Neurocomputing 396 (2020) 394–405

Contents lists available at ScienceDirect

Neurocomputing

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

The Random Neural Network with Deep Learning Clusters in Smart Search

Will Serrano*, Erol Gelenbe, Yonghua Yin

Intelligent Systems and Networks Group, Electrical and Electronic Engineering, Imperial College London, United Kingdom

ARTICLE INFO

Article history:
Received 20 November 2017
Revised 6 February 2018
Accepted 2 May 2018
Available online 24 April 2019

Keywords:
Intelligent search
World wide web
Random Neural Network
Clusters
Deep learning
Management clusters

ABSTRACT

This paper proposes a Neurocomputing application that reorders the Web results obtained from different Web Search Engines emulating the way our brain takes decisions. The proposed application is based on the Random Neural Network with Deep Learning Clusters that evaluates and adapts Web result relevance by associating independently each Deep Learning Cluster to a specific Web Search Engine. In addition, this paper presents a Deep Learning Cluster to perform as a Management Cluster that decides the final result relevance based on the inputs from each independent Deep Learning cluster. The performance of the proposed Management Cluster is evaluated when included as an additional layer to the Deep Learning Clusters. On average; the proposed Deep Learning cluster structure improves Smart Search performance.
© 2019 Elsevier B.V. All rights reserved.

1. Introduction

Our brain is formed of dense local clusters of the same neurons performing different functions which are connected between each other with numerous very short paths and few long distance connections [1]; the cluster of neurons specialization occurs due to their adaption when learning tasks. Our brain decides the different actions to be taken based on a weighted decision from the different sensorial inputs. A parallel scenario is the information available on the Internet; the Web is formed of a large amount of data where users select relevant material obtained from different sources such as Web Search Engines or Recommender Systems.

The brain retrieves a large amount of data obtained from the senses; analyses the material and finally selects the relevant information [2]. This decision can be erroneous due different external factors such as light flashes or background noise; likewise, a user needs to select relevant Web results from a search outcome that may be influenced or manipulated by a commercial interest as well as by the users' own ambiguity in formulating their requests or queries [3–6]. This paper proposes to associate the most complex biological system; our brain with the most complex artificial system represented in the Web. The connection between them is the Random Neural Network [7–9].

1.1. Related work

Deep learning applies a neural network with various computing layers that perform several linear and nonlinear transformations to model general concepts in data. Deep learning is characterized as using a cascade of l-layers of nonlinear computing modules for attribute identification and conversion; each input of every sequential layer is based on the output from the preceding layer.

Deep learning models have been used in learning to rank brief text pairs from which the main components are phrases [10]; the method is built using a convolutional neural network structure where the best characterization of text pair sets and a similarity function is learned with a supervised algorithm. The input is a sentence matrix with a convolutional feature map layer that extracts patterns, a pooling layer is then added to aggregate the different features and reduce the representation. An attention based deep learning neural network [11] focuses on different aspects of the input data to include distinct features; the method incorporates different word orders with variable weights changing over the time for both the queries and the search results where a multi-layered neural network ranks the results and provides a listwise set of results using a decoder mechanism. Deep Stacking Networks [12] are used for information retrieval with parallel and scalable learning; the design philosophy is based on the design of basic modules of classifiers and it later combination between them to learn complex functions; the output of each Deep Stacking Network is linear whereas the hidden unit's output is sigmoidal nonlinear.

* Corresponding author.
E-mail addresses: g.serrano11@imperial.ac.uk (W. Serrano), e.gelenbe@imperial.ac.uk (E. Gelenbe), y.yin14@imperial.ac.uk (Y. Yin).

https://doi.org/10.1016/j.neucom.2018.05.134
0925-2312/© 2019 Elsevier B.V. All rights reserved.

Deep learning is also used in Recommender Systems. A deep feature representation [13] learns the content information and captures the likeness and implicit association between customers and items where Collaborative Filtering is used in a Bayesian probabilistic framework for the rating matrix. A Deep learning approach [14] assigns items and customers to a vector space model in which the similarity between customers and their preferred products is optimized; the model is extended to jointly learn new features of items from different domains and user features according to their Web browsing history and search queries. The deep learning neural network maps two different high dimensional sparse features into low dimensional dense features within a joint semantic space.

This paper is an extension of a prior model presented by Serrano and Gelenbe [3], [4]. The Intelligent Search Assistant is based on the Random Neural Network (RNN) [7–9]. This is a spiking recurrent stochastic model for neural networks. Its main analytical properties are the “product form” and the existence of the unique network steady state solution. The RNN represents more closely how signals are transmitted in many biological neural networks where they actual travel as spikes or impulses, rather than as analogue signal levels. It has been used in different applications including network routing with cognitive packet networks, using reinforcement learning, which requires the search for paths that meet certain pre-specified quality of service requirements [15], search for exit routes for evacuees in emergency situations [16,17], pattern based search for specific objects [18], video compression [19], and image texture learning and generation [20].

Deep Learning with Random Neural Networks is described by Gelenbe, E. and Yin, Y. [21–24]. This model is based on the generalized queuing networks with triggered customer movement (G-networks) where customers are either “positive” or “negative” and customers can be moved from queues or leave the network. G-Networks are introduced by Gelenbe, E. [25,26]; an extension to this model is developed by Gelenbe, E. et al [27] where synchronised interactions of two queues could add a customer in a third queue. Kasun et al. [35] introduces Extreme Learning Machine based Auto Encoder (ELM-AE), which learns feature representations using singular values and is used as the basic building block for Multi-Layer Extreme Learning Machine (ML-ELM).

Advanced learning techniques applied to the cloud are described in several publications; Wang and Gelenbe [28] present different experiments that compare three on-line real time techniques for task allocation to different cloud servers; they [29] also propose an experimental system that can exploit a variety of online QoS aware adaptive task allocation schemes. Brun, O. et al. [30] address the use of Big Data and machine learning based analytics to the real-time management of Internet scale Quality of Service route optimization with the help of an overlay network. Different search models were also proposed; Gelenbe [31] propose a Search in unknown random environments, [32] describe a search in the Universe of big networks and data and [33] present time and energy in team-based search. Schmidhuber [34] reviews Deep Learning in neural networks.

1.2. Summary of contributions

This paper utilizes the Intelligent Internet Search Assistant (ISA) that acts as an interface between an individual user’s query and the different search engines presented by the authors; Serrano and Gelenbe [3], [4]. ISA acquires a query from the user and retrieves results from various Web search engines. The result relevance is calculated by applying an innovative cost function based on the division of a query into a multidimensional vector weighting its dimension terms with different relevance parameters [3]. ISA [3] adapts and learns the perceived user’s interest and reorders the retrieved snippets based in the dimension relevant centre point. ISA [3,4] learns result relevance on an iterative process where the user evaluates directly the listed results as a supervised learning.

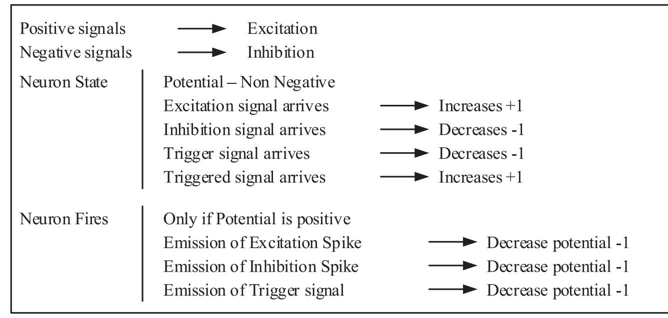


Fig. 1. Neurons.

This paper proposes the assignment of one Deep Learning Cluster per each Web Search Engine and the definition of an additional Deep Learning cluster to perform as a Management Cluster to emulate the way the brain takes decisions. The proposed Deep Learning cluster structure has evaluated ISA against other Web search engines with open user queries. Deep Learning Clusters’ Gradient Descent learning algorithm has been analyzed based on result relevance and learning speed where our proposed method assigns the best performing cluster to the other Web Search Engines to increase system speed, improve result accuracy and relevance while reducing learning time and network weigh’s computation. The Management Cluster unsupervised learning is analyzed and its performance is compared against other search engines with a new proposed quality definition, which combines both relevance and rank.

The cluster Random Neural Network with Deep Learning Clusters is described in Section 2. The ISA with Deep Learning clusters model is defined in Section 3. The application is implemented and validated in Section 4 and Section 5 respectively showing the experimental results in Section 6. Finally, conclusions are presented in Section 7 followed by the References and Appendix where the queries used by the validators are enumerated.

2. The Deep Learning Cluster Random Neural Network

2.1. The Random Neural Network

The Random Neural Network [7–9] is composed of M neurons each of which receives excitatory (positive) and inhibitory (negative) spike signals from external sources which may be sensory sources or neurons. These spike signals occur following independent Poisson processes of rates $\lambda^+(m)$ for the excitatory spike signal and $\lambda^-(m)$ for the inhibitory spike signal respectively, to neuron m ∈ {1,…M}.

2.2. Neurons

In the Deep Learning Cluster model [21,22], each neuron is represented at time $t \geq 0$ by its internal state $k_m(t)$ which is a non-negative integer. If $k_m(t) \geq 0$, then the arrival of a negative spike to neuron m at time t results in the reduction of the internal state by one unit: $k_m(t^+) = k_m(t) - 1$. The arrival of a negative spike to a neuron has no effect if $k_m(t) = 0$. On the other hand, the arrival of an excitatory spike always increases the neuron’s internal state by 1; $k_m(t^+) = k_m(t) + 1$.

If $k_m(t) > 0$, then the neuron m is defined as “excited”, and it may “fire” a spike with probability $r_m \Delta t$ in the interval $[t, t+\Delta t]$

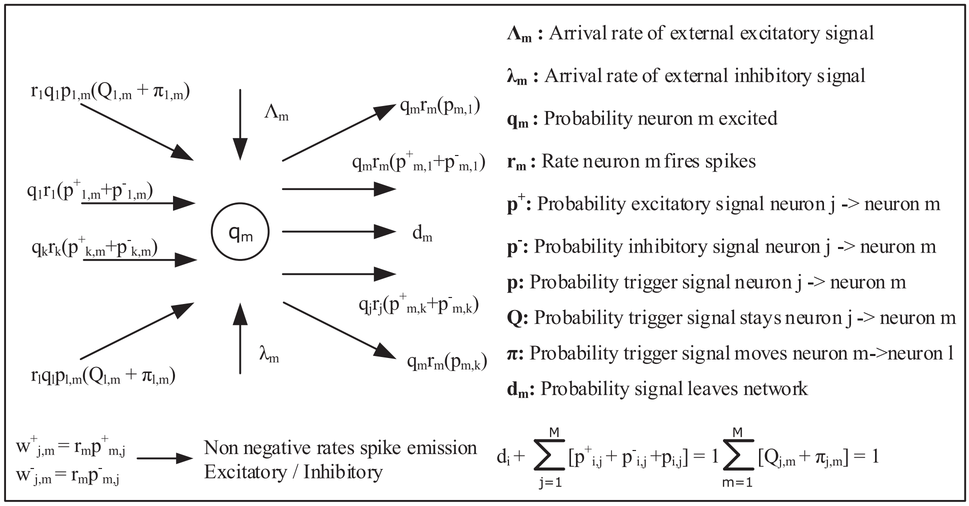


Fig. 2. Neuron interactions.

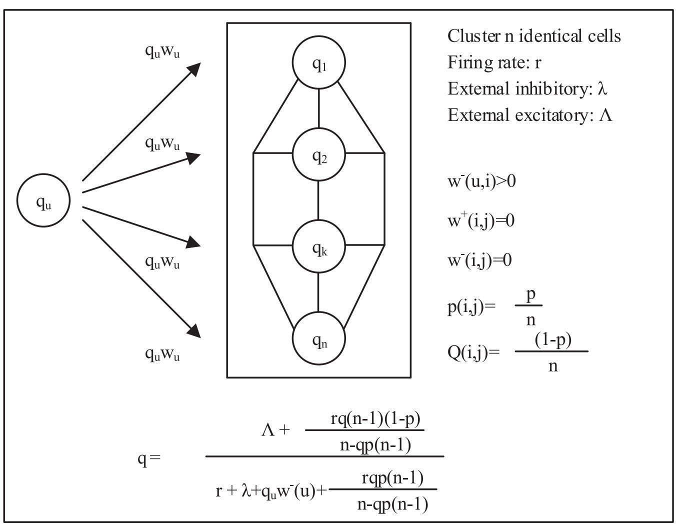


Fig. 3. Clusters of neurons.

where $r_m > 0$ is its "firing rate", so that r_m is the average firing delay of the exited m neuron.

Neurons in this model (Fig. 1) can interact in the following manner at time $t \geq 0$. If neuron i is excited ($k_i(t) > 0$) then when neuron i fires its internal state drops by 1 ($k_m(t+) = k_m(t) - 1$) and:

- It can send a positive or excitatory spike to neuron j with probability $p^+(i,j)$ resulting in $k_j(t^+) = k_j(t) + 1$;
- Or it can send a negative or inhibitory spike to neuron j with probability $p^-(i,j)$ resulting in $k_j(t^+) = k_j(t) - 1$ if $k_j(t) > 0$, else $k_j(t+) = 0$ if $k_j(t) = 0$;
- Or it can trigger neuron j with probability $p(i,j)$ resulting in $k_j(t^+) = k_j(t) - 1$ if $k_j(t) > 0$, else $k_j(t+) = 0$ if $k_j(t) = 0$ and one of two may happen. Either:
  - (A) with probability $Q(j,m)$ there is $k_m(t^+) = k_m(t) + 1$;
  - (B) or with probability $\pi(j,m)$ the trigger moves on to the neuron m and then with probability $Q(m,l)$ the sequence (A) or (B) is repeated.

2.3. Neuron interactions

The neuron interaction [21,22] defines $z(m) = (i_1, \ldots\ i_l)$ as any ordered sequence of distinct numbers $ij \in S$; $ij \neq m$; and

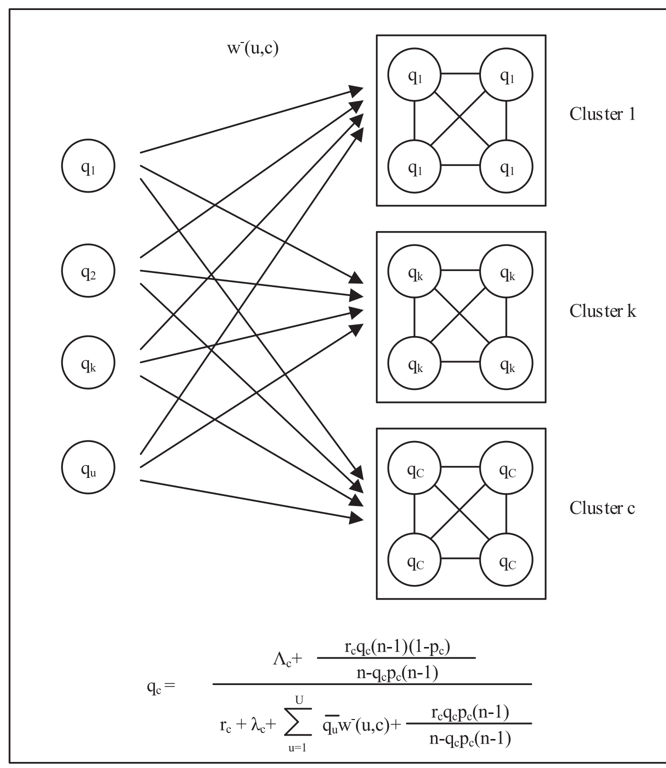


Fig. 4. Multiple clusters.

$1 \leq l \leq M-1$. It defines $q_m = \lim_{t\to\infty} \mathrm{Prob}[k_m(t) > 0]$ the probability that the neuron m is excited. It is given by the following expression:

$$q_m = \frac{\lambda^+(m)}{r(m) + \lambda^-(m)} \tag{1}$$

The model [21,22] assigns $w^+(j,i) = r(i)p^+(j,i)$ and $w^-(j,i) = r(i)p^-(j,i) \geq 0$ respectively. $\Lambda(m)$ and $\lambda(m)$ represents the arrival rates of external excitatory and inhibitory signals correspondingly (Fig. 2).

2.4. Clusters of neurons

The clusters of neurons [21,22] consider a special network $M(n)$ that contains n identically connected neurons, each which has a

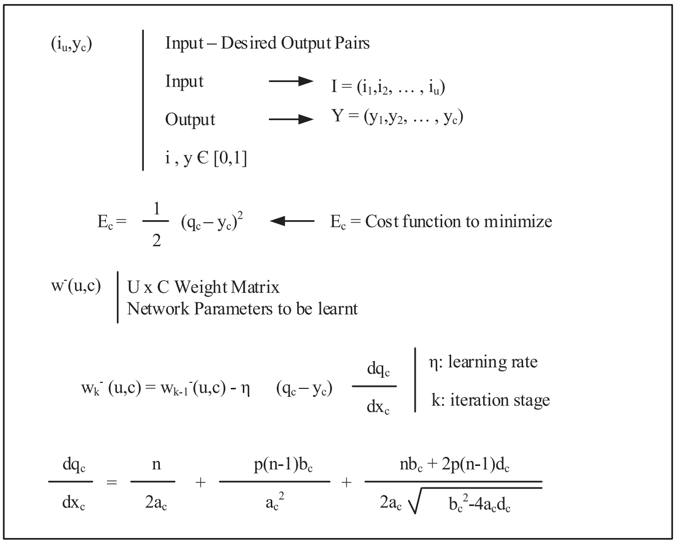


Fig. 5. Deep Learning Clusters.

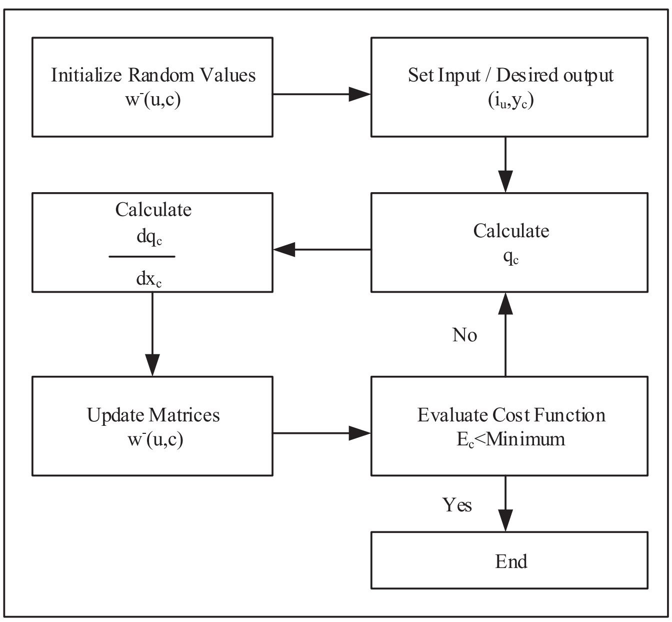


Fig. 6. Deep Learning Clusters: gradient descent iteration.

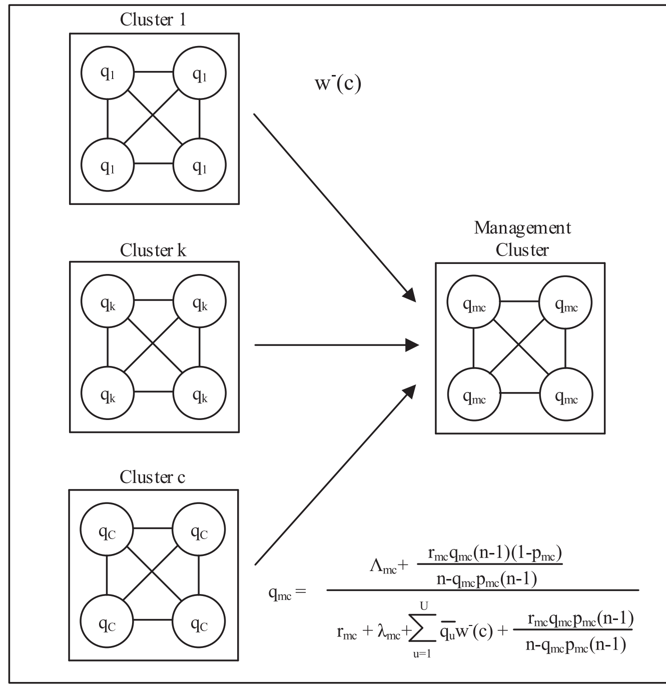


Fig. 7. Management cluster.

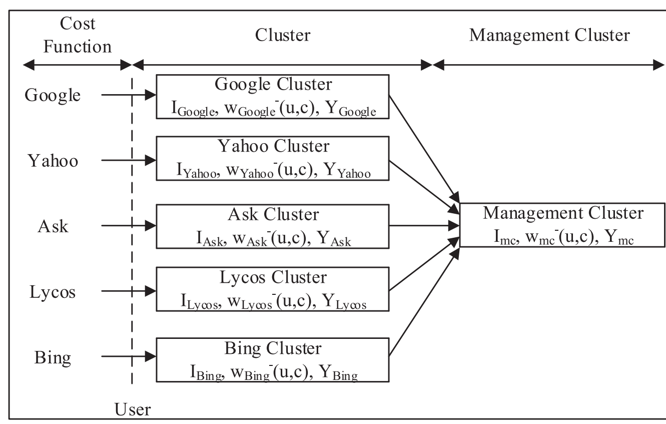


Fig. 8. ISA with Deep Learning Clusters model.

firing rate r and external inhibitory and excitatory signals Λ and λ, respectively. The state of each neuron is denoted by q, and it receives an inhibitory input from the state of some neuron u which does not belong to $M(n)$. Thus for any neuron i Є $M(n)$ there is an inhibitory weight $w^-(u) \equiv w^-(u,i) > 0$ from u to i.

For any i,j Є $M(n)$; the values of $w^+(i,j)=w^-(i,j)=0$, but all whenever one of the neurons fires (Fig. 3), it triggers the firing of the other neurons with the following values:

$$p(i,\ j) = \frac{p}{n};\ Q(i,\ j) = \frac{1-p}{n}$$

The probability a neuron becomes excited reduces to:

$$q = \frac{\Lambda + \dfrac{rq(n-1)(1-p)}{n-qp(n-1)}}{r+\lambda+q_u w^-(u)+\dfrac{rqp(n-1)}{n-qp(n-1)}} \tag{2}$$

2.5. Multiple clusters

The Deep Learning Architecture [21,22] is composed of C clusters $M(n)$ each with n hidden neurons. For the cth such cluster, $c=1, \ldots, C$, the state of each of its identical neurons is denoted by q_c. In addition, there are U input neurons which do not belong to these C clusters, and the state of the uth neuron $u=1, \ldots, U$ is denoted by $\overline{q_u}$. The cluster network has U input neurons and C clusters (Fig. 4).

Each hidden neuron in the clusters c, with c Є {1, …, C} receives an inhibitory input from each of the U input neuron. Therefore, for each neuron in the cth cluster, there are inhibitory weights $w^-(u,c) > 0$ from the uth input neuron to each neuron in the cth cluster; the uth input neuron will have a total inhibitory "exit" weight, or total inhibitory firing rate $\overline{r_u}$ to all the clusters which is of value:

$$\overline{r_u} = n\sum_{c=1}^{C} w^-(u,\ c) \tag{3}$$

Table 1
ISA cluster validation.

Query	Google	Yahoo	Ask	Bing	Lycos
1 WSE	0.5789	0.4737	0.7368	0.4737	0.7895
1 DLC	0.3158	0.7895	0.4737	0.6842	0.7895
2 WSE	0.7368	0.3684	0.7895	0.2941	0.3684
2 DLC	0.6316	0.7895	0.7368	1.0000	0.5789
3 WSE	0.5263	0.9474	0.6842	0.7059	0.9474
3 DLC	0.7368	0.7368	0.7895	0.8824	0.5789
4 WSE	0.8421	0.8947	0.8421	0.8000	0.5263
4 DLC	0.4211	0.3684	0.7895	0.2000	0.5789
5 WSE	1.0000	0.5789	0.8947	0.9333	0.8947
5 DLC	0.6316	0.1579	0.5263	0.4000	0.5789
6 WSE	0.4737	0.6842	0.8947	0.8421	0.6842
6 DLC	0.7368	0.8947	0.8421	0.8947	1.0000
7 WSE	1.0000	0.7368	0.5263	0.7368	0.7368
7 DLC	0.9474	0.8947	0.9474	0.8947	0.8421
8 WSE	0.7368	0.5263	0.8421	0.6842	0.6316
8 DLC	0.3684	0.6316	0.8947	0.4737	0.6316
9 WSE	0.8421	0.8421	0.7895	0.8462	0.8421
9 DLC	1.0000	0.6316	1.0000	0.5385	0.6842
10 WSE	1.0000	0.5789	0.9474	0.5789	0.8947
10 DLC	0.5294	0.8947	0.8421	0.8421	0.8947
11 WSE	0.8947	0.5789	0.4737	0.6316	0.4211
11 DLC	0.2105	0.5789	0.3158	0.6842	1.0000
12 WSE	0.6842	0.7368	0.8421	0.6471	0.4737
12 DLC	0.8947	0.7895	0.6842	0.5294	1.0000
13 WSE	0.6842	0.2632	0.3684	0.7692	0.6842
13 DLC	0.5789	0.7368	0.8421	0.2308	0.6842
14 WSE	0.8421	1.0000	0.7895	0.7895	0.2632
14 DLC	0.7368	0.3684	0.5789	0.7368	0.5789
15 WSE	0.5882	0.2632	0.6842	0.7368	0.2632
15 DLC	0.3529	0.2105	0.4737	0.3158	0.5789
16 WSE	0.7692	0.7895	0.6842	0.5789	0.6842
16 DLC	0.5385	0.5789	0.5263	0.9474	0.5263
17 WSE	0.4737	0.6316	0.5789	0.5789	0.6316
17 DLC	0.5263	0.3158	0.3684	0.6842	0.7368
18 WSE	0.3529	0.4737	0.4211	0.3333	0.5263
18 DLC	0.4118	0.4737	0.4737	0.2667	0.2632
19 WSE	0.5294	0.3684	0.3158	0.2308	0.1579
19 DLC	0.4118	0.7368	0.3684	0.8462	0.7368
20 WSE	0.7895	0.1579	0.6842	0.6154	0.2632
20 DLC	0.4737	0.5789	0.7368	0.6923	0.3158
21 WSE	0.5385	0.5263	0.3684	0.6316	0.5789
21 DLC	0.5385	0.8421	0.4737	0.5263	0.7895
22 WSE	0.7368	0.4737	0.4211	0.8421	0.4737
22 DLC	0.3158	0.6842	0.4211	0.4211	0.7368
23 WSE	0.5789	0.7895	0.8947	0.6923	0.8947
23 DLC	0.3684	0.3158	0.7895	0.4615	0.7895
24 WSE	0.2941	0.2105	0.6842	0.7059	0.5789
24 DLC	0.5294	0.4211	0.4737	0.5882	0.8947
25 WSE	0.3158	0.8947	0.4211	0.5789	0.6316
25 DLC	0.2632	0.5263	0.1579	0.4737	0.3684
26 WSE	1.0000	0.2632	0.3684	0.9474	0.6316
26 DLC	0.5263	0.9474	0.6316	0.6316	0.6316
27 WSE	0.5263	0.7368	0.3158	0.5333	0.8947
27 DLC	0.7368	0.4737	0.3158	0.6667	0.7895
28 WSE	0.7895	0.7368	0.6842	0.6154	0.9474
28 DLC	0.7368	0.7895	1.0000	0.4615	0.7368
29 WSE	0.8421	0.6316	0.8947	0.6316	0.4211
29 DLC	0.6316	0.6316	0.3158	0.5263	0.3158
30 WSE	0.6842	0.7895	0.4211	0.4667	0.7368
30 DLC	0.5263	0.8947	0.9474	0.6667	0.8947
31 WSE	0.3158	0.7368	0.3684	0.6316	0.7895
31 DLC	0.7368	0.5263	0.7368	0.5263	0.5263
32 WSE	0.8947	0.3684	1.0000	0.7895	1.0000
32 DLC	0.9474	0.3158	0.8947	0.7368	0.8421
33 WSE	0.8947	0.8421	0.6842	0.5455	0.4211
33 DLC	0.8421	0.7368	0.8947	0.6364	0.9474
34 WSE	0.7368	0.8421	0.7895	0.7895	0.7895
34 DLC	0.6842	0.7895	0.8947	0.5789	0.5789
35 WSE	0.4737	0.9474	0.7368	0.5263	0.5789
35 DLC	0.6316	0.7895	0.5789	0.5789	0.6316
36 WSE	0.5882	0.6842	0.4737	0.6316	0.8421
36 DLC	0.8824	0.3684	0.7368	0.4211	0.7368
37 WSE	0.9474	0.6842	0.8421	0.6842	0.6842
37 DLC	0.6316	0.7368	0.7895	0.6842	0.7368
38 WSE	0.4211	0.6842	0.5263	0.4737	0.6842
38 DLC	0.5263	0.7368	0.4737	0.4737	0.8421
39 WSE	1.0000	0.5789	0.4737	0.7368	0.9474
39 DLC	0.5263	0.5263	0.7895	0.5263	0.5789
40 WSE	0.6316	0.9474	0.6842	1.0000	0.4211
40 DLC	0.8421	0.5263	0.6842	0.6316	0.3684
41 WSE	0.8421	0.6842	0.7895	0.7368	0.8421
41 DLC	0.3158	0.7368	0.7368	0.9474	0.9474
42 WSE	0.4118	0.5789	0.7368	0.4211	0.5263
42 DLC	0.8235	0.7895	0.7368	0.8421	0.6842
43 WSE	0.6316	0.5789	0.6842	0.5789	0.5789
43 DLC	0.5263	0.5789	0.3684	0.6842	0.5789
44 WSE	0.5789	0.3158	0.8947	0.8421	0.7368
44 DLC	0.5789	0.5789	0.8947	0.6316	0.6316
45 WSE	0.7895	0.3158	0.6316	0.8421	0.5789
45 DLC	0.4737	0.3158	0.2632	0.6842	0.7895
	Google	**Yahoo**	**Ask**	**Bing**	**Lycos**
Average					
WSE	0.6846	0.6164	0.6573	0.6596	0.6421
DLC	0.5910	0.6164	0.6491	0.6167	0.6877

(continued on next page)

Table 1 (continued)

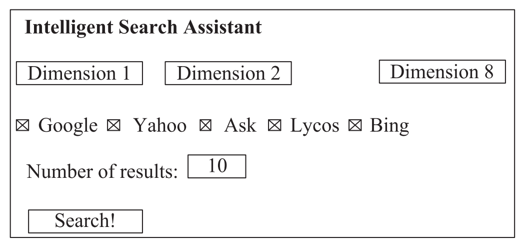


Fig. 9. Intelligent search assistant user interface.

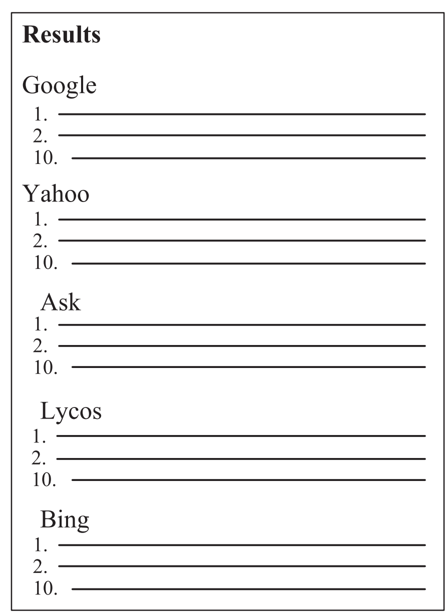


Fig. 10. Intelligent search assistant result list.

then, from (2) and (3); the probability a neuron in cluster c becomes excited is:

$$q_c = \frac{\Lambda_c + \dfrac{r_c q_c (n-1)(1-p_c)}{n - q_c p_c (n-1)}}{r_c + \lambda_c + \sum_{u=1}^{U} \overline{q_u} w^-(u,c) + \dfrac{r_c q_c p_c (n-1)}{n - q_c p_c (n-1)}} \tag{4}$$

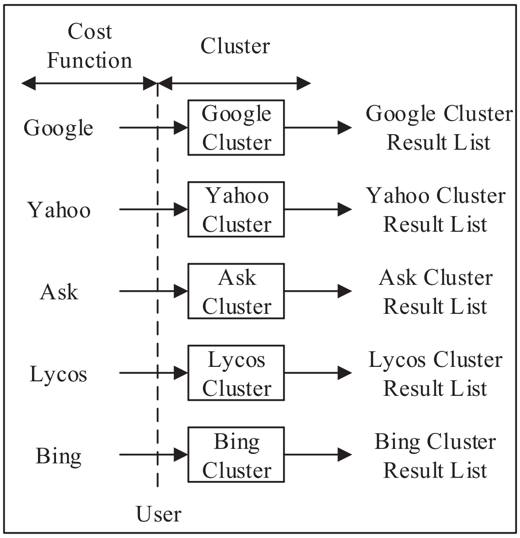


Fig. 11. Deep Learning Cluster validation.

2.6. Deep learning clusters

The learning model of the Deep Learning Clusters [21,22] defines:

- I, a U-dimensional vector $I \in [0,1]^U$ that represents the input state $\overline{q_u}$ for the neuron u;
- $w^-(u,c)$ is the $U \times C$ matrix of weights from the U input neurons to the neurons in each of the C clusters;
- Y, a C-dimensional vector $Y \in [0,1]^C$ that represents the neuron state q_c for the cluster c.

Let us now define the activation function of the cth cluster as:

$$\zeta(\mathrm{x_c}) = \frac{\mathrm{b_c}}{2\mathrm{a_c}} - \frac{\sqrt{\mathrm{b_c^2} - 4\mathrm{a_c d_c}}}{2\mathrm{a_c}} \tag{5}$$

where:

$$x_c = \sum_{u=1}^{U} \overline{q_u} w^-(u, c) \tag{6}$$

and:

$$\mathrm{y_c} = \zeta(\mathrm{x_c})$$

Gradient Descent learning algorithm optimizes the network weight parameters $w^-(u,c)$ from a set of input-output pairs (i_u, y_c):

- the input vector $I = (i_1, i_2, \ldots, i_u)$ where i_u is the input state $\overline{q_u}$ for neuron u;
- the output vector $Y = (y_1, y_2, \ldots, y_c)$ where y_c is the neuron state q_c for the cluster c.

The desired output vector is approximated by minimizing the cost function E_c:

$$E_c = \frac{1}{2}(q_c - y_c)^2 \tag{7}$$

$$\zeta(x_{mc}) = \frac{[np(\Lambda_{mc} + r_{mc}) + n(\lambda_{mc} + x_{mc}) - p(\Lambda_{mc} + r_{mc}) + r_{mc}]}{2p_{mc}(n-1)[\lambda_{mc} + x_{mc}]} - \frac{\sqrt{[np(\Lambda_{mc} + r_{mc}) + n(\lambda_{mc} + x_{mc}) - p(\Lambda_{mc} + r_{mc}) + r_{mc}]^2 - 4p_{mc}(n-1)[\Lambda_{mc} + x_{mc}]n\Lambda_{mc}}}{2p_{mc}(n-)[\lambda_{mc} + x_{mc}]} \tag{10}$$

The network learns the $U \times C$ weight matrix $w^-(u,c)$ by calculating new values of the network parameters for the input X and output Y using Gradient Descent (Fig. 5). The rule for weight update can take the generic form:

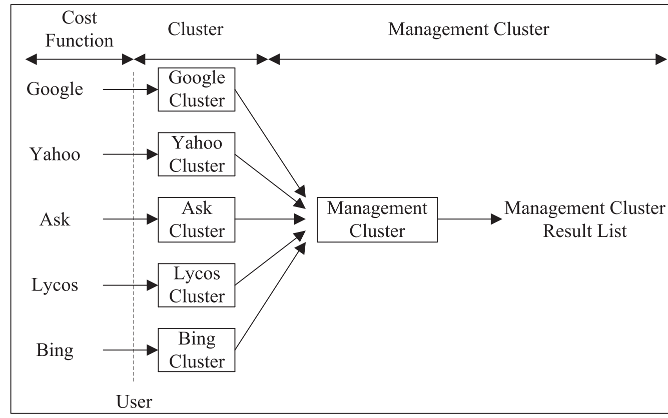


Fig. 12. Management Cluster validation.

$$\mathrm{w_k^-(u,\ c)} = \mathrm{w_{k-1}^-(u,\ c)} - \eta(\mathrm{q_c} - \mathrm{y_c})\frac{\mathrm{dq_c}}{\mathrm{dx_c}} \tag{8}$$

where η is the learning rate and k the iteration number. The derivative is calculated as:

$$\frac{\mathrm{dq_c}}{\mathrm{dx_c}} = \frac{\mathrm{n}}{2\mathrm{a_c}} + \frac{\mathrm{p(n-1)b_c}}{\mathrm{a_c^2}} + \frac{\mathrm{nb_c} + 2\mathrm{p(n-1)d_c}}{2\mathrm{a_c}\sqrt{\mathrm{b_c^2} - 4\mathrm{a_c d_c}}} \tag{9}$$

The complete learning algorithm can be specified. The weight matrix $w^-(u,c)$ with a random initialization needs first to be appropriately initiated. A value for η needs to be selected:

1. Set the input values to $I = (i_1, i_2, \ldots, i_u)$
2. Calculate q_c
3. Calculate derivative $\mathrm{d}q_c/\mathrm{d}x_c$
4. Update the weight matrices $w^-(u,c)$ following the Eq. (8) using the results of Eq. (5) and Eq. (9)
5. Evaluate the cost function E_c according to Eq. (7) using the results of Eq. (8)

This learning algorithm iterated until the value of the cost function from the network weight matrices is smaller than some predetermined value (Fig. 6).

2.7. Management cluster

This paper proposes the Management cluster model in this section:

- I_{mc}, a C-dimensional vector $I_{mc} \in [0,1]^C$ that represents the input state $\overline{q_c}$ for the cluster c;
- $w^-(c)$ is the C-dimensional vector of weights from the C input clusters to the neurons in the Management Cluster mc;
- Y_{mc}, a scalar $Y_{mc} \in [0,1]$, the neuron state q_{mc} for the Management Cluster mc.

Let us now define the activation function of the management cluster mc as:

where:

$$x_{mc} = \sum_{c=1}^{C} \overline{q_c} w^-(c) \tag{11}$$

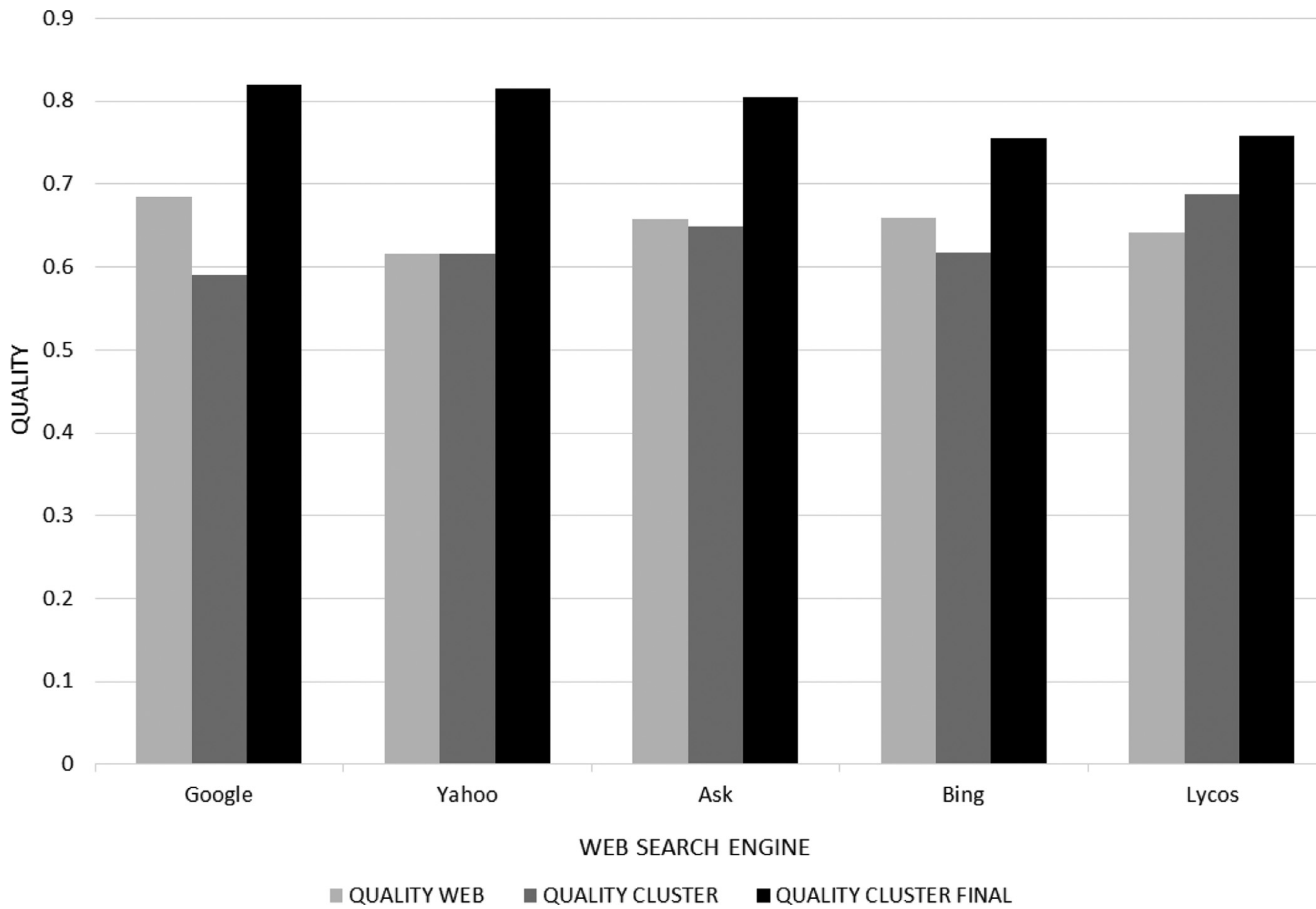


Fig. 13. Deep Learning Cluster validation – average results.

and:

$$y_{mc} = \zeta(x_{mc})$$

The input state $\overline{q_c}$ for neuron c represents the result relevance from each learning cluster; $w^-(c)$ is the C-dimensional vector of weights that represents the learning quality of each learning cluster c; y_{mc} is the final result relevance assigned by the Management Cluster (Fig. 7).

3. ISA with Deep Learning Clusters model

Our ISA associates a Deep Learning Cluster to each Web Search Engine (Fig. 8). Each Deep Learning Clusters learns its assigned Web Search Engine Relevant Centre Point [3] where the outputs y_c of the Deep Learning Clusters are the identical values as the inputs i_u.

Let's define or ISA with Deep Learning clusters as:

- I_{Google}, I_{Yahoo}, I_{Ask}, I_{Lycos}, I_{Bing}, as five U-dimensional vector I € $[0,1]^U$; one for each Web Search Engine that represents the each different Deep Learning Cluster Relevant Centre Point;
- $w_{Google}{}^-(u,c)$, $w_{Yahoo}{}^-(u,c)$, $w_{Ask}{}^-(u,c)$, $w_{Lycos}{}^-(u,c)$, $w_{Bing}{}^-(u,c)$ as five $U\times C$ matrices of Web Search Engine weights; one for each different Deep Learning;
- Y_{Google}, Y_{Yahoo}, Y_{Ask}, Y_{Lycos}, Y_{Bing}, as five C-dimensional vector Y € $[0,1]^C$ one for each Web Search Engine that represents the each different Deep Learning Cluster Relevant Centre Point.

The Deep Learning Management Cluster is defined as:

- I_{mc}, a C-dimensional vector I_{mc} € [0,1] that represents the result relevance for the Web Search Deep Learning cluster c;
- $w_{mc}{}^-(c)$ is the C-dimensional vector of weights that represents the learning quality of each Web Search Deep Learning cluster c;
- Y_{mc}, a scalar Y_{mc} € [0,1] that represents the final result relevance.

Table 2
ISA cluster validation – average values.

Web Search Engine	Quality Web Search Engine	Quality Cluster	I	Quality Cluster Final	I
Google	0.6846	0.5910	−13.67%	0.820	38.75%
Yahoo	0.6164	0.6164	0.0%	0.815	32.26%
Ask	0.6573	0.6491	−1.25%	0.805	23.96%
Bing	0.6596	0.6167	−6.50%	0.756	22.95%
Lycos	0.6421	0.6877	7.10%	0.7587	9.86%
Average	0.6520	0.6322	−2.86%	0.79094	25.56%

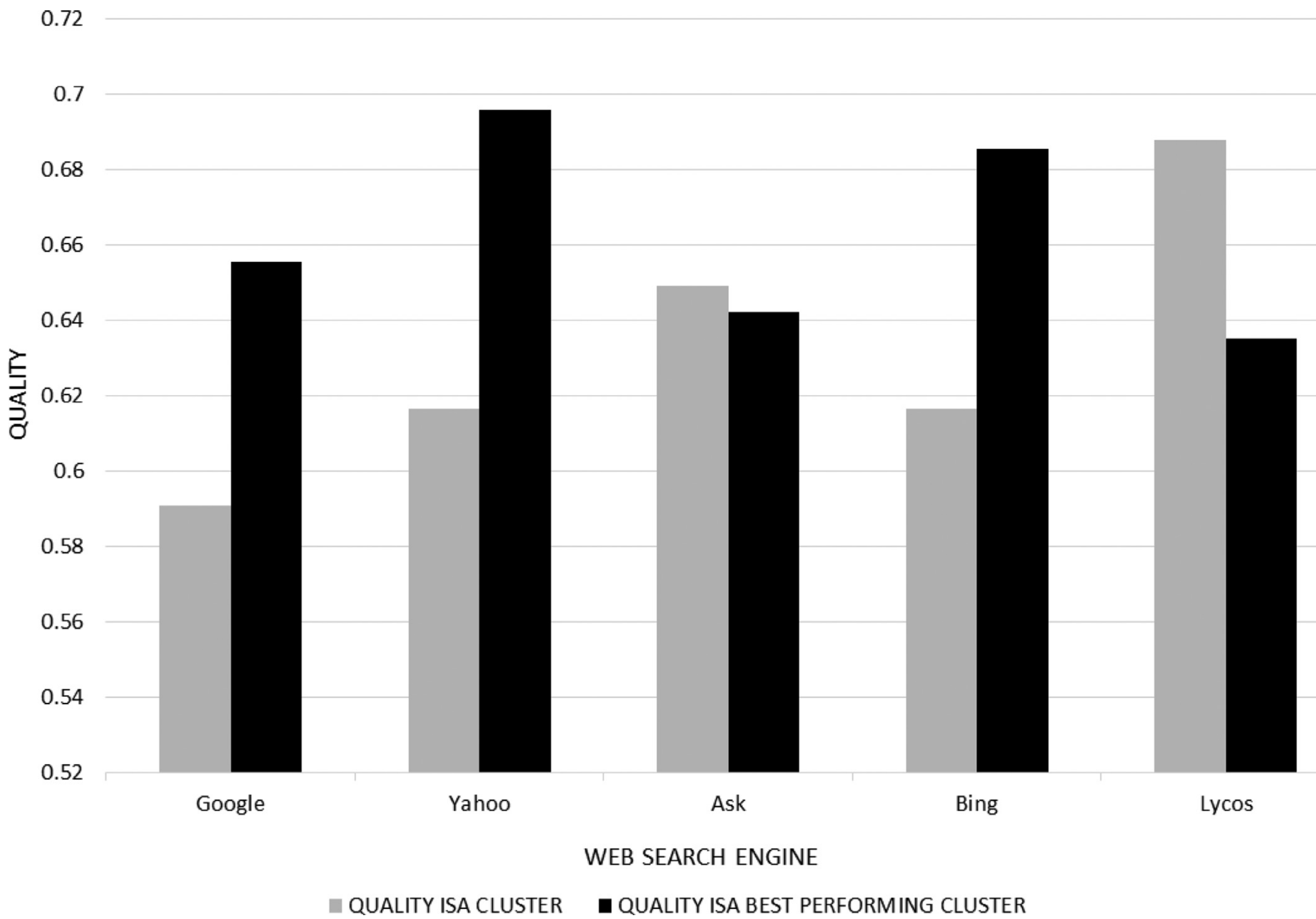


Fig. 14. Best Performing Cluster validation – average results.

A Web user in the validation transmits a search request, then our ISA associates a Deep Learning cluster to each Web Search Engine, retrieves the top N results of each Web Search Engine and rearrange them using our defined cost function independently for each Deep Learning cluster. The cost function is defined in [3]. Our ISA presents a reordered result list grouped by Web Search Engine. The user selects relevant results for each list.

Each learning cluster learns each Web Search Engine Relevant Centre Point as defined in [3] where the outputs y_c are the same values as the inputs i_u. ISA reorders each Web Search Engine cluster result list following to the minimum error to the cluster Relevant Centre Point (I_{Google}, I_{Yahoo}, I_{Ask}, I_{Lycos}, I_{Bing}).Once our ISA has assessed the best performing cluster; it applies its neural network weights to rearrange the other cluster's result lists.

4. Implementation

The Intelligent Search Assistant emulates how Web search engines work by using a very similar interface to introduce queries and display results (Fig. 9). The ISA acquires up to eight different dimensions values from the user however Web search engine and number of result options are fixed.

Our ISA has been codified to parse snippets from five Web search engines (Google, Yahoo, Ask, Lycos, Bing). The search is direct; our ISA obtains the query key words from the Web user and transmit it to the different Web search engines selected without modifying it (Fig. 10). The ISA shows to the user a rearranged list of results grouped by Web Search Engine.

Table 3
Best performing cluster validation – average values.

Web Search Engine	Quality ISA Cluster	Quality ISA Best Preforming Cluster	I
Google	0.5910	0.6554	10.88%
Yahoo	0.6164	0.6959	12.9%
Ask	0.6491	0.6421	−1.08%
Bing	0.6167	0.6854	11.14%
Lycos	0.6877	0.6351	−7.65%
Average	0.6322	0.6628	5.24%

5. Validation

A user in the experiment introduces a query. ISA assigns a learning cluster per Web Search Engine, acquires the first N results of each Web Search Engine and reorders them applying the cost function independently for each cluster. Finally ISA shows a reordered list clustered by Web Search Engine. In the validation, users have been asked to select Y relevant results per Web Search Engine, not to rank them, as they normally do using a Web search engine therefore it is considered a result is either relevant or irrelevant. The results are shown to the user on random order to avoid a biased result rank evaluation where the users indirectly follow the order shown by the selected algorithm.

Table 4
Management Cluster validation.

Query	Cost Function Q	Cluster Q	$\gamma = 1/4$	$\gamma = 1/2$	$\gamma = 1.0$	$\gamma = 2.0$	$\gamma = 4.0$
1	0.2063	0.2306	0.2298	0.2298	0.2306	0.2282	0.2259
2	0.4008	0.5584	0.5420	0.5412	0.5461	0.5437	0.5486
3	0.2351	0.3502	0.3380	0.3380	0.3380	0.3371	0.3355
4	0.2483	0.2942	0.2891	0.2891	0.2925	0.2934	0.2934
5	0.2423	0.3537	0.3435	0.3444	0.3342	0.3257	0.2781
6	0.2518	0.2878	0.2714	0.2729	0.2737	0.2729	0.2816
7	0.2620	0.3161	0.3122	0.3129	0.3129	0.3122	0.3114
8	0.2227	0.2706	0.2753	0.2753	0.2784	0.2769	0.2722
9	0.2518	0.2686	0.2677	0.2624	0.2535	0.2074	0.1711
10	0.2522	0.3102	0.2963	0.2939	0.2931	0.2816	0.2514
11	0.2769	0.2055	0.2063	0.2063	0.2063	0.2125	0.2345
12	0.2808	0.3224	0.2882	0.2857	0.2824	0.2588	0.1910
13	0.2252	0.2296	0.2225	0.2216	0.2225	0.2199	0.2207
14	0.2078	0.2973	0.2996	0.3004	0.2996	0.3027	0.3012
15	0.1682	0.2000	0.2139	0.2147	0.2171	0.2171	0.2180
16	0.2154	0.2420	0.2473	0.2473	0.2473	0.2456	0.2420
17	0.1976	0.1765	0.1906	0.1890	0.1812	0.1718	0.1686
18	0.1729	0.2934	0.2996	0.2996	0.2996	0.2996	0.3005
19	0.2044	0.1896	0.2313	0.2285	0.2202	0.2118	0.2054
20	0.3316	0.4379	0.4441	0.4441	0.4424	0.4441	0.4362
21	0.2101	0.2730	0.2890	0.2881	0.2855	0.2801	0.2757
22	0.2094	0.2518	0.2588	0.2471	0.2392	0.2220	0.1639
23	0.2128	0.2385	0.2367	0.2385	0.2367	0.2367	0.2305
24	0.3265	0.2236	0.2270	0.2253	0.2245	0.2168	0.2151
25	0.1624	0.2024	0.2188	0.2118	0.2055	0.1898	0.2025
26	0.1796	0.3396	0.3396	0.3396	0.3388	0.3380	0.3380
27	0.2109	0.2151	0.2287	0.2287	0.2321	0.2347	0.2398
28	0.3023	0.3174	0.2970	0.2988	0.3014	0.3050	0.3129
29	0.1780	0.2118	0.2204	0.2173	0.2157	0.2173	0.2180
30	0.3155	0.3418	0.3427	0.3418	0.3401	0.3350	0.3019
31	0.2047	0.1475	0.1529	0.1529	0.1624	0.1655	0.1671
32	0.2345	0.2533	0.2549	0.2557	0.2573	0.2620	0.2690
33	0.2720	0.3090	0.2979	0.2988	0.2997	0.3025	0.3090
34	0.2580	0.3224	0.3184	0.3137	0.3114	0.3114	0.2996
35	0.2078	0.2831	0.2761	0.2761	0.2761	0.2776	0.2839
36	0.2098	0.2441	0.2531	0.2539	0.2506	0.2490	0.2457
37	0.3027	0.2180	0.2588	0.2588	0.2604	0.2604	0.2627
38	0.2078	0.2259	0.2384	0.2361	0.2314	0.2267	0.1992
39	0.2047	0.2259	0.2588	0.2588	0.2580	0.2596	0.2698
40	0.2094	0.2000	0.2063	0.2047	0.2016	0.1937	0.1953
41	0.2910	0.3569	0.3592	0.3576	0.3553	0.3475	0.3122
42	0.3461	0.3576	0.3543	0.3543	0.3543	0.3543	0.3543
43	0.1945	0.2518	0.2855	0.2831	0.2792	0.2737	0.2604
44	0.2110	0.2729	0.2580	0.2494	0.2337	0.2157	0.2141
45	0.2353	0.3325	0.3114	0.3043	0.2910	0.2675	0.1945
Average	**Cost Function Q**	**Cluster Q**	**$\gamma = 1/4$**	**$\gamma = 1/2$**	**$\gamma = 1.0$**	**$\gamma = 2.0$**	**$\gamma = 4.0$**
Query	0.2389	0.2767	0.2789	0.2776	0.2759	0.2712	0.2627

5.1. Deep Learning Cluster validation

In order to measure search quality we can affirm the better learning cluster provides with a list of more relevant results on top positions (Fig. 11). The following quality description is described where within a list of N results N is scored to the first result and 1 to the last result, the value of the quality proposed is then the summation of the position score based of each of the selected results [3]. Quality, Q, can be defined as:

$$Q = \sum_{i=1}^{Y} RSE_i \tag{12}$$

where RSE_i is the rank of the result i in a particular search engine with a value of N if the result is in the first position and 1 if the result is the last one. Y is the total number of results selected by the user. The best Deep Learning cluster or Web search engine would have the largest Quality value. Normalized quality, $\bar{Q}$, is defined as the division of the quality, Q, by the optimum figure which it is when the user consider relevant all the results provided by the Web search engine. On this situation Y and N have the same value:

$$\bar{Q} = \frac{Q}{\frac{N(N+1)}{2}} \tag{13}$$

Let us define I as the quality improvement between a Deep Learning Cluster and a reference:

$$\mathrm{I} = \frac{\mathrm{QC} - \mathrm{QR}}{\mathrm{QR}} \tag{14}$$

where I is the Improvement, QC is the quality of the Deep Learning Cluster and QR is the quality reference.

5.2. Management cluster validation

The management cluster is validated with different values of $w^{-}(c)$:

$$\mathrm{w}^{-}(\mathrm{c}) = \left(\frac{1}{\mathrm{Q_c}}\right)^{\gamma} \tag{15}$$

where Q_c is Quality of Cluster c and γ the Management Cluster learning coefficient (Fig. 12).

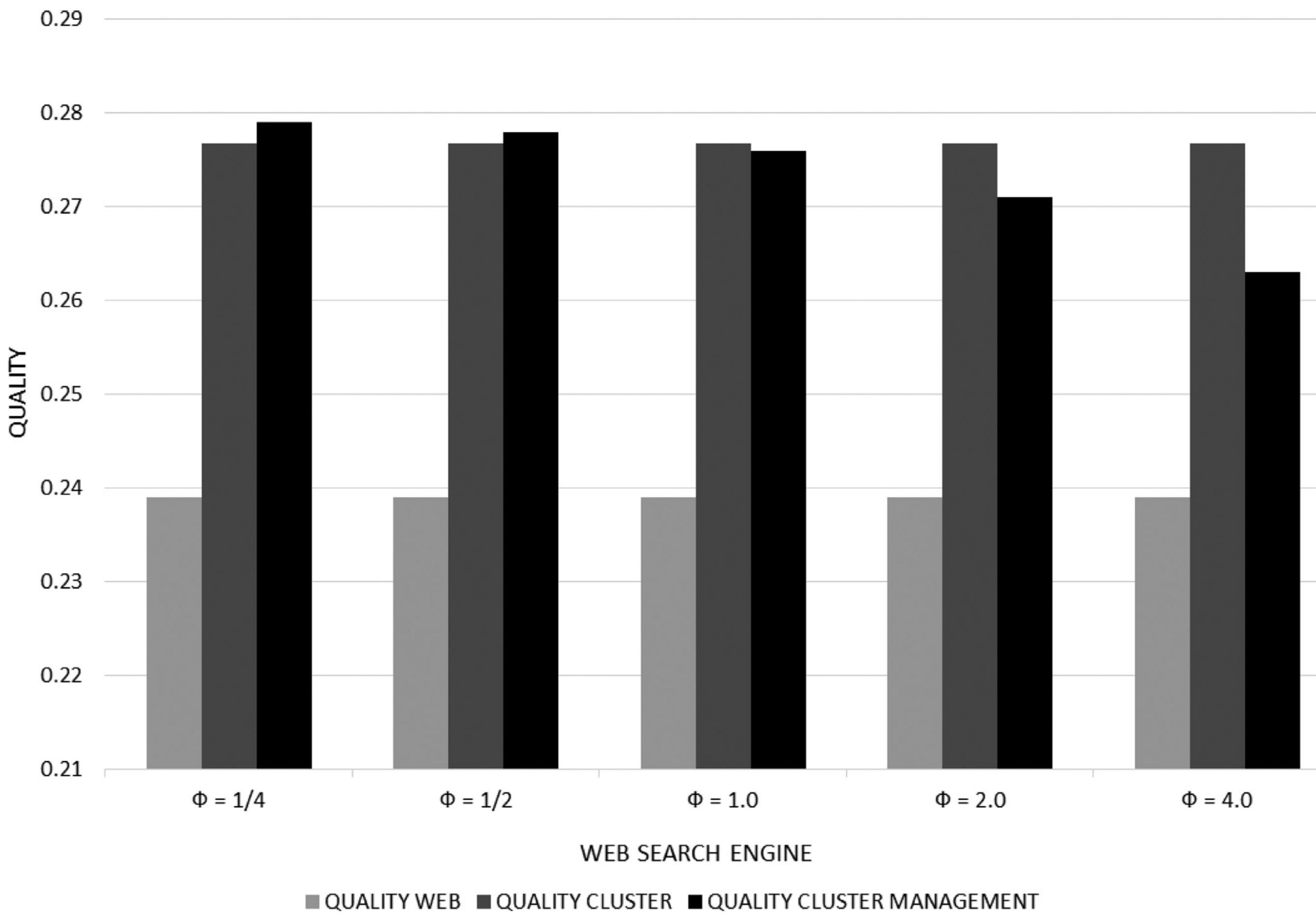


Fig. 15. Average results management cluster validation – average results.

Table 5
Management cluster validation – average values.

Learning Coefficient	Cost Function Q	Cluster Q	Management Custer Q	MC vs. Cluster I	MC vs. Cost Function I
$\gamma = 1/4$	0.2389	0.2767	0.2789	0.81%	16.75%
$\gamma = 1/2$	0.2389	0.2767	0.2776	0.34%	16.20%
$\gamma = 1.0$	0.2389	0.2767	0.2759	−0.30%	15.46%
$\gamma = 2.0$	0.2389	0.2767	0.2712	−1.97%	13.53%
$\gamma = 4.0$	0.2389	0.2767	0.2627	−5.05%	9.97%

6. Experimental results

Validators have been asked to introduce up to 8 dimensions per query. There are no rules in what users can search, however they have been advised their queries may be published. ISA acquires the first 10 results of each Web Search Engine, independently reorders them applying the cost function and finally shows a 50 result randomly reordered list joining the 10 results Web Search Engine cluster list. In the validation users have been asked to select 2 relevant results per Web Search Engine cluster list; this enables us to better compare performance between learning clusters. The proposed Deep Learning Cluster model has been validated with 45 different user queries.

6.1. Deep Learning Cluster results

The proposed Deep Learning Cluster structure has been validated with 45 different user search requests. Table 1 below shows the Normalised Quality ($\overline{Q}$) for the different Web Search Engines (WSE) and Deep Learning Clusters (DLC).

The Quality values of the different Web search engines and the Deep Learning Clusters before and after the user validation is represented in the Table 2 and Fig. 13 with the Improvement between the quality of the Deep Learning clusters after and before the user validation with the final average values.

On average; the Deep Learning clusters do not improve Web Search Engine performance before the user interaction; however, they improve Quality over 25% after the first user iteration due to their capability to learn user relevance.

Table 3 and Fig. 14 show the Quality values of the different clusters when the neural weights of the best performing cluster are selected and duplicated to each Deep Learning cluster. It is also represented the Improvement between the Learning Clusters with final average values.

When the best performing Deep Learning cluster is applied to other Web Search Engines; the search Quality improves almost 5%

on average. The best performing Deep Learning cluster can be used to teach or replace the other clusters; this improves result accuracy and relevance while reduces learning time and network weigh's computation.

6.2. Management cluster results

Table 4 shows the Quality values for the 45 different user queries at the different stages: before the user validation (Cost Function), after the user validation (Cluster) and with the Management Cluster (Management Cluster) adjusted to five different Management Cluster learning coefficients.

Table 5 and Fig. 15 show the average Quality values at the different stages: before the user validation (Cost Function), after the user validation (Cluster) and with the Management Cluster (Management Cluster) adjusted to five different Management Cluster learning coefficients. It is also represented the Improvement between the quality of the Management cluster after and before the user validation.

The Quality improvement from the Deep Learning Management Cluster is very dependent with γ with greater chance to highly degrade Quality than slightly improve it.

7. Conclusions

This paper has presented a biological inspired learning algorithm: the Random Neural Network in a Deep Learning structure with a Management Cluster. The Intelligent Search Assistant has been validated in a similar artificial environment where not all information can be processed due it is large amount: the Web.

On average; the results prove that the Deep Learning clusters outperform other Web search engines with a significant improvement after the user iteration. Cluster performance can be improved by learning from best performing clusters. The proposed application improves result accuracy and relevance while reduces learning time and network weigh's computation.

The Management cluster improves the overall Quality only when its learning coefficient is less than one; it has a detriment effect if it is equal to or greater than one. The Management Cluster needs to be the tuned to take the right decisions.

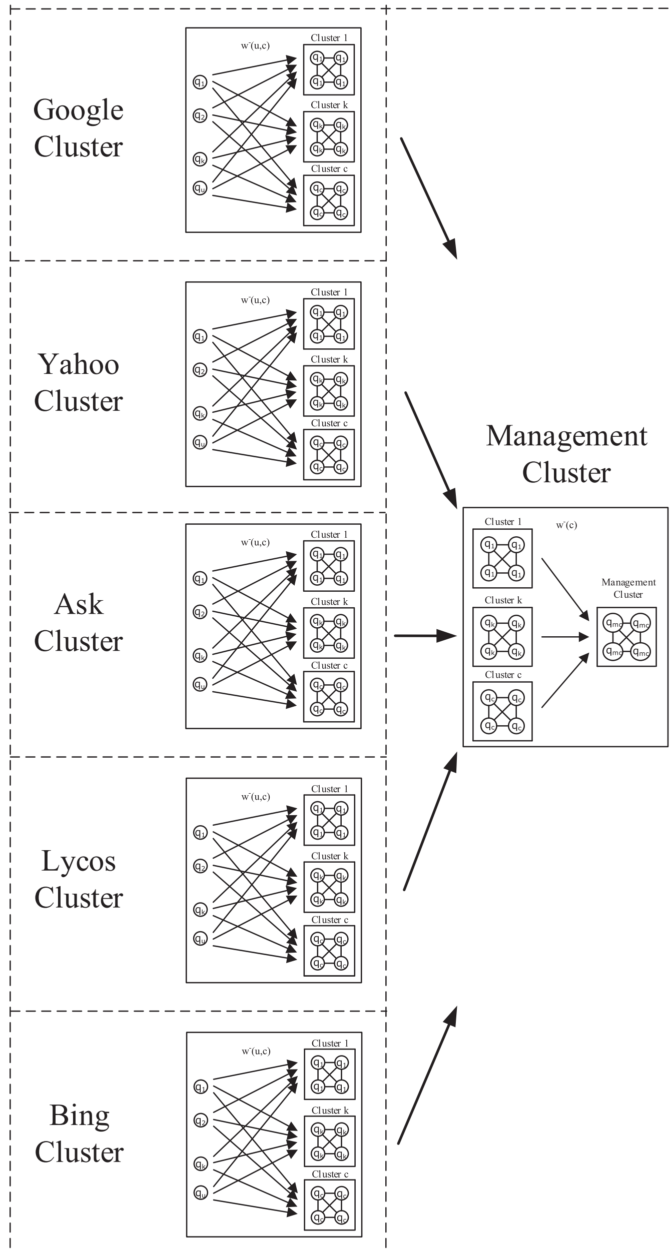
Conflict of interest

None.

Appendix. List of queries

Random Neural Network – Art Exhibitions London – Art Galleries Berlin – Night Clubs London – Night Clubs Berlin – Vegetarian restaurant Stoke Newington – Techno Festivals Europe – Indie Rock London – Flights Tokyo – Gentrification World-Best documentaries 2016 – Book shops soho – Boutique hotels Mexico city – Brownie cafe London – Haute couture catwalks 2016 – Mid Century Patterns – Recycling Center London – Vintage Arm Chair – Wedding Design Gifts Shops – Weekend Paris Break Tory Party Conference – restaurants Forest Gate – compare energy prices – mid century furniture – reclaimed wood London – Best dinner date London – Best holiday destinations – Holiday deals packages – Best rooftop bars London – cheap flights Shanghai – Film clubs Edin–urgh – fundraising jobs arts heritage Edinburgh – Holiday cottages rent Stoke on Trent 10 people – Home remedies sore throats – Volunteering opportunities Edinburgh – Brexit investments – Bristol breweries – Harmonica shops London – Narrow boats sale London – How get Irish passport – Best Pizza New York – Islington History – Southern rail overground – US presidential election forecast – Yorkshire dales walks

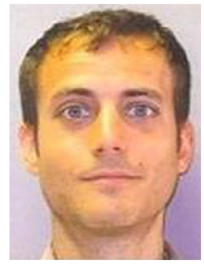
Appendix. ISA neural schematic



References

[1] D. Smith, E. Bullmore, Small-world brain networks, Neurosci.: Rev. J. Bring. Neurobiol. Neurol. Psychiatry 12 (2007) 512–523.
[2] O. Sporns, D. Chialvo, M. Kaiser, C. Hilgetag, Organization, development and function of complex brain networks, Trends Cognit. Sci. 8 (9) (2004) 418–425.
[3] W. Serrano, E. Gelenbe, An intelligent internet search assistant based on the random neural network, Artif. Intell. Appl. Innov. 475 (2016) 141–153.
[4] W. Serrano, A big data intelligent search assistant based on the random neural network, in: Proceedings of the International Neural Network Society Conference on Big Data, 2016, pp. 254–261.
[5] W. Serrano, E. Gelenbe, Intelligent search with deep learning clusters, in: Proceedings of the Intelligent Systems Conference, 2017, pp. 254–267.
[6] W. Serrano, E. Gelenbe, The deep learning random neural network with a management cluster, in: Proceedings of the International Conference on Intelligent Decision Technologies, 2017, pp. 185–195.
[7] E. Gelenbe, Random neural networks with negative and positive signals and product form solution, Neural Comput. 1 (1989) 502–510.

[8] E. Gelenbe, Learning in the recurrent random neural network, Neural Comput. 5 (1993) 154–164.
[9] E. Gelenbe, G-networks with triggered customer movement, J. Appl. Probab. 30 (1993) 742–748.
[10] A. Severyn, A. Moschitti, Learning to rank short text pairs with convolutional deep neural networks, Spec. Interest Group Inf. Retr. (2015) 373–382, doi:10.1145/2766462.2767738.
[11] B. Wang, D. Klabjan, An attention-based deep net for learning to rank, CoRR (2017) 1–8 abs/1702.06106.
[12] L. Deng, X. He, J. Gao, Deep stacking networks for information retrieval, in: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, 2013, pp. 3153–3157.
[13] H. Wang, N. Wang, D. Yeung, Collaborative deep learning for recommender systems, Spec. Interest Group Knowl. Discov. Data Min. (2015) 1235–1244, doi:10.1145/2783258.2783273.
[14] A. Mamdouh, Y. Song, X. He, A multi-view deep learning approach for cross domain user modeling in recommendation systems, World Wide Web (2015) 278–288, doi:10.1145/2736277.2741667.
[15] E. Gelenbe, R. Lent, Z. Xu, Networking with cognitive packets, in: Proceedings of the International Conference on Artificial Neural Networks, 2002, pp. 259–264.
[16] E. Gelenbe, F. Wu, Large scale simulation for human evacuation and rescue, Comput. Math. Appl. 64 (12) (2012) 3869–3880.
[17] A. Filippoupolitis, L. Hey, G. Loukas, E. Gelenbe, S. Timotheou, Emergency response simulation using wireless sensor networks, Ambient Med. Syst. 21 (2008) 1–7.
[18] E. Gelenbe, T. Koçak, Area-based results for mine detection, IEEE Trans. Geosci. Remote Sensing 38 (1) (2000) 12–24.
[19] E. Gelenbe, M. Sungur, C. Cramer, P. Gelenbe, Traffic and video quality with adaptive neural compression, Multimed. Syst. 4 (6) (1996) 357–369.
[20] V. Atalay, E. Gelenbe, N. Yalabik, The random neural network model for texture generation, Int. J. Pattern Recognit. Artif. Intell. 6 (1) (1992) 131–141.
[21] E. Gelenbe, Y. Yin, Deep learning with random neural networks, in: Proceedings of the International Joint Conference on Neural Networks, 2016, pp. 1633–1638.
[22] Y. Yin, E. Gelenbe, Deep learning in multi-layer architectures of dense nuclei, CoRR (2016) 1–10 abs/1609.07160.
[23] Y. Yin, E. Gelenbe, Single-cell based random neural network for deep learning, in: Proceedings of the International Joint Conference on Neural Networks, 2017, pp. 86–93.
[24] Y. Yin, L. Wang, E. Gelenbe, Multi-layer neural networks for quality of service oriented server-state classification in cloud servers, in: Proceedings of the International Joint Conference on Neural Networks, 2017, pp. 1623–1627.
[25] E. Gelenbe, G-networks: a unifying model for neural nets and queueing networks, Ann. Oper. Res. 48 (1994) 433, doi:10.1007/BF02033314.
[26] J. Fourneau, E. Gelenbe, R. Suros, G-networks with multiple class negative and positive customers, Model. Anal. Simul. Comput. Telecommun. Syst. (1994) 30–34, doi:10.1109/MASCOT.1994.284452.
[27] E. Gelenbe, S. Timotheou, Random neural networks with synchronized interactions, Neural Comput. 20 (9) (2008) 2308–2324.
[28] L. Wang, E. Gelenbe, in: Experiments with smart workload allocation to cloud servers, Network Cloud Computing and Applications, 2015, pp. 31–35, doi:10.1109/NCCA.2015.15.
[29] L. Wang, E. Gelenbe, Adaptive dispatching of tasks in the cloud, Cloud Comput. 6 (1) (2018), doi:10.1109/TCC.2015.2474406.
[30] O. Brun, L. Wang, E. Gelenbe, Big data for autonomic intercontinental overlays, IEEE J. Select. Areas Commun. 34 (3) (2016) 575–583.
[31] E. Gelenbe, in: Autonomous search for information in an unknown environment, Cooperative information agents, 1999, pp. 47–60. ISBN:3-540-66325-8.
[32] E. Gelenbe, O. Abdelrahman, Search in the universe of big networks and data, IEEE Netw. 28 (4) (2014) 20–25.
[33] O. Abdelrahman, E. Gelenbe, Search in non-homogenous random environments, Spec. Interest Group Measur. Eval. Perform. Eval. Rev. 39 (3) (2011) 37–39.
[34] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Netw. 61 (2015) 85–117.
[35] L. Kasun, H. Zhou, G. Huang, Representational learning with extreme learning machine for big data, IEEE Intell. Syst. 28 (6) (2013) 31–34.



Will Serrano is a Chartered Engineer and Technology Designer specializing the technical design and design management or large and high-profile infrastructure projects, from airports, rail stations to buildings and energy plants such as CrossRail, Heathrow Gatwick Airports and High Speed 2. He has delivered solutions in Local Area Networks; Wide Area Networks; Wireless LAN; Voice; Cloud and Cybersecurity. Will Serrano holds a Master's Degree in Telecommunication Systems and Networks and a PHD at Imperial College London. He is a Member of the Institution of Engineering and Technology. He is researcher with the Intelligent Systems and Networks Department at Imperial College London, UK.

Erol Gelenbe is a Fellow of IEEE, ACM and IET (UK), and a Professor in the Institute of Theoretical and Applied Computer Science of the Polish Academy of Sciences, and at Imperial College. He has introduced computer and network performance models based on diffusion approximations, and invented the Random Neural Network Model, as well as G-Networks which are analytically solvable queueing models that incorporate control functions such as work removal and load balancing. His other contributions include the concept and prototype for FLEXSIM, an object oriented discrete event simulation approach for flexible manufacturing systems, and other commercially successful projects such as the QNAP tool for the Performance Evaluation of Computer Systems and Networks. His innovative designs include the first voice-packet switch SYCOMORE, the first fibre optics random access network XANTHOS, and the Cognitive Packet Network and its adaptive routing protocol. He also designed and published the first optimal protocol for random access communications, and an optimum checkpointing scheme for databases. He has been awarded several prizes from France, the UK, Hungary and Turkey, including the 2010 IET Oliver Lodge Medal, the 2008 ACM SIGMETRICS Life-Time Achievement Award, and the 1996 Grand Prix France Telecom of the French Academy of Sciences. He was awarded Knight of the Legion of Honour and Officer of the Order of Merit of France, and Grand Officer of the Order of the Star and Commander of Merit of Italy. He is a Fellow of the French National Academy of Engineering, the Royal Academy of Belgium, the Science Academies of Hungary and Poland, and the Science Academy of Turkey. He was awarded Honoris Causa doctorates from the Universities of Liege, Belgium, Roma II, Italy, and Bogazici,Istanbul. He has graduated over 83 PhD students. His recent papers appear in the IEEE Systems Journal, IEEE Access, IEEE Trans. on Selected Areas in Communications, IEEE Trans. on Wireless Networks, Physical Review, and other journals.

Yonghua Yin received his B.E. Degree in Automation and M.S. Degree in Electronics and Communication Engineering under the supervision of Prof. Yunong Zhang from School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China in 2012 and 2014, respectively. He is currently pursuing the PhD degree under the supervision of Prof. Erol Gelenbe in the Intelligent Systems and Networks group of the Department of Electrical and Electronic Engineering at Imperial College London, UK. His research interests include neural networks, deep learning, energy packet network and control theory.