Computers & Geosciences 54 (2013) 50–65

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

The application of improved NeuroEvolution of Augmenting Topologies neural network in Marcellus Shale lithofacies prediction

Guochang Wang [a,b,*], Guojian Cheng [c], Timothy R. Carr [a]

[a] Department of Geology & Geography, West Virginia University, Morgantown, WV 26506, USA
[b] College of Earth Science, University of Chinese Academy of Sciences, Beijing 100049, China
[c] School of Computer Science, Xi'an Shiyou University, Xi'an, Shaanxi 710065, China

ARTICLE INFO

Article history:
Received 19 July 2012
Received in revised form
13 December 2012
Accepted 30 January 2013
Available online 13 February 2013

Keywords:
Marcellus Shale
Lithofacies prediction
NEAT
Node location
RCC
Organism population size evolution

ABSTRACT

The organic-rich Marcellus Shale was deposited in a foreland basin during Middle Devonian. In terms of mineral composition and organic matter richness, we define seven mudrock lithofacies: three organic-rich lithofacies and four organic-poor lithofacies. The 3D lithofacies model is very helpful to determine geologic and engineering sweet spots, and consequently useful for designing horizontal well trajectories and stimulation strategies. The NeuroEvolution of Augmenting Topologies (NEAT) is relatively new idea in the design of neural networks, and shed light on classification (i.e., Marcellus Shale lithofacies prediction). We have successfully enhanced the capability and efficiency of NEAT in three aspects. First, we introduced two new attributes of node gene, the node location and recurrent connection (RCC), to increase the calculation efficiency. Second, we evolved the population size from an initial small value to big, instead of using the constant value, which saves time and computer memory, especially for complex learning tasks. Third, in multiclass pattern recognition problems, we combined feature selection of input variables and modular neural network to automatically select input variables and optimize network topology for each binary classifier. These improvements were tested and verified by true if an odd number of its arguments are true and false otherwise (XOR) experiments, and were powerful for classification.

1. Introduction

Unconventional shale gas has successfully offset the decrease of conventional gas production, especially in U.S.A., and plays a significant role in the future fossil energy. The Marcellus Shale in the Appalachian basin (U.S.A.) is one of the most active and successful shale-gas reservoirs. It was deposited over about 2 m.y. duration in a relatively deep and anoxic water (~200 m), and hundreds of miles far away from the sediments source area (7–8 m.y. for the whole Hamilton Group and Tully Limestone; Brett and Baird, 1996). The Marcellus Shale covers most of the basin with an area about 191,000 mi^2 (500,000 km^2; Fig. 1). The average gross thickness and net thickness of the organic-rich interval (total organic carbon-TOC > 6.5%) of the Marcellus Shale are 80 ft (25 m) and 34 ft (10 m), respectively. The nano-order matrix permeability does not allow economical gas flow to the well boreholes, unless either open natural or artificial fractures are present (DOE/NETL, 2010). Unfortunately, most natural fractures were filled by minerals during diagenesis (e.g., carbonates). Therefore, artificial hydrological fracturing has become the routine operation for shale-gas reservoirs. The effectiveness of hydrologic fracture stimulation is influenced by rock geomechanical properties. The varying mineral composition of the mudrock comprising the shale-gas reservoir is the major geological factor influencing geomechanical properties and stimulation effectiveness. Simultaneously, the mineralogy affects the ratio of free to absorbed gas; higher quartz and carbonate content results in a higher free-to-absorbed gas ratio. Another key of shale-gas reservoirs is the organic matter which is usually proportional to gas content under similar kerogen maturation. We defined seven Marcellus Shale lithofacies primarily based on mineralogy and total organic carbon (TOC) from core data, and develop a model to predict them by conventional wireline logs in available logged wells. The log-predicted lithofacies provides sufficient constraints to build a 3D mudrock lithofacies. The prediction of lithofacies is a typical multi-class classification task which could be solved by neural networks.

The NeuroEvolution of Augmenting Topologies (NEAT) is a method for evolving artificial neural networks through the genetic algorithm developed by Stanley and Miikkulainen (2002a, 2002b).

* Correspondence to: 126 Brooks Hall, 98 Beechurst Avenue, PO Box 6300, Morgantown, WV 26505, USA. Tel.: +1 304 906 8794; fax: +1 304 293 6522.
E-mail addresses: w.guochang@gmail.com, w.guochang@foxmail.com (G. Wang), gjcheng.xsyu@gmail.com (G. Cheng), Tim.Carr@mail.wvu.edu (T.R. Carr).

0098-3004/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cageo.2013.01.022

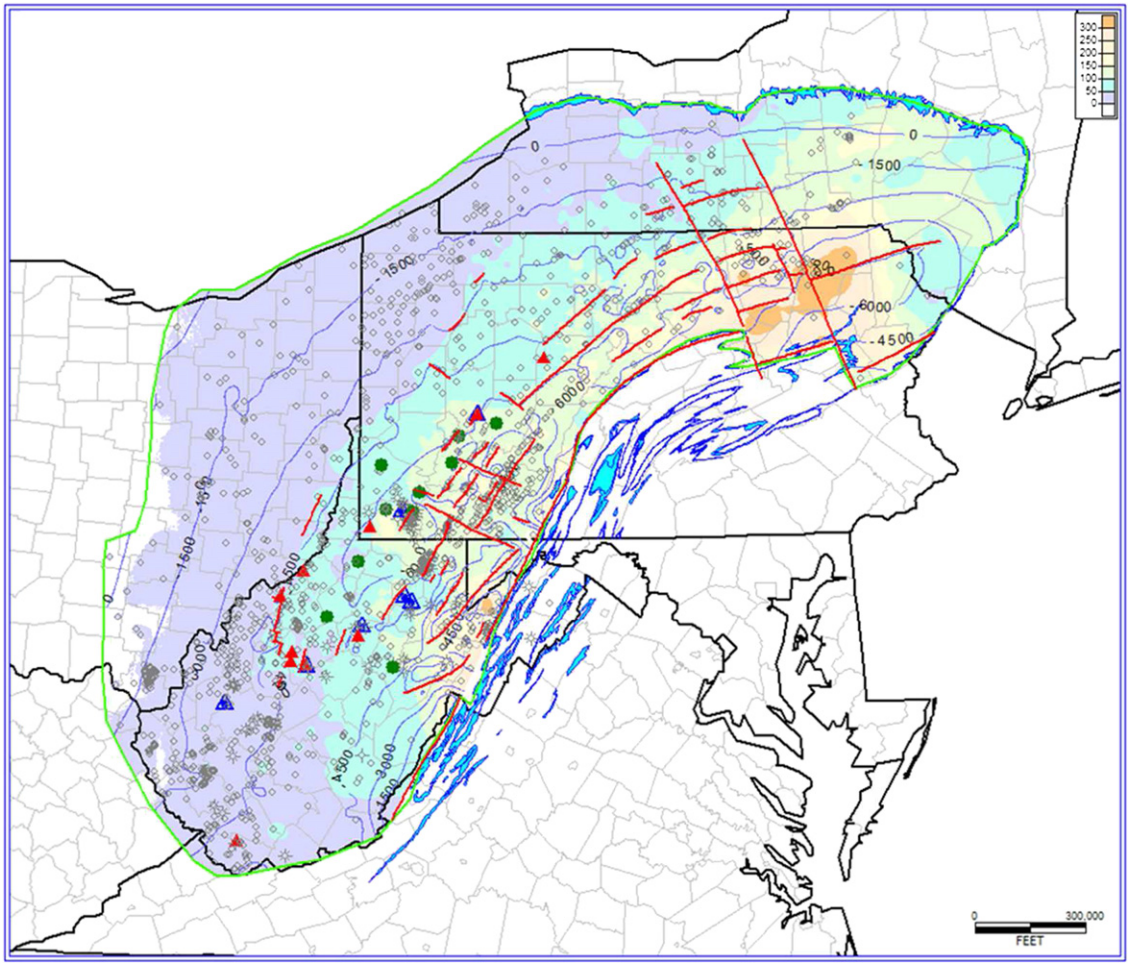


Fig. 1. Map of the Appalachian basin showing the color-filled Marcellus isopach (Contour interval is 15 m (50 ft)), structure contour of the top of Marcellus Shale (Contour interval is 500 m (1500 ft)), the location of over 3880 wells with wireline logs, and/or core data. Wells with conventional logs are gray circles; wells with core XRD and TOC data are green-filled circles; wells with pulsed neutron spectroscopy logs are red-filled triangles; and wells with both core and pulsed neutron spectroscopy logs are blue-filled circles with triangles. The blue filled areas are outcrops of Marcellus Shale. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

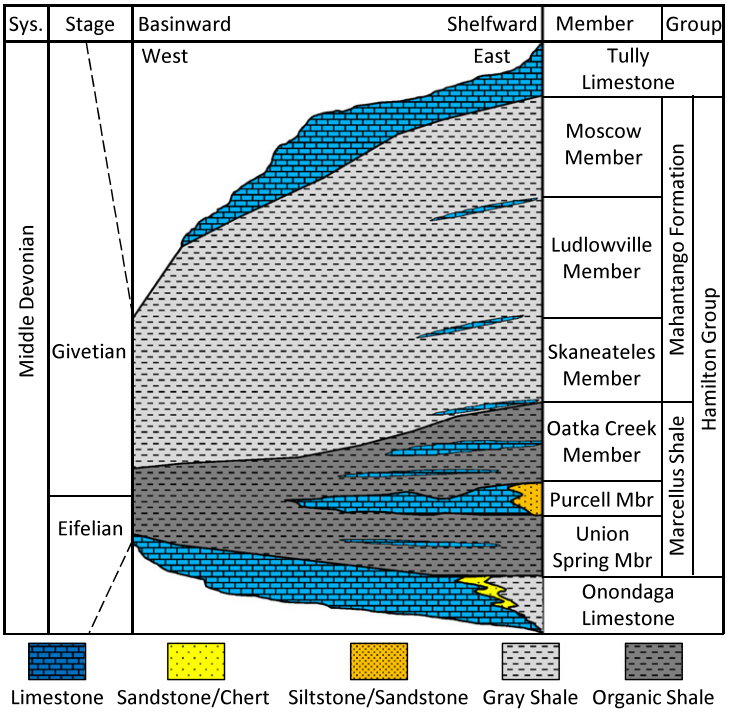


Fig. 2. Generalized representation of the Middle Devonian lithostratigraphy in the Appalachian basin showing the variation from east to west into the basin of Tully Limestone, Hamilton Group and Onondaga Limestone.

Evolving from the simplest net topology, the NEAT introduces nodes and connections into the neural network through genetic algorithm (GA), such as selection, crossover and mutation. The major advantage of NEAT is the ability to evolve and optimize the topology and weights simultaneously, which overcomes the difficulties in designing the fixed-topology methods (Stanley and Miikkulainen, 2002a, 2002b, 2004). However, the NEAT efficiency is low due to iteration count, which will activate each node several times. It becomes worse when the topology is complex or has recurrent connections. The main reason is in adopting the wrong order to activate nodes, which is controlled by node ID. Through setting a more correct order, we can activate each node once to reach the correct output. The correct order is related to the network topology. Thus, it is necessary to introduce new attributes to better characterize the features of the network topology. In addition, the population size of organisms in genetic algorithm is constant in the current NEAT version, which disobeys the natural evolution of populations. Also, at the beginning stage of evolving NEAT, the topology is usually too simple to find the best resolution. Thus, large amounts of time and computer memory are wasted if the population size is large.

To solve these problems, we added two attributes: the node location and a flag for recurrent circle (RCC). These two attributes strongly decrease the cost of time and computer memory during NEAT evaluation. In addition, the population size will not significantly increase with the generation from small to large. To solve multi-class pattern recognition tasks, we combine pairwise comparison and feature selection with NEAT to build modular network. We employ true if an odd number of its arguments are true and false otherwise (XOR) experiments to test and demonstrate these improvements of NEAT. Finally, we introduce the improved NEAT to a significant geologic topic: shale-gas reservoir characterization. The Marcellus Shale lithofacies prediction is a typical multi-class pattern recognition problem and further demonstrates the capability and efficiency of NEAT.

2. The geologic background and Marcellus Shale

The Appalachian basin is an elongated trough as a part of a foreland basin. Thrust faults were formed due to the mountain building plate collisions between the neighboring oceanic plate and the North American craton. The uplift is a result of thrust faults in the eastern area consisting of major tectonic loading that caused differential subsidence, which formed the foreland basin. Large amount of sediments were transported into the basin and built several delta systems along the shoreline. Moving basinward, reduced detrital sediments with more carbonate and high biogenetic productivity were concurrent and produced the organic-rich Marcellus Shale. Underlying the Marcellus Shale, the Onondaga Limestone was deposited in relatively quiet and shallow water (Fig. 2). The Mahantango Shale containing less organic matter in the 2nd order highstand system tract overlies the Marcellus Shale. The Marcellus Shale became thicker to northeast and thinner to west and south (Figs. 1 and 2). A thin but widespread limestone interval divides Marcellus Shale into the lower Union Spring Member and upper Oatka Creek Member.

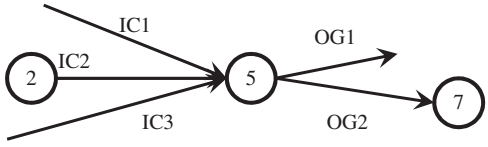
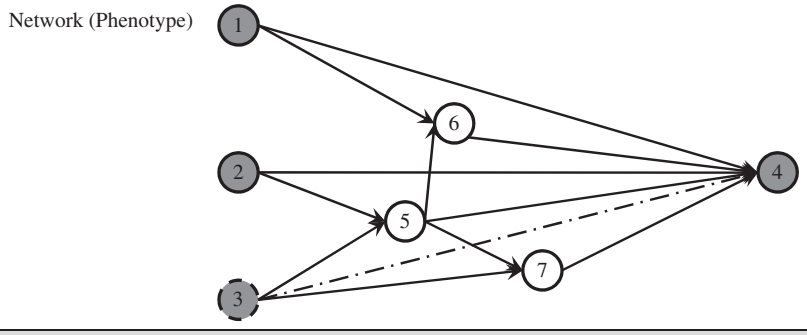


Fig. 3. The *incoming* and *outgoing* connections of a node. As for node ⑤, there are three incoming connections (IC1, IC2 and IC3) and two outgoing connections (OG1 and OG2); nodes ② and ⑦ are the *in node* and *out node* of node ⑤, respectively.

3. Layered NeuroEvolution of Augmenting Topologies

The architecture of classical neural network is organized as one input layer, one output layer and one or more hidden layers. The nodes in one hidden layer are connected with the nodes in its forward and backward layers. The network topology is organized very clearly. However, the initial topology of NEAT has only input and output nodes, and the hidden nodes and connection genes are



Genome (Genotype)

Node Genes

Node ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7
Node Type	1	1	3	2	0	0	0
Node Location	0	0	0	3	1	2	2
Node RCC	False	False	False	False	False	False	False

Connect. Genes

Innovation ID	1	2	3	4	5	6	7	8	9	10	13
In Node ID	1	2	3	2	5	1	6	5	3	7	5
Out Node ID	4	4	4	5	4	6	4	6	7	4	7
Weight	0.7	0.5	-0.8	0.9	-0.1	0.3	0.7	-0.4	1	-0.8	0.5
Enabled	True	True	False	True	True	True	True	True	True	True	True
Is_Recurrent	False	False	False	False	False	False	False	False	False	False	False

Fig. 4. An example to calculate node location in NEAT. Nodes ① and ② are input nodes; node ③ is bias node; node ④ is output node; all others are hidden nodes. The long dish dot line is disenabled connection.

randomly added into the NEAT network through genetic algorithm (GA) mutation which alters one or more gene values in a chromosome from its initial state. GA selection (selecting prospective offspring according to fitness function) and crossover (exchanging gene between two parents) algorithms result in inheritance of the good and promising topologies from former generations. The topology of NEAT can be very complex and disorderly, and current versions do not discuss the features of topologies of NEAT. This creates challenges in design of computer codes and linking to classical neural networks. As a result significant iteration is required to calculate the outputs of NEAT with an increase in costs of computer memory and time. To enhance the efficiency of NEAT and better define the topology of NEAT, two extra properties, node location and a flag of recurrent connection (RCC), are introduced into the structure of the nodes.

3.1. The location of nodes

A node commonly possesses two kinds of connections: incoming connection and outgoing connection (Fig. 3). The outgoing connection of a node links this node to another node, which is called *out node*; while an incoming connection connects a node, which is called *in node* to this node. In order to decide the output of one node, we have to first know the outputs of all its *in nodes*. The output value is incorrect as long as any one of the outputs of all the *in nodes* is unknown. When moving through all the nodes, we need to calculate the output of all the *in nodes* before activating this node. Therefore, it is necessary to possess a node attribute, which describes the relationship of nodes. We introduce the node location attribute to NEAT.

The location of node $\mathbb{A}$ is defined as the largest number of connection genes from any one of the input nodes or bias node to node $\mathbb{A}$. Take hidden node ⑥ in Fig. 4 as an example. All the ways from input and bias nodes to node ⑥ include: (1) ①->⑥; (2) ②->⑤->⑥; (3) ③->⑤->⑥. Thus the node location is two for node ⑥. There is not any connection among all input and bias nodes, thus their node location is set to zero. The location of hidden and output nodes will be changed during augmenting the NEAT topology. At the beginning of NEAT, there is no hidden node and all the output nodes are directly connected to input and bias nodes, so the output nodes have location of one. The mutation of adding nodes and connections may increase the location value of hidden and output nodes. For instance, the added connection ⑥->⑦ and added node ⑧ in ⑦−>④ connection increase location value of node ⑦ and ④ in Fig. 5a and c. However, these mutations sometimes cannot change the location of nodes (Fig. 5b and d). For convenience, we set another two rules concerning node location: (1) disenabling and enabling a connection have no effect on node location (Fig. 5e); (2) the recurrent link should not be counted as determining the location value of nodes (Fig. 5f). If an operation in GA has no effect on the NEAT topology (e.g., mutating weights), it is impossible to change the location of nodes (Table 1). On the contrary, the node location may be changed if the operation in GA

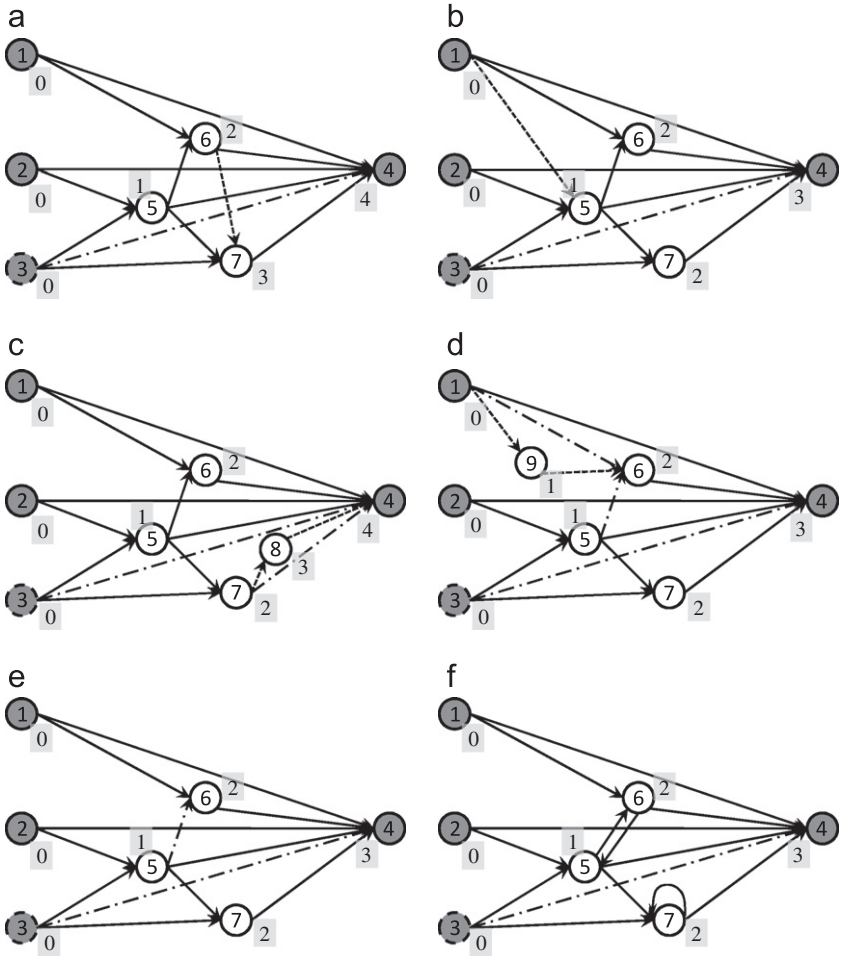


Fig. 5. Effects on node location when adding node and connection into NEAT. (a) the added connection ⑥->⑦ increases the location of node ⑦ to three and node ④ to four; (b) the added connection ①->⑤ has no effect on node location; (c) the added node ⑧ changes the location of node ④ from three to four; (d) the added node ⑨ has no effect on the node location; (e) disenabling and enabling connections will not affect the location of node; (f) the recurrent connection will not change the location of node. The shaded number indicates the location of nodes.

changes the topology of NEAT. It is necessary to re-calculate the node location only if the GA operation is possible to affect node location.

The location of nodes records the spatial relationship of all nodes, characterizing the topology feature of NEAT. The nodes with same location values are considered as locating in the same layer. For example, the location of all input nodes and the bias node is set to zero, so they are in one layer called Input Layer. Nodes ⑥ and ⑦ in Fig. 4 are located in the second Hidden Layer which consists of only hidden nodes with location value of two. When there is more than one output node, the location of output nodes could be different according to the definition. However, we arbitrarily set the location for all output nodes the same as the maximum location, so all the output nodes are in the Output Layer. Based on the node location, all the nodes can be put into corresponding layers and so the NEAT is called Layered NEAT (Fig. 6).

Table 1
Effects of various GA operations on node location and recurrent connections in NEAT.

GA operations		Node location change	Recurrent connections
Selection		No	No
Crossover	In same species	No	No
	In different species	Maybe	Maybe
Mutation	Weights	No	No
	Disenabling connection	No	No
	Re-enabling connection	No	Maybe
	Adding node	Maybe	Maybe
	Adding connection	Maybe	Maybe

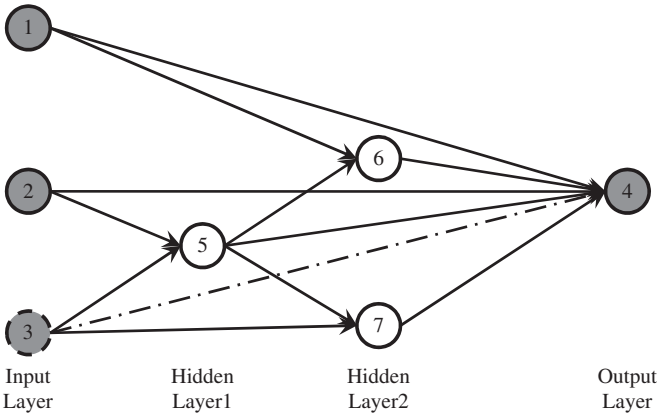
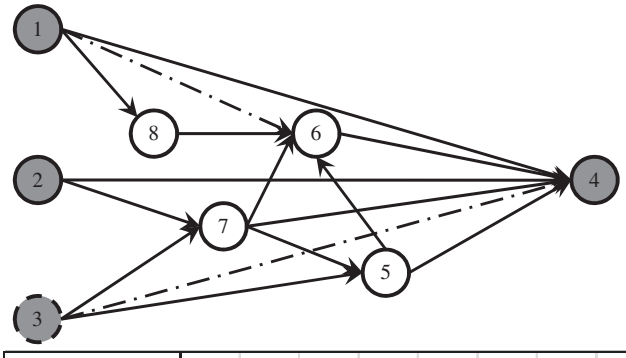


Fig. 6. The Layered NEAT according to the location attribute of nodes in Fig. 4.



Node ID	1	2	3	4	5	6	7	8
Node Type	1	1	3	2	0	0	0	0
Node Location	**0**	**0**	**0**	**4**	**2**	**3**	**1**	**1**

Fig. 7. An example to demonstrate the function of node location attribute in calculating input and output of nodes in correct order.

3.2. The function of node location attribute

3.2.1. Enhance the efficiency of NEAT

One of the most important functions of node location property is to control the order of calculating the outputs of all nodes: activating nodes. The nodes with smaller location values should be activated and calculated earlier than those having larger location values. Without node location attribute, the order to calculate nodes in original NEAT is ①->②->③->④->⑤->⑥->⑦->⑧ in Fig. 7. When we calculate the inputs of node ④ for example, the output of node ⑤, ⑥ and ⑦ is zero, and has not been updated according to their inputs. The real output of node ④ is not reached unless all the nodes connected to node ④ have been correctly calculated. Therefore, iteration is inevitable to activate all the nodes several times. The iteration times are controlled by *net_depth* (Stanley and Miikkulainen, 2002a) which equals to the location of output node. Using node location, the new order is

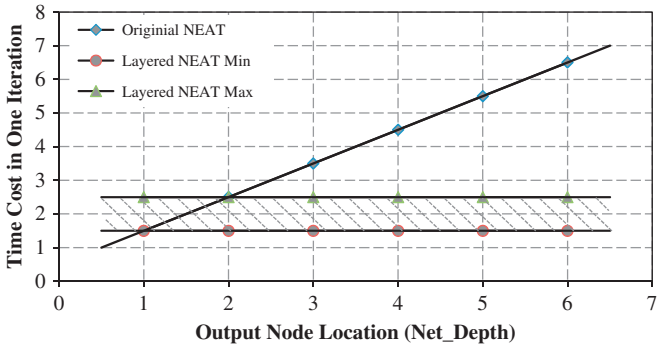


Fig. 8. The plot showing the relationship between output node location and the time cost in calculating the outputs of NEAT. For simplification, we assume that (1) the cost time of each iteration is similar in original NEAT and the Layered NEAT; (2) the time for deciding *net_depth* is about half of the time cost in an iteration for original NEAT; (3) the time for updating node location in Layered NEAT is related to the amount of connections in NEAT and is approximately equal to the time of an iteration; (4) sorting the nodes based on node location costs less than half time cost in an iteration.

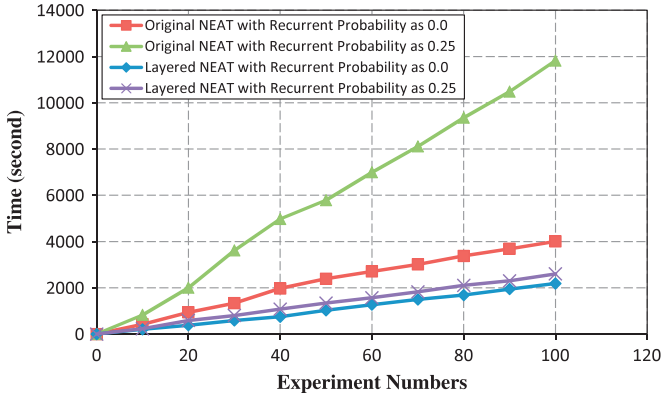


Fig. 9. The XOR experiment results showing the decreased time cost in calculating NEAT outputs with the application of node location and RCC attributes. As setting recurrent probability as zero, the layered NEAT only need one third of the time cost for original NEAT; as setting recurrent probability as 0.25, only 10% of the time cost for original NEAT is cost for Layered NEAT. This experiment is carried out on the computer: Windows XP operation system, Intel® Xeon® CPU, 1.86 GHz and 3.25 GB RAM.

①->②->③->⑦->⑧->⑤->⑥->④ in Fig. 7.Thus, as we calculate the inputs of node ④, all the nodes connected to it have received their correct outputs. This implies that we only need to activate all nodes once and the cost of time and computer memory will be reduced. Especially, significant time and computer memory are saved with the increase of *net_depth* and the complexity of NEAT topology (Fig. 8). The XOR experiment was carried out and utilized to demonstrate the ability of node location in enhancing efficiency of NEAT (Fig. 9). The running speed of the Layered NEAT was improved by a factor of about two compared to the original NEAT, when setting recurrent probability to zero (Table 2).

3.2.2. Build the linkage between NEAT and classical neural network

The classical neural network, such as back-propagation (BP), has a clear architecture: one input layer, one or more hidden layers and one output layer. The node location attribute makes it possible to link NEAT to the classical neural network. The nodes with same node location are considered to be in the same layer, so the node location value can also be considered as ID of hidden layers. In order to compare with classical neural network, we add two hidden nodes in connection ①->④, two hidden nodes in connection ②->④, and one hidden node in connection ⑤->④ (Fig. 10c). All these added hidden nodes are labeled as Ⓐ with the properties: (1) linear transfer function which is $y=x$; (2) the weight of the new added connections related to node Ⓐ is set to one and never changed except the first connection. For example, in the connection ②->④, the new added connections Ⓐ->Ⓐ and Ⓐ->④ has the constant weight of one while ②->Ⓐ has the same weight as the original connection ②->④. The weight of connections with node ③ is the bias for the corresponding node. In Fig. 5c, except nodes ⑤ and ⑦, all other nodes bias is set to zero implying no connection between them and bias node ③. The connections with weights of zero are not shown in Fig. 10c. Therefore, the NEAT in Fig. 10c has the same architecture with classical neural network, and most of the theory and conclusions for classical neural network are also suitable and correct for NEAT.

Table 2
The corresponding data used in Fig. 9 for the XOR experiment.

Experiment numbers	Recur. Prob. is 0.0		Recur. Prob. is 0.25	
	O. NEAT	L. NEAT	O. NEAT	L. NEAT
10	422.49	204.404	827.89	231.72
20	943.00	381.261	2001.47	588.13
30	1333.67	587.806	3610.79	800.65
40	1971.09	749.007	4972.39	1078.46
50	2390.38	1026.238	5786.77	1338.67
60	2708.27	1267.923	6991.27	1570.90
70	3014.74	1494.124	8111.68	1824.63
80	3378.68	1686.497	9349.99	2104.83
90	3687.84	1943.932	10481.93	2300.64
100	4008.54	2187.351	11820.31	2601.59

3.2.3. Make the topology of NEAT more clear

The node location attribute indicates the position of all the nodes in NEAT. We can put all the nodes with same location into the same layer (Figs. 6–10), better defining the topology of NEAT. When we try to compare two NEATs, the node location is very helpful and the description is easier.

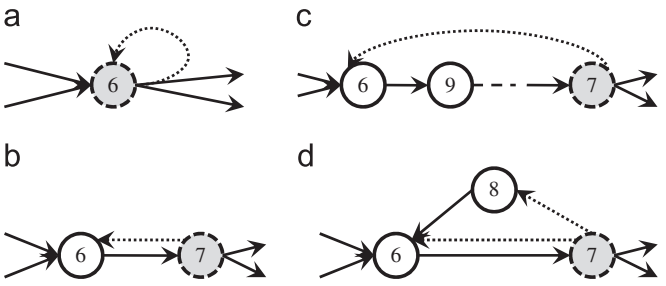


Fig. 11. The classification of recurrent circle in NEAT according the number of involved nodes. (a) The recurrent circle is formed by one node connecting with itself ⑥->⑥ and the RCC of node ⑥ is set to true; (b) two nodes connecting with each other forms a smaller recurrent circle, and the connection ⑦->⑥ is marked as recurrent connection and node ⑦ has RCC as true; (c) the recurrent circle includes more than two connections, forming one big circle; the connection ⑦->⑥ is marked as recurrent connection and node ⑦ has RCC as true; (d) a new node is added into the recurrent connection ⑦->⑥. The dish line indicates the recurrent connection and the dish circle implies the RCC of this node as true.

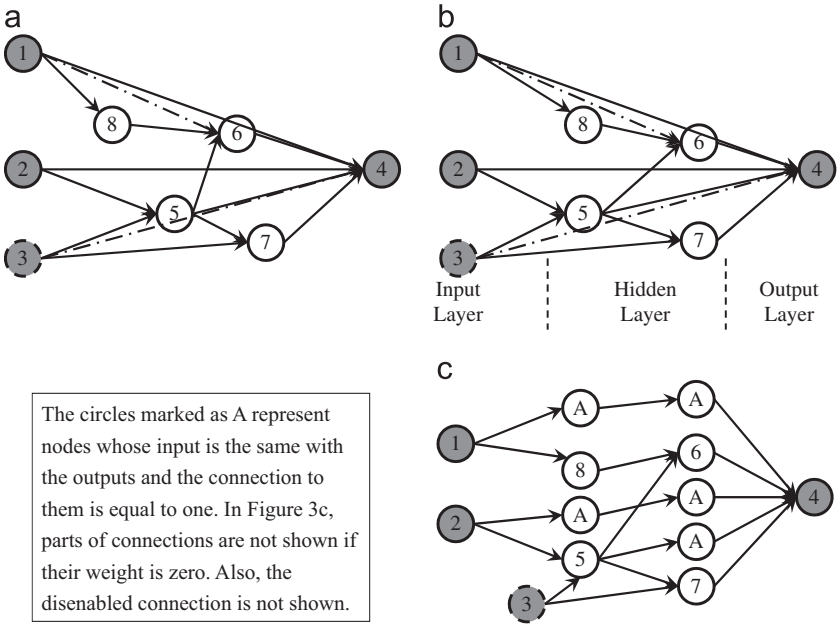
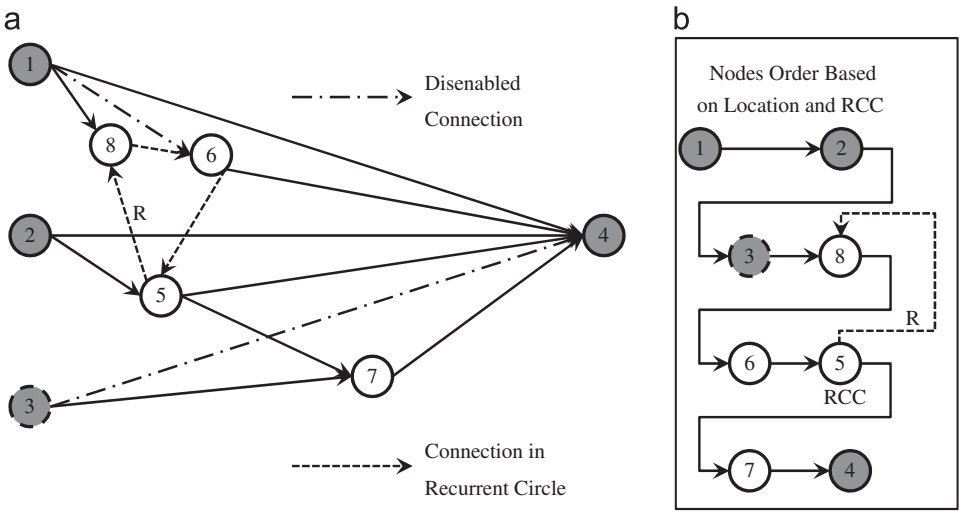


Fig. 10. Linking NEAT to classical neural network through the node location attribute. (a) NEAT without node location; (b) layered NEAT by node location; (c) the equivalent classical neural network.



c

Genome (Genotype)

Node Genes	Node ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
	Node Type	1	1	3	2	0	0	0	0
	Node Location	0	0	0	8	3	2	4	1
	Node RCC	False	False	False	False	True	False	False	False

Connect. Genes	Innovation ID	1	2	3	4	5	6	7	9	10	12	13	15	17	20
	In Node ID	1	2	3	2	5	1	6	6	3	7	5	1	8	5
	Out Node ID	4	4	4	5	4	6	4	5	7	4	7	8	6	8
	Weight	0.7	0.5	-0.8	0.9	-0.1	0.3	0.7	-0.4	1	-0.8	0.9	0.5	-0.2	0.5
	Enabled	True	True	False	True	True	False	True	True	True	True	True	True	True	True
	Is_Recurrent	False	False	False	False	False	False	False	False	False	False	False	False	False	True

Fig. 12. The process to simulate NEAT with recurrent connections. (a) the NEAT topology with recurrent cycle; (b) the new order of all the nodes according to node location and RCC attributes; (c) the corresponding genome (Genotype) of the NEAT in Fig. 10a with node location and RCC attributes.

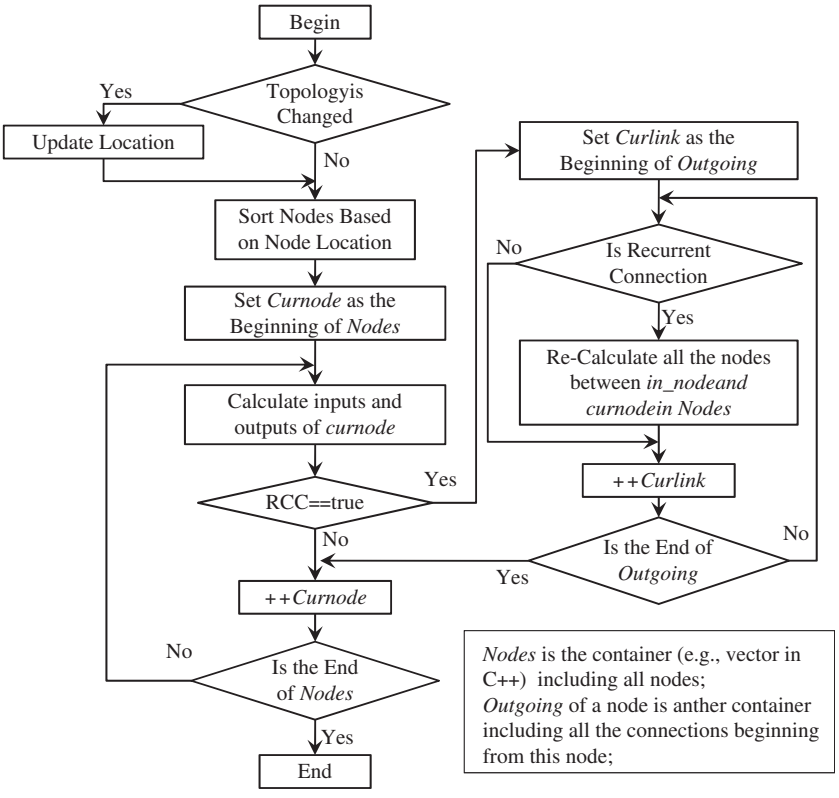


Fig. 13. The new workflow for NEAT dealing with recurrent connection through using node location and RCC attributes. "*Curnode*" stands for the *current node*; "*Curlink*" indicates the *current link*.

3.3. The RCC property of node

The recurrent neural network (RNN) is a type of neural network where connections between neurons form one or more than one directed cycle. This creates an internal state of the network which can simulate dynamic temporal behavior and the memory function in the human brain (Huang et al., 2007). RNN is very useful to process sequence of patterns and time-related problems. The connections that form directed cycles are named recurrent connection here. The recurrent connections can connect one node with itself, or with another one, or involve more than two nodes forming one or more than one large circles (Fig. 11).

The RCC is the abbreviation of recurrent circle, and labels the beginning of a recurrent circle. A recurrent circle is composed of one or more than one connection, and only one of the connections is marked as recurrent connection. As a node connecting with itself by one node (Fig. 11a), this connection is labeled as recurrent connection and the RCC of this node is set to *true*. If the recurrent circle includes over one connection (Fig. 11b and c), the connection that is last added into the NEAT is marked as recurrent connection, and the *in node* of this connection possesses a *true* value of RCC. For example, the connection ⑥->⑤in Fig. 12a is labeled as recurrent connection and the RCC of node ⑥ is set to true. As a node is added into a recurrent connection, only the first new added connection is labeled as recurrent connection while the second one is not. For example, the connection ⑦->⑧is set as recurrent connection as node ⑧ is added into the recurrent connection ⑦->⑥ in Fig. 11d.

3.4. The functions of node RCC attribute

The primary use of node RCC attribute is to deal with the calculation of recurrent circles in NEAT. In the original version of NEAT designed by Kenneth Stanley, the presence of recurrent connections causes a large *net_depth* (e.g., 100). For example, the *net_depth* is 100 rather than five in Fig. 12a due to the recurrent circle ⑤->⑧->⑥->⑤. Therefore, every node is activated for 100 times and the outputs of NEAT are not correct. The node location and RCC attributes are very powerful to solve the simulation of recurrent circles in NEAT.

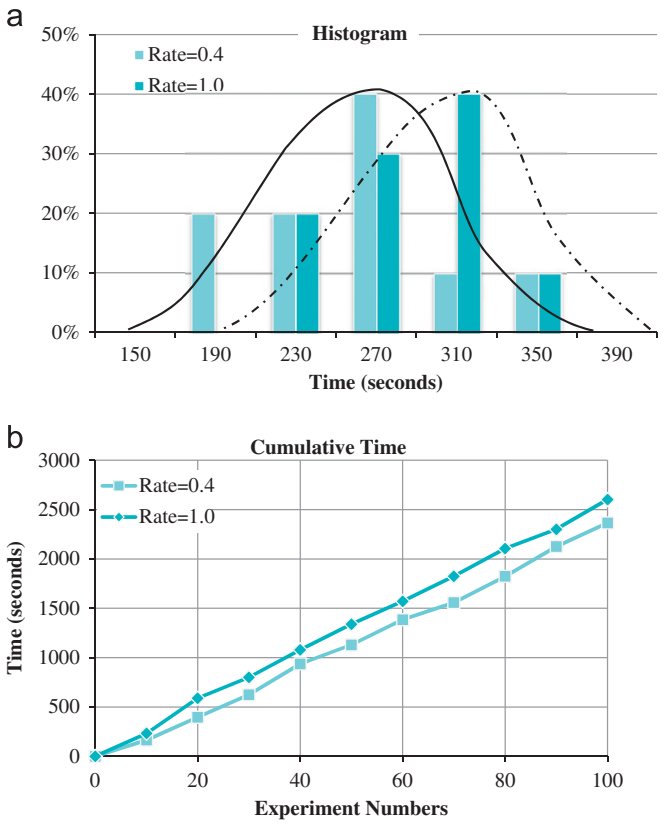


Fig. 15. The XOR experiment showing the difference of time cost with rate=0.4 and rate=1.0. (a) is the histogram; (b) is the plot with experiment numbers versus time.

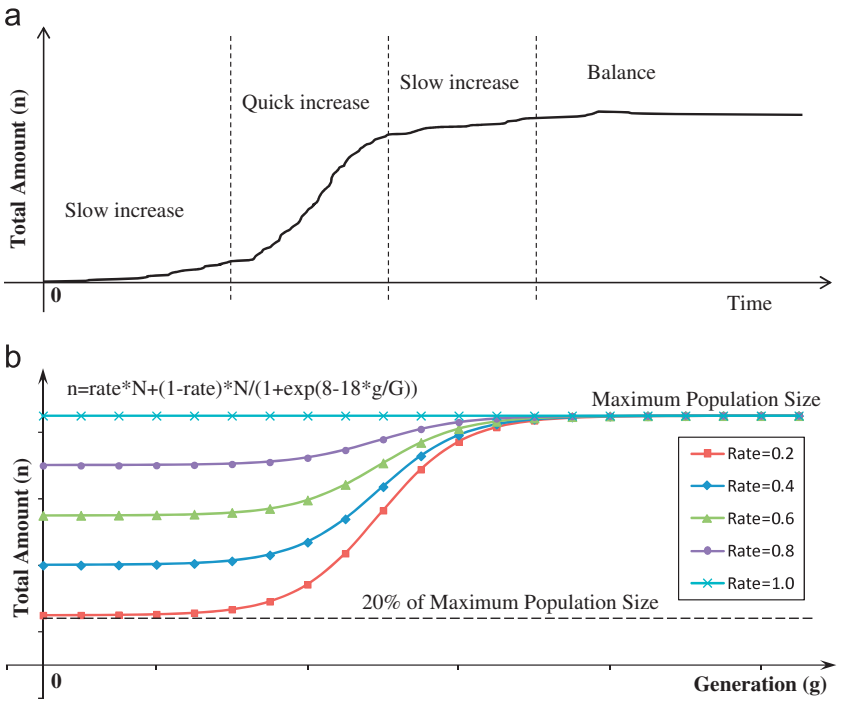


Fig. 14. The overall trend of the amount of organisms during evolution (a) and its application in genetic algorithm (b). *N* indicates the maximum population size; *G* means the maximum generation; *n* is the population size at generation *g*.

As mentioned in previous sections, the node location is used to sort the nodes with regard to their location attribute from small to large (Fig. 12b). If the RCC is set to true, the node RCC attribute marks the beginning of a recurrent circle. When moving to the node whose RCC is true, the program will search all the recurrent connections from this node and then move back to the *outnode* of these recurrent connections. Therefore, a recurrent circle leads to one more activation of all the nodes in the recurrent circle. For example, the NEAT in Fig. 12 will move back to node ⑧ when node ⑤ is activated at the first time, and nodes ⑧, ⑥and ⑤ are calculated twice in the whole loop. A more detailed workflow is shown in Fig. 13. The efficiency of NEAT with the application of node location and RCC attributes is enhanced more than four times in the XOR experiment compared to the original NEAT (Fig. 9).

4. Population size

The NEAT utilizes a genetic algorithm to optimize the weights and topology simultaneously. Thus, the design of a genetic algorithm has a significant effect on the performance and the calculation speed of NEAT. The current version of NEAT sets the population size as constant, which means that total number of individuals in each generation is the same. Clearly, this is not true for population dynamics. The growth of populations usually follows a logistic curve that is limited by available resources, such as food, water, energy, and space. The total number of organisms will reach a maximum stable value where a balance between organism and resources is created. Even though rapid fluctuations in populations are possible, the overall trend can be described by a generalized logistic function that can be modeled as a "S-shaped" curve of growth of a population. The initial stage of growth is approximately exponential; then as saturation begins, the growth slows, and at maturity, growth stops (Fig. 14a). The evolution of NEAT topology starts from minimal structure, which is similar with the simplest population and diversity structures. Generally, only if enough amounts of hidden nodes are inserted into the network, generating an adequate topology of NEAT for complex problems becomes possible. Therefore, it is not necessary to have large amount of individuals at the initial stage of NEAT evolution, which results in inefficient calculation. Along with generation increase, large population size will provide enough high-quality individuals for genetic algorithm to select and inherit good topology for the next generation. Thus, increasing the population size from a small value can not only save time for calculation, but also becomes more consistent with population dynamics. The parameter *rate* is introduced to control the initial population size and its growing

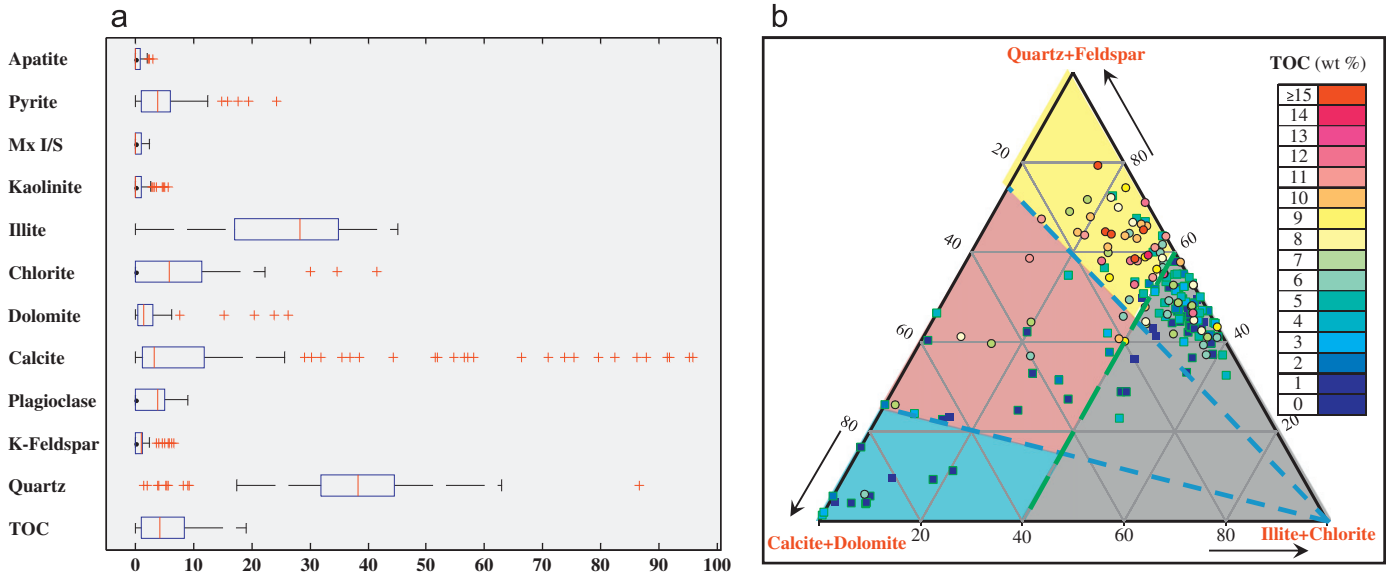


Fig. 16. The XRD and TOC data from core samples for Marcellus Shale (a) and the defined mudrock lithofacies based on these XRD and TOC data (b) in the Appalachian basin.

Table 3
A summary of characteristics of the seven Marcellus Shale lithofacies defined by core XRD and TOC data and PNS logs.

Lithofacies	Features of core data and PNS logs	Features of conventional logs					
		GR (API)	RHOB (g/m^3)	NEU (%)	ILD (ohm)	PE	URAN (ppm)
Organic Siliceous Shale (OSS)	$TOC \geq 6.5\%$ (9.7); $V_{clay} < 40\%$ (25.6); $V_{quar}/V_{carb} > 3$	(278–785)/449	(2.1–2.48)/2.39	(18.0–30.4)/24.1	(152–2505)/854	(2.7–7.9)/3.5	(20–80)/40
Organic Mixed Shale (OMS)	$TOC \geq 6.5\%$ (9.6); $V_{clay} < 40\%$ (19.8); $1/3 < V_{quar}/V_{carb} < 3$	(242–628)/424	(2.3–2.59)/2.47	(13.3–27.5)/20.3	(95–1475)/450	(3.3–6.3)/4.3	(18–73)/34
Organic Mudstone (OMD)	$TOC \geq 6.5\%$ (6.1); $V_{clay} \geq 40\%$ (47.6);	(315–793)/491	(2.4–2.64)/2.53	(21.8–29.7)/25.7	(26–414)/164	(3.4–7.0)/4.4	(17–74)/36
Gray Siliceous Shale (GSS)	$TOC < 6.5\%$ (4.0); $V_{clay} < 40\%$ (32.0); $V_{quar}/V_{carb} > 3$	(115–277)/193	(2.4–2.62)/2.59	(15.5–24.1)/19.8	(20–241)/94	(2.8–4.8)/3.5	(2–14)/8
Gray Mixed Shale (GMS)	$TOC < 6.5\%$ (2.5); $V_{clay} < 40\%$ (17.1); $1/3 < V_{quar}/V_{carb} < 3$	(87–178)/131	(2.5–2.75)/2.66	(6.3–23.4)/14.4	(24–282)/103	(3.5–5.1)/4.2	(1.5–12)/5.4
Gray Mudstone (GMD)	$TOC < 6.5\%$ (1.5); $V_{clay} \geq 40\%$ (46.9);	(139–229)/209	(2.4–2.70)/2.60	(17.2–27.7)/22.6	(19–121)/56	(3.3–5.5)/3.8	(2–15)/8.1
Carbonate Interval (CARB)	$TOC < 6.5\%$ (1.3); $V_{clay} < 40\%$ (5.6); $V_{quar}/V_{carb} < 1/3$	(24–135)/82	(2.6–2.75)/2.69	(1.8–15.2)/6.2	(61–1432)/636	(3.9–5.1)/4.7	(0–7.7)/3.1

speed in NEAT (Fig. 14b). A function that matches the trend of population size is as following:

$$n = rate \times N + \frac{(1-rate) \times N}{1+e^{(8-18g/G)}}$$

where n is the population size in generation g; N represents the maximum population size; G indicates the maximum generation; *rate* controls the trend of population size.

The smaller the rate is, the smaller the initial population size is and the faster the growing of population size is. A larger value of

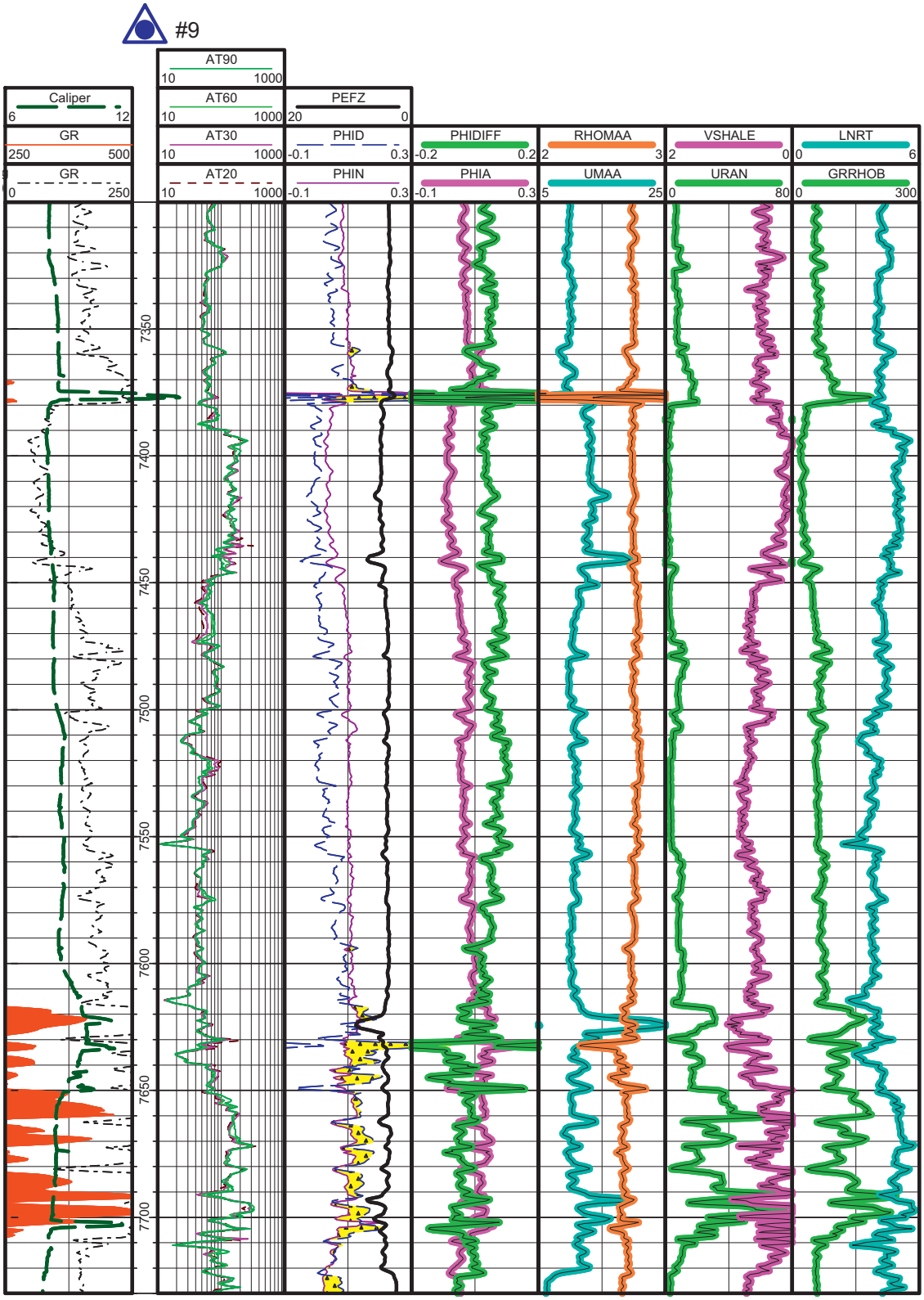


Fig. 17. One example of common log suites and the eight derived parameters used for Marcellus Shale lithofacies prediction in the Appalachian basin.

rate is recommended for simple problems while a smaller *rate* is preferred for complex problems. The function of population size growth is more effective for complex problems, which are rarely possible to solve at the beginning of evolving NEAT network. The XOR experiment was carried out again to demonstrate the function of rate in improving the efficiency of NEAT (Fig. 15).

5. The application of NEAT on black shale lithofacies classification

Core samples from deep wells provide the richest and most accurate information concerning the mineralogy and organic matter richness of black organic-rich mudrock. However, the amount of available core data is seriously limited due to the high cost of collecting and analyzing. The core samples rarely cover the entire stratigraphic intervals of interest and wells with core may be sparsely scattered across a region. Therefore, it is necessary to develop a method to leverage the sparse core samples that uses data that is more abundant and economic to provide similar, even enhanced, accuracy and resolution of rock properties. In the subsurface, wireline logs are usually sampled every half foot and thus have high vertical resolution (about 0.2–5 ft) parallel to well borehole. In the modern era, most wells are logged to record the physical properties of underground rocks, providing spatially the most abundant data sets in sedimentary basins. This is one reason that efforts have been placed on predicting reservoir properties (e.g., porosity, permeability, thickness, water saturation, and lithology) from wireline logs in conventional reservoirs (Berteig et al., 1985; Liu et al., 1992; Wong et al., 1995; Chang et al., 2000; Qi and Carr, 2006; Dubois et al., 2007; Al-Anazi and Gates, 2010). In this paper, we try to test the ability of NEAT to predict organic-rich mudrock lithofacies using conventional wireline logs.

For the Marcellus Shale lithofacies prediction, we collected about 190 sets of XRD and TOC data points from cores in 18 wells located in West Virginia and southwest Pennsylvania (Fig. 16a). The concentration by volume of quartz, carbonate and clay are derived for all the measured minerals from XRD analysis, and then plotted in the ternary plot (Fig. 16b). Mineralogy coupled with TOC data, is used to define seven mudrock lithofacies in Marcellus Shale (Table 3). Among all the core data, 157 data points also have corresponding wireline log data, so they compose one of the training data sets. Another training data set is from the pulsed neutron spectroscopy logging (PNS), which is an advanced log suite that when tied to core data can accurately model mineralogy. The PNS logs measure the concentration of various elements in rocks, and the element concentration is then modeled into mineral concentration and TOC content. The mineral concentration and TOC content can be used to classify mudrock lithofacies with the same method utilized in core samples. Compared to the conventional logs (e.g., Gamma Ray and Density), the PNS log is more expensive and only acquired in a few key wells. The Marcellus Shale lithofacies defined by core samples analysis and PNS log data are the target values in the training data sets to be used with the abundant conventional log suites.

Various kinds of wireline logs exist along with the development of logging technology. But not all of the log curves are helpful for mudrock lithofacies study as the input variables; unless the logs are commonly available and can reflect rock mineral composition, mechanical properties, and/or richness of organic matter. In the Appalachian basin, the common log suites are routinely acquired in most wells, which include gamma ray (GR), density (RHOB), compensated neutron (NEU), photo-electric (PE) and deep resistivity (ILD) logs (Fig. 17). The combination of these common log curves can be used to recognize mudrock lithofacies by integrating with the core and PNS data. In place of the direct application of conventional logs, eight petrophysical parameters derived from the conventional logs were used for Marcellus Shale lithofacies recognition. The parameters are Uranium concentration, brittleness, RHOmaa, Umaa, average porosity, porosity difference, natural logarithm of deep resistivity, and the ratio of gamma ray to density (Wang and Carr, 2012). In this research, we used the eight derived variables as the inputs of NEAT to recognize Marcellus Shale Lithofacies.

The Marcellus Shale lithofacies prediction is a typical multi-class pattern recognition problem. It is not trivial to extend binary-class learning algorithms to address multi-class problems (Ou and Murphey, 2007). A feasible approach is to decompose the problem into a set of binary classification tasks. The popular decomposing technologies include one-versus-the-rest, pairwise comparison (Hastie and Tibshirani, 1998), and error-correcting output coding (Dietterich and Bakiri, 1995). The pairwise comparison, compared with the other technologies, can avoid imbalanced

Table 4
The major parameters of NEAT used for Marcellus Shale lithofacies prediction by common logs.

	Modular Neural Network with 21 binary classifiers	
Initial topology	Input nodes	Random (between 1 and 8)
	Bias node	1
	Output nodes	1
	Transfer function	$y=1/(1+(\exp(-2.0x)))$
Training dataset	Total amount	2012
	Training subset amount	1510 (75%)
	Validation subset amount	502 (25%)
	Input variables amount	8
	Target classes amount	7
Generic algorithm setting	Population size	200
	Population increasement rate	0.4
	Maximum generation	500
	Mutate	
	Link weight probability	0.9
	Inserting node probability	0.03
	Inserting link probability	0.05
	Mate	
	Interspecies mate rate	0.001
	Multipoint mate rate	0.6
	Singlepoint mate rate	0
	Recurrent connection probability	0.25
	Linking to disconnected input node probability	0.05 and 0.2

training data, carry out parallel computation and perform better with huge training data set. One major weakness of pairwise comparison method is to build large amount of binary neural networks as m is big, which costs significant time. But, this can be overcome by developing programs to automatically build all the binary neural networks. Thus, we chose the pairwise comparison technology for NEAT to deal with multi-class classification problems.

The core- and PNS-defined mudrock lithofacies and the corresponding conventional logs compose the training dataset for the supervised-learning algorithms (e.g., genetic algorithm in NEAT). 75% of all the training data are employed to build the relationship between mudrock lithofacies and conventional logs and the other 25% is used to validate the ability of NEAT in prediction of Marcellus Shale lithofacies. With a large training dataset, the modular network is more effective than single network with multiple output nodes (Qi and Carr, 2006). Thus, we build 21 binary NEAT networks with one output node to constitute the modular NEAT network. The feature selection function of the new version of NEAT is employed to automatically evaluate and select the sensitive input variables for every binary NEAT network, and thus the initial amount of input nodes varies randomly between one and eight. The probability to involve a new input variable into the NEAT network is set to as low as 0.01. A higher probability of involving new input variables will reduce the opportunity to test various combinations of input variables due to the fact that variables are gradually added into, but never dropped from the network. The maximum generation and population size are set to 500 and 200, respectively. The recurrent connections are

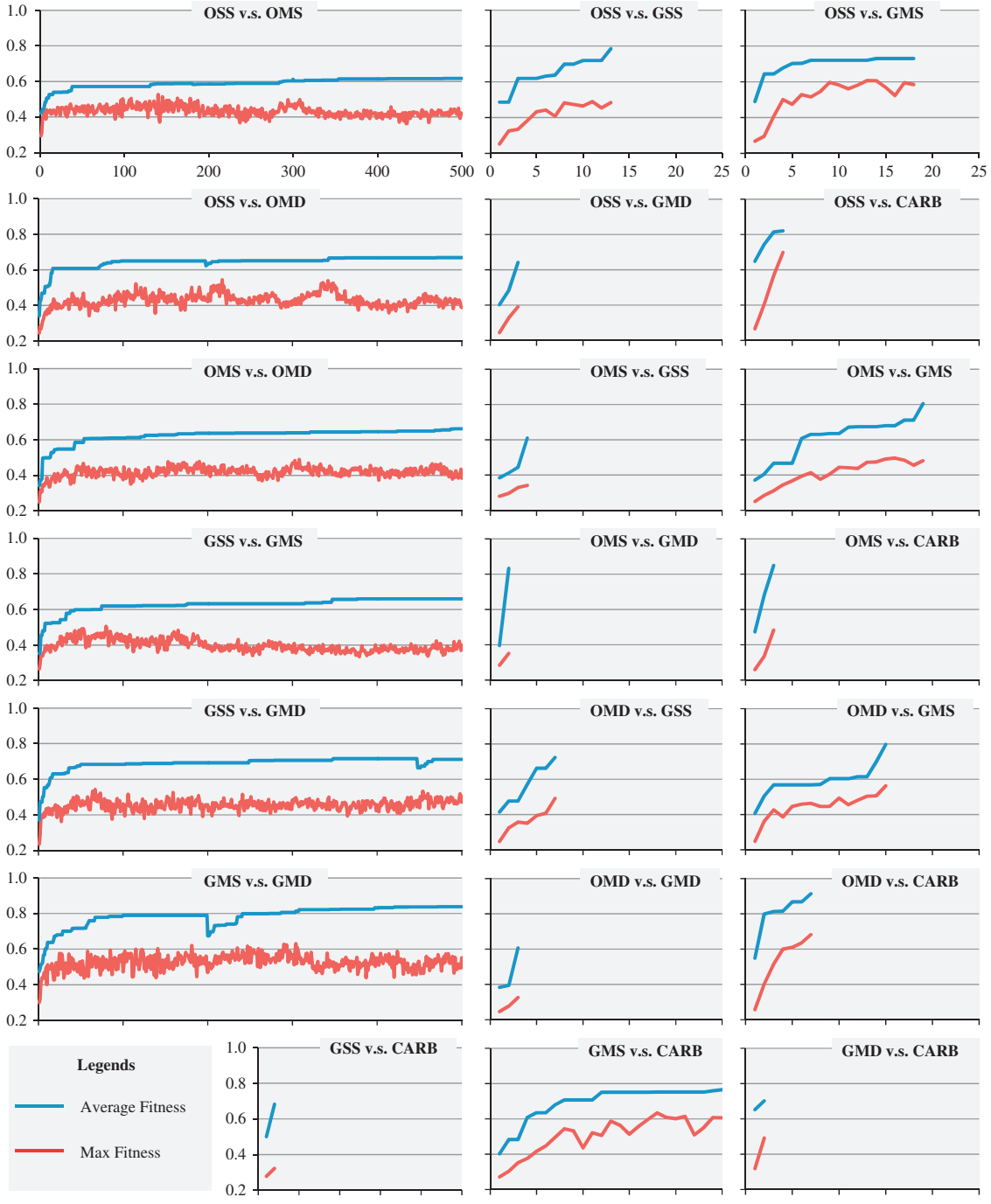


Fig. 18. The average amount of nodes and connections of all the binary NEAT classifiers for Marcellus Shale lithofacies recognition in the Appalachian basin.

introduced into the NEAT network with the probability of 25%. We suggest using a smaller probability of introducing recurrent connection, unless the network topology will be too complex. The other main parameters related to genetic algorithms are listed in Table 4.

The 21 binary NEAT classifiers are trained one-by-one through genetic algorithms and the results from all the binary networks are transferred into the predicted type of Marcellus Shale lithofacies according to specific rules, for example, the max-win rule (Hastie and Tibshirani, 1998). It is easier to optimize the topology and connection weights as the two types of mudrock lithofacies are quite different (e.g., organic siliceous shale and carbonate interval), so less connections and hidden nodes are needed with less generation of evolution. On the contrary, more connections and nodes are added into the NEAT network when the two lithofacies are similar (e.g., organic siliceous shale and organic mixed shale). Along with more training generations, more nodes and connections are added into the NEAT networks and the average fitness is gradually increasing (Figs. 18 and 19). Based on the training results (Figs. 18–20), most of the binary NEAT networks do not need hidden nodes to differentiate one lithofacies from another quite distinctive lithofacies. Only six binary NEAT networks require more hidden nodes (seven and 15), and the hidden nodes are located in at least five layers, which indicate a different design of network topology from the conventional neural networks (Fig. 20). For the conventional neural networks (e.g., BP neural network), we usually put all the hidden nodes in one or two hidden layers, and it is rare to employ more than three

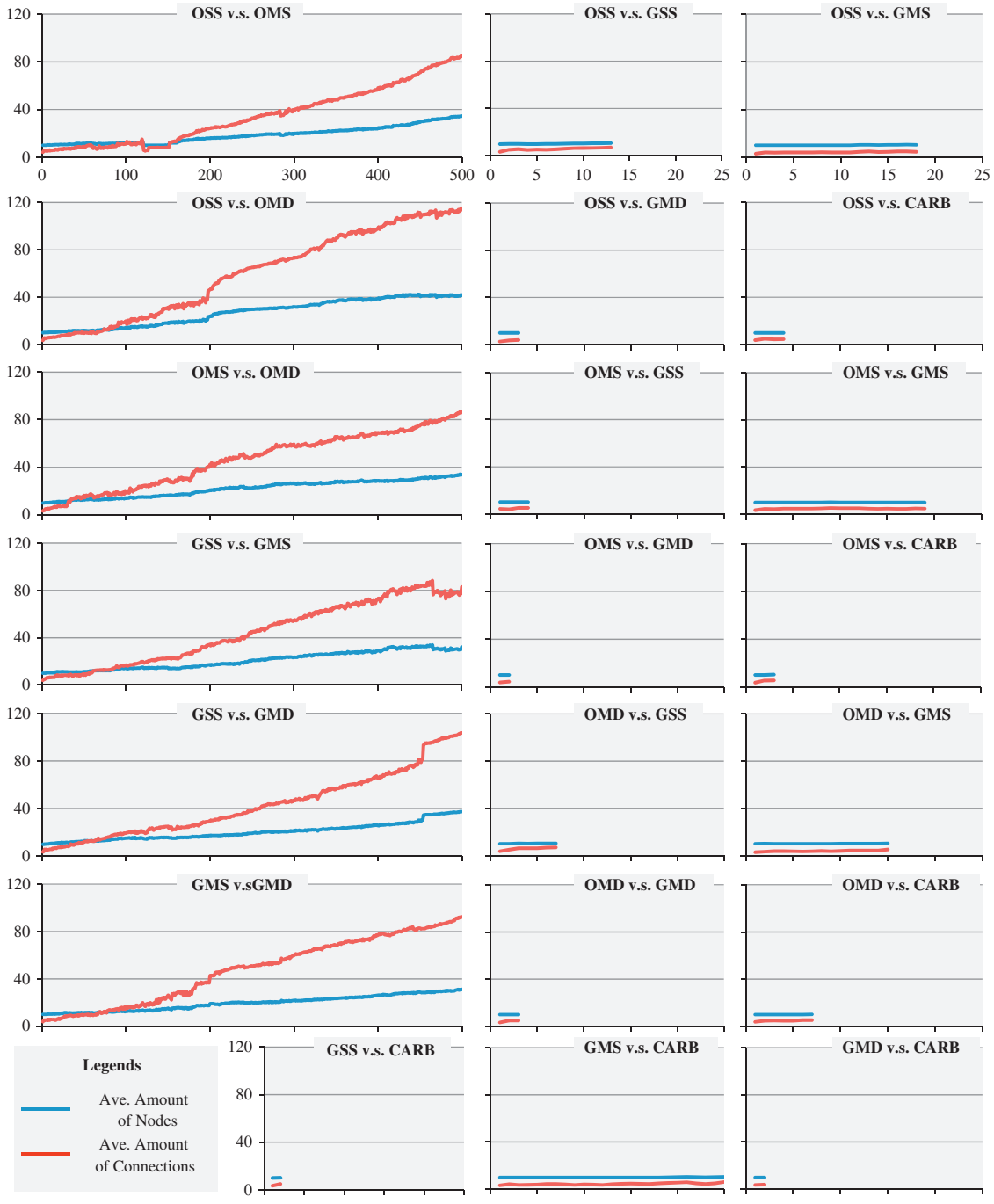


Fig. 19. The average and maximum fitness of all the binary NEAT classifiers for Marcellus Shale lithofacies recognition in the Appalachian basin.

hidden layers. However, the automatically evolved NEAT networks involve more hidden layers with less hidden nodes in each layer. It is an interesting result even though we are unsure whether this is an improved topology compared to the conventional neural network. With regard to the feature selection function, the involved input variables are different for each binary NEAT classifier (Table 5). Instead of using all the parameters, we use only porosity difference, uranium and the ratio of GR to density to classify organic siliceous shale and gray siliceous shale, which simplifies the network topology. The correct classification from the modular network is about 75% for the training part and close to 70% for the validation part (Table 6). Considering the complexity, this appears to be a good solution for predicting Marcellus Shale lithofacies.

6. Conclusions

The NeuroEvolution of Augmenting Topologies (NEAT) can evolve the topology and connection weights of the neural network at the same time through genetic algorithm. The node location attribute describes the position relationship of all the nodes, better defining the NEAT topology. Also, the efficiency of calculating the NEAT output is significantly enhanced through

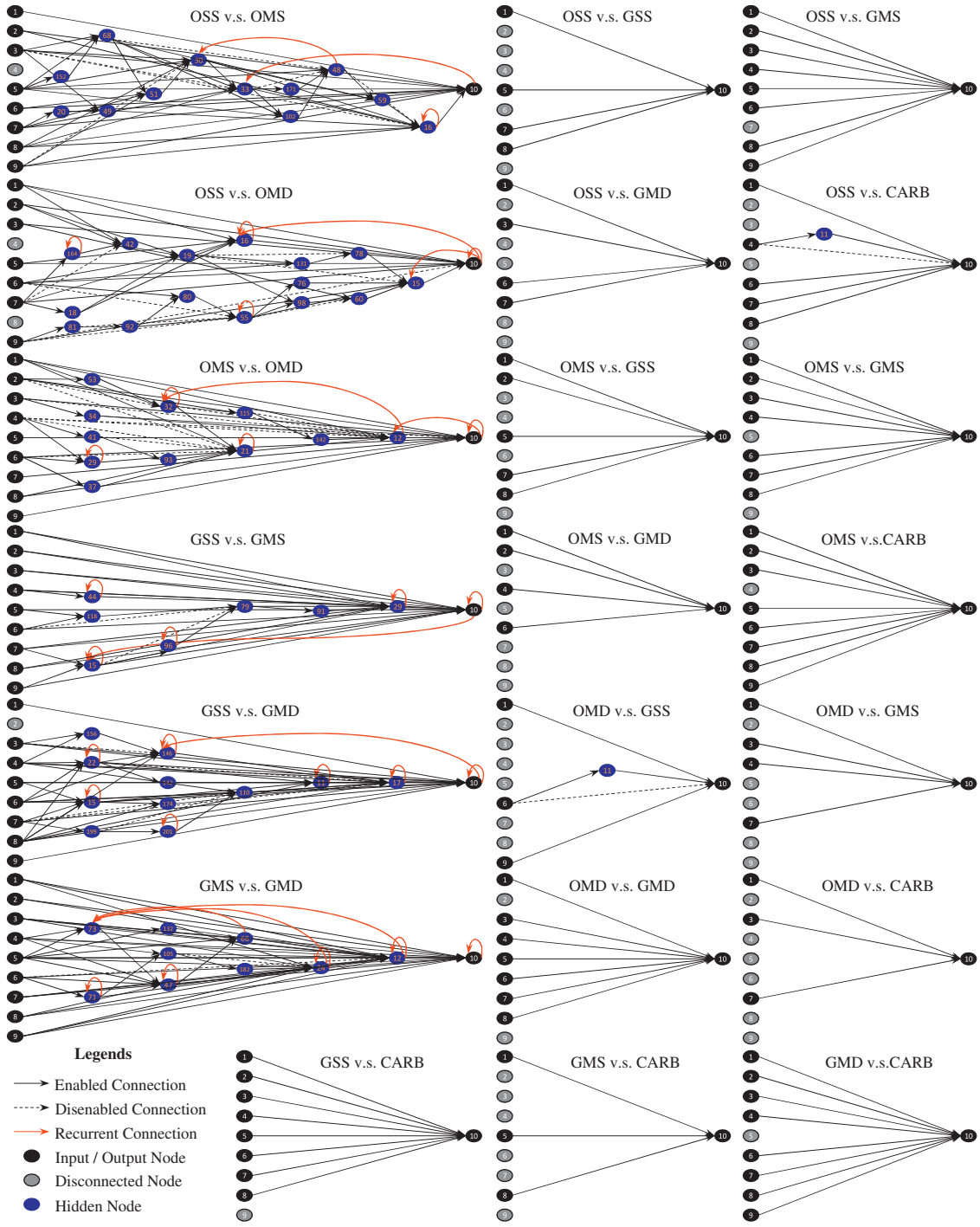


Fig. 20. The topology of all the binary NEAT classifiers for the Marcellus Shale lithofacies prediction in the Appalachian basin.

Table 5
Feature selection result of all the binary NEAT classifiers for Marcellus Shale lithofacies recognition in the Appalachian basin.

Binary Classes			Porosity Ave.	Umaa	RHOmaa	Porosity Dif.	LnILD	Uranium	GR/Den	Shale volume
OSS	***v.s.***	OMS	√	√		√	√	√	√	√
OSS	***v.s.***	OMD	√	√		√	√	√		√
OSS	***v.s.***	GSS				√		√	√	
OSS	***v.s.***	GMS	√	√	√	√	√	√		√
OSS	***v.s.***	GMD		√			√	√		
OSS	***v.s.***	CARB			√		√	√	√	
OMS	***v.s.***	OMD	√	√	√	√	√	√	√	√
OMS	***v.s.***	GSS	√			√		√	√	
OMS	***v.s.***	GMS	√	√	√		√	√	√	
OMS	***v.s.***	GMD	√		√		√			
OMS	***v.s.***	CARB	√	√		√	√	√	√	√
OMD	***v.s.***	GSS					√			√
OMD	***v.s.***	GMS		√	√			√		
OMD	***v.s.***	GMD		√	√	√	√	√	√	
OMD	***v.s.***	CARB		√				√		
GSS	***v.s.***	GMS	√	√	√	√	√	√	√	√
GSS	***v.s.***	GMD		√	√	√	√	√	√	√
GSS	***v.s.***	CARB	√	√	√	√	√	√	√	
GMS	***v.s.***	GMD	√	√	√	√	√	√	√	√
GMS	***v.s.***	CARB				√			√	
GMD	***v.s.***	CARB	√	√	√		√	√	√	√

Table 6
The confuse plot showing the results of Marcellus Shale lithofacies prediction by NEAT in the Appalachian basin. The lithofacies codes are the same with Table 3.

	Facies no.			Predicted facies					Right rate (%)
	Core facies	OSS	OMD	GSS	GMS	GMD	CARB	Core Total	
Training data set									
OSS	218	17	61					296	73.65
OMS	15	95	22		6	1		139	68.35
OMD	59	5	267		1	3		335	79.70
GSS	8	5	4	95	10	10		132	71.97
GMS	1	2	3	2	76	20	5	109	69.72
GMD	16	15	26	47	24	319	15	462	69.05
CARB					8	10	47	65	72.31
Pred total	317	139	383	144	125	363	67	**1538**	**72.63**
Pred/core	1.07	1.00	1.14	1.09	1.15	0.79	1.03	All facies	

setting the correct order of activating the nodes. In addition, the node location attribute makes it possible to compare the architecture of NEAT and conventional neural networks. The RCC attribute marks the beginning of recurrent connections, and changes the order of activating nodes when the value of RCC is true. These two new introduced attributes can improve the speed of NEAT through reduced calculation time and computer memory usage. By setting the growth rate of population size to less than one, the population size in genetic algorithm is gradually increased better following the logistic function of population growth, which can improve efficiency by reducing the number of organisms at the early stage of NEAT evolution. A small value of population size growth rate is recommended for the complex problems that are not amenable to an early solution in the beginning of NEAT evolution. The experiments of XOR demonstrated the function and effectiveness of node location, RCC and gradual growth of population size in improving the speed of NEAT calculation, especially for NEAT with recurrent connections. The feature selection is very effective to solve the multi-class pattern recognition problems, which can automatically select the sensitive variables and optimize the network topology and connection weights for each binary NEAT network. Finally, we use the improved NEAT to predict Marcellus Shale lithofacies by eight petrophysical parameters derived from conventional logs in the Appalachian basin. The training results demonstrated the ability of NEAT in solving multi-class pattern recognition problems with the new features including node location, RCC, population size growing, feature selection and modular network.

Acknowledgments

This research was supported by the U.S. Department of Energy National Energy Technology Laboratory (Activity 4.605.920.007) and National Natural Science Foundation of China (No. 698796867). Special thanks to Energy Corporation of America, Consol Energy, EQT Production and Petroleum Develop Corporation for providing data.

Appendix A. Supplementary material

Supplementary materials associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.cageo.2013.01.022.

References

Al-Anazi, A., Gates, I.D., 2010. A support vector machine algorithm to classify lithofacies and model permeability in heterogeneous reservoirs. Engineering Geology 114 (3–4), 267–277.

Berteig, V., Helgeland, J., Mohn, E., 1985. Lithofacies prediction from well data. In: Proceedings of SPWLA Twenty-Sixth Annual Logging Symposium.

Brett, C.E., Baird, G.C., 1996. Middle Devonian sedimentary cycles and sequences in the northern Appalachian basin. Geological Society of America 306, 213–241 (Special Paper).

Chang, H., Kopaska-Merkel, D.C., Chen, H., Durrans, S.R., 2000. Lithofacies identification using multiple adaptive resonance theory neural networks and group decision expert system. Computers and Geosciences 26 (5), 591–601.

Dietterich, T.G., Bakiri, G., 1995. Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2, 263–286.

DOE/NETL, 2010. Projecting the economic impact of Marcellus Shale gas development in West Virginia: A preliminary analysis using publicly available data. 〈http://www.netl.doe.gov/energy-analyses/refshelf/detail.asp?pubID=305〉.

Dubois, M.K., Bohling, G.C., Chakrabarti, S., 2007. Comparison of four approaches to a rock facies classification problem. Computers and Geosciences 33 (5), 599–617.

Hastie, T., Tibshirani, R., 1998. Classification by pairwise coupling. Annals of Statistics 26 (1), 451–471.

Huang, B.Q., Rashid, T., Kechadi, M.-T., 2007. Multi-context recurrent neural network for time series applications. International Journal of Computational Intelligence 3 (1), 45–54.

Liu, R.L., Zhou, C.D., Jin, Z.W., 1992. Lithofacies sequence recognition from well logs using time-delay neural networks. In: Proceedings of SPWLA 33rd Annual Logging Symposium.

Ou, G., Murphey, Y.L., 2007. Multi-class pattern classification using neural networks. Pattern Recognition 40, 4–18.

Qi, L., Carr, T.R., 2006. Neural network prediction of carbonate lithofacies from well logs, Big Bow and Sand Arroyo Creek fields, Southwest Kansas. Computers and Geosciences 32 (7), 947–964 〈http://linkinghub.elsevier.com/retrieve/pii/S0098300405002396〉.

Stanley, K.O. Kenneth Stanley' Website: 〈http://www.cs.ucf.edu/~kstanley/neat.html〉.

Stanley, K.O., Miikkulainen, R., 2002a. Evolving neural networks through augmenting topologies. Evolutionary Computation 10 (2), 99–127.

Stanley, K.O., Miikkulainen, R., 2002b. Efficient evolution of neural networks topologies. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02).

Stanley, K.O., Miikkulainen, R., 2004. Competitive coevolution through evolutionary complexification. Journal of Artificial Intelligence Research 21, 63–100.

Wang, G., Carr, T.R., 2012. Methodology of organic-rich shale lithofacies identification and prediction: a case study from Marcellus Shale in the Appalachian basin. Computer and Geosciences 49 (2012), 151–163.

Wong, P., Taggart, I., Gedeon, T., 1995. The use of fuzzy ARTMAP for lithofacies classifications: a comparison study. In: Proceedings of SPWLA 36th Annual Logging Symposium.