Journal of Intelligent & Fuzzy Systems 22 (2011) 83–92
DOI:10.3233/IFS-2011-0478
IOS Press

Correlation analysis and performance evaluation of distance measures for evolutionary neural networks

Kyung-Joong Kim[a], Jung Guk Park[a] and Sung-Bae Cho[b,*]
[a]*Department of Computer Engineering, Sejong University, Seoul, South Korea*
[b]*Department of Computer Science, Yonsei University, Seoul, South Korea*

Abstract. In a genetic algorithm, the search process maintains multiple solutions and their interactions are important to accelerate the evolution. If the pool of solutions is dominated by the single fittest individual in the early generation, there is a risk of premature convergence losing exploration capability. It is necessary to consider not only the fitness of solutions but also the similarity to other individuals. This speciation idea is beneficial to several application domains with evolutionary computation but it requires objective distance measures to calculate the similarity of individuals. It raises a challenging research issue to measure the distance between two evolutionary neural networks (ENN). In this paper, we surveyed several distance measures proposed for ENN and compared their performance for pattern classification problems with two different genetic representations (matrix-based and topology growing (NEAT) approaches). Although there was no dominant distance measure for the pattern classification problems, it showed that the behavioral distance measures outperformed the architectural one for matrix-based representation and. For NEAT, NeuroEdit showed better accuracy against compatibility distance measure.

Keywords: Speciation, premature convergence, distance measures, evolutionary neural networks, pattern classification, NEAT

1. Introduction

Evolutionary neural networks (ENN) exploit evolutionary computation to design the topology and weights of artificial neural networks automatically [19]. Usually, the topology of neural networks is designed by experts and the weights of the fixed structure are trained with gradient-descent learning algorithms [5]. Designing the topology is not a trivial task and the error function of the learning algorithm has to be differentiable. In addition to those limitations, the local search is likely to be trapped into local optima. The evolutionary computation (EC) is a population-based search maintaining multiple solutions simultaneously without the restriction of differentiable error equations [4]. ENN adopts the EC to optimize the topology and weights of neural networks at the same time.

Although EC is strong for global search, its population can be dominated by premature solutions in the early generation and need speciation to avoid it [12]. Because the EC gives high selection pressure to the individuals with high fitness, the population can lose its diversity in the early generation due to the premature solutions. This leads to the invention of speciation algorithms which modify the selection pressure of individuals based on the similarity to others [11]. If there are many similar individuals in the population, its

*Corresponding author. E-mail: kimkj@sejong.ac.kr (K.-J. Kim), sbcho@cs.yonsei.ac.kr (S.-B. Cho).

fitness is divided by the number of similar individuals. In this way, less fitted solutions can survive to the next generation increasing the genetic diversity of population.

Speciation has been successfully used in evolving neural networks to generate multiple diverse solutions avoiding local optima [9, 10, 17], but there is no consensus on the distance measures for the speciation of ENN. In the speciation, we need to determine the distance measure between two evolutionary neural networks but it is not a trivial problem. First of all, it is not easy to define the distance measures because there is little work on the definition of distance between neural networks. And also, there is no way to evaluate the goodness of the distance measures in a direct manner. Instead, it is just possible to see the effect of measures indirectly from the performance of ENN systems for the specific problems.

In a broad manner, there are two categories of distance measures for ENN: architectural (topology + weight) and behavioral distance measures. In the architectural measures, they calculates the similarity between two ENNs based on architectural differences (topological and weight difference). If two neural networks have the same architecture, their distance is 0. On the other hands, in behavioral measures, they consider only the outcome of the neural networks to calculate the distances. If two neural networks output the same outcome to input data although they have different architectures, their distance is 0.

As mentioned before, the comparison of distance measures is only possible indirectly based on the performance of ENN systems for specific problems and we adopted speciation-based ENN pattern classification systems. Pattern classification is one of the popular research areas of evolutionary neural networks and there are a lot of benchmarking datasets from UCI machine learning repository [2]. Also, there are several research works on exploiting speciation with evolutionary neural networks for pattern classification [1, 7, 10, 13]. Although there are several interesting problem domains for ENN, the pattern classification can be a starting position of further investigation.

In this paper, we surveyed several distance measures from literatures and compared their performance for pattern classification problems with two different neural network representations (matrix-based and topology growing approaches). In the matrix-based representation [10], the architecture of neural network is encoded directly in the matrix as entry values. Half of the matrix represents the topology and the remaining parts contain information on weights. There are several works on speciation with the matrix-based representation [1, 7, 10]. On the other hands, there are another popular ENN representations called as NEAT (NeuroEvolution of Augmenting Topologies) which grows its topology from minimal structure to complex one [17]. It adopts fitness sharing, one of the widely used speciation algorithms, to avoid local optimum and needs distance measures between two neural networks. For the two representations, we tested the accuracy of the evolved neural classifiers for pattern classification problems with different distance measures. In this way, we attempted to evaluate the distance measures for pattern classification problems with different representations.

2. Distance measures for evolutionary neural networks

Although there are several works on using the distance measures for traditional neural networks (not ENN), the distance measures have been mainly used for evolutionary neural networks. If it is not an ENN and the two neural networks have different topology, it is not trivial to map hidden neurons in on networks to one of another network. Fortunately, in ENN, there is an ID for each hidden neuron and it is possible to directly compare the hidden neurons in two different neural networks. In this reason, the distance measures have been widely used for the ENN. In Table 1, there is a summary of distance measures for ENN.

For the matrix-based representation, the outcome of neural networks is mainly used to measure the distance of two ENNs. In [1], authors proposed average output, Pearson correlation and modified Kullback-Leibler (KL) entropy methods to calculate the difference of output neuron's outcomes. They tested their methods for three representative pattern classification problems and reported that the KL entropy slightly outperforms other two alternatives. Their methods have been used in other works [7, 10].

For NEAT [17], the genome is composed of two parts (node genes, connection genes). The connection gene has in-node ID, out-node ID, weight, enable/disable, and an original historical ancestor of each gene. In this case, the distance is calculated with a simple linear combination of the matching and mismatching information. Their distance measure is called as compatibility. In [13, 14], the authors use the same representation except

Table 1
Summary of distance measures for evolutionary neural networks

Distance types	Authors	Distance measures	Contexts
Architectural distance	Kim et al. [9]	Euclidean distance	Deterministic crowding GA
	Stanley et al. [17]	Linear equation	Fitness sharing
	Jung et al. [6]	Euclidean distance	Replacement in a genetic algorithm
	Sallam et al. [13, 14]	Neuro-Edit	Fitness sharing, diversity analysis
Behavioral distance	Kim et al. [10]	Average output, Pearson correlation, Kullback-Leibler entropy	Fitness sharing
	Khare et al. [7]	Kullback-Leibler entropy	Fitness sharing
	Garcia-Pedrajas et al. [3]	Euclidean distance	Fitness sharing
	Trujillo et al. [18]	Edit distance	Fitness sharing

the historical information. The connection gene does not contain information about its ancestor in general, but a globally unique "innovation number" assigned when the gene first arose by mutation. They modified Edit Distance for the NEAT representation and the new distance measure is called as NeuroEdit. Unlike the previous two works, Trujillo et al. used the difference of actual trajectories by robot's neural controllers to calculate the similarity of ENNs [18].

In [6], a new distance measure is proposed with the consideration of redundancy in representation. Any permutation of the hidden neurons produces the same neural network with a different chromosome representation. $N!$ different representations exist for a network with n hidden neurons. In their work, they consider all permutations in calculating distances of two neural networks. This is computationally expensive because $N!$ combinations are checked.

Kim et al. [8] reported the performance evaluation of matrix-based representation with speciation for pattern classification problems. In their work, they averaged the classification accuracy for several datasets to get the rank of the distance measures. In their conclusion, the behavioral distance outperforms the architectural one and the one with much information about the outputs from output neurons and hidden neurons performs the best. However, they only considered the matrix-based representations and all datasets have relatively small number of classes (for example, 2–3 classes).

3. Methods

The purpose of this research is to review a set of distance measures published in literatures and evaluate their performance on different settings (representations) (Table 2). From our survey, we found that there are six distance measures for the matrix-based representations and two distance measures for NEAT. For the matrix-based representation, Average Output and Kullback-Leibler Entropy were proposed in [1]. There are four new measures proposed recently by authors [8]: Euclidean Distance, Hamming Distance, Euclidean Output, and Total Euclidean Output. For NEAT, the original distance measure is compatibility proposed by [17] and there is a new measure recently proposed by [13, 14].

Although there are a lot of speciation methods [11, 15], the most widely used methods are fitness sharing and deterministic crowding genetic algorithm. Fitness sharing readjusts the fitness of individuals based on the population density near the individual. The amount of fitness reduction is proportionate to the number of individuals within sharing radius. On the other hands, the deterministic crowding GA has different selection strategy without changing the fitness value. Instead of readjusting the fitness value, the method chooses the fittest one among two similar individuals (usually, a pair of children and parents). In this way, it can reduce the similarity among the individuals of the population.

Table 2
Summary of representations, speciation algorithms, and distance measures for ENN

Representation	Speciation	Distance measures	
Matrix-based representation	Fitness sharing, Deterministic crowding GA	Architectural distance measures	Euclidean distance, Hamming distance
		Behavioral distance measures	Euclidean output total euclidean output average output, Kullback Liebler entropy
NEAT	Fitness sharing	Architectural distance measures	Compatibility, NeuroEdit

The representation of evolutionary neural networks has big impact on the performance of ENN systems and the design of genetic operators (crossover and mutations). The matrix-based representation is a kind of direct encoding scheme and easy to implement. It starts from fully connected networks and applies the genetic operators like crossover and mutations (addition and deletion of arcs). In NEAT, it uses a flexible representation which allows continuous growth of the networks. It starts from the minimal structure and adds additional elements to the original networks.

3.1. Representations

3.1.1. Matrix-based representation

A matrix representation to encode the ANN is straightforward to implement and easy to apply the genetic operators [10]. When N is the total number of nodes in the ANN (including the input, hidden, and output nodes), the matrix is $N \times N$ whose entries consist of connection links and the corresponding weights. In this model, each ANN uses only forward links. In the matrix, the upper right triangle (see Fig. 1) has connection link information where '1' means that there is a connection link and '0' means that there is no connection link. The lower left triangle describes the weight values corresponding to the connection link information. The number of hidden nodes can vary within the maximum number of hidden nodes in the course of the GA operations.

There are two different genetic operators (crossover and mutation) used in the matrix-based evolutionary neural networks. The crossover operator exchanges the architectures of two ANN's in the population to search the ANN's from various architectures. In a population of ANN's, the crossover operator selects two distinct ANN's randomly and chooses one hidden node as a

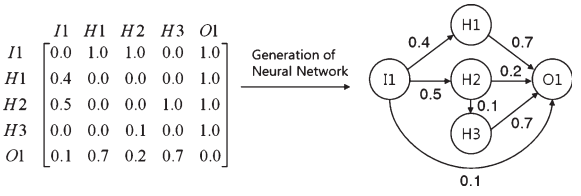

Fig. 1. In this representation, the matrix has one input neuron, at most three hidden neurons and one output neuron. In the top right corner, there is information on the connection of links between nodes. In the bottom left corner, it contains information on the weights information of links. It only allows feed-forward links. Unlike traditional neural networks, it allows the direct link from the input neuron and output neuron.

crossover point. The two ANN's exchange the connection links and the corresponding weight information of the nodes. The mutation operator changes a connection link and the corresponding weight of a randomly selected ANN from the population. It performs one of two operations: Addition of a new connection and deletion of an existing connection. The mutation operator selects an ANN from the population of ANN's randomly and chooses one connection link from it. If the connection link does not exist and the connection entry of the ANN matrix is '0', a new connection link is created. It adds the new connection link to the ANN with random weights. Otherwise, if the connection link already exists, it removes the connection link and weight information.

3.1.2. NEAT

NEAT was proposed by Stanley [16] in 2002. It has been applied to many real-world problems and there have been several variants of the standard NEAT. It starts from minimal solutions which have only links from input to output neurons (zero hidden nodes). Chromosomes are linear representations of network connectivity (Fig. 3). Each link contains in-node ID, out-node ID, weight of the connection, an enable bit, and an innovation ID which is historical marker identifying the original historical ancestor of each link. It allows easy line up of corresponding genes when two chromosomes crossover during evolution. During crossover, the links in both chromosomes with the same

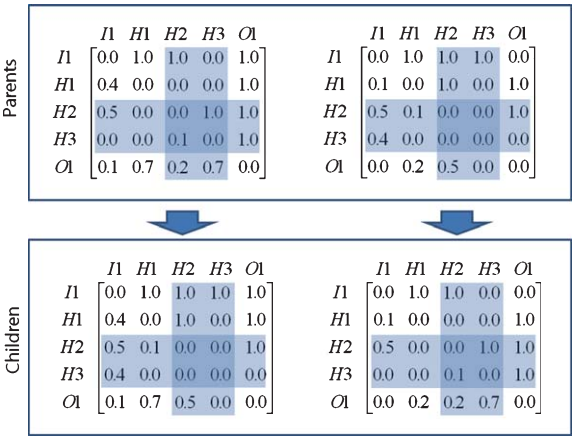

Fig. 2. Among the hidden nodes, one hidden node is randomly selected as a crossover point (in this example, H2). The hidden nodes that have larger index than the point are considered for crossover. In this example, H2 and H3 are considered and the links related to them are exchanged.

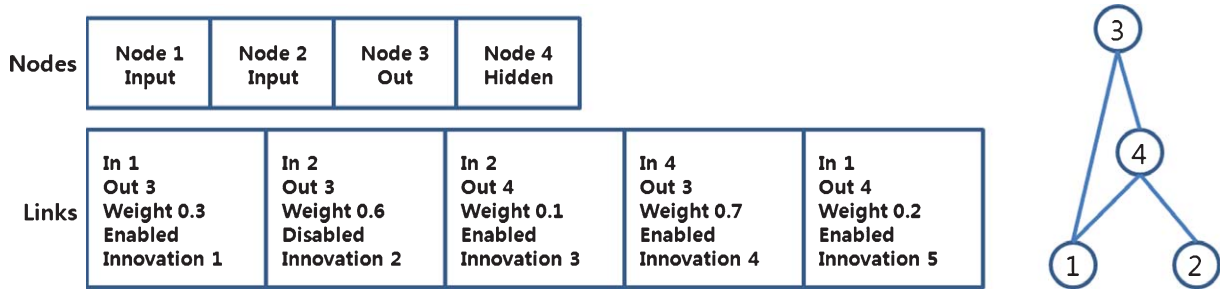

Fig. 3. NEAT maintains two separate lists of nodes and link information. In nodes, it has node ID and the type of nodes (input, output, and hidden). For each link, it contains the in-node ID, out-node ID, weight, on/off of the link, and the innovation ID.

innovation numbers are lined up. In NEAT, there are three different mutation operators (weight change, add connection, and add node).

3.2. Speciation algorithms

3.2.1. Fitness sharing

Fitness sharing decreases the fitness of individuals in a densely populated area and shares the fitness with other neural networks (Fig. 4). Therefore, it helps the genetic algorithm search a broad solution space and it generates more diverse neural networks. Given that f_i is the fitness of an individual and $sh(d_{ij})$ is a sharing function, the shared fitness f_{si} is computed as follows:

$$fs_i = \frac{f_i}{\sum_{j=1}^{\text{population size}} sh(d_{ij})}$$

The sharing function $sh(d_{ij})$ is computed using the distance value d_{ij} which means the difference of individuals i and j as follows:

```
// σ : Sharing radius
// MAX_GEN: The maximum number of generation
// POP_SIZE: A population size
// Distance (NN1, NN2): Return distance between two neural networks
// Evaluation (): Return a fitness of neural networks in a population

Initialization (); // Create neural networks randomly
                                  // Set the sharing radius as the half of the average distance of the initial
population

FOR (gen=0; gen < MAX_GEN; gen++) {
    Evaluation ();

    FOR (i=0; i < POP_SIZE; i++) {
      sh=0;
        FOR (j=0; j < POP_SIZE; j++)
            IF (Distance(NNi,NNj) < σ )    sh+=(1- Distance(NNi,NNj)/ σ );
        fitness[NNi]/=sh;
    }

    Selection ();      // Roulette-Wheel selection
    Crossover ();    // 1-point uniform crossover
    Mutation ();      // If no link, a random weight is assigned. If link, delete it.
    Elitist ();
}
```

Fig. 4. Pseudo code for fitness sharing.

$$sh(d_{ij}) = \begin{cases} 1 - \dfrac{d_{ij}}{\sigma_s}, & 0 \le d_{ij} < \sigma_s \\ 0, & d_{ij} \ge \sigma_s \end{cases}$$

Here, σ_s means the sharing radius. If the difference of the individuals is larger than σ_s, they do not share the fitness. Only the individuals who have smaller distance values than σ_s can share the fitness.

3.2.2. Deterministic crowding GA

Unlike fitness sharing, there is no additional parameter for the deterministic crowding genetic algorithm. It is based on the competition between parents and offspring paired to maximize the similarity. Among two similar neural networks, one with better fitness survives to the next generation. Figure 5 summarizes a pseudo code for the deterministic crowding algorithm. It uses the same crossover and mutation as fitness sharing.

3.3. Distance measures

In this subsection, we survey methods for distance measure which evaluate similarity between two neural networks. Distance measure methods consist of two part, Architectural distance measures and Behavioral distance measures.

Notation

N_I: The number of input neurons
N_H: The maximum number of hidden neurons
N_O: The number of output neurons
N: The number of total neurons ($N = N_I + N_H + N_O$)

```
// Shuffling (): Shuffle the population
// Survive (): Copy to the population of the next generation

Initialization (); // Create neural networks randomly

FOR (gen=0; gen < MAX_GEN; gen++) {
    Evaluation ();

    Shuffling ();
    FOR (i=0; i < POP_SIZE; i+=2) {

      P1=NN_i;       // parent 1
      P2=NN_i+1; // parent 2
      (C1,C2)=crossover(NN_i, NN_i+1);
      C1=mutation(C1); C2=mutation(C2);    // offspring

      IF(Distance(P1,C1)+Distance(P2,C2)<Distance(P1,C2)+Distance(P2,C1)){
        IF(fitness(P1)<fitness(C1)) Survive(C1); ELSE Survive(P1);
        IF(fitness(P2)<fitness(C2)) Survive(C2); ELSE Survive(P2);
      }
      ELSE{
        IF(fitness(P1)<fitness(C2)) Survive(C2); ELSE Survive(P1);
        IF(fitness(P2)<fitness(C1)) Survive(C1); ELSE Survive(P2);
      }

    }

}
```

Fig. 5. Pseudo code for deterministic crowding genetic algorithm.

$I_{ijk}, H_{ijk}, O_{ijk}$: jth input, hidden, and output neuron of ith neural network for kth sample

φ (): Output of a neuron

M: The number of training samples

V_i: A chromosome of ith neural network (weight information of matrix is converted to 1-D vector)

L: A length of the 1-D vector ($L = (N^2 - N)/2$)

3.3.1. Architectural distance measures

3.3.1.1. Euclidean distance. This is a simple Euclidean distance of two chromosome vectors.

$$d(a,b) = \sqrt{\sum_{k=1}^{M}\left(\sum_{j=1}^{N_I}(I_{ajk} - I_{bjk})^2 + \sum_{j=1}^{N_H}(H_{ajk} - H_{bjk})^2 + \sum_{j=1}^{N_O}(O_{ajk} - O_{bjk})^2\right)}$$

$$d(a,b) = \sqrt{\sum_{i=1}^{L}(V_{ai} - V_{bi})^2}$$

3.3.1.2. Hamming distance. This counts only the similarity and dissimilarity of architecture. If only one of the networks has a link, this is counted.

$$d(a,b) = \sum_{i=1}^{L} h(V_{ai}, V_{bi})$$

$h(x,y) =$

$$\left\{\begin{array}{ll} 1 & V_{ai} = 0 \text{ and } V_{bi} \neq 0,\ V_{ai} \neq 0 \text{ and } V_{bi} = 0 \\ 0 & \text{otherwise} \end{array}\right\}$$

3.3.1.3. Compatibility. Links that do not match between two neural networks are either disjoint or excess. If the link's innovation ID is larger than the other parent's maximum innovation ID, it's excess. Otherwise, it's disjoint. Compatibility of different structures in NEAT was defined as a simple linear combination of excess E and disjoint D links and the average weight differences of matching links W [16].

$$d(a,b) = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 W$$

3.3.1.4. NeuroEdit. Sallam et al. [14] proposed a new distance measure for NEAT based on EDIT distance.

3.3.2. Behavioral distance measures

3.3.2.1. Euclidean output. This calculates the Euclidean distance of outputs from output neurons.

$$d(a,b) = \sqrt{\sum_{k=1}^{M}\sum_{j=1}^{N_O}(O_{ajk} - O_{bjk})^2}$$

3.3.2.2. Total Euclidean output. Usually, the outputs from output neurons are used in the behavioral distance but we propose to use outputs from all neurons (input/hidden/output neurons).

3.3.2.3. Average output. This is the Euclidean distance of average outputs from each output neuron.

$$d(a,b) = \sqrt{\sum_{j=1}^{N_O}(\overline{O}_{aj} - \overline{O}_{bj})^2}$$

$$\overline{O}_{aj} = \frac{1}{M}\sum_{k=1}^{M} O_{ajk} \quad \overline{O}_{bj} = \frac{1}{M}\sum_{k=1}^{M} O_{bjk}$$

3.3.2.4. Kullback-Leibler entropy. This is proposed in [1]. It is called as relative entropy and modified to be used as a distance measure.

$$d(a,b) = \frac{1}{2}\sum_{k=1}^{M}\sum_{j=1}^{N_O}\left(O_{ajk}\log\frac{O_{ajk}}{O_{bjk}} + O_{bjk}\log\frac{O_{bjk}}{O_{ajk}}\right)$$

4. Experimental results

The purpose of this experimentation is to see the effect of distance measures on the performance of ENN systems. We adopted five datasets from UCI machine learning repository. For the five datasets, we evolved neural classifiers for all the combination of speciation algorithms and distance measures. For the matrix-based representation, we adopted six distance measures (Euclidean Distance, Hamming Distance, Euclidean Output, Total Euclidean Output, Average Output, and Kullback Leibler Entropy). For NEAT, there are two distance measures used (Compatibility and NeuroEdit). For each distance measure, we repeated the evolution 10 times to get the average accuracy.

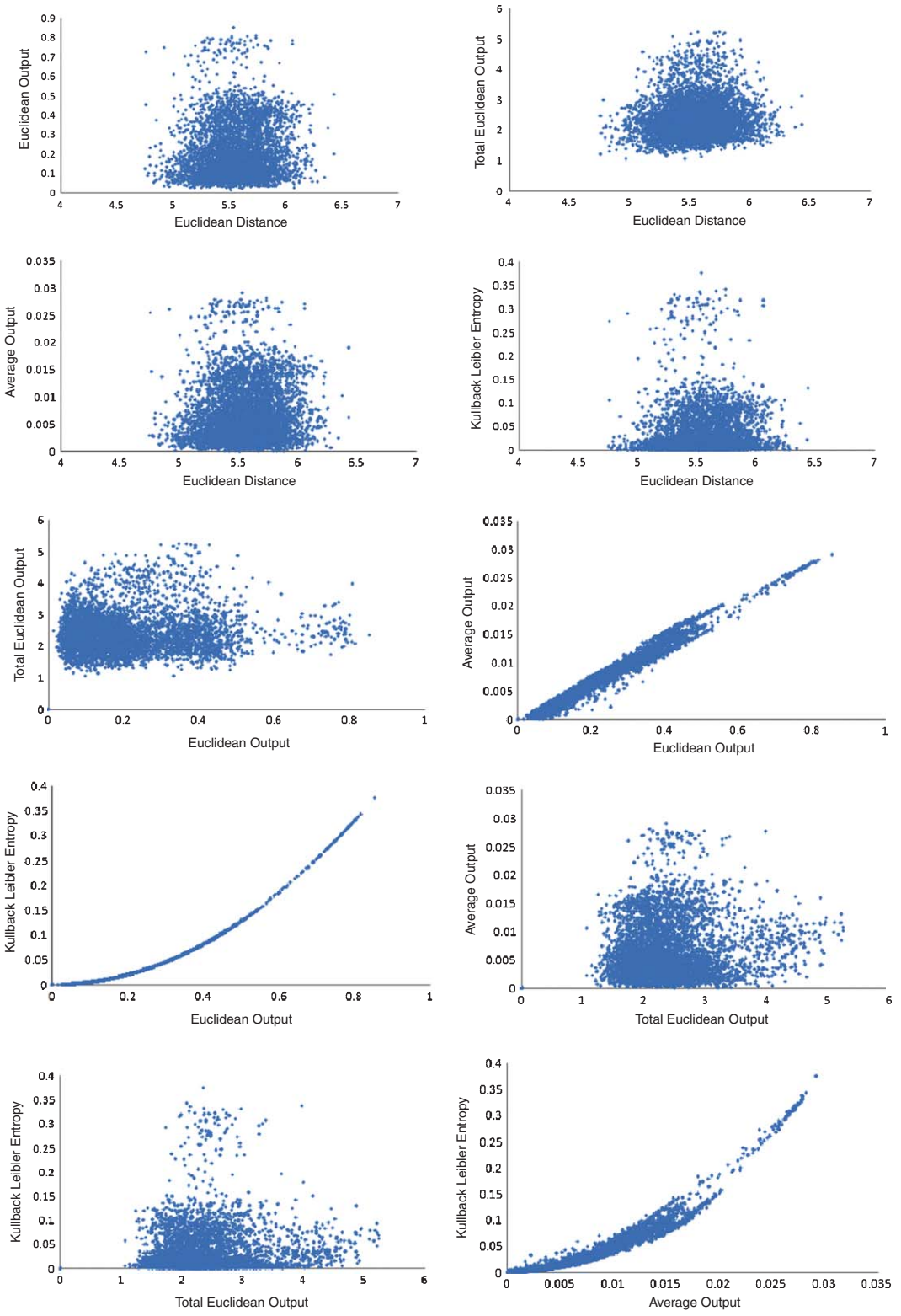

Fig. 6. Correlation analysis of distance measures for matrix-based representation (Australian Credit Card Dataset).

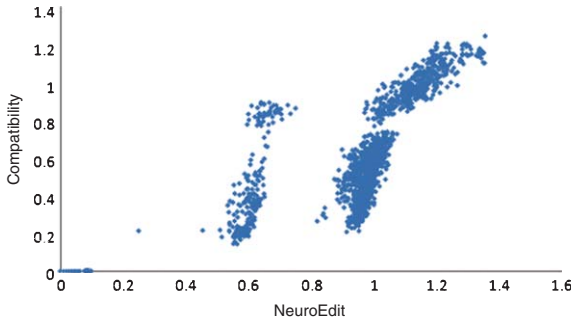

Fig. 7. Correlation analysis for NEAT distance measures (Australian Credit Card Dataset).

First of all, we did correlation analysis of distance measures to see the relationships among them. In this work, we used the population size as 100 for the two representations. For the matrix-based representation, we initially generated 100 random neural networks fully connected and calculated the distance values for the all pairs (100×99) with five distance measures. We excluded Hamming Distance because they recorded zero value for all pairs. For NEAT, we did the correlation analysis for the 100 neural networks at 100 generations because the NEAT starts from the minimal structure.

Figure 6 shows the correlation analysis of the distance measures for the matrix-based representation. It is interesting that there is linear relationship between Euclidean Output and Average Output. There is positive correlation between the Euclidean Output and the Kullback Leibler Entropy but it's not linear. Figure 7 shows the same analysis for the two distance measures of the NEAT. It shows that there is positive correlation between the two distance measures.

The second experiment is to test the classification accuracy of the evolved neural classifiers with different distance measures. Table 3 summarizes the datasets used in this experimentation. We chose binary and multi-class classification datasets. The whole data is separated to training (2/3) and test (1/3) randomly. The fitness function of the evolution is the accuracy on the training dataset. The final accuracy was measured on the test dataset.

Table 3
Summary of datasets used

Name	No of classes	No of attributes	No of samples
Australian credit card	2	14	690
Breast cancer	2	10	699
Diabetes	2	8	768
Glass	7	10	214
Soybean – large	19	35	307

Table 4 summarizes the performance of ENN-based classification systems with different distance measures. Although there were no dominant distance measures, most of the best accuracy was achieved from the behavioral distance measures. EO and TEO show relatively good performance for all datasets. Although the behavioral distance measures show improved performance, they required more computational resource than the architectural distance. The computational complexity of the behavioral distance measures is proportional to the number of training samples. In case of data mining with thousands of data, it is better to choose architectural distance measure to minimize computational cost. If not, we need additional mechanism to approximate the behavioral distance measure from sampled training data. For NEAT, the NeuroEdit outperforms the compatibility distance measures for most of datasets.

5. Conclusions

In this paper, we proposed comprehensive test of distance measures for evolutionary neural networks for different representations. Correlation analysis was used to see the relationships among distance measures

Table 4a
Performance comparison of distance measures (bold means the best accuracy for the dataset). (a) Matrix-based representation

		EU	HA	EO	TEO	AO	KL
Australian credit	FS	80.33 ± 4.7	$\mathbf{84.4 \pm 1.8}$	84.00 ± 1.4	83.90 ± 1.4	83.90 ± 1.9	83.90 ± 2.0
	DCGA	83.27 ± 1.7	83.13 ± 2.1	83.27 ± 1.7	83.41 ± 1.6	83.26 ± 1.3	83.89 ± 1.6
Breast cancer	FS	98.63 ± 0.38	98.49 ± 0.42	98.54 ± 0.68	98.54 ± 0.3	98.54 ± 0.7	98.63 ± 0.38
	DCGA	98.50 ± 0.38	98.44 ± 0.29	98.39 ± 0.31	$\mathbf{98.83 \pm 0.45}$	98.54 ± 0.38	98.39 ± 0.22
Diabetes	FS	70.99 ± 4.9	77.23 ± 2.2	74.76 ± 2.4	74.93 ± 2.6	72.16 ± 4.0	74.37 ± 3.3
	DCGA	76.49 ± 1.6	76.45 ± 1.1	77.01 ± 1.2	77.05 ± 1.9	77.23 ± 1.3	$\mathbf{77.70 \pm 2.0}$
Glass	FS	44.15 ± 5.3	49.08 ± 7.3	44.9 ± 6.8	40.77 ± 6.2	42.62 ± 6.8	46.92 ± 6.8
	DCGA	52.00 ± 7.2	53.23 ± 6.7	$\mathbf{56.92 \pm 3.6}$	52.31 ± 6.2	53.54 ± 4.5	54.15 ± 6.1
Soybean	FS	32.47 ± 5.5	34.84 ± 3.2	$\mathbf{36.67 \pm 2.5}$	33.66 ± 4.7	36.32 ± 3.2	31.93 ± 4.3
	DCGA	33.66 ± 2.4	31.61 ± 2.6	32.26 ± 3.6	32.80 ± 3.9	34.09 ± 3.7	33.23 ± 8.4

Table 4b
Performance comparison of distance measures (bold means the best accuracy for the dataset). (b) NEAT

	NeuroEdit	Compatibility
Australian Credit	**85.77 ± 0.4**	84.56 ± 1.2
Breast Cancer	**98.88 ± 0.3**	98.63 ± 0.6
Diabetes	**77.88 ± 0.6**	75.76 ± 4.2
Glass	**58.92 ± 4.9**	51.23 ± 7.0
Soybean	32.58 ± 1.9	**32.80 ± 1.6**

and we found that positive linear and non-linear relationships among several distance measures. Some of them have no correlation at all but several measures have strong correlation. Using UCI machine learning datasets, we tested the superiority of distance measures for pattern recognition problems. It showed that the behavioral distance measures outperform the architectural one for most of datasets. However, they required much more computational resource than the architectural distance measure. It is recommended to consider tradeoff between the performance improvement and the increasing of computational cost. For data mining problems with thousands of data, it is not a good choice to use the behavioral distance measures directly without sampling approaches. For NEAT, the NeuroEdit showed better accuracy than compatibility.

Acknowledgements

This research was supported by Basic Science Research Program and the Original Technology Research Program for Brain Science through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0012876) (2010-0018948). This paper is an extended version of the paper published in International Conference on Neural Information Processing 2009.

References

[1] J.-H. Ahn and S.-B. Cho, Speciated neural networks evolved with fitness sharing technique, *Proceedings of Congress on Evolutionary Computation*, 2001, pp. 390–396.

[2] A. Frank and A. Asuncion, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], School of Information and Computer Science, University of California, Irvine, CA, 2010.

[3] N. Garcia-Pedrajas, C. Hervas-Martinez and D. Ortiz-Boyer, Cooperative coevolution of artificial neural network ensembles for pattern classification, *IEEE Transactions on Evolutionary Computation* **9**(3) (2005), 271–302.

[4] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[5] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1999.

[6] S.-C. Jung and B.-R. Moon, Central point crossover for neuro-genetic hybrids, *Lecture Notes in Computer Science* **3102** (2004), 1292–1303.

[7] V. Khare and X. Yao, Artificial speciation of neural network ensembles, *Proceedings of the 2002 UK Workshop on Computational Intelligence* (2002), 96–103.

[8] K.-J. Kim and S.-B. Cho, Evaluation of distance measures for speciated evolutionary neural networks in pattern classification problems, *Lecture Notes in Computer Science* **5864** (2009), 630–637.

[9] K.-J. Kim and S.-B. Cho, Systematically incorporating domain-specific knowledge into evolutionary speciated checkers players, *IEEE Transactions on Evolutionary Computation* **9**(6) (2005), 615–627.

[10] K.-J. Kim and S.-B. Cho, Evolutionary ensemble of diverse artificial neural networks using speciation, *Neurocomputing* **71** (2008), 1604–1618.

[11] S.W. Mahfoud, Niching methods, *Evolutionary Computation 2: Advanced Algorithms and Operators*, Institute of Physics Publishing, 2000, pp. 87–92.

[12] A. Rogers and A. Prugel-Bennett, Genetic drift in genetic algorithm selection schemes, *IEEE Transactions on Evolutionary Computation* 3(4) (1999), 298–303.

[13] H. Sallam, C.S. Regazzoni, I. Talkhan and A. Atiya, Evolving neural networks ensembles NNEs, 2008 *IAPR Workshop on Cognitive Information Processing* (2008), 142–147.

[14] H. Sallam, C.S. Regazzoni, I. Talkhan and A. Atiya, Measuring the genotype diversity of evolvable neural networks, *The 6th International Conference on Informatics and Systems – Parallel and Distributed Computing and Their Applications* 2008, pp. 38–43.

[15] B. Sareni and L. Krahenbuhl, Fitness sharing and niching methods revisited, *IEEE Transactions on Evolutionary Computation* **2**(3) (1998), 97–106.

[16] K.O. Stanley and R. Miikkulainen, Evolving neural networks through augmenting topologies, *Evolutionary Computation* **10**(2) (2002), 99–127.

[17] K.O. Stanley, *Efficient Evolution of Neural Networks through Complexification*, Ph.D. Dissertation, The University of Texas at Austin, 2004.

[18] L. Trujillo, G. Olague, E. Lutton and F.F. de Vega, Discovering several robot behaviors through speciation, *Lecture Notes in Computer Science* **4974** (2008), 164–174.

[19] X. Yao, Evolving artificial neural networks, *Proceedings of the IEEE* **87**(9) (1999), 1423–1447.