2015 IEEE 18th International Conference on Intelligent Transportation Systems

Towards Robust and Accurate Traffic Prediction Using Parallel Multiobjective Genetic Algorithms and Support Vector Regression

Jiri Petrlik
Brno University of Technology
Faculty of Information Technology
IT4Innovations Centre of Excellence
Bozetechova 2, 612 66, Brno, Czech Republic
Email: ipetrlik@fit.vutbr.cz

Lukas Sekanina
Brno University of Technology
Faculty of Information Technology
IT4Innovations Centre of Excellence
Bozetechova 2, 612 66, Brno, Czech Republic
Email: sekanina@fit.vutbr.cz

***Abstract*—The support vector regression (SVR) is a very successful method in solving many difficult tasks in the area of traffic prediction. However, the performance of SVR is very sensitive to the parameters setting and the selection of input variables such as sensors providing the input data. In this paper, we describe a new method, which simultaneously optimizes the meta-parameters of SVR model and the subset of its input variables. The method is based on a multiobjective genetic algorithm. The proposed implementation is intended for a parallel environment supporting OpenMP. We evaluated the method in the tasks of data imputation, short term prediction of traffic variables and travel times prediction using real world open data. It was confirmed that the simultaneous optimization of SVR parameters and input variables provides better quality of prediction than previous methods.**

I. Introduction

The machine learning methods like neural networks or support vector regression are very successful in solving many difficult tasks in the area of traffic prediction. However, performance of machine learning methods is very sensitive to the meta-parameters settings. It is usually necessary to properly set various meta-parameters, which can be very hard and require a lot of effort. The selection of proper inputs for these methods is also a difficult task. The input variables usually come from traffic sensors, cameras or groups of probe vehicles. In real world traffic applications there are many of these candidate variables. In our previous work, we focused on the selection of proper subsets of traffic variables. We have shown that using a proper subset of the input variables for the support vector regression model, it is possible to obtain more accurate prediction which is possible even if there are many missing values in the input data. We verified these methods of feature selection in the tasks of prediction of volume, occupancy [1] and travel times [2].

One of the biggest drawbacks of our methods is that the meta-parameters of SVR remain constant for all SVR models. We set these constants according to our previous experience. However, it is not possible to find one setting of these meta-parameters that will be suitable for all SVR models. This setting should be optimised for each SVR model. In this paper, we describe a new method, which simultaneously optimizes the meta-parameters of SVR and the subset of its input variables. We evaluated the quality of our new method in three different traffic prediction tasks. The first one is the data imputation task in which the goal is to estimate missing values in traffic data. Missing values can be caused by a broken sensor, data transmission errors or other failures. The second prediction task is a short time forecasting of traffic volume and occupancy. These two variables are very useful to describe the current traffic situation. They can be utilized to autonomously modify signal plans or directly inform the drivers. The third prediction task is the estimation of travel times, in which the goal is to predict the travel time of vehicle. This information can be useful for drivers in the case they have more options how to get to the desired destination. There exist various methods for estimation of travel times. Some of them are based on traffic cameras. These cameras are connected to the text recognition software, which is capable of reading the license plates of vehicles. The differences between the time of vehicle detection at the beginning of the road segment and the end of the road segment can serve for estimation of travel time [3], [4]. Other methods are based on the regression principle. The regression models like neural networks, SVR or others can predict the travel time using traffic variables like volume, occupancy or speed [5], [6]. In our previous work, we proposed a method, which combines these two principles [2].

Intelligent transportation systems usually gather a lot of information obtained by various sensors, cameras and other data acquisition systems. It is often necessary to process gigabytes of data and predict many traffic variables in a reasonable time. The question is how to speed up algorithms to deal with these big data. It is the main reason why we parallelize our new method. Parallel methods are designed to be capable of spreading the computation among many processor cores or even many processors. This leads to much faster computation.

The rest of the paper is organized as follows. Section II provides the basic information about support vector regression and its meta-parameters. Section III describes multi-objective optimization, the multimodal NSGAII algorithm and how it is beneficial for solving real world problems. Section IV deals with our method for traffic variables prediction and describes how it differs from previously proposed methods. The results are given in Section V. Section VI provides the conclusions.

978-1-4673-6596-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ITSC.2015.360

II. Support vector regression

A. Basic description

The machine learning methods can be divided into two groups, the supervised methods and unsupervised methods. The supervised learning expects that a reasonable portion of data is available with known results. These data are used to train a model. After the model is trained, it can be used for unknown data. We will expect the training data in the form $\{(\vec{x}_1, y_1), \ldots, (\vec{x}_l, y_l)\} \subset \mathcal{X} \times \mathbb{R}$. Here $\vec{x}_i$ represents the vector of values of independent variables ($\vec{x} \in \mathcal{X}$, $\mathcal{X} \subseteq \mathbb{R}^n$) and y_i represents the value of dependent variable (desired output). As the prediction of traffic is a regression task, y is a continuous variable. The training of SVR can be considered as an optimization problem [7]. The goal of this optimization problem is to find a function $f(x)$ such that it has at most ε deviation from the correct output y_i for the given training vector $\vec{x}_i$. At the same time we want this function as flat as possible. In the case of SVR, the flatness means the minimization of vector $\vec{w}$. More precisely it, means minimization of $\frac{1}{2}||\vec{w}||^2$. For many tasks, however, this problem can be infeasible. It means there is no such $\vec{w}$, for which the deviation from the correct y_i is at most ε. Because of it, the so called "soft margin loss function" was introduced. New *slack variables* ξ and ξ^* are used to convert the infeasible problem into the feasible one. The precise formulation of SVR is shown in equation 1.

$$\begin{aligned} \text{minimize:} \quad & \tfrac{1}{2}||\vec{w}||^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*) \\ \text{subject to:} \quad & \begin{cases} y_i - < \vec{w}, \vec{x}_i > -b & \leq & \varepsilon + \xi_i \\ < \vec{w}, \vec{x}_i > +b - y_i & \leq & \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq & 0 \end{cases} \end{aligned} \tag{1}$$

Here $< \vec{w}, \vec{x}_i >$ means a dot product of vector $\vec{w}$ and $\vec{x}_i$. The regularization meta-parameter $C > 0$ controls the trade-off between the flatness and the amount of deviation larger than ε. The situation is depicted in Figure 1. Here the samples from the training set are marked as small crosses and the regression function is shown as the bold line. The permitted deviation is depicted by two thin lines. However some points from the training set don't lie in the permitted area. The sizes of slack variables ξ and ξ^* for these points are also shown in the Fig. 1.

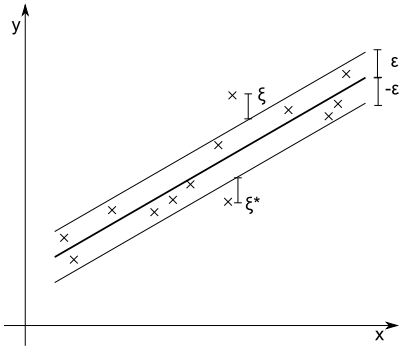

Fig. 1. Linear support vector regression.

B. Kernel functions

The linear SVR can be very useful, but many regression tasks don't have a linear character. To deal with this problem, the transformation of independent variables into more dimensional space is commonly used. To do this we need a proper function in the following form $\Phi : \mathcal{X} \rightarrow \mathcal{F}$, where $\mathcal{F}$ has more dimensions than $\mathcal{X}$. In the case of SVR, these mapping functions are called the kernels. In the past, several kernels were proposed, including polynomial kernel (eq. 2), radial kernel (eq. 3) and sigmoid kernel (eq. 4) [7].

$$K(x, y) = (x.y + 1)^p \tag{2}$$

$$K(x, y) = e^{\frac{-||x-y||^2}{2\sigma^2}} \tag{3}$$

$$K(x, y) = tanh(\kappa x.y - \delta) \tag{4}$$

C. Meta parameters

In order to obtain high quality predictions, it is necessary to properly select the kernel function and set various parameters, where regularization coefficient C is the most important one. If this coefficient were set too strictly, the problem called *overfitting* could occur. The *overfitting* appears when the training algorithm tries to find a very complex model, which is able to predict almost all values from a training set set very precisely. However, this complex model does not usually generalize well and have a quite high error for new data. The opposite problem, called *underfitting*, can appear if the regularization coefficient is set too freely. In the case of underfitting, the model is too simple to be able to predict desired variable.

III. Multiobjective evolutionary optimization

A. Multiobjective optimization

In the real world, many problems can not be sufficiently solved using only a single objective optimization algorithms, simply because it is impossible to define the quality of a candidate solution by only one objective function. Fortunately, modern multi-objective methods can consider all these criteria in a single run of the optimization algorithm. In order to compare the quality of two candidate solutions, the multi-objective optimization algorithms use the Pareto-dominance relation. According to this relation, solution a Pareto-dominates solution b, if solution a is better in at least one objective, and the solution a is not worse than b in all objective functions. Among the whole set of all possible solutions, there exists a subset of solutions called Pareto-optimal solutions. These solutions are not dominated by any other possible solution and are targets for the multi-objective optimization algorithms. The imaginary line connecting Pareto-optimal solutions is called Pareto-front and the goal of the multi-objective evolutionary optimization is to find various trade-off Pareto-optimal solutions, which are spread along the whole Pareto-front [8].

B. Multimodal NSGAII

One of the popular algorithms for multi-objective optimization is called NSGAII [9]. At the beginning this algorithm creates a set of randomly generated solutions. This set is called the parent population P_t. Each solution is represented by a string of bits called the chromosome. The quality of these

candidate chromosomes is evaluated using objective functions. Then the crossover and mutation operations are performed on P_t to create a set of new, potentially better solutions Q_t. These two subpopulations are merged together ($P_t \cup Q_t$) and good solutions are chosen to new population P_{t+1}. This process is repeated until the predefined number of iterations is reached.

A very important part of this process is the selection of solutions, which will be accommodated into the new population (P_{t+1}). The NSGAII algorithm utilizes the approach called non-dominated sorting. In this approach, solutions from union $P_t \cup Q_t$, which are not dominated by any other solution are accommodated into new population P_{t+1}. If new population P_{t+1} is not filled, new search for non-dominated solutions in $P_t \cup Q_t$ is performed. Now it is without the already selected solutions. New non-dominated solutions are again accommodated into new population P_{t+1}. This process continues until the new population P_{t+1} is not filled. When the last portion of solutions is accommodated, two potential situations can appear. First, the number of non-dominated solutions is equal to the number of remaining slots in new population P_{t+1}. In this case, all non-dominated solutions are accommodated. Otherwise, the number of non-dominated solutions is higher than the number of slots. In this case, solutions are selected using crowding distance sort, which tries to select such solutions which are widely spread along the whole Pareto front.

In the applications considered in this paper, it is very useful to obtain multiple different solutions with the same quality. The reason is that one solution (e.g. predictor) can be replaced by another if the first solution can not be used because required data are not available due a sensor failure. We call these solutions the multimodal solutions. In the context of the multi-objective optimization, two solutions are multimodal, if both have the same values of objective functions (i.e. $f_i(a) = f_i(b), \forall i = 1 \dots N$). Here $f_i(x)$ is a value of objective function i for the solution x and N is the number of optimized objectives. The modification of NSGAII capable of finding these multimodal solutions is called multimodal NSGAII [10]. The main difference is in the accommodation of new solutions into the new parent set. The multimodal NSGAII avoids of accommodating of duplicate solutions into the new population.

C. Binary genetic algorithms

The crossover operator is able to create a new candidate solution by combining two or more previously found solutions. The uniform crossover needs two parent chromosomes p_1 and p_2 and produces two children chromosomes c_1 and c_2. At first a random binary mask with the same size as parent chromosomes is generated. If the value for a given item in the mask is equal to zero, the item in c_1 will be the same as in p_1 and item in c_2 will be the same as in p_2. Otherwise the item in c_1 will be equal to the item in p_2 and the item in c_2 will be equal to the item in p_1 [11]. The Bernoulli distribution is often used to generate the mask. Bernoulli probability distribution takes value 1 with probability r and value 0 with probability $q = 1 - r$. The binary mutation generates one binary value for each gene in the chromosome according to the Bernoulli distribution. The mask is then used to decide which values in the chromosome will be changed [11].

D. Real valued genetic algorithms

In the case of chromosomes consisting of continuous values another type of crossover should be used. One of frequently used types of crossover for these chromosomes is called Simulated Binary Crossover (SBX) [12]. The SBX uses the specially defined probability distribution $P(\beta_i)$ for generating of values in children chromosomes (Equation 5). Here, parameter η_c is a non-negative number. If η_c is large, the value in the child chromosome will be near the parent value. If η_c is small, the generated value will be very different from the value in parent. The process of creating new solutions by SBX is as follows. At the beginning a random number $u_i \in [0, 1]$ is generated for each gene i. After that, the ordinate β_{q_i} is taken from the probability distribution P. The ordinate B_{q_i} is defined such that the area under the probability curve is equal to the randomly generated number u_i. The value of B_{q_i} is calculated by the equation 6.

$$P(\beta_i) = \begin{cases} 0.5(\eta_c + 1)\beta_i^{\eta_c}, & \text{if } \beta_i \leq 1 \\ 0.5(\eta_c + 1)\frac{1}{\eta_c + 2}, & \text{otherwise.} \end{cases} \quad (5)$$

$$\beta_{q_i} = \begin{cases} (2u)^{\frac{1}{\eta_c+1}}, & \text{if } u_i \leq 0.5 \\ \left(\frac{1}{2(1-u_i)}\right)^{\frac{1}{\eta_c+1}}, & \text{otherwise.} \end{cases} \quad (6)$$

Then, the obtained ordinate B_{q_i} is used to create a new children. New values of gene i in the offspring are calculated using equations 7 and 8.

$$x_i^{(1,t+1)} = 0.5\left((1 + \beta_{q_i})x_i^{(1,t)} + (1 - \beta_{q_i})x_i^{(2,t)}\right) \quad (7)$$

$$x_i^{(2,t+1)} = 0.5\left((1 - \beta_{q_i})x_i^{(1,t)} + (1 + \beta_{q_i})x_i^{(2,t)}\right) \quad (8)$$

E. Parallel genetic algorithms

Current processors contain many computational cores. This can significantly increase the performance of the machine and lower the time of computation. However, to obtain this higher performance, it is necessary to use algorithms capable of spreading the computational effort among many computational units. Fortunately there exist many approaches to parallelize genetic algorithms. The basic approach is known as the master-slave model. In this case the master process performs all operators such as selection, crossover and mutation. The only part of the algorithm which is conducted in parallel, is the evaluation of the quality of candidate solutions. In this process, chromosomes of candidate solutions are distributed to the computational units (workers), which evaluate the quality of solutions and return computed values of fitness functions back to the master [13]. Another approach is the multiple-population or island model, in which each computational unit has its own population of candidate solutions and performs all genetic operators on its own. After a predefined number of generations the most promising solutions are exchanged among the computational units simulating thus a migration [13].

IV. METHOD

A. Multiobjective search

In our previous work, we have shown that it is possible to use a multi-objective optimization to find many SVR models for prediction of traffic variables. These SVR models differ in the set of input variables that they utilize. We simultaneously optimized three objective functions: (1) the quality of prediction, which is calculated as root mean squared error (RMSE); (2) the number of input variables and (3) the portion of time, for which one of the input variables is missing. At the end we, for example, obtained SVR model, which provides very precise prediction but uses many input variables. However, this SVR is often useless because one of the values of input variables is often missing. On the other hand, we obtained a model which is less precise, but uses only a few input variables. We also obtained many compromise solutions between these two extremes. The main reason for multiple different SVRs is that we can choose a model for which the input variables are currently available. This is very usefull because traffic variables are often missing, because of various reasons, for example, broken detectors, data transmission errors etc.

We propose a method to dynamically switch the prediction model according to available data. If it is possible, we use the model with the best RMSE (minimal error). If one of desired input variables is missing, we use corresponding SVR with the second best RMSE. The process continues until we find SVR, which can be used or we run out of pre-trained SVRs [1], [2]. In our previous method, we optimized only the set of input variables, but the meta-parameters of SVR remained unchanged. However, this is far from the optimum, because for each SVR the optimal settings of the meta-parameters is different. In this work, we try to improve prediction results by simultaneous search for the optimal data inputs for SVR and the optimal meta-parameters in one run of the multiobjective-genetic algorithm.

B. Optimization of SVR Metaparameters

Two types of kernels are supported in our work: linear SVR and SVR with radial basis kernel. For linear SVR it is necessary to optimize only a regularization coefficient C. The radial kernel requires, in addition to C, to optimize the kernel parameter gamma. The whole chromosome is divided into two parts. The first part contains the information about used input variables. Each gene represents one potential input variable. If the value of gene is equal to one (true), the input variable will be used as the input for SVR model. Otherwise, the corresponding input variable will not be used. The binary part of the chromosome contains one additional bit, which defines the type of kernel (0 – the linear kernel; 1 – the radial kernel). We use uniform crossover and bit flip mutations to modify this part of the chromosome. The second part of the chromosome consists of real values, which are devoted to SVR meta-parameters. The first real value defines the value of regularization coefficient in the case that linear SVR is used. In this case the value of regularisation coefficient is equal to $2^{C_{linear}}$. The second value represents the value of regularization coefficient in the case that radial kernel is used. In this case the regularisation coefficient is equal to $2^{C_{radial}}$. Finally, the third real value is for gamma parameter ($\gamma = 2^{gamma}$). We use SBX crossover and normally distributed mutations to modify the real valued part of the chromosome. The schema of the whole chromosome is depicted in Figure 2.

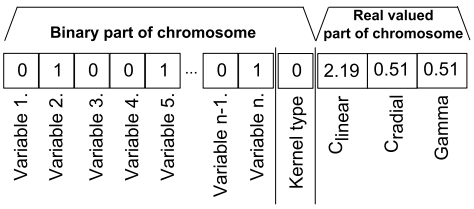

Fig. 2. Chromosome scheme.

C. Speedup by parallel implementation

One of the disadvantages of genetic algorithms is that it is necessary to evaluate the quality of a huge number of candidate solutions during the optimization process. This can be very time consuming. However, as it was mentioned earlier, it is possible to accelerate this process by parallel implementation. In our work we utilized the master-slave principle. Each computational unit, which is one processor core in our case, computes the values of fitness functions for some portion of the population. The parallel implementation is written in R language for statistical computation. We used "foreach" and "doMC" libraries. The library "doMC" is based on OpenMP technology and allows us to run fitness function evaluation on multiple processor cores. The only parallel part of our algorithm is the loop with evaluation of new candidate solutions. The main problem is a different time needed for evaluation of candidate solutions. These differences are undesirable in parallel loops because of synchronization problems. Hence we investigated the influence of SVR meta-parameters and the influence of the number of SVR inputs on the solution evaluation time. We measured the dependency between the value of regularization coefficient C and the time of SVR training for prediction of traffic variables. The results are shown in figure 3. It can be seen that the time is rapidly growing if the regularization coefficient is higher than 2^3 for linear kernel and 2^7 for radial kernel. Because of it we restricted the values in the chromosomes in the following manner. The C_{linear} gene is restricted to be within the interval $[-5, 3]$, C_{radial} in $[-5, 7]$ and gamma in $[-15, 3]$.

V. RESULTS

In order to evaluate the proposed method, we used publicly available data from Seattle (www.its-rde.net). These data are provided by the Research Data Exchange project and can be downloaded from the Internet. The Seattle data consist of datasets. In this work, we mainly focus on Seattle sensys data [14] and arterial travel times [15]. The Seattle sensys data provides the values measured on traffic detectors places on 23 intersections. This data set contains the values of traffic volume, occupancy and mean speed. The arterial travel times data which are measured by cameras and license plate recognition system contains the information about the time of vehicles trips. In order to simplify the prediction we aggregated available data into five minute intervals. We used the data from July 2011 for experiments with data imputation and short term prediction. The data from June 2011 were utilized for

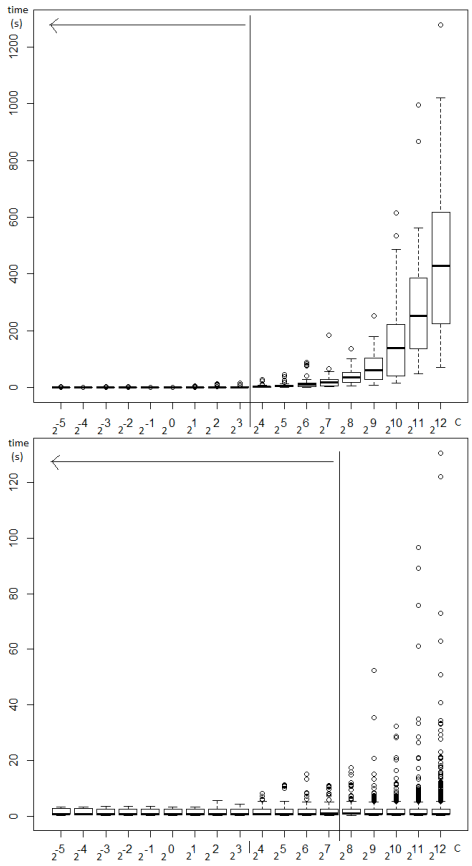

Fig. 3. Dependency between the value of regularisation coefficient (C) and the time of training: linear kernel (top) and radial kernel (bottom).

travel time prediction. The first 15 days are considered as training set and the rest is considered as test set. We used the model switching described in the previous section. The values of traffic volume are given in the number of vehicles per 5 minutes and occupancy is the portion of time for which the given place on the road is occupied by vehicles. The travel times are in seconds.

We used the same setting of the evolutionary algorithm NSGAII for each experiment. We performed 25 independent runs, each of them counting 100 generations with the population size of 40 chromosomes. Other parameters are: probability of binary crossover 70%, probability of binary mutation 1%, $\eta_c = 2$, and $\delta = 1$.

A. Data imputation

The data imputation is a process, in which we try to estimate missing values in data. The method was evaluated on the imputation of traffic volume and occupancy data on four intersections (11, 19, 22 and 23) in the dataset. For each intersection we test four sensors with the biggest mean value. Table I shows a comparison with the method which optimizes only the inputs of SVR [1]. The RMSE for method, which optimizes only the inputs, is given in the column "RMSE – Only feature selection", while the RMSE for our new method, is shown in column "RMSE – Meta-parameters". The volume is measured as the number of vehicles per 5 minutes and the occupancy as a portion of time in which the current place is occupied by vehicle. The RMSE corresponds to this variables. The improvement reached by our new method is shown in the last column. The mean improvement for data imputation is 2.29 %.

TABLE I. RMSE IN DATA IMPUTATION

Place Num.	Sensor Num.	Variable Type	Sel. RMSE Only Feature	RMSE Meta-param.	Improvement (%)
11	1	occupancy	5.81	5.42	6.73
11	17	occupancy	4.96	4.80	3.14
11	3	occupancy	5.28	4.53	14.06
11	4	occupancy	11.95	11.92	0.28
19	3	occupancy	3.09	2.90	6.20
19	4	occupancy	4.42	4.45	−0.54
19	8	occupancy	8.95	8.70	2.76
22	1	occupancy	3.86	3.69	4.59
22	3	occupancy	4.18	3.94	5.57
22	6	occupancy	2.49	2.49	0.00
22	8	occupancy	3.18	3.00	5.84
23	1	occupancy	11.92	11.72	1.68
23	2	occupancy	8.67	8.40	3.08
23	3	occupancy	10.41	10.29	1.09
23	7	occupancy	8.46	8.40	0.68
11	1	volume	4.94	4.90	0.67
11	13	volume	4.31	4.29	0.48
11	17	volume	3.74	3.65	2.45
11	3	volume	5.62	5.60	0.38
19	3	volume	5.80	5.98	−3.09
19	4	volume	6.42	6.42	0.00
19	7	volume	5.74	5.49	4.33
19	8	volume	4.77	4.78	−0.30
22	1	volume	3.92	3.84	1.99
22	4	volume	5.16	5.32	−3.17
22	7	volume	4.71	4.64	1.55
22	8	volume	4.38	4.32	1.33
23	6	volume	4.11	4.11	0.00
23	7	volume	6.36	6.19	2.66
23	8	volume	6.40	6.30	1.56
23	9	volume	4.53	4.31	5.04

B. Short Term Prediction of Traffic Variables

Table II summarizes the results for short time traffic prediction. Mean values calculated from 25 independent runs of NSGAII are reported. The format of the table is the same as for the data imputation. We predicted the values of these variables with the prediction horizon of 15 minutes. The mean improvement against [1] is 26.35 %.

C. Travel times prediction

The estimated travel time is a very useful information for drivers. This information can be shown by boards near the road or available through cellphones or other mobile devices. We try to further improve it by the optimization of meta-parameters of SVR. In Table III, "place begin" is the identifier of the start of road segment and "place end" is the identifier of the end of the segment. The column "use history" informs whether only current measured values were used for prediction (-), or a short history (15 min.) was also considered. The proposed method was compared with the method, which optimizes only the subset of input sensors [2] (column "Only Feature"). It can be seen in column "Meta-param" that our method provides only a small improvement in this case (the mean is 0.32 %).

TABLE II. RMSE FOR SHORT TERM TRAFFIC FORECASTING WITH PREDICTION HORIZON 15 MINUTES.

Place Num.	Sensor Num.	Variable Type	Sel. RMSE Only Feature	RMSE Meta-param.	Improvement (%)
11	1	occupancy	4.18	3.62	13.40
11	17	occupancy	3.65	3.20	12.34
11	3	occupancy	2.47	1.68	32.06
11	4	occupancy	9.62	9.02	6.31
19	3	occupancy	1.59	0.98	38.34
19	4	occupancy	2.49	1.62	34.93
19	8	occupancy	5.84	4.00	31.48
22	1	occupancy	2.42	1.99	17.42
22	3	occupancy	2.38	1.81	23.94
22	6	occupancy	1.13	0.53	53.14
22	8	occupancy	1.87	1.24	33.86
23	1	occupancy	7.11	6.04	15.04
23	2	occupancy	6.50	5.82	10.46
23	3	occupancy	8.02	6.24	22.11
23	7	occupancy	4.54	3.91	13.89
11	1	volume	1.51	1.24	17.97
11	13	volume	2.12	1.46	31.38
11	17	volume	1.98	1.26	36.39
11	3	volume	1.75	1.48	15.44
19	3	volume	3.19	2.15	32.41
19	4	volume	3.21	1.89	41.17
19	7	volume	3.15	2.27	28.06
19	8	volume	2.97	1.73	41.79
22	1	volume	1.84	1.36	25.90
22	4	volume	2.16	1.30	39.84
22	7	volume	2.29	1.56	31.84
22	8	volume	1.91	1.27	35.85
23	6	volume	1.80	1.39	22.58
23	7	volume	2.54	2.14	15.73
23	8	volume	2.67	2.17	18.85
23	9	volume	1.59	1.23	23.04

TABLE III. RMSE FOR TRAVEL TIMES ESTIMATION.

Place begin	Place end	Use history	Sel. RMSE Only Feature	RMSE Meta-param.	Improvement (%)
58	46	-	57.19	56.99	0.34
58	46	15 min	57.36	57.11	0.43
7	58	-	27.11	26.99	0.44
7	58	15 min	26.12	26.09	0.08

D. Parallelization

In order to analyse the speedup provided by the parallel implementation we performed 10 runs of our method for short time traffic prediction using a different number of processor cores. We measured the number of GA iterations (generations) performed in a one hour run. Fig. 4 shows that the method scales well for 2-16 CPUs.

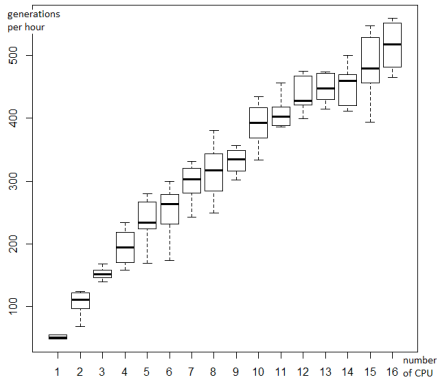

Fig. 4. Parallel speedup for short time traffic forecasting.

VI. CONCLUSIONS

In this paper, we proposed a new method for robust traffic prediction. This method is based on a multi-objective genetic algorithm and SVR. The main advantage against the other methods is that it simultaneously optimizes the input variables of SVR and SVR meta-parameters. We have shown that the method provides better prediction than methods optimizing only the set of input variables. The method is especially successful for short term traffic prediction. We also shown a parallel implementation to reduce the computational time.

ACKNOWLEDGMENT

This work was supported by the IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070 and Brno University of Technology project FIT-S-14-2297.

REFERENCES

[1] J. Petrlik, O. Fucik, and L. Sekanina, "Multiobjective selection of input sensors for svr applied to road traffic prediction," in *Parallel Problem Solving from Nature PPSN XIII*, ser. Lecture Notes in Computer Science, T. Bartz-Beielstein, J. Branke, B. Filipi, and J. Smith, Eds. Springer International Publishing, 2014, vol. 8672, pp. 802–811.

[2] ——, "Multiobjective selection of input sensors for travel times forecasting using support vector regression," in *Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2014 IEEE Symposium on*, Dec 2014, pp. 14–21.

[3] S. Takaba, T. Morita, T. Hada, T. Usami, and M. Yamaguchi, "Estimation and measurement of travel time by vehicle detectors and license plate readers," in *Vehicle Navigation and Information Systems Conference, 1991*, vol. 2, Oct 1991, pp. 257–267.

[4] K. Kanayama, Y. Fujikawa, K. Fujimoto, and M. Horino, "Development of vehicle-license number recognition system using real-time image processing and its application to travel-time measurement," in *Vehicular Technology Conference, 1991. Gateway to the Future Technology in Motion., 41st IEEE*, May 1991, pp. 798–804.

[5] C.-H. Wu, J.-M. Ho, and D. Lee, "Travel-time prediction with support vector regression," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 5, no. 4, pp. 276–281, Dec 2004.

[6] A. Dharia and H. Adeli, "Neural network model for rapid forecasting of freeway link travel time," *Engineering Applications of Artificial Intelligence*, vol. 16, no. 78, pp. 607 – 613, 2003.

[7] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[8] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001, vol. 16.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, Apr 2002.

[10] K. Deb and A. R. Reddy, "Reliable classification of two-class cancer data using evolutionary algorithms," *Biosystems*, vol. 72, no. 12, pp. 111 – 129, 2003, computational Intelligence in Bioinformatics.

[11] M. Gendreau and J. Potvin, *Handbook of Metaheuristics*, ser. International Series in Operations Research & Management Science. Springer US, 2010. [Online]. Available: http://books.google.cz/books?id=xMTS5dyDhwMC

[12] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 3, pp. 1–15, 1994.

[13] E. Cantú-Paz, "A survey of parallel genetic algorithms," *Calculateurs paralleles, reseaux et systems repartis*, vol. 10, no. 2, pp. 141–171, 1998.

[14] University of Washington Transportation Research Center, Research Data Exchange Website, Seattle Data Environment, datasets: Seattle Sensys data, Retrieved May 2013, from www.its-rde.net

[15] University of Washington Transportation Research Center, Research Data Exchange Website, Seattle Data Environment, datasets: Arterial Travel Times, Retrieved May 2013, from www.its-rde.net