Muscle Vectors as Temporally Dense "Labels"

Xiang Wu[1] and Juyang Weng[2,3,4]

[1]School of Automation
Nanjing University of Science and Technology, Nanjing, 210094, China
[2]Department of Computer Science and Engineering
[3]Cognitive Science Program
[4]Neuroscience Program
Michigan State University, East Lansing, MI, 48824 USA

***Abstract*—Consider a human who interacts with the physical world to autonomously learn in a task non-specific way through lifetime. It seems obvious that the fully autonomous learner does not have the luxury to have the mother to provide temporally dense state labels, but the context/state at every frame is beneficial (e.g., to generate attention for the next frame). How can we enable the learner to generate frame-wise contexts/states on the fly? Our past work on Developmental Network (DN-1) has shown that frame-wise state labels (e.g., stages within a phoneme) are useful to generate temporally sparse label (the type of the phoneme). However, such dense and sparse labels were handcrafted from a static data set, using human identified frame-wise equivalence. In this paper, we study a conceptually challenging problem — how to enable an autonomous learner to generate frame-wise states autonomously without human handcrafting dense labels at all. We propose that frame-wise muscle actions (e.g., producing a sound) are not only temporally dense and high-dimensional, but also natural as dense labels. However, it is unknown how a neural network can use such high-dimensional vectors as dense labels. In this work, we provide a model for this new issue and experiment with Developmental Network-2 (DN-2) for imitation of audio sequences. Our experimental results showed DN-2 can successfully emerge high-dimensional real-valued vector actions. These actions provide DN-2 with frame-wise temporal context information. This work corresponds to a key step toward our goal to enable the agent to fully autonomously generate frame-wise actions without human-provided dense labels and with only a few human-provided sparse labels.**

I. Introduction

Many research works in machine learning have been fruitfully inspired by studies of human learning. However, most of these traditional learning techniques differ from human mental development. Although they aim at providing a general framework for learning and development in some aspects, these methods do not perform autonomous development. Their computational frameworks (including rigid symbolic nodes and boundaries) are handcrafted based on the human designer's understanding of a given task at hand. In contrast, the autonomous development paradigm [1] requires that the developmental program (like functions of the genome) is decided before given any specific tasks to learn so that incremental learning takes places across an open array of tasks, skills acquired during early tasks assisting the later learning of later tasks (called scaffolding in developmental psychology [2]). As summarized in [3], there are five essential factors determine the capability of the agent in the autonomous development paradigm:

1) the sensors,
2) the effectors,
3) the computational resources,
4) the developmental program (genome),
5) the environment agent experienced.

This work focuses on 2) effectors and 5) the environment that facilitates the agent's autonomous development. In particular, we study how the actions emerged from the effectors not only directly perform in the environment but also provide context/state information to facilitate the development of the agent's behaviors.

The actions we discuss in this work correspond to real-valued and frame-wise muscle vectors. They can correspond to two types of skills [4], declarative skills (e.g., telling a story) and non-declarative skills (e.g., bike riding). The declarative skills are usually learned in classification or recognition tasks, while the non-declarative skills are often learned in robotic navigation and manipulative tasks. During learning, the actions carried out by the agent are for both declarative and non-declarative skills.

Although we used phonemes as experimental corpuses in this work, our theory and simulations on bilingual natural language acquisition [5] have shown that the frame-wise action vectors here have the potential to autonomously develop "higher" or more abstract labels for longer sequences, like in natural language acquisition. However, due to limited space, the scope of this work does not include natural languages.

A. From symbolic to emergent representations

In recent years, much effort has been focused on temporal processing problems. Addressing these problems depend on both spatial contents from the current sensory inputs and the relevant context from the attended past. Often (e.g., during robotic navigation), the agent not only acts according to the current sensory inputs but also the recent dynamic history of the agent and the environment. By recent dynamic history, we mean the state/context of the spatiotemporal context. In the remainder of this paper, state, context, action are interchangeably equivalent, unless explicitly stated otherwise.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

Hidden Markov Models (HMMs) based methods are based on symbolic states. The set of states is often formed through k-means clustering on a static set of sensory data. Although the raw data or feature data (e.g., MFCC) are vectors, the output from the clustering algorithm is symbolic. To alleviate inconsistency between state transitions, these HMM based methods use probabilities [6]. HMMs are often cascaded to construct layered probabilistic representations for the data with a hierarchical structure. In [7], the hierarchical HMMs model is used for a mobile robot to learn to track another robot's motion by observation.

The Bayesian Networks, also symbolic, can explicitly represent the causal relation since the conditioning variables are parent nodes, and dependent nodes are child nodes inside. And the links between the nodes are from causal to dependent variables. The Dynamic Bayesian Networks, which is from Bayesian Network, models the causality and variations across time series. At each time slice there exists a Bayesian Network in which parent of the current node at time t is a node at time $t-1$. The Dynamic Bayesian Networks was used to estimate the face pose from video sequences in real-time in [8].

The above computational models deal satisfactorily with high-level discrete concepts since their symbolic representations are carefully and statically designed by a human based on a static data set and a given task. Compared with these symbolic representations, the emergent representations in our work are fully grounded, natural, and fault-tolerant. By natural, we mean that the sensory vectors and motor vectors for our emergent representation are developed from raw and natural sensors (e.g., microphones) and effectors (e.g., speakers). Emergent patterns from the same sensors or effectors have distances in the neuronal feature spaces, but in the symbolic representations two symbols are treated either same or different — distances between data are lost along with the rich relations among raw sensory data and motor data. With the emergent representations, never observed patterns can be naturally processed according to their distances with observed emergent patterns. Furthermore, emergent representations do not require a human in the loop of handcrafting, suited for open-ended autonomous development from earlier simple tasks to later more complicated tasks.

Many neural networks use *emergent* representations, at least partially. The Hopfield network is a kind of recurrent network in which all the notes are binary threshold units. These units also give feedback to other units as the influence of "memory". The work in [9] utilized the Hopfield network to do simple facial expression recognition. In [10], a memristive Hopfield network was constructed to demonstrate that different patterns can be stored and retrieved successfully like associative memory behaviors.

The deep neural networks have a variable structure with the flexible connections among units in temporal trajectories. Emergent patterns are often extracted from the sensory end and mixed with symbolic representations in the higher layers. The Recurrent neural network (RNN) is meant to deal with sequences since it includes the connections along temporal trajectories. In [11], an RNN tree that contains multiple RNNs in a tree hierarchy was established to recognize the human actions from video sequences. Recently, along with Convolutional Neural Networks (CNNs) for spatial problems, Long Short-Term Memory (LSTM) RNN [12], [13] became popular in experimental studies for temporal problems as they detect latent temporal dependencies.

Networks like Hopfield Networks, CNN and LSTM partially use emergent representations. These methods are meant for classification of sensory data since their emergent representations are extracted directly from the sensory domain.

B. Developmental methods

All above methods do not learn an emergent Finite Automata (FA). In particular, they do not directly take patterns from the motor ends as contexts/states/actions, which means they do not use the concept of spatiotemporal states during processing. However, current temporal processing tasks may contain many different scenarios or involve sequential data of multiple modalities. The developmental framework which follows the autonomous development paradigm is needed to handle more complex temporal processing with frame-wise states that the framework of FA and Turing Machine (TM) entails.

Besides using emergent representations for FA and TM, the developmental methods are task non-specific and use strictly frame-wise incremental learning. Task non-specific learning allows the agent to develop through different tasks with emergent behaviors. Incremental learning is crucial so that the next sensory frame can depend on the current action — sensorimotor recursive — and learning takes place on the fly.

DN-1 is a biologically plausible developmental model of a simplified brain, and is a typical developmental method [3]. By extracting emergent patterns from sensorimotor domains, DN-1 incrementally develops through the sequence of inputs and outputs. The emergent patterns used in motor area can directly self-supervise DN-1 so that learning can take place with human teachers or without.

DN-1 has been successfully experimented with using visual modality [14], auditory modality [15], and natural language acquisition [5] using text as inputs. Although DN-1 has been tested with multiple hidden areas, there are two restrictions. (A) the boundary of each area is fixed in the sense that the number of neurons assigned to each area is fixed. (B) there are no excitatory connections between two neurons in the same hidden area. Namely, although a DN-1 can learn any Universal TM so that it has been proved that DN-1 can perform Autonomous Programming For General Purposes (APFGP) [16], its generalization power from view examples is limited by these two restrictions (A) and (B).

To address these issues, DN-2 [17] was proposed based on DN-1 by adding several new mechanisms. With new mechanisms, each neuron in DN-2 automatically develops its excitatory connections and its own inhibition zone. So the number of areas and their inter-connection relations are automatically determined by the learning experience instead of pre-

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

handcrafted. DN-2 can generate more flexible and hierarchical inner representations to abstract concrete examples better.

This work investigates how the agent automatically emerges actions for every time frame. We also analyze how these emergent frame-wise actions can liberate humans from the tedious labeling work and may help the model to emerge some sparse labels as high-level concepts. We regard actions and states are the same, and include both declarative skills and non-declarative skills.

In this paper, the automatically generable actions in our experiments are temporally dense (i.e., frame-wise for each time frame of 20ms length) and high-dimensional patterns – raw phoneme waveforms (e.g., from a speaker). This work seems to demonstrate, for the first time, that it is algorithmically possible for a machine to learn fully autonomously without a human to feed any labels to the effector end.

The remainder of the paper is organized as follows: the theory is discussed in Section II. The new mechanisms and the algorithm of DN-2 are presented in Section III. The experimental steps and results are reported in Section IV. Section V provides concluding remarks.

II. Theory

Let us first discuss what it means by DN-1 acting on temporal sequences.

A. Equivalent classes

Weng 2015 [18] proved that the control of a Turing Machine is an FA. The main ideas are: (a) Allow FA to output states so the resulting FA is called agent FA. (b) Expand the definition of the states of FA to include further agent actions — writing symbols of the Turing Machine and head motions of the Turing Machine. Therefore, although we will use actions as examples below, these actions should be considered states, writing symbols, and head motions of any Turing Machine.

A Universal Turing Machine is a general-purpose computer model of all modern digital computers. Its input has two parts, a program and the data for the program. The Universal Turing Machine simulates the program on the data. Because the program (e.g., the knowledge of a teacher) can be for any purpose, the Universal Turing Machine can be programmed for any practical purposes. Therefore, although we will use sensory inputs as examples below, these sensory inputs should be considered not just data, but also instructions of any program for any purpose. This makes sense, as for example, an auditory sensory sequence can represent rules (e.g., sound from reading a textbook).

Consider the input space of DN-1 to be $X = R^l$, where R is the set of real numbers and X consists of real-valued vectors of dimension l (e.g., images each with l pixels or sound frames each of l dimensionality). Consider the motor space of DN-1 to be $Z = R^n$, where Z consists of real-valued vectors of dimension n. For example, $l = n = 882$ for each time frame in our sound experiments: The leaner voices sound and hears sound. The "brain" neurons in DN-1 are hidden and all hidden neurons form the hidden area Y of DN-1.

According to the DN-1 theory [18], a DN-1 learns an Emergent Turing Machine, where the motor area Z corresponds to the set of state vectors of the FA controller of the Turing Machine; X corresponds to the set of input vectors of the FA. The number of neurons in DN-1 corresponds to the entries of FA that have been observed and learned. Because the number m of (hidden) Y neurons in DN-1 is finite, what does the DN-1 does for the inputs in X and the outputs in Z? Next, we use the theory of FA to reach the following new theorem. The same theorem is true for DN-2. We simply state DN, which applies to both DN-1 and DN-2.

Theorem 1 (DN equivalent classes): Suppose X and Z are represented by finite resolution real-valued vectors as in digital computers. The DN with a finite number of neurons groups all input sequences in X into a finite number of equivalent classes, meaning that all sequences from X produce the same output vector in motor space Z. Further, there is a minimum-state DN that has the fewest states among all functionally equivalent DNs. This minimum-state DN corresponds to the most-coarse partition of the sensory space X.

Proof: According to the proof of Theorem 1 in [18], using inner product distance metrics in real-valued joint space (X, Z), DN partitions the real-valued space X into a large number of equivalent classes per automata theory [19]. All vector sequences (not individual vectors) in X that belong to the same equivalent class generate the same output vector in Z since Z has a finite resolution in digital computers and, therefore, Z contains only a finite number of states. The minimum-state DN corresponds to the corresponding minimum-state FA. ■

Each sequence in X is like a spatiotemporal experience from the time the agent opens his eye in the morning of a day. Two sequences $\mathbf{x}_1$ and $\mathbf{x}_2$ belong to the same equivalent class means that $\mathbf{x}_1$ and $\mathbf{x}_2$ result in the same state/action $\mathbf{z} \in Z$. A non-minimum-state DN is like a colleague who requires a larger brain or consumes more hidden neurons to reach the same behavior performance than the minimum-state DN. Obviously, whether a DN is a minimum-state DN depends on all five essential factors in Section I.

B. Action encoding

We used the phoneme recognition experiment to demonstrate that temporally dense actions have higher entropy and can provide more context information for DN-1 to learn [15].

Let us consider how to enable the DN-2 to learn to speak by listening. Because the agent must perform continuous actions (i.e., say sound waves), we must consider how the agent learns to produce such temporally dense and high-dimensional vectors.

A new method for generating high-dimensional vectors as action supervision is needed without using dense handcrafted labels. We need to map the sound waves to a lower-dimensional space to obtain lower-dimensional patterns so that the number of Z neurons is contained. We did not take raw sensory inputs as supervised patterns directly because the raw sound waves are of very high dimensional space. In this

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

work, we use Principal Component Analysis (PCA) [20] as an encoding method to reduce the dimension of the raw source waves as action vectors. PCA has the least mean error given the dimension of PCA vectors. Specifically, we calculate the covariance matrix of all the data frames (each data frame $\mathbf{u}_i \in R^m$). Then calculate and choose the first n eigenvectors $\mathbf{e}_j$ $(j = 1, 2, ..., n)$ based on the covariance matrix. These eigenvectors form the projecting matrix $W = \{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n\}$ $(W \in R^{m \times n})$. The projecting matrix is used for generating lower-dimensional supervised action vector: $\mathbf{a}_i = W\mathbf{u}_i$ $(\mathbf{a}_i \in R^n)$. In our phoneme imitation experiments, the frame data's dimension m is 882, and the action vector's dimension n is 40.

Some characteristics of encoding sensory inputs as actions are:

1) Encoding the sensory inputs as actions avoids handcrafted labeling work. All the sensory inputs are encoded under the same rule. The generated actions can be considered as self-supervision.
2) Encoding the sensory inputs as actions implicitly links the sensory and motor area. After learning, the agent can emerge certain actions followed by a specific sensory input.
3) Encoding the sensory inputs as actions can make the agent insensibly learn to imitation. With some high-level concepts, the agent could even realize what actions it emerges.
4) Encoding the sensory inputs as actions can also provide abundant context information to the agent. These contexts are most relevant since they are directly extracted from the sensory inputs.

When encoding the sensory inputs as actions, continuous actions emerge to form a natural behavior. Then short behaviors chain together to constitute a meaningful skill. Consider the example of language acquisition in the early years. Babies start with babbling and only pronounce simple phonemes. Later, they learn words from phonemes and sentences from words. The work in [21] studied the robot's learning of speech production based on sensory context by integrating the high dimensional action space and the high dimensional context space.

As shown in Fig. 1, phoneme-level auditory streams are fed to DN-2 frame by frame as well as the corresponding actions. In our experiments below, the streams are from 5 vowels of English. Each frame spans 20ms long. During learning, DN-2 dynamically clusters to generate the dense action patterns according to the sensorimotor inputs. As intermediate states, these actions assist the agent to generate required spatiotemporal patterns leading to a successful pronunciation of the vowel.

Besides forming a natural behavior, the fine-grained level emerged actions can also assist the agent to learn some abstractive high-level concepts. In this case, DN-2 only need sparse supervision for those high-level concepts or even just some sparse reward or punishment signals (in reinforcement learning mode). These fine-grained level emerged actions could be intermediate states for "thinking". By "thinking", we mean that the emerged actions do not necessarily drive actual muscle contraction if the action pattern is weak. "Thinking" is useful for the mature DN-2 to deal with the delayed rewards.

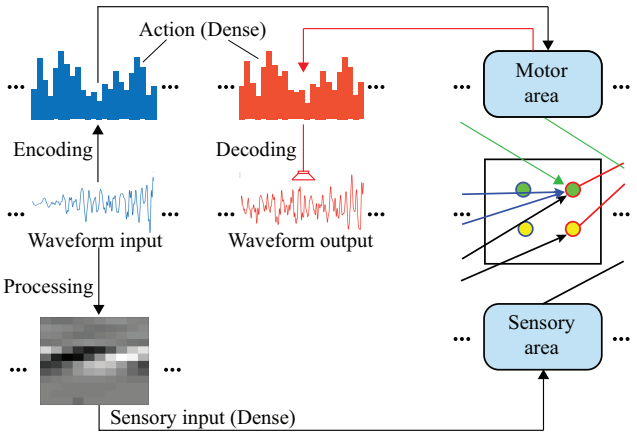

Fig. 1. During learning, the environment provides sensory input (dense) directly to the sensory area of DN-2. The sensory input (dense) is also encoded to desirable action (dense) for the motor area of DN-2. The sensory input and generated action are fed to DN-2 in each frame as the context-input pair. The DN-2 incrementally learns one frame at a time. After learning, DN-2 can emerge action to produce the waveform as output.

Let us take using DN-2 for phoneme imitation and recognition together as an example. DN-2 insensibly learns to imitate the phoneme based on the sensory inputs and corresponding contexts provided by the emerged actions. During this procedure, the sparse phoneme class label can be offered to DN-2 to link its imitation with the phoneme class concept. The supervision time is important since there are some special situations (e.g., time warping and time duration) that need to be carefully considered.

III. Developmental Network-2

DN-2 is developed from DN-1. It inherits the emergent representations, incremental learning and general-purpose learning framework from DN-1. DN-2 also follows the finite automaton logic (by extending more updates in Y area across time), and can directly perceive information from self-generated motor patterns. In DN-2, the neurons' learning follows the Lobe Component Analysis (LCA) algorithm [22].

DN-2 shares the same basic structure with DN-1. It contains three areas — X, Y and Z. X is the sensory input area which extracts the sensory information. Z is the motor area which receives supervisions or generates actions. Y is the hidden area which is "skull-closed". By "skull-closed", we mean it cannot be accessed for any direct manipulation after birth. The Y area bi-directional connects with X and Z areas to learn context-input features.

A. New mechanisms

Several biology-inspired mechanisms are included to help DN-2 to process sequences more efficiently. We just briefly introduce these new mechanisms here. The biology and neu-

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

roscience studies which support these mechanisms are listed in our previous work [23].

The Y neurons with different connections: Y neurons with different connections are initialized in DN-2. We named these neurons in the format of three binary bits according to their connection relationship with X, Y and Z areas. For each bit, "1" represents there are connections between this type of Y neuron and the corresponding area, while "0" represents no connections. For example, type 101 indicates the Y neurons having connections with the X and Z areas.

The Y neurons with different connection modes can focus on different local features more efficiently. This will help DN-2 to generate reasonable representations more quickly in the early period. In our phoneme imitation experiments, type 100 and type 111 Y neurons are mainly grown to learn different inner representations.

Local receptive fields and local inhibitions zones: The local receptive field is designed for Y neurons which have connections with X area in DN-2. The local receptive fields make these Y neurons flexibly extract different local features. The local top-k competition mechanism is used in DN-2. Each Y neuron only competes with other Y neurons which have the same connection mode. The local top-k competition ensures DN-2's inner representations contain more local features.

In the previous work [17], we take type 100 Y neurons as an example to explain how the local receptive field and refined local top-k competition mechanisms work. Type 100 Y neurons have local receptive fields with the same scale but focusing on different locations of the input domain. The refined local top-k competition zones are formed based on the locations of these neurons' receptive fields so that these neurons only compete with others having similar receptive fields. These mechanisms work together to guarantee the neuronal resources are evenly arranged for each part of the sensory regions.

Lateral connections among Y neurons: In DN-2, Y neurons having the lateral connections with other Y neurons transmit the spatiotemporal information among these neurons. When DN-2 is processing sequences, the spikes from the last frame are extracted as inputs for the neuron's lateral connections. This follows the Hebbian learning principle [24].

Synaptic maintenance mechanism is applied to the lateral connections to automatically fine-tune the connection patterns according to the statistics in the learning experience. Growing the potential connections is included in the synaptic maintenance so that DN-2 can discover new statistic dependencies. The algorithm about how to grow lateral connections is in our previous work [17]. After synaptic maintenance, the lateral connections among Y neurons record the causation relationships across time series and dynamically build the hierarchical structure inside. and shape the hierarchical structure.

B. Real value sections in motor area

In DN-2, there are several handcrafted concept zones in the motor area Z for different learning tasks. Each concept zone represents a concept class. For example, the object location and type concept zones are designed in the object detection task. This is some kind of local competition in the Z area since only the neurons in the same concept zone compete with each other to fire.

In this paper, we use the real value sections in the motor area to replace the original concept zones. Each real value section represents the real values in some range. Each firing pattern in the real value section indicates a real value. The firing patterns in all real value sections together can form a real-valued vector that may be suitable for simulating a complex signal. This kind of complex signal is the key point for driving the muscles of an agent to perform subtle and flexible actions.

In our phoneme imitation experiments, we used 40 real value sections in the motor area to emerge a 1×40 real-valued vector at each time. Each real value section contains 128 Z neurons to indicate a real value in the range from -1 to 1.

C. Work flow of DN-2

During learning, DN-2 incrementally generates optimal spatiotemporal clustering based on the incremental Hebbian learning theory. In each area, each neuron has weights to match the corresponding domain inputs. Among each type of neurons (for Y area) or each concept zone (for Z area), the local top-k competition mechanism allows k neurons with the best matches of inputs to fire and learn.

Specifically, the Y neurons receive the input pair $(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}, \mathbf{z}_{t-1})$ at time t. Each Y neuron i compute the inner-product between its weights and corresponding inputs as its response $y_{i,t}$. Among each type, the Y neurons compete according to their response values to fire. If all neurons cannot match the inputs well (the response under a threshold), a new neuron grows to learn as neuron splitting (mitosis).

At time $t+1$, the Z neurons fire according to the supervised pattern if the supervision is provided by the teacher or environment, and link the Y area's firing pattern $\mathbf{y}_t$ at time t. Without the supervision, the Z neurons complete to fire and learn based on the match with $\mathbf{y}_t$. The learning for each neuron is the incremental update of its weights and firing ages. This procedure is shown in Fig. 2.

It is demonstrated in [25] that DN-2 can incrementally learn any finite automaton (or equivalently Turing machine) one transition at a time after "birth". The Y neurons in DN-2 exactly match the current context-input vector $(\mathbf{x}_t, \mathbf{z}_t)$ and then store this pattern in their weights. And after several updates, the firing Y neurons form a pattern corresponds to an output state $\mathbf{z}_{t+n}$ $(n = 1, 2, ...)$. This procedure can be represented in the FA's transition form: $(\mathbf{x}_t, \mathbf{z}_t) \rightarrow \mathbf{z}_{t+n}$.

DN-2 uses Y neurons as clusters to approximate the samples in $Z \times X$ space. When the neuronal resource is sufficient, DN-2 learns error-free by growing Y neurons with new weight $(\mathbf{w}_x, \mathbf{w}_y, \mathbf{w}_z)$. The newborn Y neuron newly initializes a different cluster that exactly matches the new context-input sample. When the neuronal resource is limited, the Y neurons are doing optimal tessellation (in the sense of maximum likelihood) in the $Z \times X$ space.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

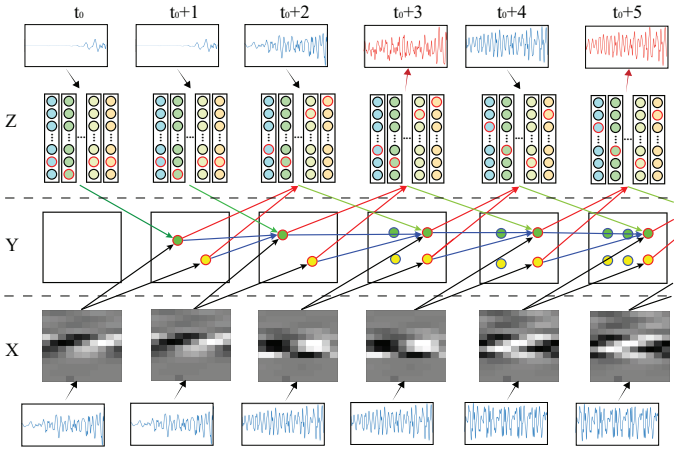

Fig. 2. The DN-2's learning procedure across time series is listed. In Y area, we just illustrate 2 types of Y neurons mainly grown in our phoneme imitation experiments. Different types of Y neurons compete to fire and learn at each moment. DN-2 incrementally initializes new Y neurons when the existing Y neurons cannot represent the context-input sample $(\mathbf{x}, \mathbf{z})$ well. The green nodes in Y area represent type 111 neurons, while the yellow nodes in Y area represent type 100 neurons. The neurons with red outlines represent firing neurons. The motor firing patterns are formed by all the real value sections. The raw waveform frames are processed into the feature patterns as sensory inputs. They are also encoded to real-valued actions as motor supervision.

D. The algorithm of DN-2

We list the outline of the DN-2 algorithm as follows.

Algorithm 1 (DN-2): Input areas: X and Z. Output areas: Z.

1) For Y area, initialize its adaptive part $N_y = (V, G)$ (where V is the synaptic weights and G is the neural ages) and response vector $\mathbf{y}$. Every Y neuron has been initialized with random weights, zero firing age and zero response as the initial state (later these Yneurons can transfer to active state). The corresponding location of every neuron is also stored. Set the total number of Y neuron to be n_y. A boundary c_y indicates the number of active neurons ($c_y \leq n_y$). Z area initializes its adaptive part N_z and the response vector $\mathbf{z}$ in similar way.
2) At time $t = 0$, supervise initial state $\mathbf{z}(t = 0)$. Input the first sensory input $\mathbf{x}(t = 0)$.
3) At time $t = 1, ...,$ repeat the following steps forever (executing steps 3a, 3b in parallel, before step 3c):
   a) All Y neurons compute in parallel:
   $$(\mathbf{y}(t), N_y') = f_y(\mathbf{p}_y, N_y) \quad (1)$$
   where $\mathbf{p}_y = (\mathbf{x}(t-1), \mathbf{y}(t-1), \mathbf{z}(t-1))$, and f_y is the Y area function to be explained below, which computes the response vector $\mathbf{y}(t)$ and updates the adaptive part N_y' of the Y area. If active Y neurons cannot match the input vector well, area Y transfers new neurons to active state. And update the boundary c_y.
   b) Components in $\mathbf{z}(t)$ are supervised if they are never fired. Otherwise, Z neurons compute Z area's response vector $\mathbf{z}(t)$ and the adaptive part N_z' in parallel:
   $$(\mathbf{z}(t), N_z') = f_z(\mathbf{p}_z, N_z) \quad (2)$$
   where $\mathbf{p}_z = (\mathbf{y}(t-1))$, and f_z is the Z area function to be explained below.
   c) Update asynchronously: $N_y \leftarrow N_y'$ and $N_z \leftarrow N_z'$. Supervise input $\mathbf{x}(t)$.

The area function f_y in Eq.(1) and area function f_z in Eq.(2) include 1) the computation of response vectors $\mathbf{y}(t)$ and $\mathbf{z}(t)$ and 2) the maintenance of adaptive parts N_y' and N_z' for Y area and Z area, respectively. The detailed steps of these area functions can be found in our previous work [17].

IV. Experiments

The experiments in this work used DN-2 for phoneme imitation through time series as the example of dense action generation. In the experiments, the dense actions are real-valued vectors which can be played as sound waveform frames. We trained DN-2 using supervised mode in this work since we want to demonstrate the feasibility of the dense action generation. We plan to teach DN-2 to concurrently imitate and recognize the phonemes later. For this task, the reinforcement learning paradigm is under our consideration.

A. Input processing and label encoding

In this work, the used audition raw data are 5 phonemes (short vowels) from the audition dataset of the 2016 Artificial Intelligent Machine Learning (AIML) Contest [26].

The lengths of these phonemes are between 210 ms to 350 ms. All the phonemes are cut into 20 ms frames. The adjacent frames have 10 ms overlap. The sampling rate of the raw waveform data is $f_s = 44.1$ KHz, so each frame has 882 elements. The training and test sequences are created by linking the 5 phoneme sequences together. There are 30 ms silence frames between the adjacent phoneme frames.

We simulated the process of the cochlea for each input frame (raw waveform data) to generate a feature pattern. Specifically, the raw waveform frame is multiplied with a series of sine functions. These sine functions have different frequencies and initial phases to simulate the hair cells at different positions in the cochlea which extract different features. For each frame, the 11×10 feature matrix is generated as well as the 1×4 volume level vector appended. The steps and explanations of this processing can be found in our previous work [15].

There are 882 real-valued components in each data frame, which is too dense to directly used as labels. We applied the PCA to the data frame to generate a real-valued vector with a lower dimension. The dimension of 40 is selected in the experiments as a tradeoff between dimensionality reduction and maintaining variance. Specifically, we use all the input frames to construct the projecting matrix W ($W \in R^{882 \times 40}$), and map the original input samples onto a lower-dimensional feature space. Later, the transposition of this projecting matrix W^T is used in the reconstructions of the waveform frames.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

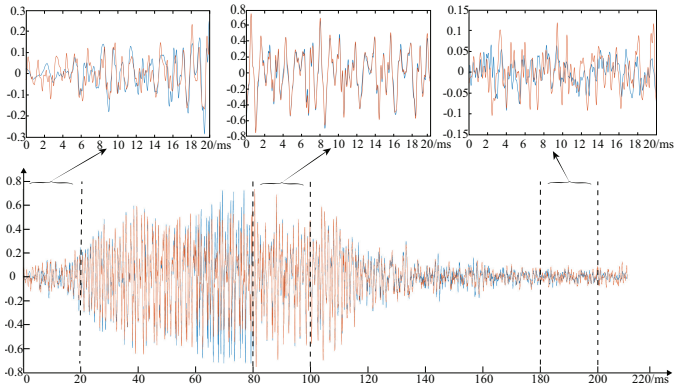

Fig. 3. The comparison of generated and original waveforms for phoneme /ʌ/ is illustrated. The blue lines represent the original input waveform, while the red lines represent DN-2's generated waveform. On the top end, the comparisons of three 20 ms segments are listed. The three segments are cut from the beginning, middle and end parts of phoneme /ʌ/, respectively.

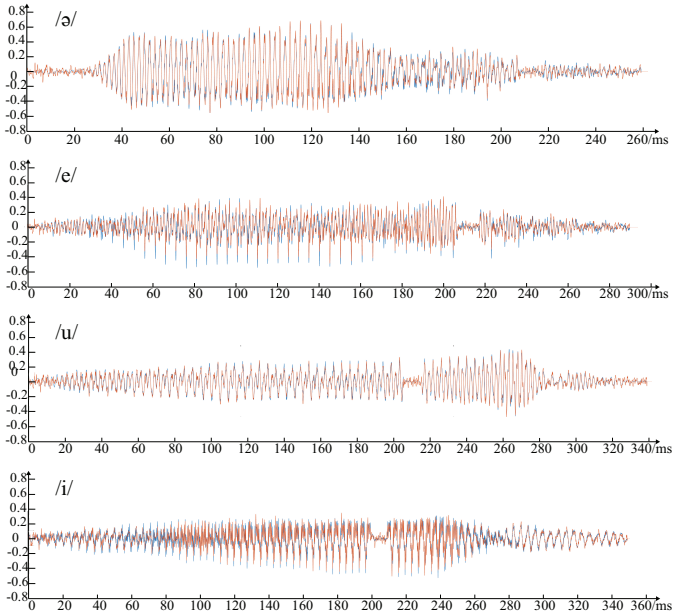

Fig. 4. The comparison of generated and original waveforms for phoneme /ə/, /e/, /u/ and /i/ are listed. The blue lines represent the original input waveform, while the red lines represent DN-2's generated waveform.

So 40 real value sections are designed in the motor area to represent the dense supervision or emergent actions. There are128 Z neurons in each real value section to approximate a real value in the range from -1 to 1.

B. Training and test procedure

In the experiments, we trained DN-2 to learn to produce the sounds (phonemes) which imitate the waveforms it "hears". During training, the type 100 and 111 Y neurons are mainly grown for leaning. At first, type 100 Y neurons are mainly grown and learn volume information. Then type 111 Y neurons are mainly grown to learn different spatial and temporal feature patterns.

After training, the test sequence (same data but without supervision) are fed to DN-2 to generate dense actions. Based on these real-valued actions, we used W^T to reconstruct the waveform frames, and further reconstruct the phonemes. We should mention that the actions generated by itself during the test also provide the context information for DN-2.

C. Results and analysis

The comparisons of phoneme waveforms generated by DN-2 and the original inputs are listed in Fig. 3 and Fig. 4. In these figures, the blue lines represent original input data, while the red lines represent DN-2's generated data. We can see clearly that at most moments the red lines cover the blue lines. This indicates that DN-2 successfully imitate the phonemes it received. In Fig. 3, we take phoneme /ʌ/ as the example to illustrate the comparisons of the beginning, middle and end frames. We can see the imitation of the beginning and end frames are not as perfect as the middle one. This is because the waveform amplitudes of the beginning and end parts are smaller which increases the difficulty of imitation.

To demonstrate type 111 neurons chaining the temporal information during learning, we illustrated the lateral connection situation of type 111 neurons in two modes. In Fig. 5, 9 type 111 neurons are randomly chosen from DN-2 to show their lateral weights as images. Each weight of the type 111 neurons is shown as the 13×12 images: row major representation. Each pixel in the weight images corresponds to a component in the weight vectors. The darker color means a higher value. Each weight ends in the first element in the final row, the remaining white area in the final row is used for completeness. We can see clearly these neurons record different Y firing patterns combined by type 100 and 111 neurons.

In Fig. 6, The lateral connections among type 111 neurons are shown. These connections form temporal relationships among the abstractions. the formed representation learns the transition across sequential data based on every context-input pair. The plotted lateral connections are grouped with the same color for each phoneme waveform.

The experimental results show that DN-2 can emerge the actions (high-dimensional real-valued vector) with self-supervision. We plan to study different dimensionality reduction and reconstruction technologies (e.g., Candid Covariance-free Incremental PCA (CCI PCA) [27]) to further reduce the error between the reconstructed data and original data.

V. Conclusions and Discussions

The emergent Turing Machine learned by a DN results in the partition of all sequences of input space into a larger number of equivalent classes. Such equivalent classes have very different sizes, including do-not-care scene elements and highly scene elements. Given first four essential factors, the fifth factor learning experience seems to be especially critical for further study.

The experimental results have demonstrated that DN-2 can perfectly imitate the sequential waveform data. The results also indicated that the motor area can be set to real value sections

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

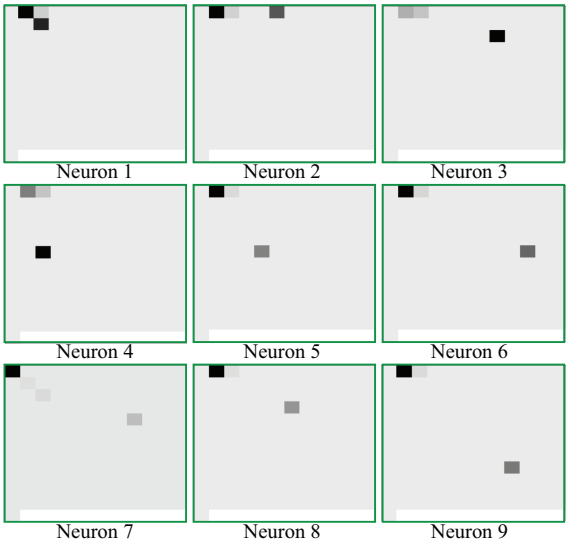

Fig. 5. The lateral weight images of 9 randomly chosen type 111 Y neurons are listed. Each weight is presented as a 13×12 image: row major representation. Each pixel corresponds to a weight component. The darker color indicates higher value. The white area in the final row is used for completeness.

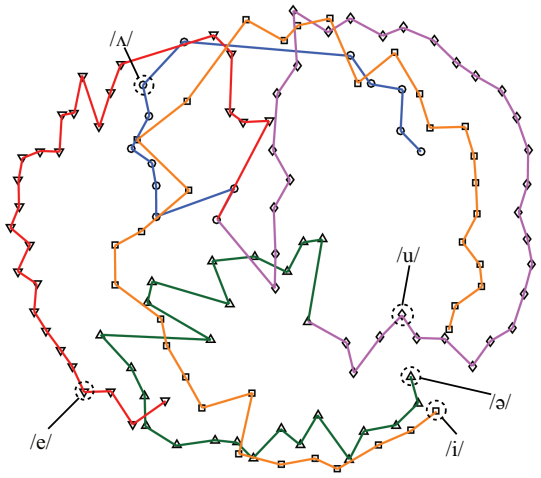

Fig. 6. Type 111 neurons' embedding and lateral connections among the same type neurons are visualized. Each type 111 Y neuron embedding colored according to the phoneme it learned. The black shapes (circles, squares, triangles, etc) represent type 111 Y neurons grown in the phoneme imitation experiments. The connection between Y neurons indicates that these neurons are laterally connected across time sequence (when a neuron is connected to multiple hidden neurons, we choose one with the strongest connection).

to approximate the real values in different ranges, but more experimental work is needed. We will use longer and more complicated auditory imitation experiments (e.g., word and sentence). Other sensory modalities are also in our plan.

References

[1] 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.

[2] D. J. Wood, J. S. Bruner, and G. Ross, "The role of tutoring in problem-solving," *Journal of Child Psychology and Psychiatry*, pp. 89–100, 1976.

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

[4] R. Sun, P. Slusarz, and C. Terry, "The interaction of the explicit and the implicit in skill learning: A dual-process approach," *Psychological Review*, vol. 112, no. 1, pp. 59–192, 2005.

[5] J. L. Castro-Garcia and J. Weng, "Emergent multilingual language acquisition using developmental networks," in *2019 International Joint Conference on Neural Networks*, (Budapest, Hungary), pp. 1–8, IEEE, July 2019.

[6] F. Camastra and A. Vinciarelli, "Markovian models for sequential data," *Neural Computing Surveys*, pp. 265–303, 2009.

[7] S. Solaimanpour and P. Doshi, "A layered hmm for predicting motion of a leader in multi-robot settings," in *2017 IEEE International Conference on Robotics and Automation*, (Singapore), pp. 788–793, May 2017.

[8] J. Scheffer, "Face pose estimation from video sequence using dynamic bayesian network," in *IEEE Workshop on Motion and Video Computing*, pp. 1–8, 2008.

[9] M. Yoneyama, A. Ohtake, Y. Iwano, and K. Shirai, "Facial expressions recognition using discrete hopfield neural networks," in *International Conference on Image Processing*, vol. 1, (Santa Barbara, CA, USA), pp. 117–120, October 1997.

[10] S. Hu, Y. Liu, Z. Liu, T. Chen, J. Wang, Q. Yu, L. Deng, Y. Yin, and S. Hosaka, "Associative memory realized by a reconfigurable memristive hopfield neural network," *Nature communications*, vol. 6, p. 7522, 2015.

[11] W. Li, L. Wen, M.-C. Chang, S.-N. Lim, and S. Lyu, "Adaptive rnn tree for large-scale human action recognition.," in *IEEE International Conference on Computer Vision*, pp. 1453–1461, 2017.

[12] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *15th Annual Conference of the International Speech Communication Association*, (Singapore), September 2014.

[13] E. Chemali, P. Kollmeyer, M. Preindl, R. Ahmed, and A. Emadi, "Long short-term memory-networks for accurate state of charge estimation of li-ion batteries," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 8, pp. 6730–6739, 2018.

[14] Z. Zheng and J. Weng, "Mobile device based outdoor navigation with on-line learning neural network: A comparison with convolutional neural network," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, (Las Vegas, NV, USA), pp. 11–18, June 2016.

[15] X. Wu, Y. Bo, and J. Weng, "Information-dense actions as contexts," *Neurocomputing*, vol. 311, pp. 164–175, 2018.

[16] J. Weng, "Meanings of data and rules emerge as actions through auto-programming for general purposes," in *IJCAI Workshop on Bringing Semantic Knowledge into Vision and Text Understanding*, (Macau, China), pp. 1–7, August 2019.

[17] X. Wu and J. Weng, "Neuron-wise inhibition zones and auditory experiments," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9581–9590, 2019.

[18] J. Weng, "Brain as an emergent finite automaton: A theory and three theorems," *International Journal of Intelligent Science*, vol. 5, no. 2, pp. 112–131, 2015.

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

[20] N. Kambhatla and T. K. Leen, "Dimension reduction by local principal component analysis," *Neural computation*, vol. 9, no. 7, pp. 1493–1516, 1997.

[21] A. Joshi and J. Weng, "Autonomous mental development in high dimensional context and action spaces," *Neural networks*, vol. 16, no. 5-6, pp. 701–710, 2003.

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

[23] X. Wu, Z. Zheng, and J. Weng, "Sensorimotor in space and time: Audition," in *2018 International Joint Conference on Neural Networks*, (Rio de Janeiro, Brazil), pp. 1–8, July 2018.

[24] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[25] Z. Zheng, X. Wu, and J. Weng, "Emergent neural turing machine and its visual navigation.," *Neural Networks*, vol. 110, pp. 116–130, 2019.

[26] "2016 artificial intelligence machine learning contest," 2016. http://www.brain-mind-institute.org/AIMLcontest/index-2016.html.

[27] J. Weng, Y. Zhang, and W.-S. Hwang, "Candid covariance-free incremental principal component analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 1034–1040, 2003.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE