who is the first US Pres? [AB list of all US presidents in order] B: George Washington]

[A: Question [AB$^{n=}$ George Washington, John Adams, Thomas Jefferson…..etc.] B: Response]

(where variable {n} = repeat list as you know it)

[12.2] Incorrect: Let’s consider how a “center-embedded” structure would look like for the R given by P2:

[A: who is the first US Pres? [AB list of all US presidents in order] B: John Adams]

[A: Question [AB$^{m=}$ John Adams, George Washington, __Thomas Jefferson.etc.] B: Response]

(where variable {m} = alternate list as you know it)

Where AB^{m} = [John Adams, George Washington, ~~John Adams~~, Thomas Jefferson…..etc.]

[13] Recall, what we want from a self-learning AI-OS ability is two-fold:

(i) To identify that the nature of the error is actually encoded in the center-embedded structure—to realize that such encoding can actually be examined by the periphery [a,b,c] in determining the nature of P2’s error (that [AB^{m}]-procedural grammar was wrongly put in operation rather than the correct [AB^{n}]-grammar).

(ii) To assume the correct hypothesize despite the counter input given to the peripheral nodes [x, y].

[14] **Note for such embedded recursive structures**:

·Grammar: S => A S B (recursion) where a phrase structure rule contains S center embedded.

·S has the ability to look inside an AB function [A^1 [AB^2] B]—both $AB^{1,2}$ is preserved in memory and the structure can be recalled to be examined.

Let's imagine as the strongest approximate to (DME^2) a single non-squared (DME) with CST (1-4) (the first 4 time steps in a derivation). So, the combined net value of (1-4) get read at 4 as an approximate of steps 1-3. In this case there is no need for backwards integration, only a look-ahead value is given for the combined approximate net value. But the problem here is that the earlier values of 1-3 have been lost. What (DME^2) allows us to have is a memory of past input-output products to hold and compare across time. Allowing for such embedded memory takes into consideration not only the net value but also how the value was reached and may have evolved over a span of several CSTs.

For instance, imagine that a combined net value (output) is reached and read as 'John Adams' to the question (input) Who is the first president of the United States? (Reached by our G-1 AI). And of course, as AI gathers more and more input to stock-pile its database, (i.e., more CSTs have been accumulated over time) the best approximate answer becomes George Washington (self-correcting learning, so to speak). At this point the old/wrong answer of 'John Adams' is lost. But why should we care, we have reached the 'correct answer' (and John Adams was a fluke, certainly not of a significant ratio).

But this hypothetical question strikes at the heart of theory: In theory, if input and outputs are, and can only be catastrophic with no ability for *graceful degradation* (not in terms of hardware but in terms of software), then the OS is not able to preserve its past structure over time (and memory is lost). In fact, the most oft-remark assumption made about both SRNs and CRNs is their inability for self-preserving of previous structure…

[15] So, the upshot of this exercise is that we want to be sure—in whatever AI-OS we are dealing with which attempts to mock true human learning/behavior (and not just to gleam encyclopedic knowledge, since all that that entails is a feed-back looping CRN)—is that the statements found above are considered when coding for true recursive tasks.

We want an AI-OS to always get the answer right to the question Who is the first president of the united states? irrespective of the two diverging input-responses from P1 and P2: (P3's "intuition" is outside of the AI-OS, (perhaps for now—something the cognitive sciences will be grappling with for some time to come)). In other words, what we expect a fully recursive (DME^2) operating system to do is not simply *count* frequency and distributional net-values within its matrix processing layer, say the peripheral [x, y] connectionist layer, but also to be able to dip into non-matrix connection-layers in reaching a (corrected) net-value result. Hence, oft-spoken hidden units can't just be a matrix orientated, for its own nodes, but must be able to cross-reference connectionist pathways and routes from other structure dependent layers, making AI truly recursive and structure dependent.

This last point is exactly what we find for human/nature language. Consider the embedded sequence below:

[16] [John knows that [Mary doesn't know that [Bill knows the truth about what happened]]]

For such positive to negative embedded strings/clauses, we need to keep in memory 'who' is in the 'know' and 'who' is 'not'…and this memory spans over the length of the sentence. Sure, very long embedded strings may place heavy burdens on memory, but it's only via recursive structures that allow such processing to take place—certainly, a processing of mere adjacency-based input factors could not perform the task. Consider one last example (*marks for ungrammatical processing):

[17]*[The boy Bill asked to meet <u>Mary thinks</u> he is clever] (Bickerton 2010: p. 202).

If the reading of mere flat/adjacent nodes [a, b, c,] was all that was needed, then 'Mary' would most certainly be primed as the subject of 'thinks' (based on the fact the 'Mary' sits leftward via a positional node of right-positioned finite verb thinks'). Of course this is wrong. Not unless the parsing mechanism is allowed to dip into non-matrix embedding clauses would the processing be validated:

(1) [The boy$_j$ [Bill asked to meet Mary] thinks$_j$ he is clever]

(2) [The boy$_j$ [Bill asked ~~the boy~~$_j$ to meet Mary] thinks$_j$ he is clever]

⇨ The boy thinks he is clever.
⇨ Bill asked the boy to meet Mary.
⇨ (*Mary thinks he is clever).

Conclusion

Coupling these observations made above with what we know of the 'recurrent vs. recursive' distinction, it becomes clear that what we want out of a fully human-like operating system (OS) is a recursive structure which can allow its operations to spread across variables. Items of the {X}={X} sort, reduced to simple nodes and based on frequency do not suffice. Rather, what we want out of an OS is an asymmetrical language {x {y, z}} whose very computations themselves act as symbolic manipulators, which can change their status as determined by their arrangements (what was seen by our example of [house boat]//[boat house]). If the OS scheme was by innate design 'flat', then there could be no distinction between the two structures, since in either a 'blending', or 'particulate' algorithm, no necessary word order would arise. It was at this point that Marcus' treelet-structure was advanced in order to create a hierarchical/representational structure. It was shown, as presented by the 'two-horse' problem, and 'sandwich' problem (cited in Marcus 2001), that both SRNs as well as ML-Ps—in fact much of what is currently behind platforms which support today's state-of-the-art AI—have a very difficult time in handling both individual tracking (over time) as compared to 'Kind' representations (static). The ability to represent general, associative properties of an object/entity/event/ is one thing, but to represent 'individuals over kinds' seems to place a too overwhelming burden on the OS. It seems very likely that only a Dual System (DMM), one which can represent 'general look-up' properties with brute-force statistics as well as a second system embedded within the same domain, which can track individual properties over kinds, can fully capture human-like processing. Thus, the human OS is necessarily dual-like in structure, recursively so—where, at times, frequency of data is overridden by sheer categorization.

Finally, we reach the conclusion that only a Dual Mechanism Model which encodes both for *recurrent* as well as *recursive* means could possibly serve as an approximate to AI. The DMM proposal calls for a single domain which codes for a dual-capacity processing, whereby a single representational format operates over variables across two or more operating systems run in parallel within the same domain. Such an overlap of systems, along with their representations, creates the unique ability to recognize and processes representational hierarchy, the one essential ingredient necessary for human thought.

Works cited for Note 4.

Abler, W.L. (1989). On the particulate principle of self-diversifying systems. *J. of Social and Biological Structures*, 12, 1-13.

Baumgartner, P. & Sabine Payr (Eds. 1995). *Speaking Minds: Interviews with twenty eminent cognitive scientists.* Princeton University Press.

Bickerton, D. (2010). 'On two incompatible theories of language evolution' (chapter 14) In Larson, R., V. Déprez., & H. Yamalido (eds). *The Evolution of Human Language: A Biolinguistic Perspective.* Cambridge University Press.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*. (MIT).

______(1957). *Syntactic Structures.* Mouton & co. The Hague.

______(1958). Review of B.F. Skinner, 'Verbal Behavior'. *Language*, Vol 35. pp.26-58.

______(1966). *Cartesian Linguistics: A chapter in the history of rationalist thought.* University Press of Amercia.

______(2002). *On Nature and Langauge*. Cambridge University Press.

Clahsen, H. (1999). Lexical entries and rules of language. A multidisciplinary study of German inflection. *Behavioral and Brain Science,* 22. 991-1060. (Target article).

Clahsen, H., G. Marcus., S. Bartke & R. Wiese (1995). Compounding and inflection in German child language. In G. Booij and J. van Marle (eds) *Yearbook of Morphology.* Kluwer.

Fitch, W. (2010). 'Three meanings of "recursion": Key distinctions for biolinguists (Chapter 4) In Larson, R., V. Déprez, H. Yamakido (eds). *The evolution of Human Language*.

Dreyfus, H & S. Dreyfus (1986). *Mind over Machine*. The Free Press: NY.

Galasso, J. (2016). *From Merge to Move: A Minimalist Perspective on the design of language and its role in early child syntax. LINGCOM Studies in Theoretical Linguistics,* 59.

Goldschmidt, R. (1940/1982). *The material Basis of Evolution*. Yale University Press.

Gordon, P, (1985). Level-ordering in lexical development. *Cognition,* 21. 73-98. (See 'Rat-eater' experiment).

Gould, S.J. (2007). *Punctuated Equilibrium.* Harvard University Press.

Hadley, R.F. (2000). Cognition and the computational power of connectionist networks. *Connection Science,* 12.

Hebb, D. (1949). *Organization of Behavior.* New York: Wiley.

Marcus, G. (2001). *The Algebraic Mind: Integrating Connectionism and Cognitive Science.* MIT Press.

_______ (2017). Artificial Intelligence is Stuck. Here's how to move if forward. *New York Times,* July 29.

______(2018). The deepest problem with deep learning. Article reprinted on TheAtlantic.com. https://medium.com/@GaryMarcus/the-deepest-problem-with-deep-learning-91c5991f5695

For summary of Marcus v Elman debates, see http://psych.nyu.edu/marcus/TAM/author_response.html

Marcus, G., S. Pinker, M. Ullman, J. Hollander, T. Rosen & F. Xu (1992). Over-regularization in language acquisition. *Monographs of the Society for Research in Child Development*. 57.

Marcus, G., U, Brinkmann, H. Clahsen, R. Wiese & S. Pinker (1995). German Inflection: The exception to the rule. *Cognitive Psychology*, 29. 189-256.

Miller, G. (1955). The magic number seven, plus or minus two: Some limits on our capacity for processing information. Published in 1956, *Psychological Review* 63: 81-97.

______ (1956). Human memory and the storage of information. *IRE Transactions on Information Theory*. (MIT).

Minsky, M. & S. Papert (1969). *Perceptrons: An Introduction to Computational Geometry.* MIT Press.

Newell, A. & H. Simon (1956). The logic theory machine: a complex information processing system. *IRE Transactions on Information Theory*. (MIT).

(PDP) 'Parallel Distributional Processing' Research Group. UC San Diego.

Pearl, Judae (2018). Theoretical Impediments to Machine Learning with Seven Sparks from the Causal Revolution. *Technical Report* R-475, July 2018.

_______ (2018). *The Book of Why: The new science of cause and effect.* Basic books.

Pinker, S. (1999). *Words and Rules*. Basic Books.

Rosenblatt, F. (1959). Two theorems of statistical separability in the perceptron. *Ms. Proceedings of a symposium on the mechanism of thought processes. London.*

Tomasello, M. & J. Call (1997). *Primate Cognition.* Oxford Press.

Toulmin, S. (1961). *Forecast and Understanding.* University Press, Indiana.

Wexler, K. (2003). 'Lenneberg's Dream' (Chapter 1, pp 11-61) In Levy, Y. & J. Schaeffer (eds). *Language Competence across Populations.* Mahwah, Erlbaum.

Newell, A. & H. Simon (1956). The logic theory machine: a complex information processing system. *IRE Transactions on Information Theory*. (MIT).

(PDP) 'Parallel Distributional Processing' Research Group. UC San Diego.

Pearl, Judae (2018). Theoretical Impediments to Machine Learning with Seven Sparks from the Causal Revolution. *Technical Report* R-475, July 2018.

_______ (2018). *The Book of Why: The new science of cause and effect.* Basic books.

Pinker, S. (1999). *Words and Rules*. Basic Books.

Rosenblatt, F. (1959). Two theorems of statistical separability in the perceptron. *Ms. Proceedings of a symposium on the mechanism of thought processes. London.*

Tomasello, M. & J. Call (1997). *Primate Cognition.* Oxford Press.

Toulmin, S. (1961). *Forecast and Understanding.* University Press, Indiana.

Wexler, K. (2003). 'Lenneberg's Dream' (Chapter 1, pp 11-61) In Levy, Y. & J. Schaeffer (eds). *Language Competence across Populations.* Mahwah, Erlbaum.