3

Linguistic Productivity and Recurrent Neural Networks

Akito Sakurai [1, 2] and Yoshihisa Shinozawa [1]
[1] Keio University, and
[2] CREST, JST
Japan

1. Introduction

Productivity is the defining property of a natural language. Any native speaker of a natural language utters a sentence that has never been heard and understands a sentence that has been heard for the first time. Chomsky claimed that the purpose of linguistics is to account for the productivity of natural languages (Chomsky, 1980).
The learnability of a productive language by computational mechanisms is hindered by the inherent nature of the language. For many years, researchers have attempted to devise a learning mechanism by which productive languages can be learnt in a manner similar to that adopted by a child learning from scratch or a student learning a second language. However, there are many problems resisted to be solved. Productivity is one of their causes.
This chapter is devoted to efforts undertaken to understand productivity in terms of language learning by means of simple but powerful methods such as neural networks, because neural networks are the simplest (maybe over-simplified) models of our brain mechanism we have obtained thus far.
It is natural to expect that a recurrent neural network (RNN) among them is capable of learning languages, specifically a subset of a natural language, because a sentence is a sequence of words and an RNN is capable of learning sequences.
The chapter consists of two parts, each of which is devoted to one of the two unmatched features of human languages—the recursive or self-embedding structure of human languages, and the syntactic or combinatorial systematicity of human languages. Both these features constitute the syntactic productivity of human languages.

2. Linguistic productivity and learnability

The difference between a natural language and other discrete symbolic systems is that a natural language is productive (Chomsky, 1959). The productivity of a language entails that a language is an infinite set of sentences; theoretically, the fact that a language is infinite refutes the learnability of a language. Strangely enough, any natural language can be learned by humans simply by hearing a finite number of, sometimes a very limited number of, sentences; furthermore, sometimes the sentences are ungrammatical (hence, they are not sentences linguistically), and it may not be indicated that they are ungrammatical (for example, refer (Pinker, 1984)).

Source: Recurrent Neural Networks, Book edited by: Xiaolin Hu and P. Balasubramaniam, ISBN 978-953-7619-08-4, pp. 400, September 2008, I-Tech, Vienna, Austria

If a natural language is learnable, it should be generated by some set of rules, which must be finite; otherwise, the language can never be learned. Gold's well-known theorems (Gold, 1967) state that even if the generating rules are finite, we would not be able to learn the language without negative examples (ungrammatical sentences).

Although productivity is the defining property of a natural language, a language should be characterized better by its learnability. Productive systems are easily built by, for example, rewriting systems; however, a learnable productive system is prohibitively difficult to build. Our natural language is, though, a proof of existence of the learnable productive system.

We could mention two key factors that consist the productivity of a language. One of the factors is systematicity and the other is recursiveness. Systematicity is related to the lexical category and recursiveness is related to the phrasal category.

Syntactic systematicity is a property by which a valid sentence remains valid when a word in it is replaced with another word belonging to the same lexical category. Since a lexical category is one of the syntactic categories and is defined as a group of words that are replaceable with each other in a sentence without invalidating the grammaticality of the sentence, the claim is a tautology. Therefore, syntactic systematicity can be better defined as a property by which the syntax is described using lexical categories. The merit of having systematicity is that the number of categories is maintained far fewer than the size of the lexicon, and therefore the rules are simpler. Since a word, specifically a noun, is coined quite easily and infinitely, systematicity ensures that an infinite number of new sentences are obtained from a sentence.

In general, systematicity is a wider concept, which has been under argument for years (for example, refer (Fodor & Pylyshyn, 1988) and the literatures citing it). We consider only syntactic systematicity, specifically weak systematicity and strong systematicity, as defined by Hadley (Hadley, 1994).

Recursiveness is a property by which a syntax is (exactly or approximately) modelled by a set of rewriting rules in which some phrasal category is directly or indirectly defined by itself. By the recursive rules, we can obtain an infinite number of different and valid sentences from a set of finite rules and lexicons. It must be noted that in recursive function theory or computational theory, the concept of recursiveness includes or is equivalent to that of repetition, which is easily observed from the definitions of recursive functions or Turing machines. Recursive rules of the form $A \rightarrow aA$ or $A \rightarrow Aa$ are called regular rules; they are not fully qualified "recursive" rules because the rules $A \rightarrow aA$ or $A \rightarrow Aa$ represent repetitions. Recursive rules generally refer to rules such as $A \rightarrow aAb$, which requires more than simply a finite state machine to parse the resultant sentences.

For language generation, when we use a symbolic representation of grammar, systematicity and recursiveness are easy to implement, and we can observe the variety or postulated infiniteness of the resultant languages.

For language learning or grammatical inference, systematicity and recursiveness pose challenges. We will first consider recursiveness, specifically the possibility or impossibility of representing recursive rules by RNNs; then, we will study recursiveness, specifically the possibility or impossibility of representing recursive rules by RNNs.

3. Recursion

In this section, we state a necessary condition: a simple RNN with two sigmoidal hidden units is a recognizer of the language $\{a^n b^n \mid n > 0\}$, whose grammar is expressed by $\{S \rightarrow aSb,$

$S \rightarrow ab\}$. Further, we show that there exists a set of parameters that satisfy the above condition. Then, we realize the language recognizer for the type of language mentioned above, although the stated condition implies the instability in learning, as has been reported in previous studies. Furthermore, the condition, contrary to its success in implementing the recognizer, implies the difficulty in obtaining a recognizer for more complicated languages.

3.1 Background

The researches conducted on the induction of a grammar that includes a recursive or self-embedding rule by neural networks have employed the following features:

- simple recurrent neural networks (SRN) adopted as the basic or core mechanism
- languages such as $\{a^n b^n \mid n>0\}$, $\{a^n b^n c^n \mid n>0\}$, which are clearly the results of, although not a representative of, recursive rules, adopted as target languages.

Here, a^n denotes a string of n-times repetition of a character a; the language $\{a^n b^n \mid n > 0\}$ is generated by a context-free grammar $\{S \rightarrow aSb,\ S \rightarrow ab\}$, and $\{a^n b^n c^n \mid n > 0\}$ is generated by a context-sensitive grammar. The adoption of these simple languages as target languages is inevitable because it is not easy to determine whether or not sufficient generalization is achieved by the learning, if realistic grammars are used.

Although these target languages were simple, it has been pointed out that they were learned but their grammars were not, because sufficient generalization by the learned network was not observed and the resultant network appeared to be almost similar to the result of rote learning. Further, the resultant networks were unstable in the sense that when they were given new training sentences, which were longer than the ones they had learned, the learned network changed to completely new networks that were more than simple refinements of the learned network.

Bodén et al. (Bodén et al., 1999; Bodén & Wiles, 2000; Bodén & Blair, 2003), Rodriguez et al. (Rodriguez et al., 1999; Rodriguez, 2001), Chalup (Chalup & Blair, 2003), and others conducted investigations during exploration for the possibility of network learning of the languages. However, they have not succeeded in clearly stating the conditions that the learned networks should satisfy.

In this section, we state a property describing SRNs with two hidden units that learned to recognize a language $\{a^n b^n \mid n>0\}$, that is, a necessary condition for an SRN to be qualified as a successful language recognizer. The stated condition implies the instability in learning. We also show the realization of the condition and obtain the recognizer for the language. However, the question— if there really exists a solution—remains unanswered.

3.2 Preliminaries

A recurrent neural network (RNN) is a network that has recurrent connections added to a feed-forward network. The calculation proceeds at first for the feed-forward part and after a single time-unit delay for the recurrent connection part; hence, the inputs for the calculations in the feed-forward part are supplemented with the outputs of the recurrent connections.

An RNN is considered to be a discrete time system. Starting with an initial state (initial outputs of the feed-forward part, i.e., outputs without external inputs), the network proceeds to accept the subsequent character in a string given to the external inputs, reaches the final state, and obtains the final external output from the final state.

A simple recurrent network (SRN) is a simple type of RNN and has only one layer of hidden units in its feed-forward part.

Rodriguez et al. (Rodriguez et al., 1999) showed that an SRN learns languages $\{a^nb^n \mid n > 0\}$ and $\{a^nb^nc^n \mid n > 0\}$. For $\{a^nb^n \mid n > 0\}$, an SRN successfully accepted a language $\{a^nb^n \mid 0 < n \leq 16\}$ after processing sentences language $\{a^nb^n \mid 0 < n \leq 11\}$. They analyzed and described the manner in which the input sentences were processed by the SRN. However, the analysis was limited to the results obtained by simulations and did not proceed into indicating how and when the learning is possible. Moreover, the existence of the language recognizers was not yet shown.
Siegelmann (Siegelmann, 1999) showed that the computational ability of an RNN is superior to that of a Turing machine, thereby implying that the recoginizers for the languages $\{a^nb^n \mid n>0\}$ and $\{a^nb^nc^n \mid n>0\}$ exist. However, based on unsatisfactory generalization obtained by the learning experiments, we would state that the possibility of learning and existence of realization is different or it might be the case that the solution does not exist seemingly contradictive to Siegelmann's result because there is a difference in formulation: the output functions of the units are piecewise linear functions; the inversibility of the function is utilized in Siegelmann's case and sigmoidal functions are utilized in standard SRN cases. It must be noted that piecewise linear functions have non-differentiable points, which makes them infeasible to utilize error-backpropagation algorithm.
On the other hand, Casey (Casey, 1998) and Maass (Maass & Orponen, 1998) showed that in noisy environments, an RNN is as powerful as finite automaton. The results suggest that we should consider infinite precision computation when we have to search the possibility of computations by an RNN or specifically an SRN.
In summary, the learnability of RNN and SRN recognizers for the language $\{a^nb^n \mid n > 0\}$ is not yet demonstrated, and moreover the existence of the recognizers is not yet proved. To conduct further research, we need to suppose that the computations performed by the RNN and SRN should be formulated with infinite precision, and the units should use sigmoidal functions. Therefore, in this research, we have adopted RNN models with infinite precision calculations and the sigmoidal function ($\tanh(x)$) as the output function for the units.
On the basis of the viewpoints mentioned above, we discuss two points in this section: a necessary condition for an SRN with two hidden units to be a recognizer for the language $\{a^nb^n \mid n>0\}$, and whether or not the stated condition is sufficient to guide us to build an SRN language recognizer.

3.3 Symbols and terminology

An SRN is a simple type of RNN and its function is expressed as follows:

$$\boldsymbol{s}_{n+1} = \sigma(\boldsymbol{w}_s \cdot \boldsymbol{s}_n + \boldsymbol{w}_x \cdot \boldsymbol{x}_n) \tag{1}$$

$$N_n(\boldsymbol{s}_n) = \boldsymbol{w}_{os} \cdot \boldsymbol{s}_n + \boldsymbol{w}_{oc} \tag{2}$$

Here, σ is a standard sigmoid function ($\tanh(x) = (1 - \exp(-x))/(1 + \exp(-x))$), which is applied componentwise.
A *counter* is a device that stores an integer and allows +1 or –1 operation and answers yes or no to an inquiry if the content is 0 (*0-test*). A *stack* is a device that allows the operations *push i* (*store i*) and *pop up* (recover the last-stored content, discard it and restore the device to its state immediately before the corresponding push operation). Clearly, a stack is a more powerful device as compared to a counter; hence, if a counter is not implementable, a stack, too, is not implementable.

To represent an input or output word, we adopt a *localist representation* or *one-hot vector*, a vector representation in which a single element is 1 and the other elements are 0. The network is trained so that the sum of the squared error (difference between the actual output and desired output) is minimized.

In learning experiments of languages, it is natural to assume there is no negative, i.e., ungrammatical sentence; therefore, we have to devise some teaching information from the sentences. Elman proposed to train an SRN as a predictor, i.e., to teach the network to predict a word when it sees a sub-string of words up to the word (refer (Elman, 1991)). By this method, if two possible outputs with the same frequency of occurrence for the same input exist in a training data, the network would learn to output the same value for two elements in the output with those for the other elements being 0; this is because the output vector should provide the minimum value of the sum of the squared error.

For the language $\{a^n b^n \mid n>0\}$, when an SRN is trained to predict the subsequent word, it behaves internally as a counter. Clearly, if SRN could count the number of *a*s and *b*s, it would be able to recognize $\{a^n b^n \mid n>0\}$. In fact, Rodriguez et al. (Rodriguez et al., 1999) and others analyzed trained networks and found that they behave like counters.

It is clear that if a network correctly predicts the subsequent character in a string in the language $\{a^n b^n \mid n>0\}$, we could see it as a counter, although its capability is limited. Let us add an auxiliary network output whose value is positive if the original network predicts only "*a*" (which happens only when the input string up to the time was $a^n b^n$ for some n) and is negative otherwise.

The modified network behaves as if it counts up for a character "*a*" and counts down for "*b*," because it outputs positive values when the number of "*a*"s and "*b*"s coincide and outputs negative values otherwise. However, the counting capability of the network may be limited because it would output any value when "*a*" is fed before the due number of "*b*"s are fed, that is, when a counting up action is required before the counter returns back to the 0-state.

As suggested by Rodriguez et al. (Rodriguez et al., 1999), we consider an SRN to be a dynamical system. For the terminology related to dynamical systems, specifically terms such as ω-limit set and stable/unstable manifold, we suggest that the readers refer to (Katok & Hasselblatt, 1996) or (Guckenheimer & Holmes, 1997). Concise definitions are provided in the Appendix of (Rodriguez et al., 1999).

A (discrete-time) *dynamical system* is represented as the iteration of a function application: $\boldsymbol{s}_{i+1} = f(\boldsymbol{s}_i)$, where $i \in N$, $\boldsymbol{s}_i \in R^n$. A point s is called a fixed point of f if $f(s) = s$. A point s is an attracting fixed point of f if s is a fixed point and there exists a neighbourhood U_s around s such that $\lim_{i\to\infty} f^i(x) = s$ for all $x \in U_s$. A point s is a repelling fixed point of f if s is an attracting fixed point of f^{-1}. A point s is called a periodic point of f if $f^n(s) = s$ for some n.

A point s is a ω-limit point of x for f if $\lim_{i\to\infty} f^{n_i}(x) = s$ for $\lim_{i\to\infty} n_i = \infty$. A fixed point x of f is hyperbolic if all of the eigenvalues of Df at x have absolute values other than one, where $Df = \partial f_i / \partial x_j$ is the Jacobian matrix of the first partial derivatives of the function f. A set D is invariant under f if for any $s \in D$, $f(s) \in D$.

Theorem 1 (Stable Manifold Theorem for a Fixed Point (Guckenheimer & Holmes, 1997))

Let f: $R^n \to R^n$ be a C^r ($r \geq 1$) diffeomorphism with a hyperbolic fixed point x. Then there exist local stable and unstable manifolds $W^{s,f}_{\text{loc}}(x)$, $W^{u,f}_{\text{loc}}(x)$, tangent to the eigenspaces $E^{s,f}_x$, $E^{u,f}_x$ of Df at x and of corresponding dimension. $W^{s,f}_{\text{loc}}(x)$ and $W^{u,f}_{\text{loc}}(x)$ are as smooth as the map f, i.e., of class C^r.

Local stable/unstable manifolds for f are defined as follows (Corollary 6.2.5 in (Katok & Hasselblatt, 1996)):

$$W^{s,f}_{\mathrm{loc}}(q) = \{ y \in U_q \mid \lim_{m\to\infty} \mathrm{dist}(f^m(y), q) = 0 \}$$

$$W^{u,f}_{\mathrm{loc}}(q) = \{ y \in U_q \mid \lim_{m\to\infty} \mathrm{dist}(f^{-m}(y), q) = 0 \}$$

where U_q represents the neighbourhood of q, and dist is the distance function. Then, the global stable and unstable manifolds for f are defined as follows:

$$W^{s,f}(q) = \cup_{i\geq 0} f^{-i}(W^{s,f}_{\mathrm{loc}}(q))$$

$$W^{u,f}(q) = \cup_{i\geq 0} f^{i}(W^{u,f}_{\mathrm{loc}}(q))$$

As defined, an SRN is a pair of a discrete-time dynamical system $\boldsymbol{s}_{n+1} = \sigma(\boldsymbol{w}_s\cdot\boldsymbol{s}_n+\boldsymbol{w}_x\cdot\boldsymbol{x}_n;)$ and an external output part $N_n(\boldsymbol{s}_n) = \boldsymbol{w}_{os}\cdot\boldsymbol{s}_n+\boldsymbol{w}_{oc}$. We simply express the former (dynamical system part) as $\boldsymbol{s}_{n+1} = f(\boldsymbol{s}_n,\boldsymbol{x}_n)$ and the external output part as $h(\boldsymbol{s}_n)$.
When an RNN (or SRN) is considered to be a counter, an input $\boldsymbol{x}_+$ plays the role of the "+1" count-up operation and another input $\boldsymbol{x}_-$ performs the "–1" or count-down operation. For simplicity, hereafter, let $f_+ = f(\cdot,\boldsymbol{x}_+)$ denote the "+1" operation and $f_- = f(\cdot,\boldsymbol{x}_-)$ denote the "–1" operation. It must be noted that when the network is used as a recognizer of the language $\{a^nb^n \mid n>0\}$, the input character "a" corresponds to $\boldsymbol{x}_+$ and "b" corresponds to $\boldsymbol{x}_-$. Further, f_- is undefined for the point outside and on the border of the square I[–1,+1] × I[–1,+1], where I[–1,+1] is the closed interval [–1,+1]; however, we do not mention it for simplicity.
D_0 is a set $\{\boldsymbol{s} \mid h(\boldsymbol{s}) \leq 0\}$, that is, a region where the counter value is 0 and which is simply connected when the network is an SRN because h is effectively a linear function. Let $D_i = f_-^{-i}(D_0)$, that is, a region where the counter value is i.
We postulate that $f_+(D_i)\subseteq D_{i+1}$. This means that any point in D_i is eligible for a state which designates that the counter content is i. This may appear to be rather demanding. An alternative approach would be that in which the point p corresponds to counter content c if and only if $p = f_-^{m1} f_+^{p1}\cdots f_-^{mi} f_+^{pi}(s_0)$ for a predefined s_0, some $m_j \geq 0$ and $p_j \geq 0$ for $1 \leq j \leq i$, and $i \geq 0$ such that $\sum_{j=1}^{i}(p_j - m_j) = c$. However, this approach has not resulted in a fruitful result.
We also postulate that the closures of D_i are disjoint. Since we defined D_i as a closed set, the postulate is natural. Our consideration was to select D_i to be closed. The postulate requires that we should maintain a margin between D_0 and D_1 and any other D_is.

3.4 Necessary condition

In this subsection, we consider only an SRN with two hidden units, i.e., all the vectors concerning $\boldsymbol{s}$ such as $\boldsymbol{w}_s$, $\boldsymbol{s}_n$, $\boldsymbol{w}_{os}$ are two-dimensional vectors.
Definition 2. D_ω is the set of the accumulation points of $\{D_i \mid i > 0\}$, i.e., $s \in D_\omega$ iff $s = \lim_{i\to\infty} s_{ki}$ for some $\boldsymbol{s}_{ki}\in D_{ki}$.
Definition 3. P_ω is the set of ω-limit points of points in D_0 for f_+, i.e., $s\in P_\omega$ iff $s = \lim_{i\to\infty} f_+^{ki}(s_0)$ for some k_i and $s_0\in D_0$. Q_ω is the set of ω-limit points of points in D_0 for f_-^{-1}, i.e., $s\in Q_\omega$ iff $s = \lim_{i\to\infty} f_-^{-k}(s_0)$ for some k_i and $s_0\in D_0$.
With regard to the results obtained by Bodén et al. (Bodén et al., 1999; Bodén & Wiles, 2000; Bodén & Blair, 2003), Rodriguez et al. (Rodriguez et al., 1999; Rodriguez, 2001), Chalup (Chalup & Blair, 2003), it is natural, at least during the first consideration, to postulate that for any x, $f_+^{i}(x)$ and $f_-^{i}(x)$ do not wonder and therefore will converge to periodic points.

Therefore, P_ω and Q_ω are postulated as finite sets of hyperbolic periodic points for f_+ and f_-, respectively. For simplicity in presentation, we postulate that P_ω and Q_ω are finite sets of hyperbolic fixed points for f_+ and f_-, respectively.
Moreover, the points in Q_ω are saddle points for f_-; hence, we further postulate that $W^{u,f^-}_{\text{loc}}(q)$ for $q \in Q_\omega$ and $W^{s,f^-}_{\text{loc}}(q)$ for $q \in Q_\omega$ are one-dimensional space and their existence is guaranteed by Theorem 1.
Postulate 4. We postulate that $f_+(D_i) \subseteq D_{i+1}$, the closures of D_i are disjoint, P_ω and Q_ω are finite sets of hyperbolic fixed points for f_+ and f_-, respectively, and $W^{u,f^-}_{\text{loc}}(q)$ for $q \in Q_\omega$ and $W^{s,f^-}_{\text{loc}}(q)$ for $q \in Q_\omega$ are one-dimensional spaces.
Lemma 5. $f_-^{-1} \circ f_+(D_\omega) = D_\omega$, $f_-^{-1}(D_\omega \cap \mathrm{I}(-1,1) \times \mathrm{I}(-1,1)) = D_\omega$ and $D_\omega \cap \mathrm{I}(-1,1) \times \mathrm{I}(-1,1) = f_-(D_\omega)$, and $f_+(D_\omega) \subseteq D_\omega$. $P_\omega \subseteq D_\omega$ and $Q_\omega \subseteq D_\omega$.
Definition 6. $W^{u,-1}(q)$ is the global unstable manifold at $q \in Q_\omega$ for f_-^{-1}, i.e., $W^{u,-1}(q) = W^{u,(f^-)^{-1}}(q) = W^{s,f^-}(q)$.
Lemma 7. For any $p \in D_\omega$, any accumulation point of $\{ f_-^{i}(p) \mid i > 0 \}$ lies in Q_ω
Proof. Since p lies in D_ω, there exist $p_{ki} \in D_{ki}$ such that $p = \lim_{i\to\infty} f_+^{ki}(p_{ki})$. Suppose q in D_ω is the accumulation point stated in the theorem statement, i.e., $q = \lim_{j\to\infty} f_-^{hj}(p)$. We set k_i to be sufficiently large for any h_j so that p_{ki} exists in any neighbourhood of q with $f_-^{hj}(p)$. Then, $q = \lim_{j\to\infty} f_-^{hj}(p_{ki}) = \lim_{j\to\infty} f_-^{hj-ki}(s_{ki})$, where k_i is a function of h_j with $k_i > h_j$. Let $s_{ki} = f_-^{-ki}(p_{ki}) \in D_0$ and $s_0 \in D_0$ be an accumulation point of $\{s_{ki}\}$. Then, since f_-^{-1} is continuous, by setting $n_j = -h_j + k_i > 0$, we get $q = \lim_{j\to\infty} f_-^{nj}(s_0)$, i.e., $q \in Q_\omega$. □
Lemma 8. $D_\omega = \cup_{q \in Q_\omega} W^{u,-1}(q)$
Proof. Let p be any point in D_ω. Since $f_-(D_\omega) \subseteq \mathrm{I}[-1,1] \times \mathrm{I}[-1,1]$ where $\mathrm{I}[-1,1]$ is the closed interval $[-1,1]$, i.e., $f_-(D_\omega)$ is bounded, and $f_-(D_\omega) \subseteq D_\omega$, $f_-^{n}(p)$ has an accumulation point q in D_ω, which is, by Lemma 7, in Q_ω. Then, q is expressed as $q = \lim_{j\to\infty} f_-^{nj}(p)$. Since Q_ω is a finite set of a hyperbolic fixed point, $q = \lim_{n\to\infty} f_-^{n}(p)$, i.e., $p \in W^{s,f}(q) = W^{u,f^{-1}}(q) = W^{u,-1}(q)$. □
Since $P_\omega \subseteq D_\omega$, the next theorem holds.
Theorem 9. A point in P_ω is either a point in Q_ω or in $W^{u,-1}(q)$ for some $q \in Q_\omega$.
It must be noted that since $q \in W^{u,-1}(q)$, the theorem statement simply states that "If $p \in P_\omega$, then $p \in W^{u,-1}(q)$ for some $q \in Q_\omega$."

3.5 An Example of a recognizer

To construct an SRN recognizer for $\{ a^n b^n \mid n > 0 \}$, the SRN should satisfy the conditions stated in Theorem 9 and Postulate 4, which are summarized as follows:

1. $f_+(D_i) \subseteq D_{i+1}$,
2. the closures of D_i are disjoint,
3. P_ω and Q_ω are finite sets of hyperbolic fixed points for f_+ and f_-, respectively,
4. $W^{u,f^-}_{\text{loc}}(q)$ for $q \in Q_\omega$ and $W^{s,f^-}_{\text{loc}}(q)$ for $q \in Q_\omega$ are one-dimensional spaces, and
5. if $p \in P_\omega$ then $p \in W^{u,-1}(q)$ for some $q \in Q_\omega$.

To find a solution as simply as possible, let us try to suppose that $p \in P_\omega$ and $q \in Q_\omega$, that is, $f_+(p) = p$ and $f_-(q) = q$. Since p cannot be the same point as q (because $f_-^{-1} \circ f_+(p) = p + \boldsymbol{w}_s^{-1} \cdot \boldsymbol{w}_x \cdot (\boldsymbol{x}_+ - \boldsymbol{x}_-) \neq p$), we have to find a way to cause $p \in W^{u,-1}(q)$.
Since it is generally very difficult to calculate stable or unstable manifolds from a function and its fixed point, we attempt to allow $W^{u,-1}(q)$ to be a "simple" manifold; if $W^{u,-1}(q)$ is simple, it is easy to define $D_0 = \{\boldsymbol{x} \mid h(\boldsymbol{x}) \geq 0\}$; on the other hand, if $W^{u,-1}(q)$ is not simple, a suitable h may not exist.

We have decided that $W^{u,-1}(q)$ is a line (if possible). Considering the function form $f_{-}(\boldsymbol{s}) = \sigma(\boldsymbol{w}_s \cdot \boldsymbol{s} + \boldsymbol{w}_x \cdot \boldsymbol{x}_{-})$, it is not difficult to observe that the line could be one of the axes or one of the bisectors of the right angles at the origin (i.e., one of the lines $y = x$ and $y = -x$). We have selected the bisector in the first (and the third) quadrant (i.e., the line $y = x$). q is selected as the origin, and p is selected arbitrarily as (0.8, 0.8).
The condition stated in Item 4 in the above is satisfied by setting one of the two eigenvalues of Df_{-} at the origin to be greater than 1 and the other eigenvalue smaller than 1. We have selected $1/0.6$ and $1/\mu$ for the two eigenvalues so that the conditions stated in Item 1 and 2 in the above are satisfied by considering the eigenvalues of Df_{+} at p for f_{+}.
The design consideration that we have ignored is the design of $D_0 = \{\boldsymbol{x} \mid h(\boldsymbol{x}) \geq 0\}$. A simple method is to make the boundary $h(\boldsymbol{x}) = 0$ parallel to $W^{u,-1}(q)$ for our intended $q \in Q_\omega$; if we do so, by setting the largest eigenvalue of Df_{-} at q to be equal to the inverse of the eigenvalue of Df_{+} at p along the normal to $W^{u,-1}$, we can obtain the points $s \in D_0$, $f_{-} \circ f_{+}(s)$, $f_{-}^{2} \circ f_{+}^{2}(s), \ldots, f_{-}^{i} \circ f_{+}^{i}(s), \ldots$ that belong to $\{a^n b^n \mid n>0\}$ and reside at approximately equal distances from $W^{u,-1}$. It is apparent that the points belonging to, say, $\{a^{n+1} b^n \mid n > 0\}$ have approximately equal distances from $W^{u,-1}$, and this distance is different from that for $\{a^n b^n \mid n>0\}$
Let $f_{-}(x) = \sigma(Ax + B_0)$, $f_{+}(x) = \sigma(Ax + B_1)$. We plan to set $Q_\omega = \{(0,0)\}$, $P_\omega = \{(0.8,0.8)\}$, $W^{u,-1} = \{(x,y) \mid y = x\}$; the eigenvalues of the tangent space of f_{-}^{-1} at (0,0) are $1/\lambda = 1/0.6$ and $1/\mu$ (where the eigenvector on $y = x$ is expanding), and the eigenvalues of the tangent space of f_{+} at (0.8,0.8) are $1/\mu$ and any value. Then, considering the derivatives at (0,0) and (0.8,0.8), it is easy to determine that

$$\frac{1}{2}A = \rho\left(\frac{\pi}{4}\right)\begin{pmatrix} \lambda & 0 \\ 0 & \mu \end{pmatrix}\rho\left(-\frac{\pi}{4}\right), \quad \frac{1}{\mu} = \left(1 - 0.8^2\right)\mu$$

where $\rho(\theta)$ is a rotation by θ. Then

$$A = \begin{pmatrix} \lambda + \mu & \lambda - \mu \\ \lambda - \mu & \lambda + \mu \end{pmatrix}$$

Next from $\sigma(B_0) = (0,0)^{\mathrm{T}}$ and $\sigma(\,(0.8\lambda, 0.8\lambda)^{\mathrm{T}} + B_1) = (0.8,0.8)^{\mathrm{T}}$, we get

$$B_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad B_1 = \begin{pmatrix} \sigma^{-1}(0.8) - 0.8\lambda \\ \sigma^{-1}(0.8) - 0.8\lambda \end{pmatrix}$$

These give us $\mu = 5/3$, $\lambda = 0.6$, $B_1 \approx (1.23722, 1.23722)^{\mathrm{T}}$.
In Fig. 1, the left-hand side image shows the vector field of f_{+}, where the arrows starting at the x end at $f_{+}(x)$ and the right image shows the vector field of f_{-}. In the left-hand pane of Fig.2, the group of points at the centre (red) correspond to $\{a^n b^n \mid n>0\}$, those at the top (blue) correspond to $\{a^{n+1} b^n \mid n > 0\}$, and those at the bottom (green) correspond to $\{a^n b^{n+1} \mid n > 0\}$. The initial point is set to $p = (0.5,0.95)$ in Fig. 2. All the points correspond to $n = 1$ to $n = 40$, and when n grows, the points group together to points on $y = -x$, forming narrow stripes, i.e., D_n, for some n. As shown in the right-hand pane of Fig. 2, numerical computations of $f_{-}^{n} \circ f_{+}^{n}$ are sensitive to small truncation errors. In the case of Mathematica, the points start straying away for $n \geq 47$.

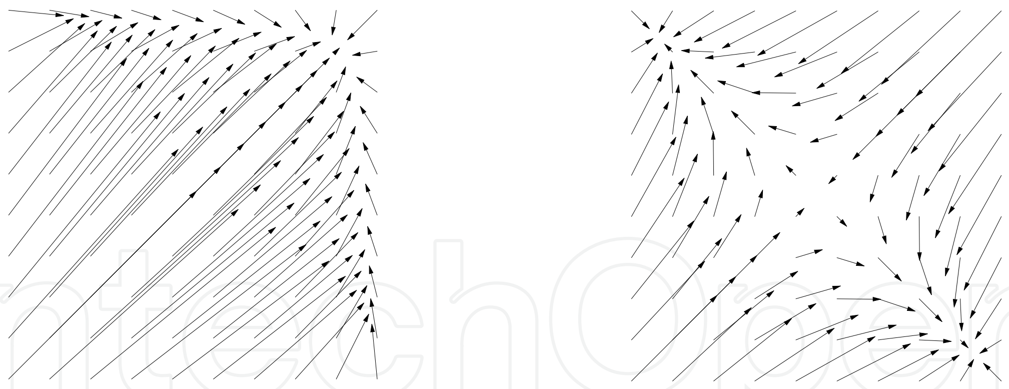

Fig. 1. Vector field representation of f_+ (left) and f_- (right)

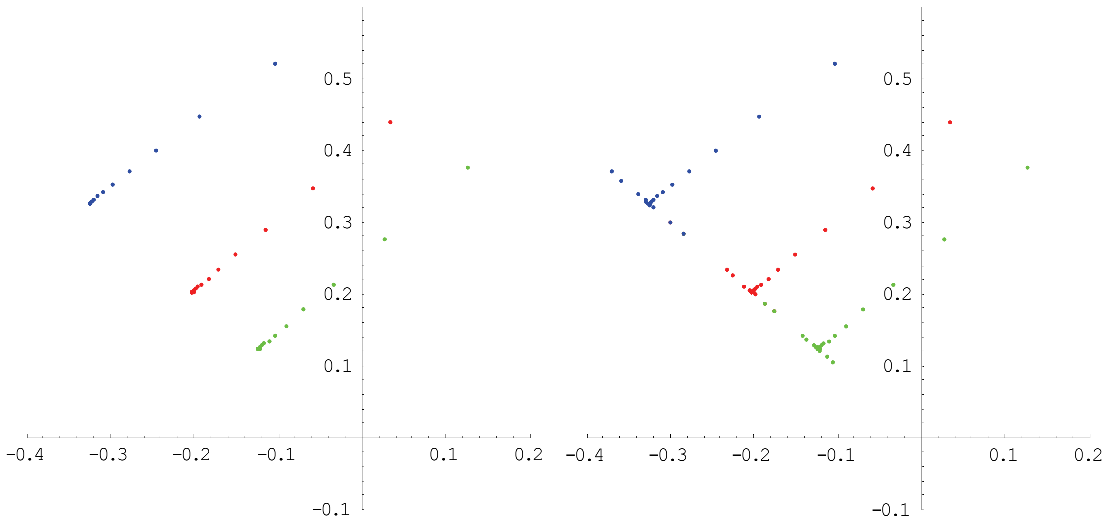

Fig. 2. On the left-hand side, $\{f_-^n \circ f_+^n(p) \mid 40 \geq n \geq 1\}$ (middle; red), $\{f_-^{n+1} \circ f_+^n(p) \mid 40 \geq n \geq 1\}$ (top; blue), and $\{f_-^n \circ f_+^{n+1}(p) \mid 40 \geq n \geq 1\}$ (bottom; green), where $p = (0.5, 0.95)$ are plotted. On the right side, the plots are identical to those on the left-hand side, except for $70 \geq n \geq 1$.

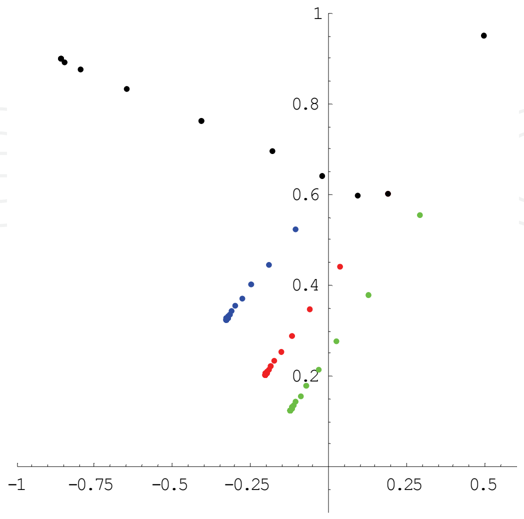

Fig. 3. Points of $\{(f_- \circ f_+)^n(p) \mid 11 \geq n \geq 1\}$, where $p = (0.5, 0.95)$ are plotted over the left-hand pane of Fig. 2, which shows that the points do not stay in D_0. The points start from p and converge at around (–0.8595, 0.8984).

3.6 Discussion for recursiveness

We obtained a necessary condition for an SRN to be used as a recognizer for the language $\{a^n b^n \mid n > 0\}$ by analyzing its proper behavior from the viewpoint of discrete dynamical systems. The stated condition supposes that the closures of D_i are disjoint, $f_+(D_i) \subseteq D_{i+1}$, and Q_ω is finite.

This suggests a possibility for the implementation of the recognizer; in fact, we have successfully built a recognizer for the language, thereby showing that the learning problem of the language has at least one solution. However, it is worthwhile to be cautious. As shown in Fig. 3, $f_{-} \circ f_{+}(p)$ does not return to around p unless there is no proper setting for D_0 to which p belongs. This means that we have to "reset" the counter when it returns to the 0-state in order to reuse it. The experimental setting used, for example, (Elman, 1990), i.e., a setting where a string *aabbaaaabbbbab*… is used is not appropriate to obtain the recognizer.

It is suggested (but not logically derived) that the instability of any solution for learning occurs due to the necessity of P_ω being in an unstable manifold $W^{u,-1}(q)$ for some $q \in Q_\omega$. Since P_ω is an attractive fixed point in the above example, $f_+^n(s_0)$ for $s_0 \in D_0$ approaches exponentially close to P_ω for n. Even a small fluctuation in the position of P_ω, since $f_+^n(s_0)$, too, is close to $W^{u,-1}(q)$, $f_-^n(f_+^n(s_0))$, which should be in D_0, is significantly disturbed. This means that even if an temporary solution is close to a correct solution, due to a small fluctuation in the position of P_ω caused by a new training data, $f_-^n(f_+^n(s_0))$ may easily be pushed out of D_0.

Since Rodriguez et al. (Rodriguez, 2001) showed that the languages that do not belong to the context-free class could be learned to some degree, we have to conduct further study on the discrepancies.

These instabilities of grammar learning by SRN mentioned above might not be visible in our natural language learning; this suggests that an SRN might not be appropriate for a model of language learning.

4. Systematicity

Hadley defines three degrees of syntactic systamaticity (Hadley, 1994). We focus on two of the degrees, namely weak systematicity and strong systematicity. According to Hadley, supposing that a training corpus is "representative" in the sense that every word (noun, verb, etc.) that occurs in some sentence of the corpus also occurs (at some point) in every permissible syntactic position, if a set of test sentences is the one that contains only grammatical sentences which are syntactically isomorphic to sentences in the training corpus, and that no new vocabulary is present, and a network is capable of successfully processing (by recognizing or interpreting) novel test sentences, then the network is said to be (at least) weakly systematic. Hadley defines that a system is strongly systematic if (i) it can exhibit weak systematicity and (ii) it can accurately process a variety of novel simple sentences and novel embedded sentences containing previously learned words in positions where they do not appear in the training corpus (i.e., the word within the novel sentence does not appear in that same syntactic position within any simple or embedded sentence in the training corpus). For subtleties in the above definition, refer to (Christiansen & Chater, 1994).

It is clear that if a training set is sufficiently large and syntactic sentence patterns are limited (for example, the length of the sentences is limited to at most ten words), we could state that a Bayesian method with lexical categories as a latent variable would provide a satisfactory result. In fact, many researches have targeted unsupervised language learning thus far. (Schütze, 1993) is an early research conducted on unsupervised language learning.

These researches show that the lexical categories could be induced from an unlabeled corpus if we adopt an approach based on symbolic paradigm and statistics. A question remains: is it possible for a simple method such as an RNN to induce lexical categories with, for example, an error-backpropagation algorithm?

If an SRN or RNN is able to learn the grammar of a language, they would be able to learn the lexical categories. However, since a large lexicon implies a large number of connections and possibly the necessity for a large number of internal nodes, it might make learning difficult for a network. Therefore, we must develop some mechanism, other than the SRN or RNN, to group similar words into clusters, which are input to the SRN or RNN. The clustering algorithm to be used may be explicit or implicit; here, “explicit” means that a clustering mechanism other than SRN or RNN is used, whereas “implicit” means that a sub-network is added to the SRN or RNN and is trained with the main SRN or RNN. (Elman, 1991) adopts the latter approach, whereas (Farkaš & Crocker, 2008) adopts the former one. Frank adopts an approach that does not use additional networks (Frank, 2006).

Farkaš and Crocker approached the problem by adding a type of self-organizing map (SOM). By means of the SOM, they successfully constructed a distributional representation of words, as done explicitly in probabilistic approaches (e.g. (Schütze, 1993)). The SOM is used not only for mapping from an input word to its distributional representation (a type of category) but also vise versa. The success of their approach clearly shows the possibility of existence of two types of networks, i.e., networks for clustering and networks for grammar.

One of the problems that remains is that since an SOM provides graded responses, it may be the case that frequency-related information creeps in the SOM output and SRN is trained not only on categories but also on frequency. Since we may not know how much of the result depends on the frequency-related information, we would be a little hesitant to say that the SRN has learned the grammar described in the lexical categories

One of the issues that we have to consider when we adopt the approach of “graded responses” such as statistics and neural networks with real value outputs is the performance criteria for the results. In this chapter, we consider an RNN with linear input functions and sigmoidal activation functions, which assume values between 0 and 1. Although the words are represented by localist representation or one-hot vector and therefore for an input, exactly one node is 1 and others are 0, the outputs assume real values, that is, possibly all the nodes have graded predictions for words.

Networks are trained so that an output designates the subsequent word. A training data set may be ambiguous in a sense that the possible number of subsequent words after a sequence may exceed one. Then, as usual, the output values are considered to be a distributional representation of the subsequent words or the likelihood of the network prediction. Suppose that there are training sequences that consist of a common prefix for a string of words but more than one word comes after the prefix. If a network is properly trained, we could expect that the network’s output prediction is uniform for the subsequent words after the common prefix string. If some word appears more often than another word, the output activation for the former word must be higher than that for the latter word, although the activations may not be proportional to their frequency.

Consequently, in a manner different from many neural network classification researches, which employ the sum of errors for each training samples, we must measure the distributional differences.

Measuring the degree of systematicity in a trained network is a problem.
Elman did not conduct a direct test to observe whether the trained network exhibited weak systematicity or equivalent property, because Elman's pioneering study was conducted before Hadley's proposal of weak systematicity. However, Elman showed the result of analysis on activations of hidden units, which effectively showed the categorization of words. The dendrogram in (Elman, 1991) shows the hierarchical structure among words and the similarity between word groups, although it is constructed on the basis of activations summed over all the contexts before the target word. Therefore, there remains the question whether the network really learned the hierarchy or the result is simply a result of statistical analysis of the network's behaviour. In other words, although the network was trained to predict a word, it had not acquired the capability to predict a category (or distribution of possibilities of words) of the subsequent word.
Frank, in (Frank, 2006), adopted a criterion that compares the sum of activations for words in the expected category and the unexpected category. However, a malicious network may not predict a word in an incorrect category but predict only one word among many words in the correct category.
One and only one alternative is to measure the distributional responses in terms of, for example, χ^2 distance or KL divergence.
We have another problem concerning training data. To examine the systematicity of an RNN, Frank selected a rather difficult problem. In (Frank, 2006), the training data include data of type N1 V1 N1 and N2 V2 N2, where N1 and N2 represent the noun categories and V1 and V2 verb category, whereas the test data is of N2 V1 N2 and N1 V2 N2, requiring that a network should induce that words in N1 and N2 are in the same category. The requirement appears to be rather demanding, because no suggestion of equating the position of N1 in the first sentence and N2 in the second and that of V1 and V2 is provided. Since the network inevitably learns the frequencies of words, it may learn to predict N2 after V2, irrespective of the first N1, or to predict N2 in the third place when N2 appears in the first place regardless of V1 in the second place.
It is better to select simpler settings for the training data and test data.
In this section, we examine the learnability of systematicity in a simpler situation. It is hypothesized that the systems of categories exist *a priori*; hence, learners need to only select the best one among them. Suppose that SRNs are furnished with an input conversion sub-network that converts an input word into its category and undergoes learning to predict the subsequent word. We found that an SRN with a conversion sub-network consistent with the grammatical categories of words in input sentences has the smallest learning error in terms of predicting frequency. Consequently, we can recover correct categorization of words by observing the SRN outputs, calculating learning errors, and selecting the SRN with the least learning error.

4.1 Experimental settings: network architecture

In systematicity experiments, SRNs are augmented with a new input layer (Fig. 4 (left)) which corresponds to the primary networks in (Bodén, 2004). The connection weights between the first and second input layers are fixed during learning with 0 (disconnected) or 1 (connected). The activation function of the second input layer is linear and that of the hidden and output units is the standard sigmoid function $1/(1 + \exp(-x))$

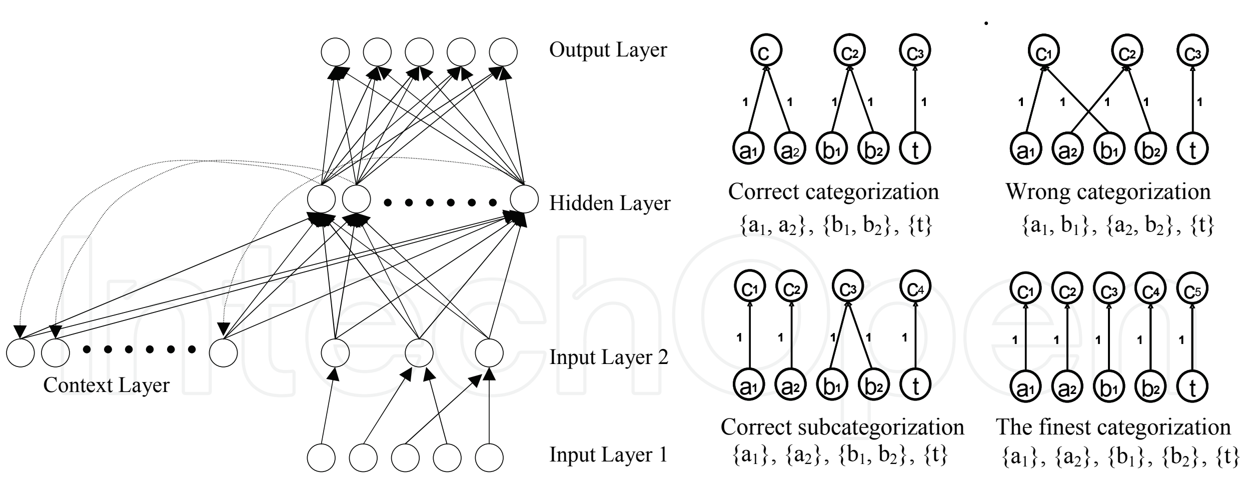

Fig. 4. Architecture experimented (left) and categorization examples (right).

A localist representation or the so-called one-hot vector (a binary vector with a single 1) is adopted for input words as well as categories. The connections between Input Layer 1 and Input Layer 2 convert the former to the latter. These connections are set and fixed to 1 or 0 according to the conversion, because a word belongs to a category or not (the corresponding weight is 1 or 0, respectively). As shown in Fig. 4 (left), the information that the SRN part of the network would know is limited to the categories of the input words.

In the current experiment, the category refers to a mutually exclusive partition of words. Since words and categories are represented by localist representation, each node in Input Layer 1 has exactly one outgoing connection to Input Layer 2, and each node in Input Layer 2 has incoming connections from the nodes designating words of the category. In other words, in this experiment, we suppose that the word in the example sentences belongs to only one category or is derived from at most one non-terminal symbol in derivation. Examples of the categorization are shown in Fig. 4 (right).

As many networks as the possible categorizations were prepared. Then, all the networks were trained while observing their learning errors. It must be noted that the network is robust to overtraining and that a network with the least learning error might possibly be a network with the least generalization error because the networks accept only categorized inputs so that a network with incorrect categorization connections receives inconsistent learning data.

4.2 Experimental settings: target language and learning method

In the current experiments, the task is to learn a diversified centre-embedded language (DCEL) {abt, aabbt, aaabbbt | a = a_1 or a_2, b = b_1 or b_2} (Bodén, 2004), where “t” is the terminal of a string and “a” and “b” correspond to categories. Hereafter, the terms character and string are used instead of word and sentence; further, the density of a category is the number of terminal symbols derived from a non-terminal symbol or equivalently the number of elements or characters belonging to the category; for example, the density of “a” in the above example is 2. The length of the language is the length of the longest sequence in the language (excluding the termination symbol “t”); hence, the length of the language in the above example is 6.

A simple error-backpropagation algorithm with the squared error function is adopted for the learning as in SRN experiments. The learning rate is 0.10 and no acceleration is used.

The generalization capability of the learned networks is evaluated by the Kullback-Leibler divergence (KLd) between two distributions: expectation and realized network output. The former distribution is obtained by supposing the uniform distribution for the above three strings and the characters in the same categories (e.g. {a_1,a_2}). {1/4, 1/4, 1/4, 1/4, 0} is an example distribution for {a_1,a_2,b_1,b_2,t} that appears after a string a_1a_2 or a_2a_1. The latter distribution is obtained by normalizing the network outputs to 1. It must be noted that KLd is used only for evaluation and never used for learning.
The learning data were obtained by first randomly generating a set of strings of terminal symbols, and then randomly re-sampling from the set until the due number of strings is obtained. 90,000 strings were used for each run of training and 10,000 strings were used to calculate the resultant learning error and the generalization error.
15 training sessions for each network were conducted by changing the learning data and its presentation order; the average and the standard deviation of the resultant learning and generalization errors were obtained. Categorizations were varied for each network.
The parameters were varied; the length of the language was 6, 8, and 10; the number of training data was 30 and 60; the density of category was 2, 3, 4, and 5.

4.3 Results

Figs. 5 and 6 summarize some of the results obtained; L denotes the length of the language and D denotes the density of the correct category for "a" and "b" (set equal); Nc denotes the number of categories including "t"; N denotes the number of learning examples. An error bar shows the standard deviation averaged over experiments.

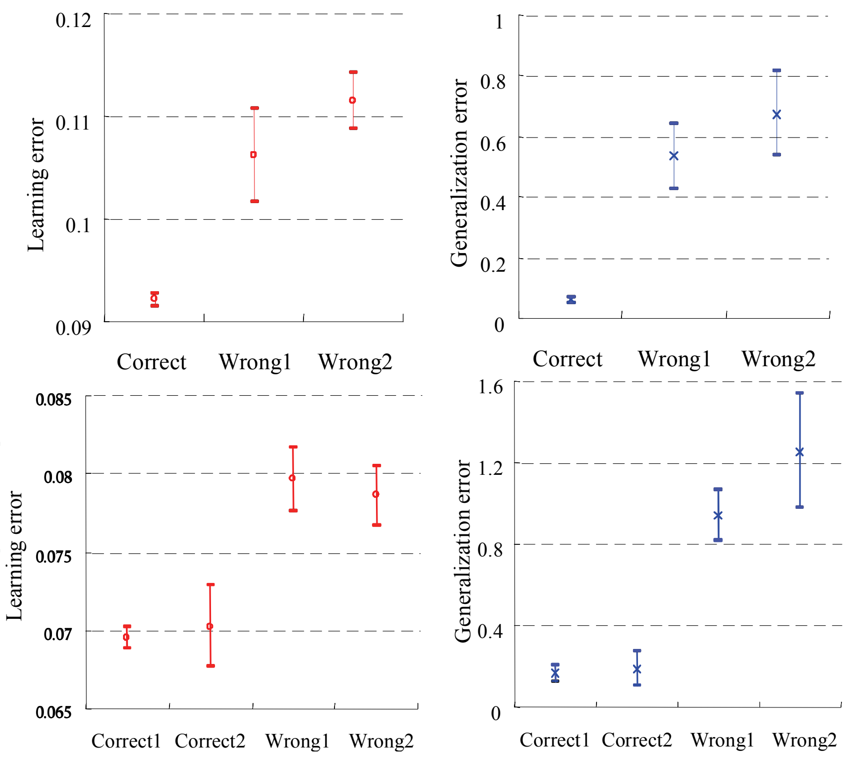

Fig. 5. Top two figures. L = 6, D = 2, Nc = 3, and N = 60. On the horizontal axis, Correct is {a_1,a_2}, {b_1,b_2}, {t}, Wrong1 is {a_1}, {a_2,b_1,b_2}, {t} and Wrong2 is {a_1,b_1}, {a_2,b_2}, {t}. Bottom two figures. L = 8, D = 4, Nc = 4, and N = 60. On the horizontal axis, Correct1 is {a1,a2,a3,a4}, {b_1,b_2}, {b_3,b_4}, {t}, Correct2 is {a_1,a_2,a_3,a_4}, {b_1,b_2,b_3}, {b_4}, {t}, Wrong1 is {a_1,a_2,b_1}, {a_3,a_4,b_2}, {b_3,b_4}, {t} and Wrong2 is {a_1,a_2,b_3,b_4}, {a_1,b_1}, {a_2,b_2}, {t}

The correct categorization consistently gives the smallest of the resultant learning and generalization errors. The correct sub-categorization also gives the smallest values as shown in Fig. 5 (bottom). It is observed that the two different sub-categorizations give the similar learning errors and generalization errors.
In Fig. 6, some results of the experiments for categories with higher density are shown. The difference between the correct categories and the wrong categories in these cases is clearer than that that found in the cases in Fig. 5.

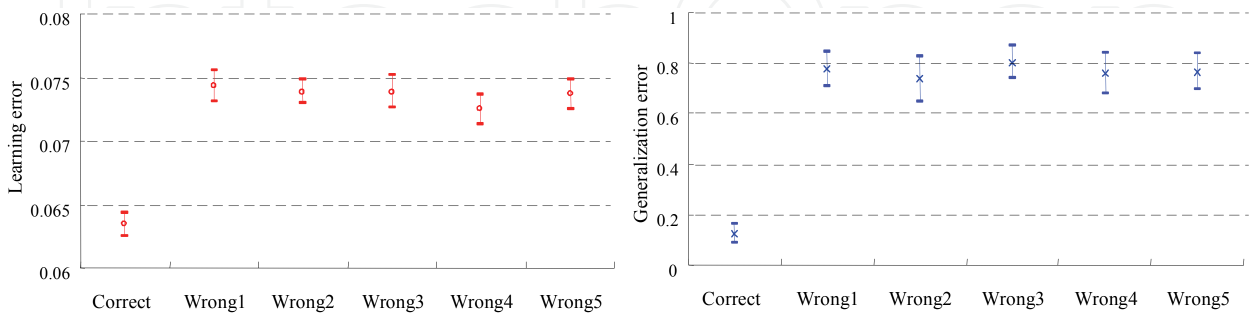

Fig. 6. L = 10, D = 5, Nc = 3, and N = 60. On the horizontal axis, Correct is $\{a_1,a_2,a_3,a_4,a_5\}$, $\{b_1,b_2,b_3,b_4,b_5\}$, $\{t\}$; Wrong1 is $\{a_1,a_2,a_3,b_1,b_2,b_3\}$, $\{a_4,a_5,b_4,b_5\}$, $\{t\}$; Wrong2 is $\{a_1,a_2,b_3,b_4,b_5\}$, $\{a_3,a_4,a_5,b_1,b_2\}$, $\{t\}$; Wrong3 is $\{a_1,a_2,b_1,b_2\}$, $\{a_3,a_4,a_5,b_3,b_4,b_5\}$, $\{t\}$; Wrong4 is $\{a_1,a_2\}$, $\{a_3,a_4,a_5,b_1,b_2,b_3,b_4,b_5\}$, $\{t\}$; Wrong5 is $\{a_1,b_1\}$, $\{a_2,a_3,a_4,a_5,b_2,b_3,b_4,b_5\}$, $\{t\}$.

4.4 Discussion on systematicity

Networks with systematicity that conforms to the systematicity in the target language can be differentiated from the networks without it, because networks with correct categorizations or sub-categorizations can be found by referring to their learning error, simple and observable quantity, as stated in the previous section.
To acquire word meanings (or the extent of category of objects/movements/events that a word means), it has been argued that humans use a lot of tactics that constrain generalizations to an appropriate level (e.g., (Markman, 1990)). It is noteworthy that no workable model is possible if we require that humans should interact with external environments to use the tactics.
Acquisition of grammatical categories of word is a different problem than acquisition of word meanings, because grammatical categories exist only for grammars, which suggests that humans use different tactics. Pinker suggests that humans utilize bootstrapping interactions between syntax and semantics development as a key factor (Pinker, 1984). However, if the semantics should concern with external environments, no workable model is possible.
Conventional approaches have not yet provided satisfactory results. For example, although distributional clustering is one of the promising approaches, the clusters corresponding to grammatical categories that the algorithm provides are unsatisfactory (Clark, 2000). However, if we could utilize the fact that grammatical categories exist only for grammars, as in the case, it may be possible to induce grammatical categories solely from written texts.
In the current experiments, based on these observations, we proposed to hypothesize the existence of innate systems of categorizations. Although the method and results reported here are still very primitive, they suggest that the hypothesis is promising.

5. Conclusion

Productivity is the key property of a natural language. Learnability is an equally important property, since productivity without learnability will not help the transfer of a language beyond generations. Linguistic productivity is supported by recursiveness described in terms of phrasal categories and systamaticity described in terms of lexical categories.
Recursiveness is realizable by an SRN; however, the realized function is limited to a counter capable of counting up and down just once. Hence, for example, parsing consecutive embedded sentential phrases requires a reset of counters or stacks. The difficulty of learning of recursiveness is observed experimentally and theoretically.
Human learns recursive grammar in his/her first language which requires stacks to parse. We have to determine an alternative mechanism or appropriate a priori knowledge along with an additional method that embeds the knowledge in the learning mechanism.
Systematicity is learnable with an RNN by using added sub-networks, which is equivalent to one-stage learning algorithms composed of clustering of words into categories and learning of grammars written in lexical (and phrasal) categories. Learnability is observed when frequency information that exist in training corpus and influence learning results is not transferred to the SRN; this shows the possibility of grammatical induction when systematicity exists.
The difficulty in learning systematicity in more realistic situations is rooted in ambiguous words and varied contexts. An ambiguous word is a word, for example, which could be a noun in a sentence but a verb in another sentence. Varied contexts are contexts which induce the same category but have very different forms, for example, a noun could be followed by almost any lexical category.
Research is being conducted on the learnability of productive grammars.

6. References

Bodén, M.; Wiles, J.; Tonkes, B. & Blair, A. D. (1999). Learning to predict a context-free language: Analysis of dynamics in recurrent hidden units, *Proceedings of ICANN'99*, pp. 359-364, Edinburgh, 1999

Bodén, M. & Wiles, J. (2000). Context-free and context-sensitive dynamics in recurrent neural networks, *Connection Science*, Vol. 12, No. 3-4, (Dec. 2000), pp. 197-210

Bodén, M. & Blair, A. (2003). Learning the dynamics of embedded clauses, *Applied Intelligence*, Vol. 19, No. 1-2, (July 2003), pp. 51-63

Bodén, M. (2004). Generalization by symbolic abstraction in cascaded recurrent networks, *Neurocomputing*, Vol. 57, pp. 87-104

Casey, M. (1998). Correction to proof that recurrent neural networks can robustly recognize only regular languages, *Neural Computation*, Vol. 10, No. 5, (July 1998), pp. 1067-1069

Chalup, S. K.. & Blair, A. D. (2003). Incremental training of first order recurrent neural networks to predict a context-sensitive language, *Neural Networks*, Vol. 16, pp. 955-972

Chomsky, N. (1959a). A Review of B.F. Skinners verbal behavior, *Language*, Vol. 35, pp. 26-58, Reprinted in: Chomsky, N. (1965). *Aspects of the theory of syntax*, M.I.T. Press, Cambridge, Mass

Chomsky, N. (1959b). On certain formal properties of grammars, *Information and Control*, Vol. 2, pp. 137–167

Chomsky, N. (1980). *Rules and representations*, Columbia University Press, New York

Christiansen, M. H. & Chater, N. (1994). Generalization and connectionist language learning, *Mind and Language*, Vol. 9, pp. 273-287

Clark, A. (2000). Inducing syntactic categories by context distribution clustering, *Proceedings of CoNLL-2000 and LLL-2000*, Lisbon, Portugal, (Sept., 2000)

Elman, J. L. (1990). Finding structure in time, *Cognitive Science*, Vol. 14, pp. 179-211

Elman, J. L. (1991). Distributed representations, simple recurrent networks, and grammatical structure, *Machine Learning*, Vol. 7, pp. 195-224

Elman, J. L. (1995). Language as a dynamical system, In: *Mind as Motion: Explorations in the Dynamics of Cognition*, Port, R. F. & van Gelder, T. (Eds.), pp. 195—223, MIT Press, Cambridge, MA

Farkaš, I. & Crocker, M. W. (2008). Syntactic systematicity in sentence processing with a recurrent self-organizing network, *Neurocomputing*, Vol. 71, No. 7-9, pp. 1172-1179

Fodor, J. A. & Pylyshyn, Z. (1988). Connectionism and cognitive architecture: a critical analysis, *Cognition*, Vol. 28, pp. 3-71

Frank, S. L. (2006). Learn more by training less: Systematicity in sentence processing by recurrent networks, *Connection Science*, Vol. 18, pp. 287-302

Gers, F. A. & Schmidhuber, J. (2001). LSTM Recurrent Networks Learn Simple Context Free and Context Sensitive Languages, *IEEE Transactions on Neural Networks*, Vol. 12, No. 6, pp. 1333-1340

Gold, E. M. (1967). Language identification in the limit, *Information and Control*, Vol. 10, No. 5, pp. 447-474.

Guckenheimer, J. & Holmes, P. (1997). *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*, Corr. 5th print., Applied Mathematical Sciences, 42, Springer

Hadley, R. (1994). Systematicity in connectionist language learning, *Mind and Language,* Vol. 9, No. 3, pp. 247–272

Hopcroft, J. & Ullman, J. (1979). *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley

Iwata, A.; Shinozawa, Y. & Sakurai, A. (2007). A Characterization of simple recurrent neural networks as a language recognizer, *Proceedings of 14th International Conference on Neural Information Processing*, (Nov. 2007), Kitakyushu, Japan

Katok, A. & Hasselblatt, B. (1996). *Introduction to the Modern Theory of Dynamical Systems*, Cambridge University Press

Maass, W. & Orponen, P. (1998). On the effect of analog noise in discrete-time analog computations, *Neural Computation*, vol. 10, pp. 1071-1095

Markman, E. M. (1990). Constraints children place on word meanings, *Cognitive Science*, Vol. 14, pp. 154-173

Pinker, S. (1984). *Language learnability and language development*, Harvard University Press, (1984), Cambridge

Rodriguez P.; Wiles J. & Elman J. L. (1999). A Recurrent neural network that learns to count, *Connection Science*, Vol. 11, No. 1, (March 1999), pp. 5-40

Rodriguez, P. (2001). Simple recurrent networks learn context-free and context-sensitive languages by counting, *Neural Computation*, Vol. 13, No. 9, pp. 2093–2118

Schmidhuber, J.; Gers, F. & Eck D. (2002). Learning nonregular languages: a comparison of simple recurrent networks and LSTM, *Neural Computation*, Vol. 14, No. 9, (Sept. 2002), pp. 2039-2041

Schütze, H. (1993). Part-of-speech induction from scratch. *Proceedings of ACL 31*, pp. 251–258, Ohio State University

Siegelmann, H. T. (1998). *Neural Networks and Analog Computation: Beyond the Turing Limit*, Birkhauser, ISBN 0-8176-3949-7, Boston

Suhara, Y. & Sakurai, A. (2006). Generalization by categorical nodes in recurrent neural networks, In: *International Congress Series*, Vol. 1291, pp. 161-164

Van derVelde, F.; Van derVoort van der Kleij, G. T. & De Kamps, M. (2004). Lack of combinatorial productivity in language processing with simple recurrent networks, *Connection Science*, Vol. 16, pp. 21–46

Wiles, J.; Blair, A. D. & Bodén, M. (2001). Representation beyond finite states: alternatives to push-down automata. In: A Field Guide to Dynamical Recurrent Networks, Kolen, J.F. & Kremer, S.C. (Eds.), pp. 129-142, IEEE Press, New York