Meanings of “Data” and “Rules” Emerge as Actions through Auto-Programming for General Purposes

Juyang Weng[1,2]
[1] Dept. of Computer Science and Engineering, Cog. Sci. Program, and Neurosci. Program
Michigan State University, East Lansing, MI, 48824 USA
[2] GENISAMA LLC, 4700 S. Hagadorn Rd, Suite 107-F, East Lansing, MI 48823 USA
weng@cse.msu.edu, info@genisama.com

Abstract

By definition, semantics is a branch of linguistics and logic concerned with meaning. The computer vision community has been using text as a form of linguistic description about visual meanings of a natural world. This endeavor is well accepted because indeed humans often use text to describe a visual world. However, the goals of this endeavor seem to be more tractable if we consider text synthesis as a special case of action generation. Indeed, much visual information is difficult to explain clearly by only text (e.g., visual subtlety between a white rock and a white paper ball, both of a fist size). However, actions (e.g., for car driving, avoid the former but run over the latter) are often more directly related to the meanings demanded by the task at hand. This paper shows that learning actions includes not only semantics and text synthesis, but also enables a machine, biological and artificial, to learn auto-programming for general purposes (APFGP). A simple but powerful learning engine—Developmental Network (DN)—automatically generates action patterns as numerical vectors for its APFGP. This work breaks the artificial barrier between “rules” and “data” in the well-established framework of Universal Turing Machines (UTM).

1 Introduction

In computer vision, Cresceptron appeared in ICCV 1993 [Weng *et al.*, 1993] and IJCV 1997 [Weng *et al.*, 1997], as far as the author knows, is the first deep learning CNN (Convolutional Neural Networks) for 3D worlds. The key ideas are as follows: delay abstraction of a 3D object in a cluttered 3D scene till the last layer of CNN by a class label, expressed by text. Within the CNN, visual features represented by neurons are not holistic, but rather distributed, non-symbolic, and numerical in nature.

The Cresceptron approach to deep learning for 3D has been adopted by many later publications for 3D, e.g., [Riesenhuber and Poggio, 1999; Fei-Fei, 2005; Serre *et al.*, 2007; Krizhevsky *et al.*, 2012; LeCun *et al.*, 2015; Jordan and Mitchell, 2015] without crediting the source Cresceptron. Cresceptron cited Neocognitron [Fukushima, 1980] that does not learn and deals with only 2D. The first max-pooling like mechanism in Cresceptron, confirmed by [Schmidhuber, 2015] reflected a great defect that Cresceptron and many later publications faced—representations in later layers are still sensory in nature. By this, we mean that each neuron in all later layers all centers at a pixel location, i.e., without abstraction other than being “rough” in visual features.

In Where-What Network 1 (WWN-1) [Ji *et al.*, 2008] emergent representations abstract using motor signals. By abstract, we mean that each feature has explicitly two components, sensory and motor/action. As soon as motor becomes part of the criterion for matching, the corresponding neuronal feature is no longer sensory, but sensorimotor instead. Sensorimotor is not new in symbolic representations as we will see with Finite Automata (FA) below, but for neuronal representation, WWN-1 seems to be the first (neuronal sensorimotor).

These sensorimotor-based and competition-based emergent representations are fundamentally different from feedback control in classical control theories because classical control theories do not self-generate representations at all.

However, this work argues that symbols and text pose great limitations that have not been exposed clearly in the literature. By symbols, we mean a set of handcrafted set Σ of finite number of symbols.

In an *ungrounded* network, its human programmer handcrafts a mapping $f : \Sigma \mapsto R^n$ which maps each symbol $\sigma \in \Sigma$ in a symbolic set Σ to a unique vector $\mathbf{x}$ in the vector space R^n of a static dimensionality n. This mapping converts relations in $\Sigma \times \Sigma$ between symbols like (σ_1, σ_2) into vector-to-vector relations like $(\mathbf{x}_1, \mathbf{x}_2)$. Therefore, numerical techniques such as error back-propagation are used for minimization of approximation errors where neurons with weights become interpolators. This is effective if the vectors are very sparse in the vector space R^n. But natural vectors are dense:

Computer vision systems using CNN are *grounded*. Namely, they deal with many images each of which is a vector $\mathbf{x}$ in the vector space $X = R^m$, where m can be considered the number of pixels in an image. However, many computer systems still assume the above handcrafted mapping $f : \Sigma \mapsto R^n$. A CNN generates features from R^m to Σ or from R^m to R^n. DN generates emergent representations not from X alone like CNN, but from both X and R^n. Not feeding R^n into an expended version of X as in [Mnih *et al.*, 2015] to become a feed-forward network, but rather a DN grows representations from X forward and from R^n backwards using an emergent network that has both deep and shallow connections. This two-way development of representations in DN has a natural and optimal representations: more "concrete" near $X = R^m$ and more "abstract" near $Z = R^n$ and having any degree of mixture in between.

However, because the set Σ is static, the learner is not able to automatically learn and discover new concepts beyond the statically given set Σ. Another problem is that only the first order logic on Σ has tractable algorithms but higher order logics do not yet. Nevertheless, humans do not seem to have problems to deal with higher order logic. [Weng, 2012] argued that there needs to be a new kind of problem formulation that goes beyond pattern recognition using a static symbolic set so that a new kind of learning systems could accomplish the grand goals of task-nonspecificity [Weng *et al.*, 2001], e.g., machine discovery of new knowledge beyond that Σ.

Toward this end, we address a very challenging problem that a biological brain faces: Do not require a human programmer to handcraft a set of symbols, neither for sensor nor for motor, as explained below.

1. Sensor: the learner directly receives sensory inputs (e.g., images) from a sensor (e.g., camera) where each image is represented by a vector $\mathbf{x} \in X$ which contains a projection (i.e., patches) of many 3D objects, such as in a busy street scene.

2. Motor: Each motor vector is not "pure" like a "pure" vector that corresponds to a clean class label "human". Each motor vector $\mathbf{z} \in Z$ contains multiple subvectors each of which corresponds to the action of (e.g., saying) a certain concept, action, goal, intent, etc., in a way similar to, but not the same as, how images are projected from a natural world.

Namely, both the image vectors and the motor vectors are not pure, i.e., not monolithic. A major difference between a sensor and a motor is that a vector of the former is from the extra-body 3D world which the leaning agent can only partially change; but a vector from the latter is from the body of the learning agent which the learning agent can apply much change (e.g., actions) limited by its own body constraints.

This paper explains how such a learner autonomously learns from the natural physical world, via real time sensory vectors $\mathbf{x} \in X$ and motor vectors $\mathbf{z} \in Z$, with or without human teachers in the environment. The DN generates hidden neurons with weight vectors $\mathbf{y} \in Y$ using optimal statistics. An emergent Universal Turing Machine (UTM) is learned so that the agent gradually learns how to Auto-Program for General Purposes (APFGP) in the natural world from infancy to adulthood. The programmer of the learning engine DN must enable the agent to learn skills from simple to complex, and early learned skills assist later learning of new skills.

None of the following sections should be skipped if one wants to gain a basic degree of understanding.

2 Basic Formulations

A developmental learning agent has a set of sensors $(S_1, S_2, ..., S_l)$ and a set of motors $(M_1, M_2, ..., M_m)$. In the following, we will consider one sensor only and one motor only because the DN algorithm is the same: It looks at a grand sensory image consisting of l sensory sub-images, and a grand motor image consisting of m motor sub-images. Each sensor provides only an image vector $\mathbf{x}(t)$ at time t. Because each vector is a sensed image from a cluttered world, it is not know what objects are in each image or how many. Each component in the sensory vector corresponds to a pixel on retina, a hair cell in cochlea, a touch neuron in skin, and so on.

Likewise, we do not require that each motor (e.g., muscle) vector in each motor corresponds to a symbolic action or a class label. Instead, each motor provides only a motor vector $\mathbf{z}(t)$ at time t. Each component of the motor vector corresponds to a muscle neuron, but many muscle neurons may fire together to show a complex pattern of actions, such as speak English or sing a song through vocal tract.

The above general consideration leads to an interesting but important outcome: The DN algorithm is independent of the number of sensors and the number of motors. The DN algorithm is task-nonspecific and modality-nonspecific. By modality, we have sensory modality (vision, audition, touch, etc.) By motor modality, we have leg, mouth, arm, etc. For simplicity, we assume that each neuron has an initial sensory receptive field and an initial motor receptive field, both of which dynamically change though lifetime.

3 Finite Automata: Symbolic

Let us start from Finite Automata (FA) since they are simply and powerful. Then, we will remove symbols.

Skills that are declarable [Sun *et al.*, 2005] (e.g., through English) can be declared verbally through muscles in the vocal tract. Non-declarable skills (e.g., riding a bike) are also executed through muscles. These numerical networks, biological or artificial, do not necessarily have symbols inside.

Next, consider the "state" of a motor vector as a symbolic "state" of a Finite Automaton (FA). The transition function of an FA takes the current state $q \in Q$ and the current input $\sigma \in \Sigma$ and maps the state-input pair (q, σ) to the next state $q' \in Q$, but for convenience we write the pair of q and σ vertically as a column vector:

$$\begin{bmatrix} q \\ \sigma \end{bmatrix} \rightarrow \begin{bmatrix} q' \\ \sigma' \end{bmatrix}. \tag{1}$$

where σ' is the next input, or the predicted next input when the input is absent. This new representation of FA, q at top and σ at bottom, is useful for our discussion below about internal representations in the hidden area Y in Eq. (10).

Why FA? In computer science, the Universal Turing Machines were widely recognized as a model for any general purpose digital computers (Von Neumann computers). However, Weng [Weng, 2015] recently proved that the control of any Turing Machine is an FA. The main idea in the proof [Weng, 2015] was to allow each state in FA to include also actions—write a symbol onto the TM tape and move the read-write head of the TM tape. Thus, learning any FA is sufficient for any Universal TM and sufficient for "general purposes".

4 Emergent FA: Non-Symbolic

Below, we bridge a large gap between symbols and vectors. We enable DN to do abstraction that we handcraft symbols for, but without a human in the loop of handcrafting any set of symbols.

We consider the "skull" to be the enclosure boundary of DN (brain). Inside the skull is "internal" and outside the skull is "external". Running at discrete times $t = 0, 1, 2, 3, ...$, the sensory area X of DN takes input patterns $\mathbf{x}(t)$ and the muscles area Z of DN takes state patterns $\mathbf{z}(t)$. State/action $\mathbf{z}$ is taught by the environment but very often self-supervised.

5 Data and Rules Unified

In a Universal TM, the tape contains two sections separated by a special coding. The first section has a series of rules equivalent to any computer program. The second section consists of a series of data for the program to apply to.

The following Theorem states that in an Emergent FA (EFA), the rules and data are unified so that there is no need for the EFA to distinguish rules from data or vice versa.

Theorem 1 (Rules and Data) *In an EFA, each sensory (motor) image contains rules and data and there is not need for the EFA engine DN to treat rules and data differently.*

Proof: In a symbolic UTM, regardless the UTM is operating on the rule part, operating on the data part, or transit between them, the UTM transition always corresponds to a transition of EFA. Reading rules and data corresponds to reading an input $\mathbf{x}$ in EFA [Weng, 2015]. A transition in UTM corresponds to generating a new state in EFA. The rules and data are not distinguished in EFA. So, with EFA, the rules are data are unified and are treated as simply transitions. End of proof.

This is surprising: The teacher—the physical world including humans—does not need to tell rules from data. Unlike a UTM whose programs are handcrafted by humans, an emergent UTM learns programs directly from the physical world.

6 GENISAMA: Eight Requirements

The meanings of the acronym GENISAMA are as follows:

Grounded: All patterns $\mathbf{z} \in Z$ and $\mathbf{x} \in X$ are from the external environment (i.e., the body and the extra-body world), not from any symbolic Turing Machine tape.

Emergent: All patterns $\mathbf{z} \in Z$ and $\mathbf{x} \in X$ emerge from activities (e.g., images). All hidden vectors $\mathbf{y} \in Y$ emerge automatically from $\mathbf{z} \in Z$ and $\mathbf{x} \in X$.

Natural: All patterns $\mathbf{z} \in Z$ and $\mathbf{x} \in X$ are natural from real sensors and real effectors, without using any task-specific encoding used by all prior networks of FA and TM.

Incremental: The machine incrementally updates at times $t = 1, 2, ...$ Namely DN uses $(\mathbf{z}(t), \mathbf{x}(t))$ for updating itself and discards $(\mathbf{z}(t), \mathbf{x}(t))$ before taking the next $(\mathbf{z}(t+1), \mathbf{x}(t+1))$. We avoid storing images for offline batch training (e.g., as in ImageNet) because the next image $\mathbf{x}(t+1)$ is unavailable without first generating and executing the agent action $\mathbf{z}(t)$ which typically alters the scene that determines $\mathbf{x}(t+1)$. Batch data sets are not action-dependent.

Skulled: As the skull closes the brain to the environment, everything inside the Y area (hidden brain) are initialized at $t = 0$ and off limit to environment's direct manipulation.

Attentive: In every cluttered sensory image $\mathbf{x} \in X$ only the attended parts correspond to the current attended symbol set s. New here is the attention to cluttered motor image $\mathbf{z} \in Z$ so that the attended parts correspond to the current state symbol q (e.g., firing muscle neurons in the mouth and arms). Two symbols correspond to a pattern (not necessarily connected, as in $s = \{\text{car2}, \text{pedestrian1}\}$). Note: The attention here for $\mathbf{x}$ is about the cluttered sensory world, consistent with the literature [Moran and Desimone, 1985; Olshausen *et al.*, 1993] but the attention in [Graves *et al.*, 2014; Graves *et al.*, 2016] is about the structured internal memory instead, different from the neuroscience literature.

Motivated: Different neural transmitters have different motivational effects, e.g., resulting in (a) avoiding pains, (b) seeking pleasures, (c) speeding up learning of important events that cause pain and pleasure, and (d) uncertainty- and novelty-based neuronal connections (synaptic maintenance for auto-wiring) and behaviors (e.g., curiosity).

Abstractive: Each learned concept (e.g., object type, location, scale, relation, goal, intent) in Z are abstracted from concrete examples in $\mathbf{z} \in Z$ and $\mathbf{x} \in X$, invariant to other concepts learned in Z. E.g., the location (in image) is invariant to type and scale; the type concept "dog" is invariant to location and scale. Let us note that invariance is different from correlation: dog-type and dog-location are correlated (e.g., dogs are typically on ground).

7 Many-to-Many Mapping

Symbol-to-vector: To deal with sensing from the real world, we must deal with sensory vectors in $X = R^n$ where R is the set of real numbers and n is the dimension of sensor (e.g., the number of pixels in an image from the sensor). Similarly, we must deal with motor vectors in $Z = R^m$ where m is the number of muscles (or the number of muscle neurons). DN extends the input space Σ to become a vector sensory space X, and the state space Q to a vector motor space Z:

$$\Sigma \equiv X; \quad Q \equiv Z \tag{2}$$

where $\equiv$ means each element in the set of symbols on one side corresponds to a unique vector in the set of vectors on the other side.

Many vectors: However, one symbol may correspond to many possible vectors from a sensor of the real world. For example, many image patches of a cat at different image locations in a natural scene correspond to the same cat. Furthermore, each image of a cluttered scene may contain multiple objects each represented by a different symbol in Σ. Therefore, we extend Eq. (2) to the following two expressions:

$$\Sigma \lhd X; \quad Q \lhd Z \tag{3}$$

where $\lhd$ represents many to many correspondences, but left-hand side is a much smaller symbolic set than the right-hand side vector set.

Similarly many actions $\mathbf{z} \in Z$ (e.g., pronounced sounds for "cat" with different accents) correspond to the same symbol in Q (e.g., as a symbol "cat"). Thus, $Q \lhd Z$.

Furthermore, $Z \rhd Q$ means a many-to-many mapping, because from an image $\mathbf{z} \in Z$, $q \in Q$ depends on attentions to $\mathbf{z}$. Likewise, $X \rhd \Sigma$ means also a many-to-many mapping.

Framewise pattern recognition: Suppose X is lower (more concrete) and Z is higher (more abstract). Then, establishing a bottom-up framewise mapping

$$\delta_f : X \mapsto Q \tag{4}$$

is a *spatial pattern recognition* problem. Why?

For clarity, we first define a subvector $\mathbf{x}' \in X$ of $\mathbf{x} \in X$, denoted by $\mathbf{x}' \subseteq \mathbf{x}$, if $\mathbf{x}'$ contains only a subset of components in $\mathbf{x}$ and all the other missing components are marked as "absent" denoted by $*$. For example $(*, 1, 2, *, 0) \subset (0, 1, 2, 4, 0)$.

Because an image from a cluttered scene may contain multiple objects to be classified and each object type may have multiple instances, δ_f is only a mapping but not a function because a function must be defined on every vector $\mathbf{x} \in X$ and the value of $\delta_f(\mathbf{x})$ must be unique. Therefore, the frame-wise pattern recognition does not perform well for images from a cluttered scene, because at least the output depends on which object is attended from many objects.

Context-based pattern recognition: A context-based pattern recognition problem is to construct a mapping:

$$\delta_c : Q \times X \mapsto Q. \tag{5}$$

Many time-varying video analysis and speech recognition problems, including many Markov models, belong to this framework. The major limitation of this framework is that Q is static, not emergent.

Although the context is difficult to pre-handcraft, context-based pattern recognition framework does not deal with cluttered scenes well either because the output depends on which moving objects to attend from many moving objects.

Emergent-context emergent-input: DN, along with its embodiments Where-What Networks (WWN-1 [Ji *et al.*, 2008] to WWN-9 [Solgi and Weng, 2015]), aim at the following framework, called ECEI framework. The ECEI framework extends the symbolic set Q to an emergent vector space Z, to fully learn an emergent finite automaton that realizes the mapping:

$$\delta_{\text{ecei}} : Z \times X \mapsto Z \tag{6}$$

or, if X is also predicted,

$$\delta_{\text{ecei}} : Z \times X \mapsto Z \times X. \tag{7}$$

This framework, looks intimidating as it is highly recurrent: $Z \to Z \times X \to Z$. But, the reader will see below that it is not intimidating if we unfold the time, as time in a life.

Unfolding time: We should treat X and Z external as they can be "supervised" by the physical environment as well as "self-supervised" by the network itself. We add the internal area Y to be hidden—cannot be directly supervised by external teachers. Furthermore, we should unfold the time t and each area A as a function of time denoted by $A(t)$, and allow the network to have three areas X, Y, and Z that learns incrementally through time $t = 0, 1, 2, ...$:

$$\begin{bmatrix} X(0) \\ Y(0) \\ Z(0) \end{bmatrix} \to \begin{bmatrix} X(1) \\ Y(1) \\ Z(1) \end{bmatrix} \to \begin{bmatrix} X(2) \\ Y(2) \\ Z(2) \end{bmatrix} \to ... \tag{8}$$

where $\to$ means neurons on the left links to the neurons on the right. This is a highly recurrent network, not a black box.

8 Developmental Networks

The hidden Y area corresponds to the entire "brain". In the following, we assume the brain has a single area but it will emerge many areas and subareas.

The brain takes input from vector $(\mathbf{z}, \mathbf{x})$, not just sensory $\mathbf{x}$ but also motor $\mathbf{z}$, to produce an internal response vector $\mathbf{y}$ which represents the best match of $(\mathbf{z}, \mathbf{x})$ with one of many internally stored patterns of $(\mathbf{z}, \mathbf{x})$:

The winner-take-all learning rule, which is highly nonlinear and simulates parallel lateral inhibition in the internal (hidden) area Y of DN is sufficient to prove in [Weng, 2015] that a DN that has sufficient hidden neurons learns any Turing Machine (TM) perfectly, immediately, and error-free.

The n neurons in Y give a response vector $\mathbf{y} = (y_1, y_2, ...y_n)$ of n neurons in which only the best-matched neuron fires at value 1 and all other neurons do not fire giving value 0:

$$y_j = \begin{cases} 1 & \text{if } j = \operatorname*{argmax}_{1 \le i \le n} \{ f(\mathbf{t}_i, \mathbf{z}, \mathbf{b}_i, \mathbf{x}) \} \\ 0 & \text{otherwise} \end{cases} \qquad j = 1, 2, ...n, \tag{9}$$

where f is a function that measures the similarity between the top-down weight vector $\mathbf{t}_i$ and the top-down input vector $\mathbf{z}$ as well as the similarity between the bottom-up weight vector $\mathbf{b}_i$ and the bottom-up input vector $\mathbf{x}$. The value of similarity is the inner product of their length-normalized versions [Weng, 2015]. Corresponding to FA, both the top-down weight and the bottom-up weight must match well for f to give a high value as inner product.

The response vector $\mathbf{y}$ the hidden Y area of DN is then used by Z and X areas to predict the next $\mathbf{z}$ and $\mathbf{x}$ respectively in discrete time:

$$\begin{bmatrix} \mathbf{z} \\ \mathbf{x} \end{bmatrix} \to \mathbf{y} \to \begin{bmatrix} \mathbf{z}' \\ \mathbf{x}' \end{bmatrix} \tag{10}$$

where $\to$ denotes the update on the left side using the left side as input. The first $\to$ above is highly nonlinear because of the top-1 competition so that only one Y neuron fires (i.e., exactly one component in binary $\mathbf{y}$ is 1). The second $\to$ consists of simply links from the single firing Y neurons to all firing neurons on the right side.

The expression in Eq. (10), is extremely rich as illustrated in Fig. 1: Self-wiring within a Developmental Network (DN) as the control of GENISAMA TM, based on statistics of activities through "lifetime", without any central controller, Master Map, handcrafted features, and convolution. (a) Each feature neuron has six fields in general. S: Sensory; M: motoric; L: lateral; R: receptive; E: effective; F: field. But simulated neurons in X do not have Sensory Receptive Field (SRF) and Sensory Effective Field (SEF) because they only effect Y and those in Z do not have Motor Receptive Field (MRF) and Motoric Effective Field (MEF) because they only receive from Y. (b) The resulting self-wired architecture of DN with Occipital, Temporal, Parietal, and Frontal lobes. Regulated by a general-purpose Developmental Program (DP), the DN self-wires by "living" in the physical world. The X and Z areas are supervised by physics, including self, teachers, and other physical events. Through the synaptic maintenance, some Y neurons gradually lost their early connections (dashed lines) with X (Z) areas and become "later" (early) Y areas. In the (later) Parietal and Temporal lobes, some neurons further gradually lost their connections with the (early) Occipital area and become

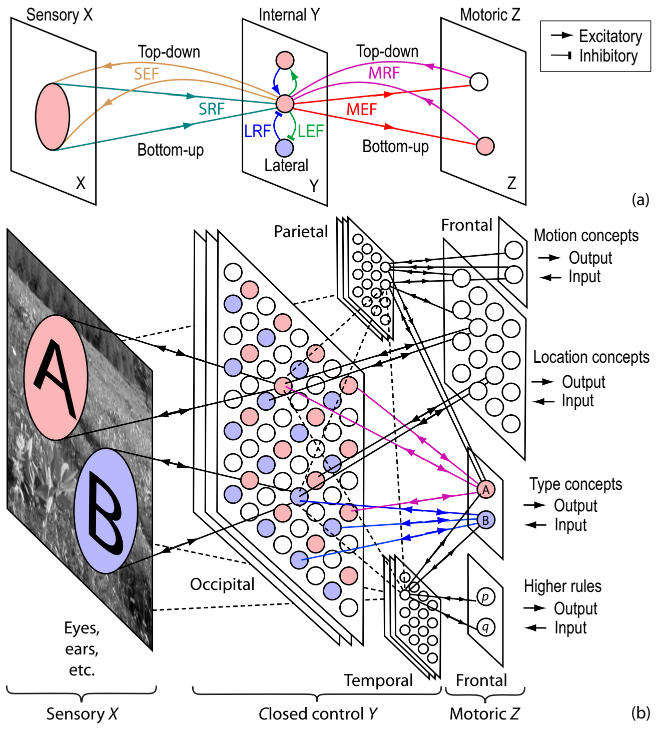

Figure 1: Brain Y is theoretically modeled as the two-way bridge of the sensory bank X and the motor bank Z.

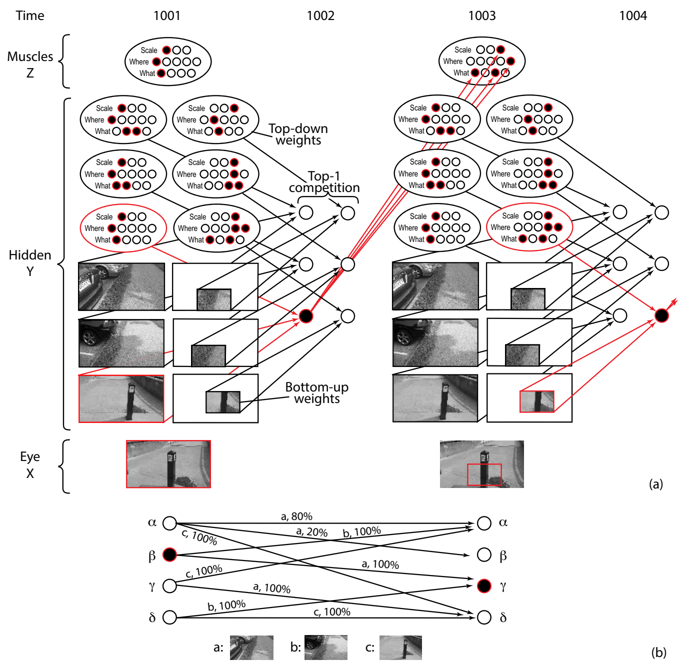

Figure 2: Put intuitively, a DN (a) has top-down weights as context for attention but a traditional neural networks in (b) does not. This DN incrementally learns a finite automaton that is the control of a Universal Turing Machine — amounting to auto-programming for general purposes.

rule-like neurons. These self-wired connections give rise to a complex dynamic network, with shallow and deep connections instead of a deep cascade of areas. Object location and motion are non-declarative concepts and object type and language sequence are declarative concepts. Concepts and rules are abstract with the desired specificities and invariances. DN does not have any static Brodmann areas.

Like the transition function of the FA in Eq. (1), each prediction of $\mathbf{z}'$ in Eq. (10) is called a *transition*. but now in real-valued vector, without any symbols. The same $\mathbf{y}$ can also be used to predict the binary (or real-valued) $\mathbf{x}' \in X$ in Eq. (10). The quality of prediction of $(\mathbf{z}', \mathbf{x}')$ depends on how state $\mathbf{z}$ abstracts the external world sensed by $\mathbf{x}$. The more mature the DN is in its "lifetime"' learning, the better its predictions.

The above vector formalization is simple but very powerful in practice. Illustrated in Fig. 2(a), the pattern in Z can represent the binary pattern of any abstract concept — context, state, muscles, action, intent, object type, object group, object relation. However, as far as DN is concerned, they mean the same— a firing pattern of the Z area!

Namely, unified numerical processing-and-prediction in DN amounts to any abstract concepts above. In symbolic representations, it is a human to handcraft every abstract concept as a symbol; but DN does not have a human in the "skull". it simply learns, processes, and generates vectors. In the eyes of a human outside the "skull", the DN gets smarter and smarter.

Consider learning. Suppose human society together with mother nature as a teacher is a huge and complex TM, including a Universal Turing Machine.(UTM) Because its control is an FA, represented by a huge FA transition table having r rows and c columns. At each time t, $t = 1, 2, ...,$ only the winner Y neuron fires at response value 1 and incrementally updates its weight vector $(\mathbf{z}_i, \mathbf{x}_i)$ as the vector average of attended part of $(\mathbf{z}, \mathbf{x})$. This is called the incremental Hebbian learning rule. Then, the i-th Y neuron memorizes perfectly the i-th distinct input pair $(\mathbf{z}, \mathbf{x})$ observed in life, because the teacher TM has no errors. When the teacher has errors the DN is optimal in the sense of maximum likelihood, as proved in [Weng, 2015].

The Z area was taught the next (binary) response vector $\mathbf{z}'$ using the same incremental Hebbian learning rule.

Suppose there are at most rc transitions in TM. The FA needs at most rc hidden neurons to perfectly learns the TM. Because a universal TM is a TM, a DN with sufficient hidden neurons can learn any university TM perfectly, immediately and error-free.

Because any universal TM is a general purpose computer, so is the corresponding DN. However, DN is emergent (i.e., skull closed) by automatically learning from the real physical world. In contrast, a universal TM is human handcrafted. Therefore, we have established in theory that a DN can automatically learn from the real physical world to gradually become a general purpose machine that auto-programs.

In the AIML Contest 2016, we let each DN to take one of three sensory modalities while it learns in "lifetime": We let the $\mathbf{x}$ be the image at each time instance and $\mathbf{z}$ be landmark location-and-type and action of navigation, the DN became a vision-guided navigation machine (see Fig. 3). We let $\mathbf{x}$ be the frame of firing pattern of hair cells in cochlea at each

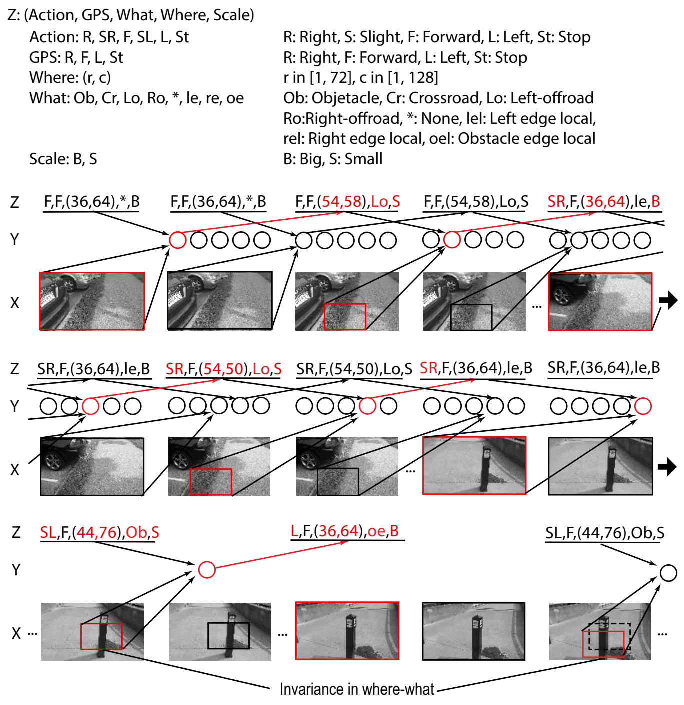
Z: (Action, GPS, What, Where, Scale)
Action: R, SR, F, SL, L, St
GPS: R, F, L, St
Where: (r, c)
What: Ob, Cr, Lo, Ro, *, le, re, oe
Scale: B, S
R: Right, S: Slight, F: Forward, L: Left, St: Stop
R: Right, F: Forward, L: Left, St: Stop
r in [1, 72], c in [1, 128]
Ob: Objetacle, Cr: Crossroad, Lo: Left-offroad
Ro:Right-offroad, *: None, lel: Left edge local,
rel: Right edge local, oel: Obstacle edge local
B: Big, S: Small
Z
Y
X
F,F,(36,64),*,B
F,F,(36,64),*,B
F,F,(54,58),Lo,S
F,F,(54,58),Lo,S
SR,F,(36,64),le,B
SR,F,(36,64),le,B
SR,F,(54,50),Lo,S
SR,F,(54,50),Lo,S
SR,F,(36,64),le,B
SR,F,(36,64),le,B
SL,F,(44,76),Ob,S
L,F,(36,64),oe,B
SL,F,(44,76),Ob,S
Invariance in where-what

Figure 3: A task-specific and modality-specific example of how a task-nonspecific and modality-nonspecific engine learns. A DN learns concepts like where, what, scale and navigation actions, as a motor vector in Z (for text), while learning an attention sequence that allows a context-matched patch in the input image to win. Any arbitrary attention sequence can be learned in a similar way. Namely, the DN has its own attention skill that is learned from past experience, a major advance in computer vision in terms of learned top-down attention. The text can be in natural language.

time instance and $\mathbf{z}$ be the dense stages and the sparse type of sounds, the DN became a auditory-recognizer machine. We let $\mathbf{x}$ be a time frame of vector of word (either English or French) and $\mathbf{z}$ be the language kind and meaning of each sentence context, the DN became a bilingual language learner. Thus, AIML Contest is the first contest that demonstrated task-nonspecific and modality-nonspecificity.

9 Theorems

[Weng, 2015] has proved: (1) The control of a TM is an FA. Thus, the emergent FA can learn a emergent UTM for APFGP. (2) The DN is always optimal in the sense of maximum likelihood conditioned on the number of neurons and the learning experience. When there are neurons in the hidden brain to be initialized, the learning is further error-free.

10 Motivation

Motivation is very rich. It has two major aspects (a) and (b) in the current DN model. All reinforcement learning methods other than DN, as far as we know, are for symbolic methods (e.g., Q-learning [Sutton and Barto, 1998; Mnih *et al.*, 2015]) and are in aspect (a) exclusively. DN uses concepts (e.g., important events) instead of the rigid time-discount in Q-learning to avoid the failure of far goals.

(a) Pain avoidance and pleasure seeking to speed up learning important events. Signals from pain (aversive) sensors release a special kind of neural transmitters (e.g., serotonin [Daw *et al.*, 2002]) that diffuse into all neurons that suppress Z firing neurons but speed up the learning rates of the firing Y neurons. Signals from sweet (appetitive) sensors release a special kind of neural transmitters (e.g., dopamine [Kakade and Dayan, 2002]) that diffuse into all neurons that excite Z firing neurons but also speed up the learning rates of the firing Y neurons. Higher pains (e.g., loss of loved ones and jealousy) and higher pleasure (e.g., praises and respects) develop at later ages from lower pains and pleasures, respectively.

(b) Synaptic maintenance—grow and trim the spines of synapses [Wang *et al.*, 2011; Guo *et al.*, 2015]—to segment object/event and motivate curiosity. Each synapse incrementally estimates the average error β between the pre-synaptic signal and the synaptic conductance (weight), represented by a kind of neural transmitter (e.g., acetylcholine [Yu and Dayan, 2005]). Each neuron estimates the average deviation $\bar{\beta}$ as the average across all its synapses. The ratio $\beta/\bar{\beta}$ is the novelty represented by a kind of neural transmitters (e.g., norepinephrine, [Yu and Dayan, 2005]) at each synapse. The synaptogenic factor $f(\beta, \bar{\beta})$ at each synaptic spine and full synapse enables the spine to grow if the ratio is low (1.0 as default) and to shrink if the ratio is high (1.5 as default). See Fig. 1(b) for how a neuron can cut off their direct connections with Z to become early areas in the occipital lobe or their direct connections with the X areas to become latter areas inside the parietal and temporal lobes. However, we cannot guarantee that such "cut off" are 100% based on the statistics-based wiring theory here.

11 Experiments

The experimental results [Weng *et al.*, 2018] of DN include (1) vision that includes simultaneous recognition and detection and vision-guided navigation on MSU campus walkways, (2) Audition to learn phonemes with a cochlea, and (3) acquisition of English and French in an interactive bilingual environment. Because of the maximum likelihood optimality in Sec 9, there are no local minima problems for these highly nonlinear systems and the performance data were impressive for the three large data sets [Weng *et al.*, 2018].

12 Conclusions

"Data" and "Rule" are treated in a unified way by DN. Although surprising, this is necessary for Auto-Programming for General Purposes. These results have been proved mathematically and experimentally. However, much remains to be done. Full GENISAMA on a system has yet to be realized.

References

[Daw *et al.*, 2002] N. D. Daw, S. Kakade, and P. Dayan. Opponent interactions between serotonin and dopamine. *Neural Networks*, 15(4-6):603–616, 2002.

[Fei-Fei, 2005] L. Fei-Fei. Visual recognition: Computational models and human psychophysics. Technical Report PhD thesis, California Institute of Technology, Pasadena, California, 2005.

[Fukushima, 1980] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[Graves *et al.*, 2014] A. Graves, G. Wayne, and I. Danihelka. Neural Turing machines. Technical report, Google DeepMind, London, UK, Dec. 10, 2014. arXiv:1410.5401.

[Graves *et al.*, 2016] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwinska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, A. P. Badia, K. M. Hermann, Y. Zwols, G. Ostrovski, A. Cain, H. King, C. Summerfield, P. Blunsom, K. Kavukcuoglu, and D. Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476, 2016.

[Guo *et al.*, 2015] Q. Guo, X. Wu, and J. Weng. Cross-domain and within-domain synaptic maintenance for autonomous development of visual areas. In *Proc. the Fifth Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics*, pages +1–6, Providence, RI, August 13-16 2015.

[Ji *et al.*, 2008] 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*, pages 61–66, Monterey, CA, Aug. 9-12, 2008.

[Jordan and Mitchell, 2015] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349:255–260, July 17 2015.

[Kakade and Dayan, 2002] S. Kakade and P. Dayan. Dopamine: generalization and bonuses. *Neural Network*, 15:549–559, 2002.

[Krizhevsky *et al.*, 2012] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.

[LeCun *et al.*, 2015] Y. LeCun, L. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[Mnih *et al.*, 2015] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[Moran and Desimone, 1985] J. Moran and R. Desimone. Selective attention gates visual processing in the extrastrate cortex. *Science*, 229(4715):782–784, 1985.

[Olshausen *et al.*, 1993] B. A. Olshausen, C. H. Anderson, and D. C. Van Essen. A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *Journal of Neuroscience*, 13(11):4700–4719, 1993.

[Riesenhuber and Poggio, 1999] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–1025, 1999.

[Schmidhuber, 2015] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[Serre *et al.*, 2007] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(3):411–426, 2007.

[Solgi and Weng, 2015] M. Solgi and J. Weng. WWN-8: Incremental online stereo with shape-from-x using life-long big data from multiple modalities. In *Proc. INNS Conference on Big Data*, pages 316–326, San Francisco, CA, August 8-10, 2015.

[Sun *et al.*, 2005] R. Sun, P. Slusarz, and C. Terry. The interaction of the explicit and the implicit in skill learning: A dual-process approach. *Psychological Review*, 112(1):59–192, 2005.

[Sutton and Barto, 1998] R. S. Sutton and A. Barto. *Reinforcement Learning*. MIT Press, Cambridge, Massachusetts, 1998.

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

[Weng *et al.*, 1993] J. Weng, N. Ahuja, and T. S. Huang. Learning recognition and segmentation of 3-D objects from 2-D images. In *Proc. IEEE 4th Int'l Conf. Computer Vision*, pages 121–128, May 1993.

[Weng *et al.*, 1997] J. Weng, N. Ahuja, and T. S. Huang. Learning recognition and segmentation using the Cresceptron. *International Journal of Computer Vision*, 25(2):109–143, Nov. 1997.

[Weng *et al.*, 2001] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, 291(5504):599–600, 2001.

[Weng *et al.*, 2018] J. Weng, Z. Zheng, X. Wu, J. Castro-Garcia, S. Zhu, Q. Guo, and X. Wu. Emergent Turing machines and operating systems for brain-like auto-programming for general purposes. In *Proc. AAAI 2018 Fall Symposium: Gathering for Artificial Intelligence and Natural Systems*, pages 1–7, Arlington, Virginia, October 18 - 20 2018.

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

[Weng, 2015] J. Weng. Brain as an emergent finite automaton: A theory and three theorems. *International Journal of Intelligent Science*, 5(2):112–131, 2015. received Nov. 3, 2014 and accepted by Dec. 5, 2014.

[Yu and Dayan, 2005] A. J. Yu and P. Dayan. Uncertainty, neuromodulation, and attention. *Neuron*, 46:681–692, 2005.