Short-term traffic flow forecasting based on two-tier k-nearest neighbor algorithm

Hou Xiaoyu, Wang Yisheng, Hu Siyu

*a Beijing STONE Intelligent Transportation System Integration Co., Ltd. Beijing 100081, China

Abstract

The K-nearest Neighbor algorithm does not require a priori knowledge and its forecasting results are better than those of the linear model algorithm. However, its computing speed is low and its parameter adjustment method is not flexible enough. Based on the traditional K-nearest neighbor algorithm, this paper proposes a two-tier K-nearest neighbor algorithm. Combined with the actual traffic flow, it calibrates the algorithm parameter to improve the calculation speed and the accuracy of the algorithm.

© 2013 The Authors. Published by Elsevier Ltd.
Selection and peer-review under responsibility of Chinese Overseas Transportation Association (COTA).

Keywords: Two-tier K-nearest neighbor algorithm; short-term forecasting; traffic flow; state vector; historical sample database.

1. Introduction

Traffic flow forecasting means to calculate traffic flow parameters of time $t + \Delta t$ when time $t$. Usually, those parameters which reflect the state of traffic are flow, speed and occupancy. The essence of traffic flow forecasting is to forecast those traffic flow parameters. Traffic flow forecasting usually includes long-term forecasting and short-term forecasting. The time interval $\Delta t$ of short-term forecasting is less than 15 minutes [1].

Traffic system is a complex large system which is nonlinear, time-varying and participated by people. Those bring difficulties to short-term forecasting. Now, common short-term forecasting methods include history average [2], time-series, Kalman filter, non-parametric regression, neural networks [3] and synthetic model, etc. The process of non-parametric regression algorithm is simple. It does not require prior knowledge and can get the calculated result very fast [4], so take non-parametric regression algorithm to do traffic flow forecasting.

* Hou Xiaoyu. Tel.: 18611747416; fax: 010-62124578.
E-mail address: houxiaoyu@sits.com.cn
The most commonly used non-parametric regression algorithms are algorithm based on K-nearest neighbor and algorithm based on kernel function (In this paper, the K-nearest neighbor algorithm is shorthand for the non-parametric regression algorithm based on K-nearest neighbor, the kernel function algorithm is shorthand for the non-parametric regression algorithm based on kernel function and the two-tier K-nearest neighbor algorithm is shorthand for the non-parametric regression algorithm based on two-tier K-nearest neighbor). Based on the K-nearest neighbor algorithm, this paper introduces the process of the two-tier K-nearest neighbor algorithm and describes how to establish the historical sample database of this algorithm. Then, this paper discusses the algorithm parameters with actual traffic flow data detected by RTMS in Beijing 3rd Ring Rd.

2. The two-tier K-nearest neighbor algorithm

2.1. The process of K-nearest neighbor algorithm

The general process of non-parametric algorithm used for traffic flow forecasting is shown as following.

a) Pretreat the real-time data, which means to judge whether the data is error or not. When the data is error, correct the data.
b) Using the corrected data, construct the current state vector.
c) With the current state vector and historical data, establish the historical sample database.
d) Use the historical sample database to perform pattern matching. At the same time, calculate the distance between the vector in historical sample database and current state vector in order to judge whether the current state vector is typical sample data or not. If the current state vector is typical sample data, put it into historical sample database.
e) Get the final forecasting results with the pattern matching results.

The process as shown below:

Fig. 1. The process of non-parametric algorithm
There are many kinds of non-parametric regression algorithm. The most important and commonly used are the K-nearest neighbor algorithm and the kernel function algorithm. Compared with the Kernel function algorithm, the K-nearest neighbor algorithm can get the results faster, so it is more suitable for traffic flow forecasting.

When calculates, the K-nearest neighbor algorithm does not need all samples to be involved in forecasting. It uses only the K-th nearest data.

For example, set the current state vector is \( X \) and the samples from historical sample database are \( X_{\text{hi}}, i = 1, ..., n \). The process of this algorithm is as follows:

First, calculate the Euclidean distance \( D_i \) between \( X \) and \( X_{\text{hi}} \).

Second, sort \( D_i \) from small to large and sort \( X_{\text{hi}} \) according to \( D_i \). The new historical samples array are \( X_{\text{ki}} \). Their corresponding weight values are \( W_{ki}, i = 1, ..., n \)

\[
W_{ki}(x, X_1, ..., X_n) = C_i, \quad i = 1, ..., n
\]  

Third, the weight values of the K-th nearest historical samples are the biggest, set their values \( C = 1 \). For the others, set \( C = 0 \). The final formula of the K-nearest neighbor algorithm used for short-term forecasting is as follows:

\[
X(t + 1) = g(X(t)) = \sum_{i=1}^{K} X_{\text{hi}}(t + 1)/ K
\]  

K is the number of the selected nearest historical samples.

2.2. The process of two-tier K-nearest neighbor algorithm

Based on the K-nearest neighbor algorithm, the two-tier K-nearest neighbor algorithm improves the pattern matching section, adding the state pattern matching step. The algorithm ensures the distance and trend between current state vectors and the historical samples is minimum.

There is a time series \( X \). \( X = \{x(1), x(2), \ldots, x(n)\} \). The current value is \( x(n) \), and the forecasting value is \( x(n + 1) \). The steps of the two-tier K-nearest neighbor algorithm are as follows:

Step 1: Construct the \( l \)-dimensional current state vector \( X_{\text{now}} \),

\[
X_{\text{now}} = (x(n-l+1), x(n-l), x(n))
\]  

And the \( l \)-dimensional historical state vector is \( X_{\text{past}}(i) \)

\[
X_{\text{past}}(i) = (x_{\text{past},i}(n-l+1), x_{\text{past},i}(n-l), x_{\text{past},i}(n)), \quad i = 1, 2, ..., m
\]  

\( m \) is the total number of historical state vector which involved in the calculation

Step 2: When \( 1 \leq i \leq m \), calculate the Euclidean distance \( D(i) \) between \( X_{\text{past}}(i) \) and \( X_{\text{now}} \). Define the corresponding subscript of distance \( D(i) \) is \( D\text{tag}(i) \). \( D\text{tag}(i) = i \).

Step 3: Sort \( D(i) \) from small to large and select the first \( M \). The sorted \( D(i) \) is \( D(D\text{tag}(i)), 1 \leq i \leq M \)

Step 4: In order to ensure that historical state vector has the same trend with the current state vector, define state mode vector to storage the trend. The state mode vector configuration method is as follows:

Take Sequence \( X \) as an example. When \( 1 \leq j \leq n-1 \), set

\[
d(i) = \begin{cases} 
0, & x(j) > x(j+1) \\
1, & x(j) < x(j+1), \quad j = 1, 2, ..., n-1 \\
2, & x(j) = x(j+1)
\end{cases}
\]  

(5)
The state mode vector is $P = (d(1), \ldots, d(n-1))$

Follow the steps described above, building $l$-dimensional mode vector $P_{now}$ of the current state vector

$$P_{now} = (d_{now}(n-l), \ldots, d_{now}(n-1))$$  \hspace{1cm} (6)

Similarly, build $l$-dimensional historical state mode vector $P_{past}(i)$

$$P_{past}(i) = (d_{past,i}(n-l), \ldots, d_{past,i}(n-1)) \quad 1 \leq i \leq m \hspace{1cm} (7)$$

Step 5: Calculate the Euclidean distance $H(i)$ between the historical state mode vector and the current state mode vector.

Step 6: Sort $H(i)$ from small to large and select the first $K$ of the historical state vectors involved in forecasting. The selected vectors are

$$X_{past}(Dtag(Htag(1))), X_{past}(Dtag(Htag(2))), \ldots, X_{past}(Dtag(Htag(k)))$$  \hspace{1cm} (8)

Step 7: Using the next value $X_{now}'(Dtag(Htag(i))) = (X_{past,Dtag(Htag(i))}(n+1),1 \leq i \leq k$ of $X_{past}(Dtag(Htag(i))),1 \leq i \leq k$ and their corresponding Euclidean distance $D(Htag(i)),1 \leq i \leq k$ to calculate the final result $X_{now}'(n+1)$

$$X_{now}'(n+1) = \sum_{i=1}^{k} \frac{D(Htag(i)) \ast X_{past}'(Dtag(Htag(i)))}{\sum_{i=1}^{k} D(Htag(i))}, \sum_{i=1}^{k} D(Htag(i)) \neq 0 \hspace{1cm} (9)$$

When $\sum_{i=1}^{k} D(Htag(i)) = 0$, 

$$X_{now}'(n+1) = \frac{\sum_{i=1}^{k} X_{past}'(Dtag(Htag(i)))}{k} \hspace{1cm} (10)$$

2.3. The process of creating historical sample database

Traffic state historical sample database is a collection of the historical traffic data. Only the historical samples cover almost all possible traffic status, the algorithm can get enough samples when searching in order to get best result.

But with the time passing by, the amount of historical samples increases. This wastes a lot of storage space and decrease the search efficiency. So, in this section, we will focus on the construction method of the historical sample database. The historical sample database must completeness and typical. It does not affect the search efficiency of the forecasting algorithm and meets the real-time requirements.

The process of creating historical sample database as shown below
Concrete steps to create a historical sample library as follows

1) Pretreat the data
   - Exclude error data
   - Fill missing data

2) Establish the historical sample database
   The establishment of the historical sample database dependents on the dimensions choice of the current state vector. Concrete steps:
   - Select vector U from historical data. Put it into the historical sample database. Then, select another vector V.
   - Calculate the Euclidean distance D between U and V
   - If D is less than the threshold value Y, then V is the repeat state vector. If not, put V into the historical sample library.
   - Loop calculating till all the historical data are dealt with.

3) Establish the sample center database
   The efficiency of two-tier K-nearest neighbor algorithm depends on the database size and the storage structure of the traffic state historical sample database. In order to improve the computing speed of the algorithm, cluster the historical sample database and store the cluster center in the sample center database. Because the short-term forecasting algorithm has high requirements on the computing speed, take the grid-based method to cluster.

4) Judge the intensity of each cluster
   In traffic historical sample database, the cluster intensity is different. In the high intensity areas, if the value of k is relatively small, it will lose a certain number of similar nearest neighbor. This will reduce the forecasting accuracy. In the low intensity areas, if the value of k is large, the non-nearest neighbor data will be contained in. This will also reduce the forecasting accuracy [5]. In order to avoid such a situation, define the intensity M to ensure the completeness of the historical sample library.

   Intensity M: The minimum number of the historical state vector contained in the clustering grid.
   Only when the sample size meets the intensity requirement, the historical sample database is complete. If the sample size can not meet the intensity requirements, change the K value to avoid the interference of the non-nearest neighbor.
3. Simulation Analysis

In this paper, based on the two-tier K-nearest neighbor algorithm, studied the short-term forecasting for speed. Simulation data collected by RTMS in Beijing 3rd ring Rd. Using data from 2012-6-4 to 2012-6-10 to build historical sample database and the time interval was 5 minutes. In order to improve the computing speed, chose grid clustering to cluster. Then, implemented the algorithm with C# and verified the algorithm with data from 2012-6-11 06:00:00 to 2012-6-11 23:59:59. Use MRE (Mean Relative Error) to evaluate and compare simulation results:

\[
MRE = \frac{1}{N} \sum_{i=1}^{N} \frac{|X_i(t+1) - \hat{X}_i(t+1)|}{X_i(t+1)}
\]

(3)

\(X_i(t+1)\) is the actual value of the next time. \(\hat{X}_i(t+1)\) is the forecasting value of the next time. \(N\) is the number of samples.

Set current state vector is \((x_i-2, x_i-1, x_i)\). Choose different K to forecast. The corresponding MRE are as shown below:

<table>
<thead>
<tr>
<th>K</th>
<th>MRE (%)</th>
<th>K</th>
<th>MRE (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>6.726251</td>
<td>10</td>
<td>6.151915</td>
</tr>
<tr>
<td>3</td>
<td>6.3492</td>
<td>11</td>
<td>6.13569</td>
</tr>
<tr>
<td>4</td>
<td>6.302381</td>
<td>12</td>
<td>6.137661</td>
</tr>
<tr>
<td>5</td>
<td>6.259648</td>
<td>13</td>
<td>6.146449</td>
</tr>
<tr>
<td>6</td>
<td>6.214307</td>
<td>14</td>
<td>6.153</td>
</tr>
<tr>
<td>7</td>
<td>6.214266</td>
<td>15</td>
<td>6.1635</td>
</tr>
<tr>
<td>8</td>
<td>6.1938</td>
<td>16</td>
<td>6.17001</td>
</tr>
<tr>
<td>9</td>
<td>6.18279</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Fig 3 The trend of the MRE in different K](image)
Thus, as $K$ increases, MRE gradually reduced. When $K=11$, the MRE of the data collected in whole 3rd Ring Rd is minimum. Then, the MRE curve appeared slow upward trend. So compared with the value of $K=11$, there is no significant change. Considering the calculation speed and the prediction accuracy, we finally choose $k=11$ in order to get the best forecasting results.

When $k=11$, select 3044 detector to do further analysis. The position of the 3044 detector as shown below:

![Fig. 4. The position of the 3044 detector](image)

The calculated results contrast as follows:

![Fig. 5. The calculated results when k=2, 11, 16](image)
As can be seen from the figure, comparing with other values, when \( K=11 \), the forecasting value is closer to the actual value. The final forecasting results show that the two-tier K-nearest neighbor algorithm has higher precision. It is suitable for traffic flow forecasting.

4. Conclusion

Based on the K-nearest neighbor algorithm, this paper proposes two-tier K-nearest neighbor algorithm. With the actual data, verifies the practicality of the algorithm. The analysis result shows that the algorithm can meet the real-time, accuracy and reliability of the short-term traffic flow forecasting. In the future, we will improve the algorithm in the following areas:

1) Changing the dimension of the state vector, verify in what dimension, algorithm is most accurate.
2) Improve the efficiency of the algorithm. To meet the Real-time requirements of the short-term traffic flow forecasting.
3) Constantly improve the historical sample database, to improve the algorithm accuracy.

Acknowledgements

The authors would like to acknowledge the National High Technology Research and Development Program of China (863 Program), since this work was supported by the Program under Grant No. 2011AA110302.

References


