A Fast and Efficient Route Finding Method for Car Navigation Systems with Neural Networks

Mehdi Hashemzadeh[a], Mohammad. R. Meybodi[b]
[a]*Department of Electrical & Computer, Islamic Azad University, Qazvin Branch, Qazvin, Iran*
[b]*Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran*
E-mail: hashemzadeh@qazviniau.ac.ir

Abstract

In this paper we have proposed a new route finding method for car navigation systems meanwhile, have utilized learning power and high speed of neural networks in our proposed method. Our suggested procedures miss all limitations on standard algorithms planned for all these systems. In this way, despite of optimal rout finding between the origin and destination the parameters have been adopted than regarding roads traffics conditions, their limits and coincidence of routes in which all driver requests can simply be carried out. We believe that, this has been the first time discussing about neural networks on car navigation systems. Therefore, it could be the start up for widespread investigations.

Keywords: Car Navigation System, Neural Network, Route Finding.

1. Introduction

Car navigation systems are devices that are installed on cars using GPS signals, and digital road maps showing cars position and finally assisting drivers on finding routes between origin and destination. The provision of real-time information on traffic congestion began in major urban areas in most countries. Those systems that have access to this information can use them in their routing.

From 1995 highly advanced car navigation systems have been introduced to the market so far. These systems from the possibilities of drivers' guidance have had efficient digital maps, and also their sending orders have gained considerable audio and vision developments [1]. Up to now a great deal of dull problems related to these systems have been solved such as accurate positioning, collecting complete information of digital maps in huge volume and also direct guidelines to be able to assure drivers all have been included. But still some problems incase of routing algorithms and their optimal route calculations are exist.

Route selection problems in car navigation systems are search problems for finding an optimal route from a starting point to a destination on a road map [2]. It appears that for finding the same optimal route to be the shortest route within origin and destination. And it is common that someone uses some standardized algorithm for search the roadway graphs. But if you expect the systems to act out like a driver who knows all routes in city and is aware of all traffic limitations (like No turning left or right, No U-turn and …) and simultaneously the connected information of the traffic conditions of roads should be sent out to him, this algorithms will not be able to solve the problem. Of course some techniques have been used in this aspect and some progresses have been achieved. And some algorithms have been proposed which those have just some theoretical viewpoints due to some problems that will be discussed here later on. Thus, those can hardly be implemented.

In this paper we are advising a procedure which that uses the learning power and memorization of neural networks. In fact we produce possible aspects that can be applied in routing in a city can all be anticipated already, and then we find the reply for those cases. So we train the neural network. After having trained the network we are using it on time. It must be mentioned that, the neural networks are working with highly speed during its answer time.

In second section the problems and limitations of routing algorithms in these systems are being mentioned. In third section our proposed method will be planned and achieved experimental results will be discussed and finally in forth section the advantages of already expressed view-points will be discussed.

2. Problems and limitations of exiting methods and algorithms

From the majority of initial investigations carried-out and represented algorithms in this manner, they all have been on the basis of Dijkstra algorithm and A* algorithm [3]. In this one the size and scale of the search-graph is very important to have high speed calculation. Normally the road graphs are met of greater sizes (The number of the nodes and edges are more). So, the most of the investigations in case of optimizations have been done to reducing search time. Due to mobility of vehicles and their passengers' security and approaching in time orders prior to passing on intersections in wanted routes, the routing functions have been done in real-time.

As to A* algorithms, a RTA* and a LRTA* have been proposed for real-time applications [3]. They usually use the direct distance between the next crossing and the destination as a heuristic function. Hence, they get closer optimal from the point of distance. In some proposed methods the calculation time of the Dijkstra like algorithms have been reduced by restricting a search to within the area where traffic congestion has changed, but this method can not perform global searches. And also Time-Constrained Search (TCS) has been proposed [4]. It can achieve a closer optimal reply in a specified time. Some other various techniques are expressed for improving the speed of processing [5, 6]. Some of proposed algorithms use graph partitioning to reduce space for search in which numerous procedures have been expressed in this aspect [7, 8].

Although some of suggested algorithms are able to find out the responses in real-time, but in contrast they meet low qualities in routes they calculate. If we disregard the above-mentioned point, still some problems and limitations are exiting in route finding on car navigation systems, they cause implementations of the methods to be impossible. That some of these limitations are as follow:

Rotational limitations on intersections

As an example if we consider Figure 1.a in existing intersection, the out come graph will be in form of Figure 1.b. From the point of Dijkstra algorithm and A* algorithm to go from node d to node a trough node b, will be possible. But it will be impossible due to the sign of no left. The implementation of these limitations that their numbers are a lot will be impossible in some algorithms like A*. And if it is supposed to investigate any nodes in any limitations,

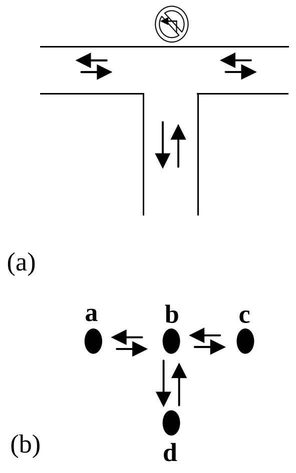


Figure 1. A sample intersection

the search time will be really high. And also this information in case of all intersections and streets must be stored and then loaded during the working time into the system's memory.

Problems of graph portioning based algorithms

Most of the defined algorithms are using graph partitioning in some ways like dividing roadway graphs into several partitions of the zones, and only do their search within zones and for transferring between zones they use permanent previously defined links. These fixed links are usually regarded on freeways. These algorithms have basically two difficulties. First if these all vehicles are supposed to use same systems, after some short period of time these permanent links will be congested. Secondly as per integrity of urban environmental structure there will be no possibility roadway graph on graph partitioning. Because the nodes usually are distributed in all parts of the city.

Non possibilities of the existing algorithms for implementation of comparative searching

In the suggested algorithms as per various drivers' requests the routing can not be carried out so far. In real world the drivers usually regard other parameters reaching to their destination. As an example during their voyage they want to cross special location such as gas-station, banks, restaurants, and so on. Or they may want to pass through streets with longer routes, meanwhile they keep sufficient quietness on their own trips. And other items as well. But implementing of all these parameters are approximately impossible on the proposed algorithms.

Difficulties with regarding the traffic conditions information

The algorithms are regarded on traffic information are mostly have theoretical aspects, and usually their implementations are impossible. Though some of them are implemented but still they lack possibility and they include some difficulties. One of their difficulties is regarding the traffic conditions on all links. In practice, information of traffic conditions are switched only on some of the streets, which those usually are in center of the city. But the defined algorithms during change on traffic conditions of a street usually force all links to do searching for finding the wanted link.

3. Our proposed method

Regarding to our described difficulties on second section and if we want these systems work actually and not be theoretical ones. Even the driver who comes for the first time into the town could drive like other resident drivers. The best solution is training the system to find the reply for any request. Therefore all traffic patterns are regarded according to the traffic conditions to be able to find possible optimal route between two exiting intersection on roadway graphs and due to exiting traffic conditions and parameters in every pattern we determine optimal reply. We use these various aspects as input and output vectors for training neural networks. And we use learning power and immediate reply on neural networks. Actually we reserve all possible requests and their replies. It may first appear that why we are not using a database for this. Then we notice that due to all possible aspects huge database is needed. Due to hardware limitations and specially low capacity of the systems' memory, the implementation will not be possible. And also reply speed of neural networks is very high against search of databases. Of course in this case more details will be described in the next sections.

3.1. Details on proposed method

The input pattern (input vector) of the neural network consists of components for determining origin and destination points and also are being dispatched on streets traffic conditions, that are being used in routing and every another parameter, that we want to have. For example in Figure 2 you will find 6 intersections, 13

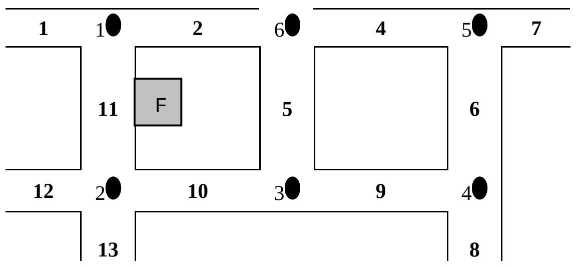


Figure 2. A sample map of streets

streets, a sign regarding no turning left, and a gas-station. Supposing streets 2, 3, 4 and 5 equiped to sending traffic conditions.

For displaying origin and destination intersection we represent them as bipolar (-1, +1) representation. Considering traffic information, if a link is congested, the component for that link in the input vector will be +1, otherwise it will be -1. And in case of gas-station requesting we will put +1 in component of input vector which is belong to the gas-station request. Since the map of Figure 2 has 6 intersections therefore 3 bits will be enough for representation on origin and destination. Table 1 shows the binary and bipolar representation of intersections.

The input pattern (input vector) of the neural network has eleven components. The first three are used for representing origin and the second three will be used for destination. Components 7, 8, 9 and 10 are determined respectively for streets 2, 3, 4 and 5 which have traffic information system. And the eleventh is considered for determining gas-station ($X[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}]$).

As an example the path of the intersection 5 towards intersection 1 as per gas-station request and also jam of traffic on streets 3 and 4 all can be represented as vector X[+1,-1,+1,-1,-1,+1,-1,+1,+1,-1,+1].

Table 1. Binary and Bipolar representation of intersections

Intersection NO	Binary Representation	Bipolar Representation
1	001	-1-11
2	010	-11-1
3	011	-111
4	100	1-1-1
5	101	1-11
6	110	11-1

The output vector consists of components numbering of all streets. For instance for this map we should have 13 components in the output vector. The component of each street that is in the reply route will be +1, and those streets that are not on the route, their components will be -1. So regarding to the case that mentioned above, if we select streets 6, 9, 10 and 11 as reply route, the output vector will be: Y[-1,-1,-1,-1,-1,+1,-1,-1,+1,+1,+1,-1,-1].

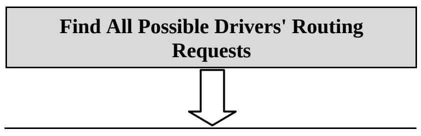
Or for another example if we consider that origin is intersection 4 and destination is intersection 6, and the links 2 and 5 are met of high traffic volume, so due to limitation on turning no left sign that is already on intersection 5, out route will be links 9 and 5. Therefore input and output vectors will be:

X[+1,-1,-1,+1,+1,-1,+1,-1,-1,+1,-1],
Y[-1,-1,-1,-1,+1,-1,-1,-1,+1,-1,-1,-1,-1].

So we produce all possible cases on the map and their suitable replies. And they are being stored as input and output vectors to train the neural network. (In simulated sample, a computer program have been made which the user only sees the names of the origin and destination, and jammed traffic streets and then enters the streets which are in the selected route. So, the program produces input and output vectors. More explanations have been presented on section 3.3) After production of all cases and training vectors, we train neural network. The only thing that is needed after neural network training is storing weight matrix on car navigation system. During system's function the input pattern is constituted as per drivers' request and then delivers to neural network. So, the network carries out with high output speed and the streets on route are specified. Of course in this arrangement, streets order must be determined. In our implemented system, we used the information which those stored about each intersection and its streets to determine the orders of selected streets in the reply route. And any time new information gets transferred to roadway traffics, car navigation system immediately regards present location and new conditions will produce new pattern and its reply. You can find a general view of our proposed method in Figure 3 and Figure 4.

3.2. Backpropagation neural networks

One of the most powerful neural networks that has the high capability for storing and learning the patterns, is Backpropagation neural networks. In this section some explanations are represented for this type of neural networks.



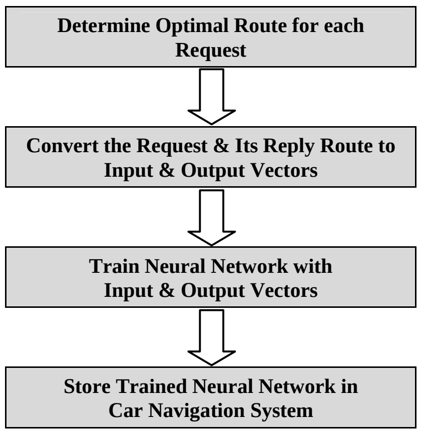

Determine Optimal Route for each Request
Convert the Request & Its Reply Route to Input & Output Vectors
Train Neural Network with Input & Output Vectors
Store Trained Neural Network in Car Navigation System

Figure 3. Training of neural network

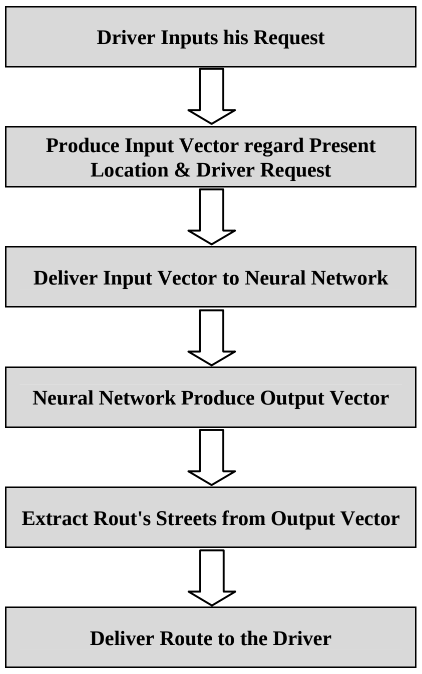

Driver Inputs his Request
Produce Input Vector regard Present Location & Driver Request
Deliver Input Vector to Neural Network
Neural Network Produce Output Vector
Extract Rout's Streets from Output Vector
Deliver Route to the Driver

Figure 4. Working of car navigation system

Backpropagation is a training method. It is simply a gradient descent method to minimize the total squared error of the output computed by the net. The very general nature of the backpropagation training method means that a backpropagation net (a multilayer, feedforward net trained by backpropagation) can be used to solve problems in many areas. Applications using such nets can be found in virtually every field that uses neural nets for problems that involve mapping a given set of inputs to a specified set of target outputs (that is, nets that use supervised training) [9].

The training of a network by backpropagation involves three stages: the feedforward of the input training pattern, the calculation and backpropagation of the associated error, and the adjustment of the weights. After training, application of the net involves only the computations of the feedforward phase. Even if training is slow, a trained net can produce its output very rapidly [9].

Although a single-layer net is severely limited in the mapping it can learn, a multilayer net (with one or more hidden layers) can learn any continuous mapping to an arbitrary accuracy. More than one hidden layer may be beneficial for some applications, but one hidden layer is sufficient [9].

There are sufficient descriptions in case of these networks' structure and also training of standard Backpropagation in [9]. A lot of changes and developments have been achieved dealing with Backpropagation neural networks. And also different methods have been done for progressing speed of training and its accuracy. You can find some of these achieved progresses have been applied in our suggestion in [10, 11, 12, 13, 14, 15].

3.3. Experimental results

We have implemented an actual map based upon information as per our suggestion. In that map we aimed 1000 intersections and 2000 streets. First we have stored all intersections numbering 1 up to 1000. So for displaying intersections on origin and destination 10 bits will be enough (2^{10}=1024). And the city's streets are specified numbering started 1 up to 2000. And also three items are considered for requesting gas-station, bank and department-store between origin and destination. So, 10 components of input vector considered for origin point, 10 others for destination and three others considered for gas-station, bank and department-store. And supposing, there are 77 streets for sending out traffic information. In this case our input vector will have 100 components. And the output vector will be the same as total numbers of streets. So, our output vector will have 2000 components. By using a computer program we produced 100000 various cases and their training vectors.

Table 2. Experimental results with various learning rates

Learning rate	Number of Epochs	Correct Response
0.05	60000	99.25%
0.25	60000	85%
0.5	60000	91%

Table 3. Experimental results with various number of epochs

Learning rate	Number of Epochs	Correct Response
0.05	10000	58%
0.05	60000	99.25%
0.05	70000	90%

In a Backpropagation network which has one hidden layer, we have used 100 neurons on input layer and 100 neurons on hidden layer and finally 2000 neurons will be on output layer. From the offered suggestion done by Nguyen-Widrow [9], we have used initial weights and also we have used bipolar sigmoid activation function as activation function. Learning of the network for 100000 input and output patterns have been researched based on learning rate and the number of epochs in training phase. Therefore some results have been achieved according to the Table 2 and Table 3. With learning rate 0.05 and the number of epoch equal 60000 we get correct answer for 99.25% of tested patterns.

4. Discussion

If we take the defined difficulties into consideration in section II, and if we want the systems to assist drivers efficiently, then there will be only some routes in actual world in which the drivers adopt in front of various conditions. So, they will be stored in the system, and they will be used in necessitated time. We can discuss about benefits of our proposed method in four aspects.

First, other algorithms are able merely to consider the cost or distance of links in their searching. And only little progress have been achieved updating the links costs during traffic condition changes. Meanwhile our suggested method offering do not have any limitations from the point of parameters number or exiting measures. It is enough to add one component in the input vector, when we want to add a new parameter for our searching. It may only elongate the network training, in which there is no relationship on response time within driving. It must be mentioned that any method have not been represented yet in spite of reaching to destination to have taken any other various parameters into account.

Second, as it was said due to limitations of traffics, in which it outnumbers, practically implementing the graph searching algorithms in actual world is not possible. But our proposed method is out of this category.

The third stays with high speed of the neural networks during reply time. In practice it is the most important factor in neural network for storing patterns.

Forth, if it is supposed to store the input and output patterns on a database, first it should be loaded in memory, but that one itself has a problem due to limitation on hardware, especially lack of memory large capacity. But in case of neural network only we should load weight matrix in the memory. Furthermore speed of neural network during reply time in front of searching the database is very high, because the information volume on database is really high. And with increasing one intersection in city the possible aspects with other intersections as per number of records on database will be increased. Which on our proposed method as far as it has been regarded the number of intersection square power 2, you will not be able to see any changes on the dimension of the weight matrix. For instance there is the possibility for 10 bits determining origin and destination points on input pattern totalling up to 1024, if it outnumbers 1025 between intersections, it will be enough to add only 1 component on input pattern. Thus 2^{11} for 2048 intersections will be sufficient. In this case increasing one component would not have any effectiveness dealing weight matrix on neural network reply.

As on our suggestion we can use outmost techniques to achieve the best routes during learning phase of neural network, which there is not any relationship with the system's real-time application. We believe that, this has been the first time discussing about neural networks on car navigation systems. Therefore, it could be the start up for widespread investigations. Our future works would involve with accuracy of the neural network for this application. And also investigation on advanced neural network for application of the car navigation systems all will be effective developing the our proposed method

5. References

[1] C. Brenner and B. Elias, "Extracting Landmarks for Car Navigation Systems using Existing GIS Databases and Laser Scanning," ISPRS Archives, Vol. XXXIV, Part. 3/W8, Munich, Sept. 2003.

[2] T. Nakamura, "Head Start Route Selecting Algorithm for Reducing Driver's Waiting Time," The Second World Congress on Intelligent Transport Systems, pp. 2031-2036, 1995.

[3] R. E. Korf, "Real-Time Heuristic Search," Artificial Intelligence, Vol. 42, No. 2-3, pp.189-211, 1990.

[4] H. Hiraishi, H. Ohwada and F. Mizoguchi, "Intercommunicating Car Navigation System with Dynamic Route Finding," Proc.of IEEE/IEEJ/JSAI Int. Conference on Intelligent Transportation Systems, pp.284-289, 1999.

[5] I. Chabini and S. Lan, "Adaptation of the A* Algorithm for the Computation of Fastest Paths in Deterministic Discrete-Time Dynamic Networks," IEEE Trans. Intelligent Transportation Systems, Vol. 3, pp. 60-74, Mar. 2002.

[6] E. P. F. Chan and N. Zhang, "Finding Shortest Paths in Large Network Systems," Proc. of the 9th Int. Conference on Advances in Geographic Information Systems, ACM Press New York, NY, USA, 2001.

[7] I. Flinsenberg, "Graph Partitioning for Route Planning in Car navigation Systems", Proc. of the 11th IAIN World Congress, Smart Navigation Systems and Services , Berlin, Germany, Oct. 2003

[8] K. Kim, Y. Seungwon, and K. C. Sang, "A Partitioning Scheme for Hierarchical Path Finding Robust to Link Cost Update," Proc. of the 5th World Congress on Intelligent Transportation Systems, Seoul, Korea, Oct. 1998.

[9] L. Fausett, "Fundamentals of Neural Networks, Architectures, Algorithms, and Applications," Prentice Hall, Upper Saddle River, New Jersey 07458, © 1994.

[10] X. G. Wang, Z. Tang, H. Tamura, M. Ishii and W. D. Sun, "An Improved Backpropagation Algorithm to Avoid the Local Minima Problem," Neurocomputing, Vol. 56, pp. 455-460, 2004.

[11] K. Eom, K. Jung, H. Sirisena, "Performance Improvement of Backpropagation Algorithm by Automatic Activation Function Gain Tuning Using Fuzzy Logic," Neurocomputing, Vol. 50, pp. 439-460, 2003.

[12] M. Mandischer, "A Comparison of Evolution Strategies and Backpropagation for Neural Network Training," Neurocomputing, Vol. 42, pp. 87-117, 2002.

[13] B. Walczak, "Neural Networks with Robust Backpropagation Learning Algorithm," Analytica Chimica Acta, Vol. 322, pp. 21-29, 1996.

[14] G. D. Magoulas, M N. Vrahatis, G. S. Androulakis, "Effective Backpropagation Training with Variable Stepsize," Neural Networks, Vol. 10, No. 1, pp. 69-82, 1997.

[15] A. V. P. Espinoza, J. B. O. Mere, F. J. M. D. Pison, A. G. Marcos, "TAO-Robust Backpropagation Learning

Algorithm," Neural Networks, Vol. 18, pp. 191-204, 2005.