Brains as Naturally Emerging Turing Machines

Juyang Weng
Department of Computer Science and Engineering
Cognitive Science Program, and Neuroscience Program
Michigan State University
East Lansing, Michigan, 48824, USA
Email: http://www.cse.msu.edu/~weng/

***Abstract*—It has been shown that a Developmental Network (DN) can learn any Finite Automaton (FA) [29] but FA is not a general purpose automaton by itself. This theoretical paper presents that the controller of any Turing Machine (TM) is equivalent to an FA. It further models a motivation-free brain — excluding motivation e.g., emotions — as a TM inside a grounded DN — DN with the real world. Unlike a traditional TM, the TM-in-DN uses natural encoding of input and output and uses emergent internal representations. In Artificial Intelligence (AI) there are two major schools, symbolism and connectionism. The theoretical result here implies that the connectionist school is at least as powerful as the symbolic school also in terms of the general-purpose nature of TM. Furthermore, any TM simulated by the DN is grounded and uses natural encoding so that the DN *autonomously* learns any TM directly from natural world without a need for a human to encode its input and output. This opens the door for the DN to fully autonomously learn any TM, from a human teacher, reading a book, or real world events. The motivated version of DN [31] further enables a DN to go beyond action-supervised learning — so as to learn based on pain-avoidance, pleasure seeking, and novelty seeking [31].**

I. Introduction

Our computational theory [31] of brain and mind includes two major parts rooted in the rich literature about biological brains [15], [7]: (A) dynamically emerging, motivation-free circuits and functions, and (B) motivation based on such circuits and functions.

The computation in the former (A) is carried out by target-precise neuron-to-neuron signal transmissions. Weng & Luciw 2012 [37] and Weng et al. 2013 [38] presented a computational theory for such brain circuits to process information, spatial and temporal, respectively, using their distributed, emergent, and non-symbolic representations. As reviewed in those two articles, such brain circuits are also fundamentally different from many existing neural networks cited therein — the brain circuits are not only locally recurrent as many neural networks, but also globally recurrent in the sense that they all use motor as input concepts. As explained in Weng & Luciw [39] the brain motors (or actions) correspond to all possible concepts that a human can learn and express, from conception, through prenatal life, birth, childhood, infancy, and adulthood — such as location, type, scale, temporal context, goal, sub-goal, intent, purpose, price, ways to use, and so on. These concepts are used by the brain circuits as states, like states in a Finite Automaton (FA) [12], but such an FA is emergent and non-symbolic to be explained below.

The computation in the latter (B) is based on target-imprecise diffusion of neural transmitters that diffuse across brain tissue. Weng et al. 2013 [41] proposed a model for how reinforcement learning is carried out in such emergent brain circuits through two types of transmitter systems — serotonin and dopamine. Wang et al. 2011 [27] presented a model about how individual neurons use two other types of transmitter systems — acetylcholine and norepinephrine — to automatically estimate uncertainty and novelty so that each neuron can decide where it gets inputs from. These four types of neural transmitter systems — serotonin, dopamine, acetylcholine and norepinephrine — along with other neural transmitters but seemingly relatively less important than these four types [16], amount to what we know as motivation. Various emotions are special cases of motivation [15], [7].

To be focused, this paper will not further discuss the motivation part of a biological brain and will instead concentrate on the former — (A) the basic brain circuits and functions. In other words, the theory below models any emotion-free brain.

This theory here does not claim that the TM capable brain model is indeed complete for an emotion-free brain. Most likely, the opposite is true because there is no widely accepted and rigorous definition of a natural phenomenon such as a brain and, therefore, there is always some limitation for any theory to explain a natural phenomenon. As such, as any theory can only approximate a natural phenomenon but can never exhaust such an approximation. The Newtonian physics is a good example because it is refined by the relativity theory.

In the following Section II, we discuss the relevant important concepts and review the prior studies. Section III presents DN. We extend FA as a temporal machine in Section IV so as to pave the way toward our new theory of Emergent TM which is presented in Section V. Section VI briefly discusses experiments of DN including motived DNs such as pain avoidance and pleasure seeking. Section VII provides concluding remarks and discussion.

II. Relevant Studies and Concepts

All computational networks fall into two categories, Symbolic Networks (SNs) and Emergent Networks. The former category uses symbolic representations and the latter uses emergent representations. See the review for symbolic models and emergent models in Weng 2012 [32] which tried to clarify some common misconceptions on representations.

The class of SN [23] includes Finite Automata (FA), Markov Chains, Markov Models, Hidden Markov Models (HMM), Partially Observable Markov Decision Processes (POMDP), Belief Networks, Graphical Models, and all other

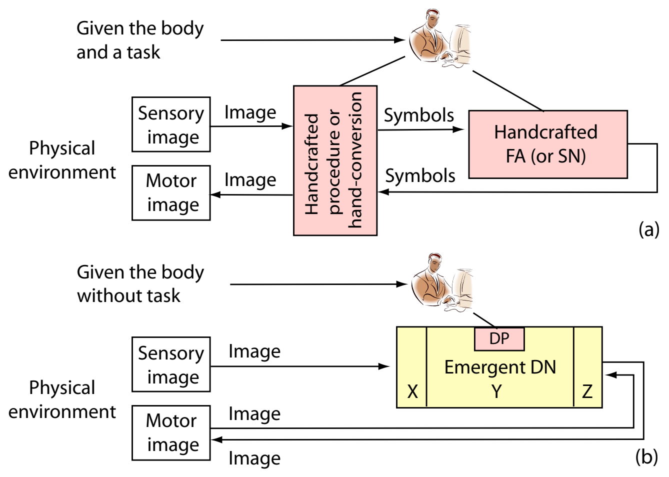

Fig. 1. Comparison between a symbolic FA (or SN) and an emergent DN. (a) Given a task, an FA (or SN), symbolic, handcrafted by the human programmer using a static symbol set. (b) A DN, which incrementally learns the FA but takes sensory images directly and produces motor images directly. Without given any task, a human designs the general-purpose Developmental Program (DP) which resides in the DN as a functional equivalent of the "genome" that regulates the development — fully autonomous inside the DN.

networks that use at least some symbolic representations. The HMM and other probability-based models in the above list are symbolic because they add probability to the symbolic FA basis and therefore the basic nature of their representations are still symbolic — adding probability does not change the nature of symbolic representation. We will use FA as an example for SN because any SN includes FA as its basis.

The class of Emergent Network includes all neural networks that use exclusively emergent representations, such as Feedforward Networks, Hopfield Networks, Boltzmann Machines, Restricted Boltzmann Machines, Liquid State Machines, Reservoir Computing, and the newer Developmental Networks (DNs) [30]. However, traditional neural networks are not as powerful and complete as DN, because they do not have the logic of FA as explained in [30].

The major differences between a Symbolic Network (SN) and a Developmental Network (DN) are illustrated in Fig. 1.

Marvin Minsky 1991 [19] and others correctly argued that symbolic models are logical and clean, while connectionist models are analogical and scruffy. Neural networks are called emergent networks here but excluding networks that are not emergent — symbolic or partially symbolic. Michael Jordan 2014 [8] correctly raised fundamental questions that many researchers have not paid sufficient attention and effort to. The logic capabilities of emergent networks, as a category, are still unclear. This paper further addresses some fundamental issues that Michael Jordan recently raised [8].

Computationally, feed-forward connections serve to feed sensory features [21] to motor area for generating behaviors. It has been reported that feed-backward connections can serve as class supervision [10], attention [3], and storage of time information.

The framework of Finite Automata (FA) plays a major role in our theory about the brain. The work of Weng 2011 [30] and 2013 [33] was not the first to relate a network with an FA. Some researchers used neural networks to batch-compile a special kind of FA as discussed below.

Frasconi et al. 1995 [5] used a feed-forward network to explicitly compute the state transition function $\delta : Q \times \Sigma \mapsto Q$ of an FA. Their network required (1) a special canonical binary coding of the states so that the Hamming distance is 1 between any source state q and any target state q', (2) an additional intermediate state is added if the source state q and target state q' are the same, (3) the entire state transition function δ is known *a priori* so that their algorithm can directly compute all the weights as a batch (i.e., compiled, instead of learned incrementally). This compiled network uses a layer of logic-AND nodes followed by a layer of logic-OR nodes. Frasconi et al. 1996 proposed a radial basis function as an alternative batch-compiled feed-forward network for the above logic network [6] since a finite number of samples is sufficient for completely characterizing the FA due to its symbolic nature. Omlin & Giles 1996 [22] proposed a second-order network for computing the state transition function of a fully given FA. By 2nd order, the neuronal input contains the sum of weighted multiplications (hence the 2nd order), between individual state nodes and individual input nodes. There does not seem to be known evidence that a biological neuron uses such a product. The network Omlin & Giles 1996 is also statically "programmed" by a human programmer based on a fully given FA. They positively contributed to neural network studies.

Using a similar approach, other researchers used neural networks to simulate a TM. Hyötyniemi 1996 proved that his handcrafted recurrent network can compute a specially encoded TM that although simple but functionally equivalent to any TM. The simplicity of the TM implies that such network techniques are probably not practically efficient.

As far as we know, the first naturally emergent Turing Machine was proposed by Weng 2014 [34] in which an emergent TM was defined in terms of the corresponding FA-in-DN, based on the mathematically proved capability of a DN to learn any large and complex FA. This paper focuses on this kind of Emergent Turing Machines.

The work Graves et al. 2014 [9] is interesting as it mentioned Turing Machine while doing neural networks. They used a neural network controller to perform copy into memory, sorting, and associate recall from a memory [9] that corresponds to a matrix of synaptic weights, as examples of operations typical for TMs. Although they called their system "Neural Turing Machine", they have not established, or intended to establish, that all their operations are sufficient for simulating any TM. In addition, they regarded an attention process as to read from and write to, selectively, some synapses of a subset of neurons. This is different from attention studied in neuroscience and psychology — attending natural objects in cluttered scenes.

In addition to the similarity to Hopfield Network [13] and LSTM [11] cited therein, Graves et al. 2014 [9] appeared to have a series of mechanism similarities with DN 2011 [30] along with DN embodiments Where-What Networks, WWN-1 2008 [14] to WWN-7 2013 [43]. To facilitate understanding their conceptual relation, let us see some of the similarities with correspondence of concepts: (1) The finite state machine (i.e. FA) was mentioned once, but not explained, in [9] but emergent FA in DN had been published in Weng 2011 [30] and the proofs published as a conference paper 2013 [33], and enriched in a journal paper 2014 [34]. (2) The $N \times M$ memory in [9] corresponds to, respectively, the synaptic weight vectors of N neurons each of which has M input components in [30]. (3) The similarity measure of [9, Eq.(6)] is the same as that of [30, Eq.(1)]. The softmax in [9] appears to correspond to the max operation in Cresceptron [35]. But DN improved the max mechanism in Cresceptron: Max in Cresceptron is replaced by the more general cross-feature local top-k competition not restricted to the same feature in the domain of the max operation since many features are not relevant to the current context. In terms of shift in [9], we are not aware of any evidence that the brain uses location-based addressing in addition to content-based addressing.

In our Emergent TM, the TM tape becomes the real world, not the memory matrix in [9]. In contrast to the error back-prop learning, DN uses biologically plausible Hebbian learning. From the proof for FA-in-DA [33], [34] and the presentation of Emergent TM here, the mechanisms in the motivation-free DN seem to be sufficient for any motivation-free TM.

The above FM and TM studies aimed at successfully computing the transition function of FA or TM using a programmed network and a special rigid encoding of input and output. However, they have the following limitations.

1) **Batch processing**: They do not learn incrementally — taking one image frame and action frame at a time to update the system and discard the frame before taking the next frame.
2) **Special sensory encoding**: the system requires a special encoding for input that is necessary for network to establish the mapping. They do not learn directly from natural images from the real world (i.e., natural images). Thus, learning is not autonomous because a human must be in the loop of sensory encoding.
3) **Special output/action encoding**: the system requires a special encoding for actions that is necessary for network to establish the mapping. They do not learn directly from natural actions from the real world (i.e., signals from robot actuators). Thus, learning is not autonomous because a human must be in the loop of output/action encoding.
4) **Compiled internal representations**: The term "compiled" means the programmer, as a task-understood central controller, must use the meaning of the encoded input and output. The internal representations are therefore not emergent because any autonomous mergence of internal representations requires the absence of a task-understood central controller.
5) **Non-developmental:** Do not co-learn while a teacher FA is operating. If FA operation represents real-time operation of a parent, teacher, or real-world event, do not imitate while such a teacher FA operates. This also implies that the learning system is not able to improve while it is already operating. Thus, earlier and simpler learned skills are not used to learn newer and more sophisticated skills while performing. Learning autonomy is not done. Teaching for human level performance is therefore so expensive that it appears to be impossible.

Our DN aims at autonomous development , which implies that every neuron in the biological brain must *automatically* figure out its roles for its cell type in the brain [30] — absence of any central controller. Handcrafted problem-specific modules — human as a central controller — do not satisfy this developmental requirement. In practice, such handcrafted problem-specific modules greatly increase the cost of product development and result in a brittle system in uncontrolled natural settings.

As far as we know, the DN in Weng 2011 [30] was the first general-purpose emergent FA in DN that have all the following properties although some earlier neural networks have some of these properties.

1) **Incremental learning** together with space and time optimality in the sense of ML (maximum likelihood). The system takes one-pair $(\mathbf{x}, \mathbf{z})$ of sensory pattern $\mathbf{x}$ and motor pattern $\mathbf{z}$ at a time to update the network and discard the pair immediately after.
2) **Natural sensory input without special encoding**. E.g., each input frame can be a natural visual image or touch image. Thus, sensory input is emergent, emerging naturally from the real world.
3) **Natural output/action without special encoding**. The output can directly drive natural effectors. It also allows the motor area to have subareas where each subarea represents either an abstract concept (location, type, scale, etc.) or natural muscle actions (e.g., driving a car or riding a bicycle), Thus, motor input is emergent, emerging naturally from the real world and the body.
4) **Fully emergent representations**. Not only the above 2) and 3) are emergent, but also the internal representation of the huge brain Y.
5) **Developmental.** Earlier and simpler learned skills are not used to learn newer and more sophisticated skills while performing. The learning inside the network is fully autonomous from the inception time, hopefully relieving human programmers from handcrafting internal representations and serve as the central controller.
6) **A unified area function** that does not need interactive computation at each time and does not have local minima in its high dimensional and nonlinear optimization. Because of the use of Voronoi regions and top-k competition, the formulation of complex nonlinear optimization leads to the computationl of an incremental mean which is a linear problem.
7) **ML optimization** at each discrete time, conditioned on the limited computational source (i.e., number of neurons) and limited learning experience (i.e., agent age and the limited intelligence of the teaching environment).

Explained in Weng 2012 [31], the DN model is inspired by biological brains, especially brain anatomy (e.g., [4], [24]) and brain physiological experiments (e.g., [3], [1]). But we will use computational language in the following material, so that the material is understandable by an analytical reader.

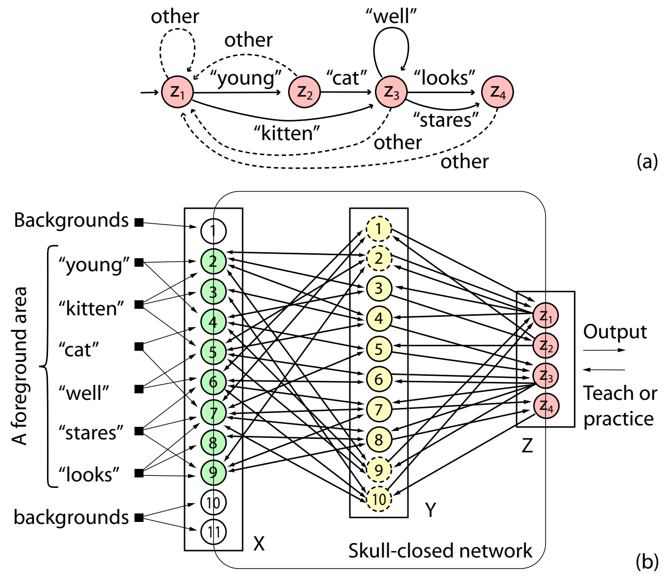

Fig. 2. Conceptual correspondence between an Finite Automaton (FA) with the corresponding DN. (a) An FA, handcrafted and static. (b) A corresponding DN that simulates the FA. It was taught to produce the same input-output relations as the FA in (a). A symbol (e.g., z_2) in (a) corresponds to an image (e.g., $(z_1, z_2, ..., z_4) = (0, 1, 0, 0)$) in (b).

The FA-based brain theory in [30] whose full proofs were available publically in [33]:

Theorem 1 states that for any FA that operates in real time, there is an emergent DN that learns the FA incrementally. It observes one state-and-input pair from the FA at a time, learns immediately and becomes error-free for all the FA transitions that it has learned, regardless how many times a transition has been observed — one is sufficient but more results in better optimality in the real world. The DN is equivalent to the part of FA that correspond to all transitions that have demonstrated so far.

Theorem 2 establishes that if the FA-learned DN is frozen — computing responses only but not updating its adaptive parts, the frozen DN is optimal in the sense of maximum likelihood when it takes inputs from infinitely many possible cases in the world.

Theorem 3 asserts that the FA-learned DN, if it is allowed to continue to learn from infinitely many possible cases in the world, is optimal in the sense of maximum likelihood.

III. Algorithm of DN

The DP self-programs the logic of the world into a DN based on DN experiences in its physical activities. The DP is small but DN is typically huge. A DN has its area Y as a "bridge" for its two banks, X and Z, as illustrated in Fig. 2(b).

Biologically, a DP algorithm models the collective effects of some genome properties of the cells of the nervous system — neurons and other types of cells in the nervous system [2], [15], [7]. Thus, in nature, the DP is a result of evolution across many generations of a species. The DP seems to be a more systematic and direct way to understand natural intelligence than studying the concrete behavior responses from the brain of a child or adult.

In artificial intelligence, a DP algorithm is the result of human understanding of the development of natural intelligence followed by a human DP design based such understanding. This approach, known as developmental approach [40], [31], short-cuts the long and expensive process of cross-generation evolution.

Some parameters of DP (e.g., the number of cells in Y) could be experimentally selected by a genetic algorithm, but the DP as a whole seems to be extremely expensive for any artificial genetic algorithm to reach without handcrafting (e.g., see the handcrafted area function below).

Human design of DP algorithm [40] seems to be a more practical way to reach human-like mental capabilities and human-level performance in robots and computers for two main reasons: (1) Fully automatic development of intelligence (i.e., task-nonspecific and fully automatic learning) is the approach that the natural intelligence takes and has demonstrated successful. (2) The design of the DP algorithm is a clean task, in contrast to traditional AI — modeling intelligence itself — which is a muddy task [28], [31].

The quality in a human-designed DP, when the DP is widely used in the future, greatly affects all the capabilities in the developmental robots and computers that use the DP.

In the DN, if Y is meant for modeling the entire brain, then X consists of all receptors and Z consists of all effectors — muscle neurons and glands. Additionally, the Y area of the DP can also model any Brodmann area in the brain and if so, the X and Z correspond to, respectively, the bottom-up areas and top-down areas of the Brodmann area. From the analysis below, we can also see that the Y area of the DN can model any closely related set of neurons — Brodmann area, a subset, or a superset.

The most basic function of the area Y seems to be prediction — predict the signals in its two vast banks X and Z through space and time.

Algorithm 1 (DP): Input areas: X and Z. Output areas: X and Z. The dimension and representation of X and Y areas are hand designed based on the sensors and effectors of the species (or from evolution in biology). Y is the skull-closed (inside the brain), not directly accessible by the outside.

(A) At time $t = 0$, for each area A in $\{X, Y, Z\}$, initialize its adaptive part $N = (V, G)$ and the response vector $\mathbf{r}$, where V contains all the synaptic weight vectors and G stores all the neuronal ages. For example, use the generative DN method discussed below.

(B) At time $t = 1, 2, ...$, for each A in $\{X, Y, Z\}$ repeat:

1) Every area A performs mitosis-equivalent if it is needed, using its bottom-up and top-down inputs $\mathbf{b}$ and $\mathbf{t}$, respectively.
2) Every area A computes its area function f, described below,
$$(\mathbf{r}', N') = f(\mathbf{b}, \mathbf{t}, N)$$
where $\mathbf{r}'$ is its response vector and N and N' are the adaptive parts of the area defined above, before and

after the area update, respectively. Note that $\mathbf{r}$ is not part of the domain of f because f is the model for any area A, not just for an individual neuron of A. Thus, f does not use iterations, efficiently approximating lateral inhibitions and internal excitations.

3) For every area A in $\{X, Y, Z\}$, A replaces: $N \leftarrow N'$ and $\mathbf{r} \leftarrow \mathbf{r}'$.

The DN must update at least twice for the effects of each new signal pattern in X and Z, respectively, to go through one update in Y and then one update in Z to appear in X and Z.

In the remaining discussion, we assume that Y models the entire brain. If X is a sensory area, $\mathbf{x} \in X$ is always supervised. The $\mathbf{z} \in Z$ is supervised only when the teacher chooses to. Otherwise, $\mathbf{z}$ gives (predicts) motor output.

The area function f which is based on the theory of Lobe Component Analysis (LCA) [36], a model for self-organization by a neural area. Each area A has a weight vector $\mathbf{v} = (\mathbf{v}_b, \mathbf{v}_t)$. Its pre-response vector is:

$$r(\mathbf{v}_b, \mathbf{b}, \mathbf{v}_t, \mathbf{t}) = \frac{\mathbf{v}_b}{\|\mathbf{v}_b\|} \cdot \frac{\mathbf{b}}{\|\mathbf{b}\|} + \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|} \cdot \frac{\mathbf{t}}{\|\mathbf{t}\|} = \dot{\mathbf{v}} \cdot \dot{\mathbf{p}} \quad (1)$$

which measures the degree of match between the directions of $\dot{\mathbf{v}} = (\mathbf{v}_b/\|\mathbf{v}_b\|, \mathbf{v}_t/\|\mathbf{v}_t\|)$ and $\dot{\mathbf{p}} = (\dot{\mathbf{b}}, \dot{\mathbf{t}}) = (\mathbf{b}/\|\mathbf{b}\|, \mathbf{t}/\|\mathbf{t}\|)$.

To simulate lateral inhibitions (winner-take-all) within each area A, only top k winners among the c competing neurons fire. Considering $k = 1$, the winner neuron j is identified by:

$$j = \arg \max_{1 \le i \le c} r(\mathbf{v}_{bi}, \mathbf{b}, \mathbf{v}_{ti}, \mathbf{t}). \quad (2)$$

The area dynamically scale top-k winners so that the top-k respond with values in $(0, 1]$. For $k = 1$, only the single winner fires with response value $y_j = 1$ and all other neurons in A do not fire. The response value y_j approximates the probability for $\dot{\mathbf{p}}$ to fall into the Voronoi region of its $\dot{\mathbf{v}}_j$ where the "nearness" is $r(\mathbf{v}_{bj}, \mathbf{b}, \mathbf{v}_{tj}, \mathbf{t})$.

All the connections in a DN are learned incrementally based on Hebbian learning — cofiring of the pre-synaptic activity $\dot{\mathbf{p}}$ and the post-synaptic activity y of the firing neuron. If the pre-synaptic end and the post-synaptic end fire together, the synaptic vector of the neuron has a synapse gain $y\dot{p}$. Other non-firing neurons do not modify their memory. When a neuron j fires, its firing age is incremented $n_j \leftarrow n_j + 1$ and then its synapse vector is updated by a Hebbian-like mechanism:

$$\mathbf{v}_j \leftarrow w_1(n_j)\mathbf{v}_j + w_2(n_j)y_j\dot{\mathbf{p}} \quad (3)$$

where $w_2(n_j)$ is the learning rate depending on the firing age (counts) n_j of the neuron j and $w_1(n_j)$ is the retention rate with $w_1(n_j) + w_2(n_j) \equiv 1$. Note that a component in the gain vector $y_j\dot{\mathbf{p}}$ is zero if the corresponding component in $\dot{\mathbf{p}}$ is zero.

The simplest version of $w_2(n_j)$ is $w_2(n_j) = 1/n_j$ which corresponds to:

$$\mathbf{v}_j^{(i)} = \frac{i-1}{i}\mathbf{v}_j^{(i-1)} + \frac{1}{i}\dot{\mathbf{p}}(t_i), i = 1, 2, ..., n_j, \quad (4)$$

where t_i is the firing time of the post-synaptic neuron j. The above is the recursive way of computing the batch average:

$$\mathbf{v}_j^{(n_j)} = \frac{1}{n_j}\sum_{i=1}^{n_j} \dot{\mathbf{p}}(t_i) \quad (5)$$

The initial condition is as follows. The smallest n_j in Eq. (3) is 1 since $n_j = 0$ after initialization. When $n_j = 1$, the initial value of $\mathbf{v}_j$ on the right side of Eq. (3) is used for pre-response competition to find this winner j but the initial value of $\mathbf{v}_j$ does not affect the first-time updated $\mathbf{v}_j$ on the left side since $w_1(1) = 1 - 1 = 0$.

In other words, any initialization of weight vectors will only determine who win (i.e., which newly born neurons take the current role) but the initialization will not affect the distribution of weights at all. In this sense, all random initializations of synaptic weights will work equally well — all resulting in weight distributions that are computationally equivalent. Biologically, we do not care which neurons (in a small 3-D neighborhood) take the specific roles, as long as the distribution of the synaptic weights of these neurons lead to the same computational effect. This neuronal learning model leads to the following conjecture.

Conjecture 1: In a small 3-D neighborhood (e.g., of a hundred nearby neurons), neural circuits are so different across different biological brains that mapping the detailed neuron wiring of brain is not informative at the level of individual neuron.

The NIH Connectome program aims to "map the neural pathways ... about the structural and functional connectivity of the human brain. ... resulting in improved sensitivity, resolution, and utility, thereby accelerating progress in the emerging field of human connectomics." The DN theory and the above conjecture predict that such an NIH program is not as scientifically useful as the NIH program hoped in terms of understanding how the brain works and future studies of abnormal brain circuits. For the brain, "more detailed connectomics data" seems to be not as productive as more complete and clear theories.

IV. FA as a Temporal Machine

In this section, we present an FA as a temporal machine, although traditionally an FA is a logic machine, driven by discrete event of input.

As we need a slight deviation from the standard definition of FA, let us look at the standard definition first.

Definition 1 (Language acceptor FA): A finite automaton (FA) M is a 5-tuple $M = (Q, \Sigma, q_0, \delta, A)$, where Q is a finite set of states, consists of symbols. Σ is a finite alphabet of input symbols. $q_0 \in Q$ is the initial state. $A \subset Q$ is the set of accepting states. $\delta : Q \times \Sigma \mapsto Q$ is the state transition function.

This classical definition is for a language acceptor, which accepts all strings x from the alphabet Σ that belongs to a language L. It has been proved [12] that given any *regular language* L from alphabet Σ, there is an FA that accepts L, meaning that it accepts exactly all $\mathbf{x} \in L$ but no other string not in L. Conversely, given any FA taking alphabet Σ, the language L that the FA accepts is a regular language. However, a language FA, just like any other automata, only deals syntax not semantics. The semantics is primary for understanding a language and the syntax is secondary.

We need to extend the definition of FA for agents that run at discrete times, as follows:

Definition 2 (Agent FA): A finite automaton (FA) M for a finite symbolic world is a 4-tuple $M = (Q, \Sigma, q_0, \delta)$, where Σ and q_0 are the same as above and Q is a finite set of states, where each state $q \in Q$ is a symbol, corresponding to a set of concepts. The agent runs through discrete times $t = 1, 2, ...,$ starting from state $q(t) = q_0$ at $t = 0$. At each time $t-1$, it reads input $\sigma(t-1) \in \Sigma$ and transits from state $q(t-1)$ to $q(t) = \delta(q(t-1), \sigma(t-1))$, and outputs $q(t)$ at time t, illustrated as $q(t-1) \xrightarrow{\sigma(t-1)} q(t)$.

The inputs to an FA are symbolic. The input space is denoted as $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_l\}$, which can be a discretized version of a continuous space o input. In sentence recognition, the FA reads one word at a time. The number l is equal to the number of all possible words — the size of the vocabulary. For a computer game agent, l is equal to the total number of different percepts.

The outputs (actions) from a language acceptor FA are also symbolic, $A = \{a_1, a_2, ..., a_n\}$ which can also be a discretized version of a continuous space of output. For a sentence detector represented by an FA, when the FA reaches the last state, its action reports that the sentence has been detected.

An agent FA is an extension from the corresponding language FA, in the sense that it outputs the state, not only the acceptance property of the state. The meanings of each state, which are handcrafted by the human programmer but are not part of the formal FA definition, are only in the mind of the human programmer. Such meanings can indicate whether a state is an accepting state or not, along many other meanings associated with each state as our later example will show. However, such concepts are only in the mind of the human system designer, not something that the FA is "aware" of. This is a fundamental limitation of all symbolic models. The Developmental Network (DN) described below do not use any symbols, but instead (image) vectors from the real-world sensors and real-world effectors. As illustrated in Fig. 2, a DN is grounded in the physical environment but an FA is not.

Fig. 3 gives an example of the agent FA. Each state is associated with a number of cognitive states and actions, shown as text in the lower part of Fig. 3, reporting action for cognition plus a motor action. The example in Fig. 3 shows that an agent FA can be very general, simulating an animal in a micro, symbolic world. The meanings of each state in the lower part of Fig. 3 are handcrafted by, and only in the mind of, the human designer. These meanings are not a part of the FA definition and are not accessible by the machine that simulates the FA.

Without loss of generality, we can consider that an agent FA simply outputs its current state at any time, since the state is uniquely linked to a pair of the cognition set and the action set, at least in the mind of human designer.

V. Emergent Turing Machines

It has been proved [12] that an FA with n states partitions all the strings in Σ into n sets. Each set is called equivalence class, consisting of strings that are indistinguishable by the FA. Since these strings are indistinguishable, any string x in the same set can be used to denote the equivalent class, denoted as $[x]$. Let Λ denote an empty string. Consider Fig. 3. The

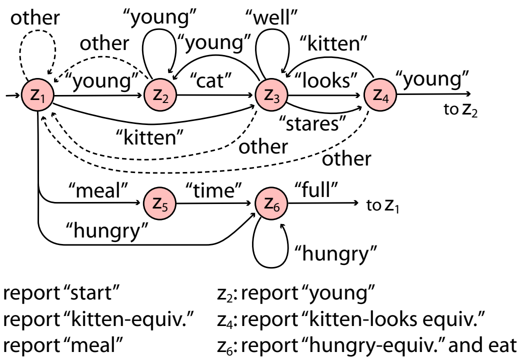

Fig. 3. An FA simulates an animal. Each circle indicates a context state. The system starts from state z_1. Supposing the system is at state q and receives a symbol σ and the next state should be q', the diagram has an arrow denoted as $q \xrightarrow{\sigma} q'$. A label "other" means any symbol other than those marked from the out-going state. Each state corresponds to a set of actions, indicated below the FA. The "other" transitions from the lower part are omitted for brevity.

FA partitions all possible strings into 6 equivalent classes. $[\Lambda] =$ ["calculus"] as the agent does not know about "calculus" although it is in Σ. All the strings in the equivalent class $[\Lambda]$ end in z_1. All strings in the equivalent class ["kitten" "looks"] end in z_4, etc.

From the above discussion, we can see that the key power of an FA is to lump very complex equivalent (q, σ) contexts into equivalent classes.

A Turing Machine (TM) [12], [18] is a 5-tuple $T = (Q, \Sigma, \Gamma, q_0, \delta)$, where Q is the set of states, Σ and Γ are the input and tape alphabets, respectively, with $\Sigma \subseteq \Gamma$, q_0 is the initial state, and δ is the transition function:

$$\delta : Q \times (\Gamma \cup \{\Delta\}) \rightarrow (Q \cup \{h\}) \times (\Gamma \cup \{\Delta\}) \times \{R, L, S\}$$

where Δ is the blank symbol not in Γ, h denotes the halt state, and R, L, S denote the head motion, right, left, and stationary, respectively. Consider the following two definitions:

1) Define Q' to include also the tape write action w and the head move action m:

$$Q' = (Q \cup \{h\}) \times (\Gamma \cup \{\Delta\}) \times \{R, L, S\}.$$

Each state in Q' is a three tuple (q, w, m) where w and m can be empty.

2) Let $\Sigma' = \Gamma \cup \{\Delta\}$.

The above transition function δ for TM becomes the transition function δ' of an FA: $\delta' = Q' \times \Sigma' \mapsto Q'$.

Therefore, the controller of any TM is an FA. A grounded DN can learn the FA perfectly. It takes input $\sigma \in \Sigma'$ from the real word and its action can include head write and head motion. A TM is not grounded, but the DN is grounded: A TM senses from, and acts on, a tape but a DN senses from, and acts on, its real-world physical environment.

The completeness of agent FA-in-DN can be described as follows. Given a vocabulary Σ' representing the elements of a symbolic world, a natural language L is defined in terms of Σ' where the meanings of all sentences (or events) in L

are defined by the set of equivalent classes, determined by Q' of FA-in-DN. When the number of states is sufficiently large, a properly learned FA-in-DN can sufficiently characterize the cognition and behaviors of an agent living in the real physical world with vocabulary Σ'.

This argument is based on the following observation: As long as the context state $q(t-1)$ is properly learned so that it contains all the information that is necessary and sufficient for generating the following states. Then $q(t-1)$ with sensory input $\sigma(t-1)$ correctly selected from a cluttered scene should be sufficient to generate the next state: $q(t-1) \overset{\sigma(t-1)}{\longrightarrow} q(t)$.

The Chomsky hierarchy [18] after the work of Norm Chomsky in particular and the automata and languages theory in classical computer science [12], [18] in general regard only Turing Machines as general-purpose programming machine because they mainly consider only the syntax of a computer language, not the rich semantics that a symbol can represent. However, a symbolic state q and an input symbol σ can practically represent any set of meanings. Yet, the meanings of general purpose with Turing Machines and FA-in-DN are different: With a TM, it means what kind of sequence of computations the TM program can represent. With the latter FA-in-DN, it means the richness of meaning any symbol (q and σ) can represent so that the FA-in-DN can represent any emergent state-based agent that has a finite memory.

In particular, it is important to note that a state can remember very early event [38], [31]: E.g., an event needed by $q(t)$ can be contained in $q(t-1)$, $q(t-2)$, etc.

But FA-in-DN goes beyond the symbolic AI, because it automatically develop internal representations — emergent.

VI. Experimental Results

Due to the focused theoretical subject here and the space limitation, detailed experimental results of DN are not included here. The DN has had several versions of experimental embodiments, called Where-What Networks (WWNs), from WWN-1 [14] to WWN-7 [43]. Each WWN has multiple areas in the Z areas, representing the location concept (Location Motor, LM), type concept (Type Motor, TM), or scale concept (Scale Motor, SM), and so on.

A learned WWN can simultaneously detect and recognize learned 3-D objects from new unobserved cluttered natural scenes [17], [37].

The function of this space-time machine DN differs depending on the context information in its Z area [39]. If there is no Z signal at all, the WWN is in an (emergent) free-viewing mode and it detects any learned object from the cluttered scene and tells its location from LM, type from TM, and scale from SM. If the LM area fires representing a location (intent or context), the WWN recognizes the object near that intended location from the cluttered scene and tells its type from TM and scale from SM. If the TM area fires representing an object type (intent or context) the WWN finds (i.e., detects) an intended object type from the cluttered scene and tells its location from LM and scale from SM.

A WWN can also perform autonomous attention: If the DN suppresses the firing neuron that represents an object type in TM, the WWN switches attention from one object type to another object type that barely lost in the previous Y competition — feature-based autonomous attention. If the DN suppresses the firing neuron in LM, the WWN switches attention from one object location to another object location that barely lost in the previous Y competition — location-based autonomous attention.

The WWN has also performed language acquisition for a subset of natural language and also generalized and predicted [20]. For example, predict from one person Joe to his hierarchical properties such as male and human, and predict from Penguin to its hierarchical properties such as non-flying and bird.

The WWNs have versions that are motivated, such as pain avoidance and pleasure seeking, so that its learning does not need to be supervised [41]. The learned tasks include object recognition under reinforcement learning and autonomous foraging (wandering around) in the presence of a friend and an enemy.

However, the experimental results from such DN experiments are difficult to understand and to train without a clear theoretical framework here that links DNs with the well-known automata theory and the mathematical properties presented as the three theorems that have been proved here.

VII. Conclusions and discussion

In conclusion, the controller of a TM is an FA. This paper presents a theory that a DN interactively and incrementally learns a naturally emerging TM by imitating a teacher TM that is operating. The new theory builds on the established theory about FA in DN. How much such naturally emergent TM can explain the logic of a brain is still left for future work, although the general-purpose nature of TM is widely recognized.

This theory gives a developmental TM. By developmental, we mean that the model regards brain areas should automatically emerge from activities, instead of fully specified rigidly by the genome. This view is supported by a great deal of cross-modal plasticity found in mammalian brains, from eye deprivation by Torsten N. Wiesel and David H. Hubel [42], to the *auditory* cortex that processes *visual* information by Mriganka Sur et al. [25], to the reassignment of modality — visual cortex is reassigned to audition and touch in the born blind as reviewed by Patrice Voss [26].

Therefore, it appears that a valid brain model at least should *not* assume a static existence of — genome rigidly specified — Brodmann areas. This static existence has been prevailing in almost all existing biologically inspired models for sensorimotor systems. Instead, a brain model should explain the emergence, and known plasticity of, brain areas. DP enables areas to emerge in DN and adapt but we have not experimentally conduct such studies. The genome provides the power of cells to move and connect. The genome also plays a major role in early and coarse connections of a brain. However, fine connections in the brain seem to be primarily determined by the statistics of activities from the conception of the life all the way up to the current life time.

Acknowledgements

The author would like to thank Hao Ye at Fudan University who carefully proof-read the proofs presented here and raised two gaps that I have filled since then. The author would also like to thank Z. Ji, M. Luciw, K. Miyan, N. Wagle, Z. Zheng, and other members of the Embodied Intelligence Laboratory at Michigan State University; Q. Zhang, Y. Wang, X. Song, Q. Guo, X. Wu, W. Zhang, and other members of Fudan University whose work have provided experimental supports for the theory presented here.

References

[1] N. P. Bichot, A. F. Rossi, and R. Desimone, "Parallel and serial neural mechanisms for visual search in macaque area v4," *Science*, vol. 308, pp. 529–534, 2006.

[2] N. A. Campbell, J. B. Reece, L. A. Urry, M. L. Cain, S. A. Wasserman, P. V. Minorsky, and R. B. Jackson, *Biology*, 9th ed. San Francisco: Benjamin Cummings, 2011.

[3] R. Desimone and J. Duncan, "Neural mechanisms of selective visual attention," *Annual Review of Neuroscience*, vol. 18, pp. 193–222, 1995.

[4] D. J. Felleman and D. C. Van Essen, "Distributed hierarchical processing in the primate cerebral cortex," *Cerebral Cortex*, vol. 1, pp. 1–47, 1991.

[5] P. Frasconi, M. Gori, M. Maggini, and G. Soda, "Unified integration of explicit knowledge and learning by example in recurrent networks," *IEEE Trans. on Knowledge and Data Engineering*, vol. 7, no. 2, pp. 340–346, 1995.

[6] ——, "Representation of finite state automata in recurrent radial basis function networks," *Machine Learning*, vol. 23, pp. 5–32, 2006.

[7] M. A. Gluck, E. Mercado, and C. Myers, Eds., *Learning and Memory: From Brain to Behavior*, 2nd ed. New York: Worth Publishers, 2013.

[8] L. Gomes, "Machine-learning maestro michael jordan on the delusions of big data and other huge engineering efforts," *IEEE Spectrum*, Online article posted Oct. 20, 2014.

[9] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," Google DeepMind, London, UK, Tech. Rep., Dec. 10, 2014, arXiv:arXiv:1410.5401.

[10] G. E. Hinton, S. Osindero, and Y.-. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*. Boston, MA: Addison-Wesley, 2006.

[13] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences of the USA*, vol. 79, no. 8, pp. 2554–2558, 1982.

[14] Z. Ji, J. Weng, and D. Prokhorov, "Where-what network 1: "Where" and "What" assist each other through top-down connections," in *Proc. IEEE Int'l Conference on Development and Learning*, Monterey, CA, Aug. 9-12, 2008, pp. 61–66.

[15] E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. Siegelbaum, and A. J. Hudspeth, Eds., *Principles of Neural Science*, 5th ed. New York: McGraw-Hill, 2012.

[16] J. L. Krichmar, "The neuromodulatory system: A framework for survival and adaptive behavior in a challenging world," *Adaptive Behavior*, vol. 16, no. 6, pp. 385–399, 2008.

[17] M. Luciw and J. Weng, "Where What Network 3: Developmental top-down attention with multiple meaningful foregrounds," in *Proc. IEEE Int'l Joint Conference on Neural Networks*, Barcelona, Spain, July 18-23, 2010, pp. 4233–4240.

[18] J. C. Martin, *Introduction to Languages and the Theory of Computation*, 3rd ed. Boston, MA: McGraw Hill, 2003.

[19] M. Minsky, "Logical versus analogical or symbolic versus connectionist or neat versus scruffy," *AI Magazine*, vol. 12, no. 2, pp. 34–51, 1991.

[20] K. Miyan and J. Weng, "WWN-Text: Cortex-like language acquisition with What and Where," in *Proc. IEEE 9th Int'l Conference on Development and Learning*, Ann Arbor, August 18-21, 2010, pp. 280–285.

[21] B. A. Olshaushen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, June 13, 1996.

[22] C. W. Omlin and C. L. Giles, "Constructing deterministic finite-state automata in recurrent neural networks," *Journal of the ACM*, vol. 43, no. 6, pp. 937–972, 1996.

[23] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, New Jersey: Prentice-Hall, 2010.

[24] M. Sur and J. L. R. Rubenstein, "Patterning and plasticity of the cerebral cortex," *Science*, vol. 310, pp. 805–810, 2005.

[25] L. VonMelchner, S. L. Pallas, and M. Sur, "Visual behaviour mediated by retinal projections directed to the auditory pathway," *Nature*, vol. 404, pp. 871–876, 2000.

[26] P. Voss, "Sensitive and critical periods in visual sensory deprivation," *Frontiers in Psychology*, vol. 4, p. 664, 2013, doi: 10.3389/fpsyg.2013.00664.

[27] Y. Wang, X. Wu, and J. Weng, "Synapse maintenance in the where-what network," in *Proc. Int'l Joint Conference on Neural Networks*, San Jose, CA, July 31 - August 5, 2011, pp. 2823–2829.

[28] J. Weng, "Task muddiness, intelligence metrics, and the necessity of autonomous mental development," *Minds and Machines*, vol. 19, no. 1, pp. 93–115, 2009.

[29] ——, "Three theorems: Brain-like networks logically reason and optimally generalize," in *Proc. Int'l Joint Conference on Neural Networks*, San Jose, CA, July 31 - August 5, 2011, pp. 2983–2990.

[30] ——, "Why have we passed "neural networks do not abstract well"?" *Natural Intelligence: the INNS Magazine*, vol. 1, no. 1, pp. 13–22, 2011.

[31] ——, *Natural and Artificial Intelligence: Introduction to Computational Brain-Mind*. Okemos, Michigan: BMI Press, 2012.

[32] ——, "Symbolic models and emergent models: A review," *IEEE Trans. Autonomous Mental Development*, vol. 4, no. 1, pp. 29–53, 2012.

[33] ——, "Establish the three theorems: DP optimally self-programs logics directly from physics," in *Proc. International Conference on Brain-Mind*, East Lansing, Michigan, July 27 - 28 2013, pp. +1–9.

[34] ——, "Brain as an emergent finite automaton: A theory and three theorems," *International Journal of Intelligent Science*, 2014, received Nov. 3, 2014 and accepted by Dec. 5, 2014.

[35] J. Weng, N. Ahuja, and T. S. Huang, "Learning recognition and segmentation using the Cresceptron," *International Journal of Computer Vision*, vol. 25, no. 2, pp. 109–143, Nov. 1997.

[36] J. Weng and M. Luciw, "Dually optimal neuronal layers: Lobe component analysis," *IEEE Trans. Autonomous Mental Development*, vol. 1, no. 1, pp. 68–85, 2009.

[37] ——, "Brain-like emergent spatial processing," *IEEE Trans. Autonomous Mental Development*, vol. 4, no. 2, pp. 161–185, 2012.

[38] J. Weng, M. Luciw, and Q. Zhang, "Brain-like temporal processing: Emergent open states," *IEEE Trans. Autonomous Mental Development*, vol. 5, no. 2, pp. 89 – 116, 2013.

[39] J. Weng and M. D. Luciw, "Brain-inspired concept networks: Learning concepts from cluttered scenes," *IEEE Intelligent Systems Magazine*, vol. 29, no. 6, pp. 14–22, 2014.

[40] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen, "Autonomous mental development by robots and animals," *Science*, vol. 291, no. 5504, pp. 599–600, 2001.

[41] J. Weng, S. Paslaski, J. Daly, C. VanDam, and J. Brown, "Modulation for emergent networks: Serotonin and dopamine," *Neural Networks*, vol. 41, pp. 225–239, 2013.

[42] T. N. Wiesel and D. H. Hubel, "Comparison of the effects of unilateral and bilateral eye closure on cortical unit responses in kittens," *Journal of Neurophysiology*, vol. 28, pp. 1029–1040, 1965.

[43] X. Wu, Q. Guo, and J. Weng, "Skull-closed autonomous development: WWN-7 dealing with scales," in *Proc. International Conference on Brain-Mind*. East Lansing, Michigan: BMI Press, July 27-28, 2013, pp. 1–8.