Proceedings of the 2007 IEEE Symposium on
Foundations of Computational Intelligence (FOCI 2007)

Random Hypergraph Models of Learning and Memory in Biomolecular Networks: Shorter-Term Adaptability vs. Longer-Term Persistency

Byoung-Tak Zhang

Abstract— **Recent progress in genomics and proteomics makes it possible to understand the biological networks at the systems level. We aim to develop computational models of learning and memory inspired by the biomolecular networks embedded in their environment. One fundamental question is how the systems rapidly adapt to their changing environment in a short period (learning) while performing persistently through the longer time span (memory). We study this issue in a probabilistic hypergraph model called the hypernetworks. The hypernetwork architecture consists of a huge number of randomly sampled hyperedges, each corresponding to higher-order micromodules in the input. We find that a system consisting of a large number of a wide range of heterogeneous low-dimensional components has a fairly competitive chance of long-term survival (memory, persistency) and short-term performance (learning, adaptability) as opposed to a system consisting of a small number of high-dimensional, fine-tuned, complex components. Empirical evidence is offered to support these findings and theoretical explanations are given.**

I. Introduction

Cells are highly complex information processing systems. Much progress has been made in elucidating the molecular mechanisms of individual cells during the last decades. In particular, recent progress in genomics and proteomics makes it possible to understand the biological units and processes at the systems level. These include gene regulation networks [7], metabolic pathways [5], signal transduction networks [2][8], and other protein interaction networks [9]. For example, Figure 1 shows an example regulatory network in a human cell [1], where a small number of genes interact more closely than others and the whole network builds an Internet-like hub structure.

These findings may shed light on the design principles of biological information processing systems which might be useful for building artificial information processing systems. For example, in contrast to the typical problem settings in machine learning, the biological learning systems are faced with continuous exposure to the environment. Simple organisms such as bacteria should strive to adapt to the changes in their resources of both beneficial (food) and harmful (poison) nature to swim and to survive [4]. Also, it is interesting to know how the biochemical networks maintain the stability while adapting to the cellular environmental changes. The life time of each molecule is very limited but the system or the population of the biomolecules maintains its robustness [14].

Biointelligence Laboratory (http://bi.snu.ac.kr/), School of Computer Science and Engineering and Graduate Programs in Bioinformatics, Brain Science, and Cognitive Science, Seoul National University, Seoul, 151-744 Korea. E-mail: btzhang@bi.snu.ac.kr

We aim to develop computational systems by mimicking the organizational and processing principles underlying the biomolecular networks. We ask the fundamental question of how the organisms adapt to the short-term environmental change (learning issue) while maintaining the longer-term persistency in performing a task (memory issue). Here we hypothesize that a key to the biological solution to this challenge is the diversity of the variety, size, and complexity of the molecular species and structures in the cell (and the organism). The biochemical signal transduction networks, for example, consist of various molecular species interacting with each other massively yet specifically [12]. The molecules are relatively simple, but their massive interaction in a network may exhibit complex, emergent behavior [2].

We use the random hypergraph models [6] to study the hypothesis that the diversity of the system in its organizational structure plays an important role in their adaptability in a short term and survivability in a long term. In Section II, we define the hypernetwork architecture. In Section III, we use the random graph process to build the hypernetwork structures that model the training data coming from the environment. The resulting network structures and their performances are examined in Section IV to study the relationship between the structural and functional properties. Section V summarizes the results and discusses future work.

II. The Hypernetwork Architecture

Hypernetworks are a generalization of the hypergraphs. A hypergraph is an undirected graph G whose edges connect a non-null number of vertices [3], i.e. $G = (X, E)$, where $X = \{x_1, x_2, ..., x_n\}$, $E = \{E_1, E_2, ..., E_m\}$, and $E_i = \{x_{i_1}, x_{i_2}, ..., x_{i_k}\}$. E_i is called the hyperedges. Mathematically, E_i is a set and its cardinality (size) is $k \geq 1$, i.e., the hyperedges can connect more than two vertices while in ordinary graphs the edges connect up to two vertices, i.e., $k \leq 2$. A hyperedge of cardinality k will be referred to as a k-hyperedge.

For example, a hypergraph may consist of seven vertices $X = \{x_1, x_2, ..., x_7\}$ and five hyperedges $E = \{E_1, E_2, E_3, E_4, E_5\}$ each having a different cardinality. A hypergraph can be represented as an incidence matrix. The incidence matrix of a hypergraph $G = (X, E)$ is a matrix $[a^i_j]$ with m rows that represent the hyperedges of G and n

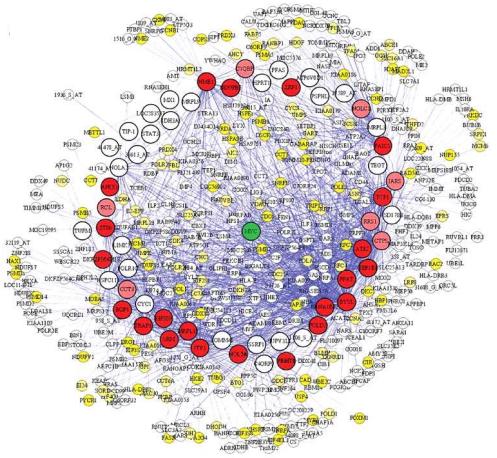

Fig. 1. An example of regulatory networks in the cell. This *MYC* subnetwork in a human B cell shows some micromodules, i.e. groups of a small number of genes accounting for many of the connections. The network contains 56 genes directly connected to *MYC*, the hub, and 444 genes connected thorough an intermediate. The size of each circle is proportional to the number of the gene interactions. Source: *Nature Genetics* [1]

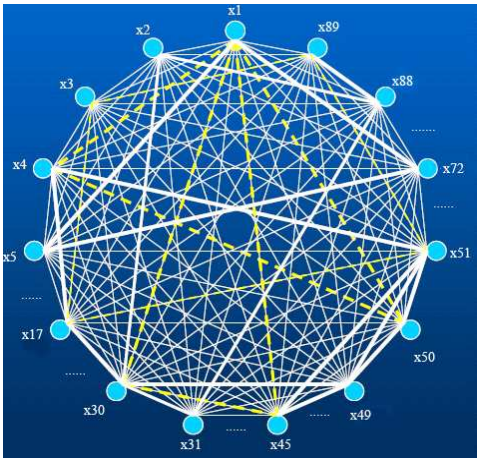

Fig. 2. A hypernetwork $H = (X, E, W)$ is a weighted hypergraph consisting of a set X of vertices, a set E of hyperedges, and a set W of weights. In contrast to ordinary graphs, a hypergraph consists of edges of cardinality $1 \leq k \leq n$, where $n = |X|$. In this figure, each hyperedge E_i is represented as a polygon connecting the vertices in it. The thickness of each hyperedge is proportional to its weight w_i.

columns that represent the vertices of G, such that $a^i_j = 1$ if $x_j \in E_i$ and $a^i_j = 0$ if $x_j \notin E_i$.

A hypernetwork is a hypergraph that is augmented with a weight value to each hyperedge (Figure 2). Formally, we define a hypernetwork as a triple $H = (X, E, W)$, where $X = \{x_1, x_2, ..., x_n\}, E = \{E_1, E_2, ..., E_m\}, W = \{w_1, w_2, ..., w_m\}$, where $E_i = \{x_{i_1}, x_{i_2}, ..., x_{i_k}\}$. A k-hypernetwork consists of a set X of vertices, a subset E of $X[k]$, and a set W of hyperedge weights, where $E = X[k]$ is a set of subsets of X whose elements have precisely k members. A hypernetwork H is said to be k-uniform if every edge E_i in E has cardinality k. A hypernetwork H is k-regular if every vertex has degree k. Note that an ordinary graph is a 2-uniform hypernetwork with $w_i = 1$.

From the biological network point of view, the hyperedges in a hypernetwork can be viewed as building blocks, such as for pathways, motifs, modules, and circuits [11][13][16]. For example, the set of a small number of genes in Figure 1 can be represented as a hyperedge in the hypernetwork model shown in Figure 2. In the hypernetwork model, the coupling strength of the vertices in the hyperedge is represented by the weight value associated with the hyperedge. In this sense the random hypernetwork structure can be used to model and identify primitive "micromotifs" or higher-order "micromodules" to build massively-interacting "microcircuits".

Formally, the hypernetworks can be used as a probabilistic memory to store a data set $D = \{\mathbf{x}^{(n)}\}_{n=1}^{N}$ so that they can be retrieved later by content, where $\mathbf{x}^{(n)}$ denotes the n-th pattern to store. We define the energy of the hypernetwork as

$$\mathcal{E}(\mathbf{x}^{(n)}; W) = \sum_{i=1}^{|E|} w_{i_1 i_2 ... i_{|E_i|}} x^{(n)}_{i_1} x^{(n)}_{i_2} ... x^{(n)}_{i_{|E_i|}}, \qquad (1)$$

where W represents the parameters (hyperedge weights) for the hypernetwork model. Note that $x^{(n}_{i_1} x^{(n)}_{i_2} ... x^{(n)}_{i_{|E_i|}}$ is a combination of $k = |E_i|$ elements of the data item $\mathbf{x}^{(n)}$ which is represented as a k-hyperedge in the network.

Then, the probability of the data being generated from the hypernetwork is given as Gibbs distribution

$$P(\mathbf{x}^{(n)}|W) = \frac{1}{Z(W)} \exp\left\{-\mathcal{E}(\mathbf{x}^{(n)}; W)\right\}, \qquad (2)$$

where $\exp\left\{-\mathcal{E}(\mathbf{x}^{(n)}; W)\right\}$ is called the Boltzmann factor and the normalizing term $Z(W)$ is expressed as

$$Z(W) = \sum_{\mathbf{x}^{(m)}} \sum_{i=1}^{|E|} w_{i_1 i_2 ... i_{|E_i|}} x^{(m)}_{i_1} x^{(m)}_{i_2} ... x^{(m)}_{i_{|E_i|}}. \qquad (3)$$

In effect, the hypernetwork represents a probabilistic model of the data set using a population of hyperedges and their weights.

It is interesting to note the relationship of the hypernetwork architecture and the traditional neural network or machine learning models. To see the connection, we define the potential function associated with the hyperegdes as

$$\Phi_{E_i}(\mathbf{x}) = \prod_{x_j \in E_i} x_j = x_{1_1}, x_{i_2}, ..., x_{i_{|E_i|}}, \qquad (4)$$

where $k = |E_i|$ is the size of the ith hyperedge. When the variables x_i are binary-valued, this potential function computes a logical conjunction of the variables. Using this potential function, the energy function can be expressed as

$$\mathcal{E}(\mathbf{x}; W) = \sum_{i=1}^{|E|} w_{i_1 i_2 ... i_{|E_i|}} \Phi_{E_i}(\mathbf{x}), \qquad (5)$$

which is a weighted sum of the potential functions. The weighting has a smoothing effect to make the logical decisions more robust [15]. If the x-variables take real values,

this results in a higher-order polynomial networks [18]. Thus the hypernetwork structure in this case is a generalization of the conventional polynomial networks where the degree and order of the polynomials are arbitrary. Allowing only k-hyperedges, the hypernetwork represents a polynomial network of order k.

The similarity to the multilayer perceptron is to be seen by defining a sigmoid potential function, i.e.,

$$\Phi_{E_i}(\mathbf{x}) = \frac{1}{1+\exp\left\{-\sum_{x_j \in E_i} w_j x_j\right\}}, \tag{6}$$

where $k_i = |E_i|$ is the size of receptive field of the sigmoid unit. In general, different hyperedges have different k_i and k_i is smaller than the number of inputs n, i.e. $k_i < n$, $k_j < n$, and $k_i \neq k_j, i \neq j$ for most of i and j. This heterogeneous structure of the hypernetwork of sigmoid units is contrasted with the conventional sigmoid neural networks where the structure is homogeneous with a fixed receptive field size n for all the neurons.

The hypernetwork can emulate the radial basis function networks by defining a Gaussian potential function:

$$\Phi_{E_i}(\mathbf{x}) = \exp\left\{-\frac{1}{\sigma_i}||\mathbf{x}^{(i)} - \mathbf{x}||_{E_i}\right\}, \tag{7}$$

where $||\mathbf{x}^{(i)} - \mathbf{x}||_{E_i} = \sum_{x_j \in E_i} x_j^{(i)} x_j$. Again, the hypernetwork takes heterogeneous radial basis functions where the dimension of the basis function is variable.

Figure 3 compares the output functions for the hyperedges of different potential functions. Also shown is the effect of the size of hyperedges. When the hyperedges are small, i.e. for small $k = |E_i|$, the receptive fields are narrow and thus the hypernetwork builds a representation consisting of low-dimensional, general components (micromodules). When the hyperedges are large, the hypernetwork builds a representation consisting of high-dimensional, specialized components. To see the profile of distribution we consider the histogram of k-hyperedges within a hypernetwork. We define the *hypergram* of a hypernetwork $H = (X, E, W)$, denoted $\mathbf{h}^{(k_1 \ldots k_2)}$, as the weight spectrum of the k-hyperedges for $k = k_1, ..., k_2$:

$$\mathbf{h}^{(k_1 \ldots k_2)} = \{(w_{k_1}, w_{k_1+1}, ..., w_{k_2}) | w_i = |E_i|, E_i \in E\}, \tag{8}$$

where E_i are the hyperedge set of the hypernetwork H. In Section IV, we anlyze the hypergrams on various problems.

III. Generating Random Hypernetworks from Data

For a k-hypergraph, the number of possible edges are

$$|E| = C(n,k) = \frac{n!}{k!(n-k)!}, \tag{9}$$

where $n = |X|$ and $C(n,k)$ denotes the number of cases to choose k elements from a population of n elements. If we denote the set of all graphs as Ω, its size is

$$|\Omega| = 2^{C(n,k)}. \tag{10}$$

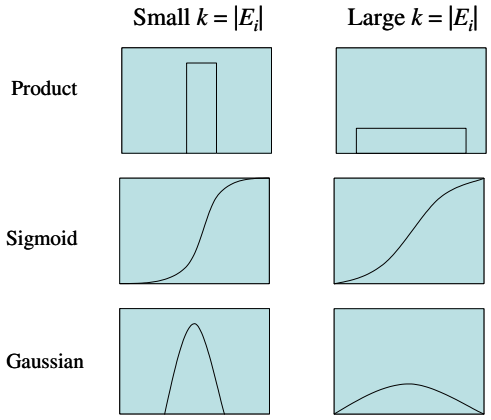

Fig. 3. Three examples of potential (basis) functions to be associated with the hyperedges. The potential functions with small k-hyperedges receive inputs from a narrow range (in dimensions) while those with large k-hyperedges observe a wide range of the input space. Thus, changing the parameter k in the random hypernetworks has an effect of varying the receptive-field size in neural networks.

Since a hypernetwork is a multiset of hypergraphs, the size of k-hypernetworks $H^{(k)}$ is the multiples of $|\Omega|$. For the class of $(0,n)$-hypernetworks, the number of possible hyperedges is the multiples of

$$|E| = \sum_{k=0}^{n} C(n,k) = \sum_{k=0}^{n} \frac{n!}{k!(n-k)!} = 2^n, \tag{11}$$

and the size of the space of $(0,n)$-hypernetworks is

$$|\Omega| = 2^{\kappa \cdot 2^n}, \tag{12}$$

where κ is the maximum number of copies of an hyperedge to appear in a hypernetwork.

We now introduce a stochastic method to search this space. A random graph is a graph constructed by a random procedure [6]. A random graph model chooses a graph at random, with equal probabilities, from the set of all 2^n graphs whose vertex set is $[n] = \{1, 2, ..., n\}$. We consider a probability space

$$(\Omega, \mathcal{F}, \mathcal{P}), \tag{13}$$

where Ω is the set of all graphs with vertex set $[n]$, $\mathcal{F}$ is the family of all subsets of Ω, and to every $\omega \in \Omega$ we assign its probability as

$$\mathcal{P}(\omega) = 2^{-C(n,k)}. \tag{14}$$

The probability space can be viewed as the product of $C(n,k)$ binary spaces. It is a result of $C(n,k)$ independent tosses of a fair coin, i.e. Bernoulli experiments.

The random hypernetworks can be generated by a binomial random graph process. Given a real number p, $0 \leq p \leq 1$, the binomial random (hyper)graph, denoted by $\mathcal{G}(n,p)$, is defined by taking as Ω the set of all hypergraphs on vertex set $[n]$ and setting

$$P(G) = p^{|E(G)|}(1-p)^{C(n,k)-|E(G)|}, \tag{15}$$

where $|E(G)|$ stands for the number of edges of G. The hypernetworks are generated by repeating the random hypergraph process.

Alternatively, we can generate the random hypergraphs using a uniform random graph process. Given an integer M, $0 \leq p \leq C(n,k)$, the uniform random hypergraph, denoted by $\mathcal{G}(n,M)$, is defined by taking as Ω the family of all graphs on the vertex set $[n]$ with exactly M edges, and as $\mathcal{P}$ the uniform probability on Ω,

$$P(G) = C(C(n,k),M)^{-1}, \tag{16}$$

where $G \in \Omega$. The hypernetworks are then generated by repeating the random hypergraph process. The binomial random graph model and the uniform random graph model are known to be asymptotically equivalent if $C(n,k)p$ is close to M [6].

One difference in our approach from the typical random graph process is that we want to build a model of an environment rather than simply generate a random graph structure. This is where the role of training data $D = \{\mathbf{x}\}$ comes in. The basic idea is that, instead of inserting the structure of hyperedge, the random graph process inserts into the hypernetwork a hyperedge that is instantiated by the training sample. The whole procedure for building a k-hypernetwork that fits a given data set D is summarized as follows.

- 1. Start with the initial hypernetwork $H = (X, E, W) = (\emptyset, \emptyset, \emptyset)$.
- 2. Get a training sample $\mathbf{x} \in D$. Generate a hypernetwork $H' = (X', E', W')$ as follows:
  - Generate hyperedges (duplication permitted), E_i, of cardinality k from $\mathbf{x}$ by a random hypergraph process.
  - $E' \leftarrow E' \cup \{E_i\}$.
  - $W' = \{w_i | w_i = |E_i|, i = 1, ..., |E'|\}$.
  - $X' = \{x_j | x_j \in E_i \in E'\}$.
- 3. $H \leftarrow H \cup H'$.
- 4. Go to step 2 if not terminated.

The repeated application of the random graph process for the samples in the training set D results in resampling the training data onto the hypernetwork structure. When the problem is a supervised learning task, we can modify the random process to add an error correction procedure. It can be shown that this biased random process performs gradient search to find maximum-likelihood parameters for the training data set. To see this, given a set $D = \{\mathbf{x}^{(n)}\}_{n=1}^{N}$ of n independently and identically distributed examples, we consider the likelihood of the parameters W:

$$P(D|W) = \prod_{n=1}^{N} P(\mathbf{x}^{(n)}|W), \tag{17}$$

where $P(\mathbf{x}^{(n)}|W)$ has the form in Eqn. (2) and W consists of the weights or the number of copies of the hyperedges of size k. Taking the logarithm of the likelihood we obtain

$$\begin{aligned} & \ln P(D|W) \qquad (18) \\ = \; & \ln \prod_{n=1}^{N} P(\mathbf{x}^{(n)}|W) \\ = \; & \sum_{n=1}^{N} \left\{ \left[\sum_{i_1,...,i_{|E_i|}} w_{i_1...i_{|E_i|}} x_{i_1}^{(n)} ... x_{i_{|E_i|}}^{(n)} \right] - \ln Z(W) \right\}. \end{aligned}$$

We take the derivative of the log-likelihood

$$\frac{\nabla}{\nabla w_{i_1,...,i_{|E_i|}}} \ln \prod_{n=1}^{N} P(\mathbf{x}^{(n)}|W) \tag{19}$$

which leads to a learning rule (see [17] for derivation)

$$N \left\{ \left\langle x_{i_1} ... x_{i_{|E_i|}} \right\rangle_{Data} - \left\langle x_{i_1} ... x_{i_{|E_i|}} \right\rangle_{P(\mathbf{x}|W)} \right\}, \tag{20}$$

where the two terms are defined as

$$\left\langle x_{i_1} ... x_{i_{|E_i|}} \right\rangle_{Data} = \frac{1}{N} \sum_{n=1}^{N} \left[x_{i_1}^{(n)} ... x_{i_{|E_i|}}^{(n)} \right] \tag{21}$$

$$\left\langle x_{i_1} ... x_{i_{|E_i|}} \right\rangle_{P(\mathbf{x}|W)} = \sum_{\mathbf{x}} \left[x_{i_1} ... x_{i_{|E_i|}} P(\mathbf{x}|W) \right] \tag{22}$$

Eqn. (20) suggests that maximum-likelihood is achieved by reducing the difference between the average frequencies of the hyperedges in the data set and in the hypernetwork model, as was described above.

IV. Adaptability vs. Persistency

The random hypernetwork architecture is inspired by the complex, heterogeneous organization of biomolecular networks in nature. We are interested to understand and simulate the process by which a complex randomized system organizes itself to a structured system to perform a task persistently while adapting to be robust against temporary perturbations from the environment. For empirical study, here we use various data sets as a surrogate for the environment. These include 3760 digit images (of 64 bits each), 165 face images (480 bits), and 120 gene expression samples (12600 bits) [19]. The characteristics of these data sets interesting to us in this study is they represent highly noisy and corrupted environments. Based on the data we build hypernetworks by the procedure described in the previous section, and the structural and functional properties of the networks are examined to see what factors are crucial. In particular, we anlyze the hypergrams to see the relationship between the diversity of the hyperedges and the performance of hypernetworks for a wide range of tasks.

We use the hypernetwork model consisting of hyperedges with simple product potential functions. The random graph process generates random subsets of variables whose values are sampled from the training data. The use of a large number of relatively simple yet heterogeneous hyperedges may test the potential role of diversity and flexibility of representation

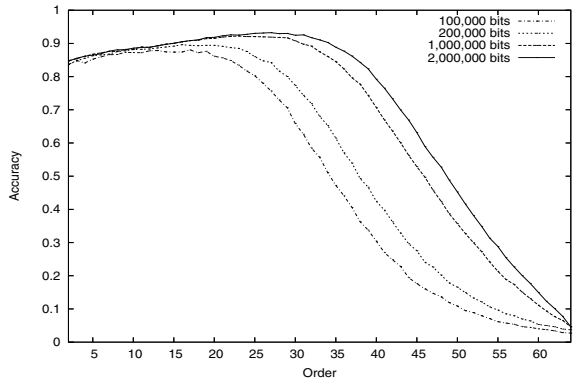

Fig. 4. The hypergram for the digit image data showing the performance profile for k-hypernetworks, i.e. uniform hypernetworks of k-hyperedges, for $k = 1, ..., 64$. The four curves are for the different settings of the maximum network size. The overall result indicates that microcircuits consisting of hyperedges of size $k = 1$ to 20 (30 for large networks) constructed out of 64-bit images contain high information content for this specific data set.

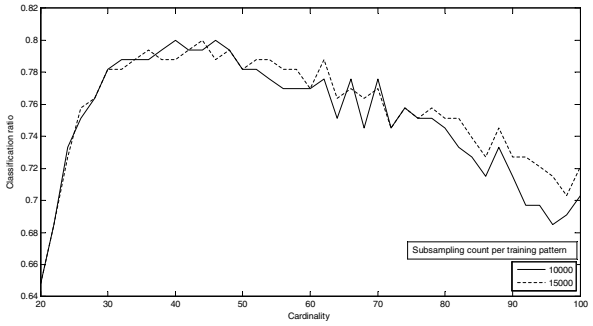

Fig. 5. The hypergram for the face image data showing the performance profile for k-hypernetworks, i.e. uniform hypernetworks of k-hyperedges, for $k = 20, ..., 100$. The two different curves are for the different settings of the maximum network size. The overall result indicates that microcircuits consisting of hyperedges of size $k = 20$ to 50 constructed out of 480-bit images contain high information content for this specific data set. The general shape of the hypergram for this data is very similar to that for the digit data, except for the regions of low-k values.

in adaptivity of the whole system. This can test the hypotheses that the organizational complexity of biological networks is a source of its adaptability and survivability.

Figure 4 shows the hypergram for the face image data showing the performance profile of k-hypernetworks for $k = 1, ..., 64$. The training data for this problem contains 2630 examples and the test set contains 1130 examples of 64-bit image. It is interesting that the hypernetworks show a good performance in the low-k range, meaning there are useful modules in the low-dimensional subspace. The overall result suggests that microcircuits consisting of hyperedges of size $k = 1$ to 20 (or 30 for large network sizes) contain high information content for this specific data set. Note that the performance degrades for $k > 20$ (and $k > 30$ for large networks). This seems attributed to the fact that as k grows the probability of a test sample being matched to a training

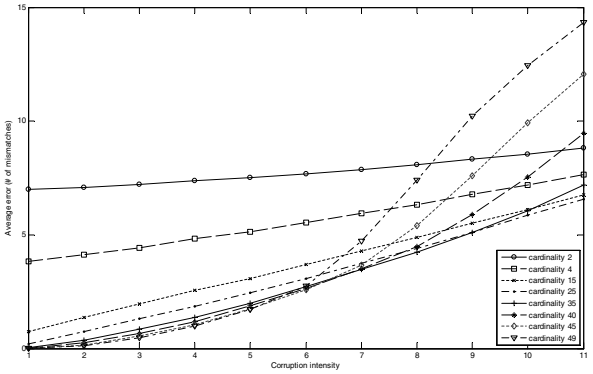

Fig. 6. The effect of corrupted inputs on the persistent behavior of the hypernetworks, compared for various sizes of micromodules, i.e. hyperedges of size ranging $k = 2$ to 49 for the digit image data. The hypernetworks consisting of larger micromodules (hyperedges of larger cardinalities) are more susceptible to data corruption (or environmental perturbations). The hypernetworks consisting only of very small micromodules, say $k = 2$ in this specific setting, lack the ability to adapt and perform well.

sample gets smaller.

For this problem, $|X| = 64$ and the size of the full search space is $|E| = 2^{64}$. According to this simulation result, for $k = 20$, the k-hypernetwork consisting of $|E(H^{(k)})| = 100,000$ hyperedges $(= 2,000,000/20)$ achieves a good performance. The effectivenss of this search can be expressed as the ratio $r = \frac{|E(H^{(k)})|}{|E|} = \frac{100,000}{2^{64}} << 1$. This suggests that, though the full search space is intractably big, the random hypernetwork manages to handle this problem. We have compared the performance of hypernetworks with the state-of-the-art machine learning models, such as multilayer perceptrons, naive Bayes classifiers, decision trees, k-nearest neighbors, and support vector machines, and obtained very competitive results. The best performance of 95.1 % was obtained by k-nearest neighbors with $k = 3$, and the hypernetworks obtained 94.4 % with uniform hyperedge size of 33. The hypernetworks outperformed all the other methods.

Figure 5 shows a similar trend for the face image data set. This task consists of 150 face images for training and 15 test images, each consisting of 480 bits (Yale face data). The hypernetworks generated by the random graph process biased by the training set achieved a competitive performance for the range of $k = 20$ to 50. The complete search space is of size 2^{480}. For k-hypernetworks, the effective ratio of search for $k = 30$ is $r = \frac{|E(H^{(k)})|}{|E|} = \frac{1,500,000}{2^{480}} << 1$, where the number 1,500,000 comes from 10,000 hyperedges $\times$ 150 images (of 15 people).

The effect of noise in sampling and performance was investigated. Biological cells seem robust against environmental noise and perturbations [9][12]. We test this effect in our surrogate data by randomly corrupting the digit images. Figure 6 shows the classification error (y-axis) vs. the corruption intensity (x-axis) for various k values. As expected, the error rate increases as the corruption intensity increases. The error increases more dramatically for large k-hypernetworks than for small k-hypernetworks. This implies

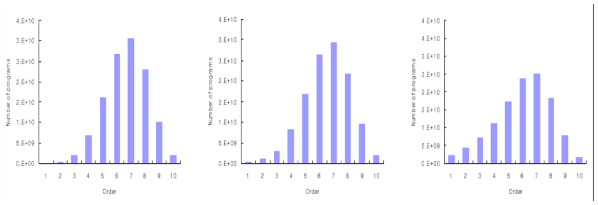

Fig. 7. Evolution of the hypergrams for learning the gene expression data. The left panel shows the histogram of k-hyperedges in the initial hypernetwork. The next two right panels show the hypergrams for the next two stages of evolutionary learning. As learning proceeds, the frequency of small hyperedges tends to increase while that of large hyperedges tends to decrease. No explicit complexity penalty was enforced.

that high k-hypernetworks are more susceptible to data corruption. Conversely, the low k-hyperedges are more robust to build a reliable system when faced with noisy environments.

The diversity of the system components can be increased further by incorporating hyperedges of various k's. Figure 7 shows the evolution of the hypergrams in the complete 10-hypernetwork. The data set came from microarray experiments. Ten genes were selected to build a hypernetwork classifier for disease diagnosis. The learning process started with a complete, but randomly instantiated hypernetwork and the hyperedges in the current network are amplified or retracted depending on their match with the hyperedges sampled from the training data. The training proceeded by repeatedly sweeping the training set to sample more hyperedges. Each sweep constitutes an epoch. The figure shows the general trend that, as learning proceeds, the frequency of small hyperedges tends to increase while that of large hyperedges tends to decrease. This seems to conform to the Occam's razor principle, i.e. simple models should be preferred to complex models when everything else is the same [18]. This is especially interesting since the biased random graph process tends to reduce the complexity of the system even though we did not enforce any explicit complexity penalty. This seems because smaller hyperedges are less susceptable to the noise than the larger ones.

V. Conclusion

We introduced the random hypernetwork architecture inspired by biomolecular networks in cells. The network consists of a large number of heterogeneous "micromodules" (i.e. hyperedges) which interact with each other in a massive way to build potential "microcircuits". We analyzed the structural properties of the hypernetworks to cope with a short-term change (adaptability) and a long-term change (persistency) in the environment, where the environment was simulated by a sequence of training examples. For intelligent organisms, including humans, learning is a capability useful to cope with the first kind of perturbations, and memory is a faculty necessary to deal with the second kind of change.

Our analysis suggests that large micromodules are advantageous to agile learning of specific examples while small micromodules are useful to keep the hypernetwork system stable and persistent for a wide range of inputs in the longer term. This kind of specific yet stable behavior is also found in interaction and transcriptional regulatory networks in cells [9] and neuronal synapses [8][10]. This observation offers an important lesson for designing computational learning systems. Most of neural network architectures are designed to consist of high-dimensional, fine-tuned components of homogeneous structure, which is not necessarily the best strategy to build a computational intelligence system that prospers and survives across the entire life. Applied back to biology, it would be also interesting to see if the random hypernetworks identify motifs and modules of real biological networks.

Acknowledgements

The author would like to thank Joo-Kyung Kim, Sun Kim, and Ha-Young Jang for performing simulations. This research was supported by MOST (NRL 2002-2007), MICE (MEC 2000-2009), and BK21-IT.

References

[1] Basso, K., Margolin, A.A., Stolovitzky, G., Klein, U., Dalla-Favera, R., and Califano, A., "Reverse engineering of regulatory networks in human B cells," *Nature Genetics*, 37(4): 382-390, 2005.

[2] Bhalla, U.S. and Iyengar, R., "Emergent properties of networks of biological signaling pathways," *Science*, 283(5400):381-7, 1999.

[3] Berge, C. *Graphs and Hypergraphs*, North-Holland Publishing, Amsterdam, 1973.

[4] Bren, A./ and Eisenbach, M., "How signals are heard during bacterial chemotaxis: Protein-protein interaction in sensory signal propagation," *Journal of Bacteriology*, 182(24):6865-6873, 2000.

[5] Cho, D.-Y., Cho, K.-H., and Zhang, B.-T., "Identification of biochemical networks by S-tree based genetic programming," *Bioinformatics*, 22(13):1631-1640, 2006.

[6] Janson, S., Luczak, T., and Rucinski, A., *Random Graphs*, Wiley, 2000.

[7] Lee, D.I., Rinaldi, N.J., Robert, F., ..., and Young, R.A., "Transcriptional regulartory networks in *Saccharomyes cerevisiae*," *Science*, 298:799-804, 2002.

[8] Lisman, J., Schulman, H., and Cline, H., "The molecular basis of CaMKII function in synaptic and behavioural memory", *Nature Review Neuroscience*, 3(3):175-90, 2002.

[9] Maslov, S. and Sneppen, K., "Specificity and stability of topology in protein networks," *Science*, 296:910-913, 2002.

[10] Miller, P., Zhabotinsky, A.M., Lisman, J.E., Wang, X.J., "The stability of a stochastic CaMKII switch: dependence on the number of enzyme molecules and protein turnover," *PLoS Biology*, 3(4):e107, 2005.

[11] Milo, R., Shen-Orr, S., Itzkovitz, S., Kashitan, N., Chklovskii, D., and Alon, U., "Network motifs: simple building blocks of complex networks," *Science*, 298:824-827, 2002.

[12] Shengupta, A.M., Djordijevic, M., and Shraiman, B.I., "Specificity and robustness in transcription control networks," *Proc Natl Acad Sci U S A*, 99(4):2072-7, 2002.

[13] Shilling, C.H. and Palsson, B.O., "The underlying pathway structure of biochemical reaction networks," *Proc Natl Acad Sci U S A*, 95:4193-4198, 1998.

[14] Shouval, H.Z., "Clusters of interacting receptors can stabilize synaptic efficacies," *Proc Natl Acad Sci U S A*, 102(40):14440-5, 2005.

[15] Valiant, L., "Robust logics", *Proc. ACM Symposium on the Theory of Computing* (STOC 99), pp. 642-651, 1999.

[16] Wolf, D.M. and Arkin, A.P. "Motifs, modules and games in bacteria," *Curr Opin Microbiol*, 6(2):125-34, 2003.

[17] Zhang, B.-T. and Kim, J.-K., "DNA hypernetworks for information storage and retrieval," *Lecture Notes in Computer Science*, DNA12, 4287:298-307, 2006.

[18] Zhang, B.-T. Ohm, P., and Mühlenbein, H., "Evolutionary induction of sparse neural trees," *Evolutionary Computation*, 5(2):213-236, 1997.

[19] Zhang, B.-T. Yang, J.-S., and Chi, S.-W., "Self-organizing latent lattice models for temporal gene expression profiling," *Machine Learning*, 52(1/2):67-89, 2003.