Research Progress Report

Botao Zhu

January 7, 2019

1 Reading and Research Activities

1.1 Summary

In this section, some applications of applying Deep Learning into the field of routing were introduced.

[1] proposed an adaptive, energy-efficient, and lifetime-aware routing based on Q-learning algorithm, which can deal with challenges such as the large propagation delay and the stringent power in the condition of underwater sensor network (Remarks:need to study Q-learning). In order to reduce the complexity of computation for finding a global optimal path in WSN, the authors used Deep Learning(DL) to relieve the path search burden.

[2] proposed a DL approach to control traffic in heterogeneous network. First, they collected enough parameters by executing the traditional routing protocols such as Open Shortest Path First(OSPF). And then, they used a supervised training method to process the data. During this period, a greedy layer-wise training method was used to initialize and a backpropagation algorithm was used to fine-tune. During the phase of running, the DL model needs to be run K rounds, which is the number of hops, by the source node for finding the optimal path from the source node to the destination node. The history of the traffic patterns of all the routers work as the input and only one router is chosen as the output.

[3] proposed a DL-based routing scheme. It aims to solve the following problems: (1) mitigate the traffic overhead imposed on BSs in disaster situations, (2) extend BS coverage through collaboration with wireless ad-hoc nextworks, (3) establish the maximum number of communication connections in situations in which network resources are insufficient and communication links are unstable. To begin with, the node degrees are divided into five stages using DL algorithm. Then, Viterbi algorithm was used to generated a virtual router in terms of the node degrees. At the end, the route was established by an IP-based routing procedure.

[4] used a Deep-Reinforcement Learning(DRL) to optimize routing by designing and training a DRL agent, which can provide routing configurations that tend to minimize the network's delay.

However, because source nodes need to run many models [2], which will consume high power, [5] integrated [2]'s method with programmable routers and achieved good performance. It proposed a DL-based routing table construction method for a GPU-accelerated SDR. They adopted supervised Deep Belief Architectures(DBA) to compute the subsequent nodes with the traffic patterns of the edge routers as the input. And then, they presented unique characterizations of inputs and outputs based on the traffic patterns at the edge routers. They demonstrated how the trained DBA can predict the nodes, the benefits of the DL-based routing strategy in terms of lower signaling overhead as well as fast convergence and effectiveness of proposed DL-based solution compared to a benchmark routing method through both analysis and extensive. Next section will introduce clearly.

1.2 Deep Learning Based Routing Strategy

1.2.1 Input and Output Design

The traffic pattern is served as the input to the DL structure and processed for routing path decision as the output. The traffic pattern at each router can be defined as the number of inbound packets of the router during each time interval,such as $\beta\triangle$t. Therefore, by assuming that a network comprises of N routers, we can use a matrix of β rows and N columns to represent the traffic patterns of all

routers. The simulation results demonstrate that it is accurate enough to set the value of β to 1. The output layer can be designed to give the next node similar to the distributed routing strategy because of its lower complexity and higher tolerance. In the vector, only a single element has the value of 1, the order of which represents the next node. So, we can use two N dimensional vectors to represent the input and output.

1.2.2 Deep Learning Structure Design

The authors choose the DBA as the deep learning structures as shown in Figure 1.

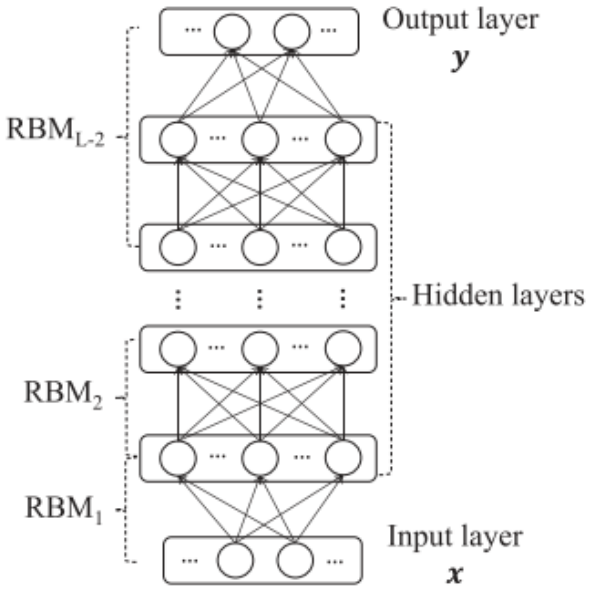


Figure 1: Considered L-layer DBA

DBA can be also seen as a stack of $(L-2)$ Restricted Boltzmann Machines(RBM) and a logistic regression layer as the top layer. The structure of RBM consists of two layers, the visible layer, $\mathbf{v}$, and the hidden layer,$\mathbf{h}$, shown in Figure 2. While training an RBM, sets of unlabeled data are given

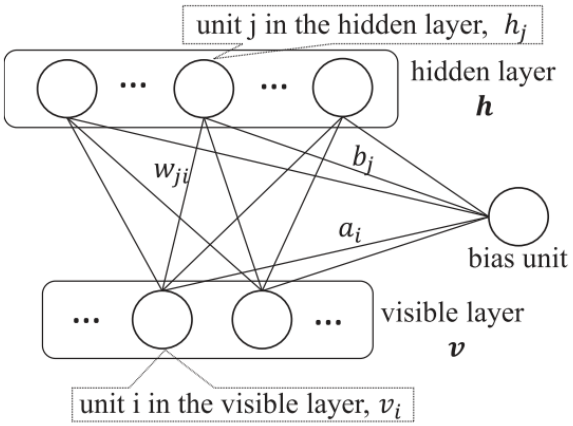


Figure 2: RBM

to the visible layer, and the value of the weights and biases are repeatedly adjusted until the hidden

layer can reconstruct the visible layer. To mathematically model the training process, they use a log-likelihood function of the visible layer given as follows.

$$l(\theta, a) = \sum_{t=1}^{m} \log p\left(v^t\right) \tag{1}$$

where θ denotes the vector consisting of all the values of the weights and biases of the hidden layer. Since RBM is a particular form of log-linear Markov Random Field, the energy function, $E(v,h)$, and the joint probability function, $p(v,h)$, are defined as follows:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j h_j w_{ji} v_i \tag{2}$$

$$p(v,h) = \frac{e^{-E(v,h)}}{Z} \tag{3}$$

$$Z = \sum_v \sum_h e^{-E(v,h)} \tag{4}$$

where v_i and h_j are the unit i in the visible layer and the unit j in the hidden layer respectively. Z represents the normalizing constant partition function. Also, the relationship between $p(v)$ and $p(v,h)$ can be expressed as follows:

$$p(v) = \sum_h (v,h) \tag{5}$$

However, the complexity of the calculation is extremely high. To solve this problem, they use the Gibbs Sampling method to sample the values of v and h to approximate the real values since the conditional distribution probability of one layer can be calculated. As each unit is independent on the other units in the same layer, when one layers is fixed, the conditional distribution probability of the other layer can be calculated as follows:

$$p(v|h;\theta,a) = \prod_i p(v_i|h;\theta,a) \tag{6}$$

$$p(h|v;\theta,a) = \prod_j p(h_j|v;\theta,a) \tag{7}$$

Since input units are continuous, they use Gaussian probability distribution to model the traffic patterns. Equations(2) and (6) should be revised as follows:

$$E(v,h) = -\sum_i \frac{(v_i - a_i)^2}{2{\delta_i}^2} - \sum_j b_j h_j - \sum_i \sum_j \frac{v_i}{\delta_i} h_j w_{ji} \tag{8}$$

$$p(v_i|h;\theta,a) = N\left(a_i + \delta_i \sum_j h_j w_{ji}, {\delta_i}^2\right) \tag{9}$$

where δ_i is the value of the variance for the unit v_i. However, in Figure 3, we can see that the visible layer of RBM_{L-2} consists of not only RBM_{L-3}'s hidden layer but also the output layer of the DBA, y. Its hidden layer is the top hidden layer of the DBA and energy function is expressed as follows.

$$E(v,h,y) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum k c_k y_k - \sum_i \sum_j h_j w_{ji} v_i - \sum_j \sum_k h_j w_j k y_k \tag{10}$$

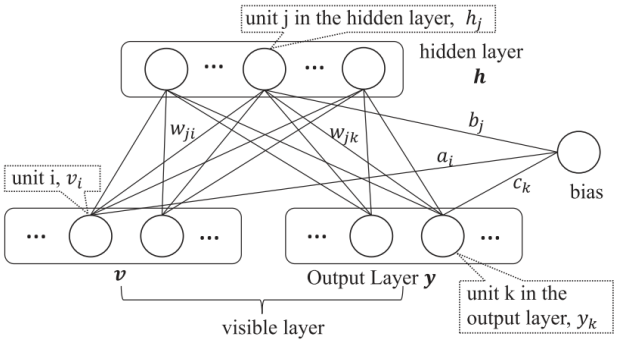


Figure 3: Last RBM

where y represents the vector in the output layer. c_k is the bias of the unit y_k. w_{jk} represents the weight of the link connecting the units h_j and y_k. The conditional distribution of the concatenated vector consisting of v and y is,

$$\begin{aligned} p\left(v,y|h;\theta,a\right) &= p\left(v|h;\theta,a\right)p\left(y|h;\theta,a\right) &(11)\\ &= \prod_i p\left(v_i|h;\theta,a\right)\prod_k p\left(y_k|h;\theta,a\right) &(12)\end{aligned}$$

The purpose of supervised training is to minimize the difference between the output of the DBA and the labeled output y. They use the cross-entropy cost function to measure the difference between the output of the DBA, denoted by $h_\theta\left(x\right)$, and the labeled output y.

$$C\left(\theta\right) = -\frac{1}{m}\sum_{t=1}^{m}\left(y^t\log\left(h_\theta\left(x^t\right)\right)+\left(1-y^t\right)\log\left(1-h_\theta\left(x^t\right)\right)\right)+\frac{\lambda}{2}\sum_{l=2}^{L}\sum_{j=1}^{n_l}\sum_{i=1}^{n_{l-1}}\left({w_{ji}}^l\right)^2 \quad (13)$$

$\left(x^t,y^t\right)$ is the tth training data. $h_\theta\left(x^t\right)$ denotes the output of the DBA. $C\left(\theta\right)$ consists of two parts: the difference between the output of DBA and the labeled output, keeping the training process from overfitting. Using the gradient descent to update θ to minimize the value of $C\left(\theta\right)$. η_{bp} is the learning rate in the back-propagation process.

$$\theta := \theta - \eta_{bp}\frac{\partial C\left(\theta\right)}{\partial\theta} \quad (14)$$

1.2.3 The Procedures of the Proposed Deep Learning Based Routing Strategy

Initialization Phase The goal of the initialization phase is to obtain the labeled data which consist of the input vector and the corresponding output vector. They approach a number of available data set sources, such as the center for applied internet data analysis, and extract the traffic information and relevant routing paths.

Training Phase The training phase consists of two steps: the loop of the Greedy Layer-Wise training to train each RBM and the following backpropagation process to fine-tune the weights of links between the layer. Let N and I denote the total number of routers and the number of inner routers. So,every edge router needs to train $(N-I-1)$ DBAs while each inner router needs to train $(N-I)$ DBAs. Then, every edge router needs to send its θ of $(N-I-1)$ to other $(N-I-1)$ edge routers. Every inner router needs to send its θ of $(N-I)$ DBAs to all the edge routers. Therefore, each edge router obtains θ of all the DBAs of all the routers in the network, and the number of sets

of θ is $(N-I)(N-1)$.

Algorithm 1. Supervised Train DBA

Input: $(x,y) = \{(x^t, y^t) | t = 1, ..., m\}, \eta_{CD}, \eta_{bp}, L, n = (n_1, ..., n_L)$

Output: θ

1: **for** $i = 1, ..., L-2$ **do**
2: TrainRBM $(u^i, \eta_{CD}, n_i, n_{i+1})$
3: **end for**
4: Fine-tuneDBA$((x,y), \theta, \eta_{bp})$
5: **return** θ

Running Phase Every edge router can utilize the next node information to construct the whole paths from itself to all the other edge routers. They use an array of N elements, $\mathcal{TP}[N]$, to save the numbers of inbound packets of N routers in the network to represent the traffic patterns, and $\theta[N-I][N-1]$ to save the parameters of all the DBAs in the network. Another array $\mathcal{ER}[N-I]$ is used to save the sequence numbers of the edge routers in the network since they are not continuous.

Algorithm 2. Running Phase

Input: $\mathcal{TP}[N], \theta[N-I][N-1], \mathcal{ER}[N-I], sr$

Output: $\mathcal{NR}[N][N-I-1]$

1: $D \leftarrow \mathcal{ER}[N-I] - sr$
2: **while D** $\neq 0$ **do**
3: $d \in \mathbf{D}$
4: $s \leftarrow sr$
5: **repeat**
6: $\theta' \leftarrow \theta[s][d]$
7: $nr \leftarrow$ run DBA with θ' and $\mathcal{TP}[N]$
8: $\mathcal{NR}[s][d] \leftarrow nr$
9: $s \leftarrow nr$
10: **until** $nr = d$
11: $D \leftarrow D - d$
12: **end while**
13: **return** $\mathcal{NR}[N][N-I-1]$

Running algorithm 2, each edge router can obtain the outputs of DBAs to construct the paths to $(N-I-1)$ edge routers. And then using a matrix, $\mathcal{NR}[N][N-I-1]$, to save the results of these DBAs that can be used to build the whole paths to all the other edge routers. Figure 4 is the routing model and Table 1 is an example of the routing table.

Table 1: Routing table of R_3

Dest	Path
R_1	$R_3 \rightarrow R_2 \rightarrow R_1$
R_2	$R_3 \rightarrow R_2$
...	...
R_{16}	$R_3 \rightarrow R_7 \rightarrow R_{11} \rightarrow R_{15} \rightarrow R_{16}$

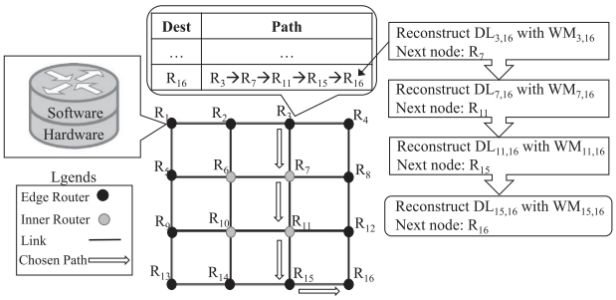
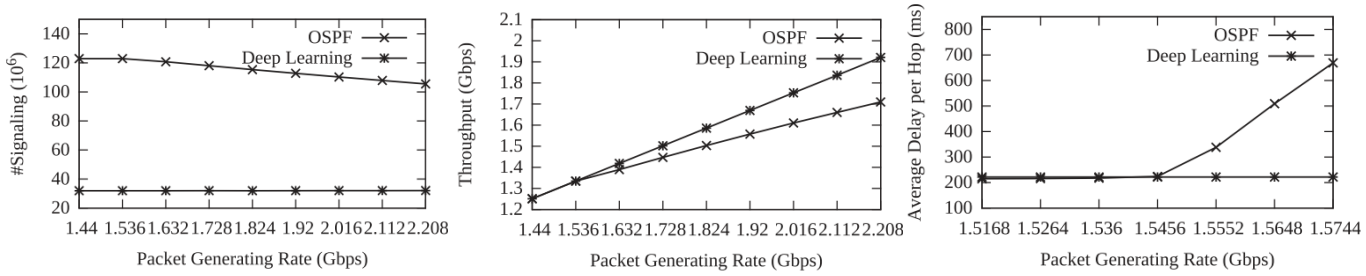


Figure 4: Routing model

1.2.4 Network Performance Analysis

They compared the proposed deep learning method with OSPF by the network signaling overhead, throughput and average delay.



(a) Comparison of signaling overhead (b) Comparison of throughput (c) Comparison of average delay

Figure 5: Comparison of network performance under different network loads

2 Objectives for the Next 2 Weeks

2.1 Reading

Reading papers foused on ML-based or DL-based routing.

2.2 Course

Studying chapter 1 and chapter 2 of Neural Networks and Deep Learning, **Coursera**. `https://www.coursera.org/learn/neural-networks-deep-learning`

2.3 Code

Studying the classic routing protocol: LEACH and using Matlab to implement.

3 Advisor's Comments

Bibliography

[1] T. Hu and Y. Fei, "Qelar: A machine-learning-based adaptive routing protocol for energy-efficient and lifetime-extended underwater sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 6, pp. 796–809, June 2010.

[2] N. Kato, Z. M. Fadlullah, B. Mao, F. Tang, O. Akashi, T. Inoue, and K. Mizutani, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 146–153, June 2017.

[3] Y. Lee, "Classification of node degree based on deep learning and routing method applied for virtual route assignment," *Ad Hoc Networks*, vol. 58, pp. 70–85, April 2017.

[4] G. Stampa, M. Arias, D. Sanchez-Charles, V. Muntés-Mulero, and A. Cabellos, "A deep-reinforcement learning approach for software-defined networking routing optimization," *CoRR*, vol. abs/1709.07080, 2017. [Online]. Available: http://arxiv.org/abs/1709.07080

[5] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1946–1960, Nov 2017.