Optimization of Support Vector Regression Models for Stormwater Prediction

**Patrick Koch, Wolfgang Konen,
Oliver Flasch, and Thomas Bartz-Beielstein**

Cologne University of Applied Sciences
E-Mail: {patrick.koch | oliver.flasch |
wolfgang.konen | thomas.bartz-beielstein}@fh-koeln.de

Abstract

In this paper we propose a solution to a real-world time series regression problem: the prediction of fill levels of stormwater tanks. Our regression model is based on *Support Vector Regression* (SVR), but can easily be replaced with other data mining methods. The main intention of the work is to overcome frequently occuring problems in data mining by automatically tuning both preprocessing and hyperparameters. We highly believe that many models can be improved by a systematic preprocessing and hyperparameter tuning. The optimization of our model is presented in a step-by-step manner which can easily be adapted to other time series problems. We point out possible issues of parameter tuning, e.g., we analyze our tuned models with respect to overfitting and oversearching (which are effects that might lead to a reduced model generalizability) and present methods to circumvent such issues.

1 Introduction

In environmental engineering stormwater tanks are installed to stabilize the load on the sewage system by preventing rainwater from flooding the system and by supplying a base load in dry periods. Heavy rainfalls are the most common reason for overflows of stormwater tanks, causing environmental pollution from wastewater contaminating the environment. To avoid such situations, the effluent of the stormwater tanks must be controlled effectively and possible future state changes in the inflow should be recognized as early as possible. This problem can be defined as a classical time series regression problem of predicting a stormwater tank fill level at time t from a fixed window of past rainfall data from time t back to time $t - W$ (for a fixed window size W) and will be referred to as the *stormwater problem* in the remainder of this paper.

A model that predicts fill levels by means of rainfall data can be an important aid for the controlling system. Special sensors (Fig. 1) record time series data which can be used to train such a model.

Although many methods exist for time series analysis [1], ranging from classical statistical regression to computational statistics, such methods often require time-consuming investigations on the hyperparameter selection and preprocessing of the data. Besides that, the results are often worse than special-purpose models which are designed from scratch for each new problem. This situation is of course very unsatisfying for the practitioner in environmental engineering, because new models have to be created and parameters have to be tuned manually for each problem.

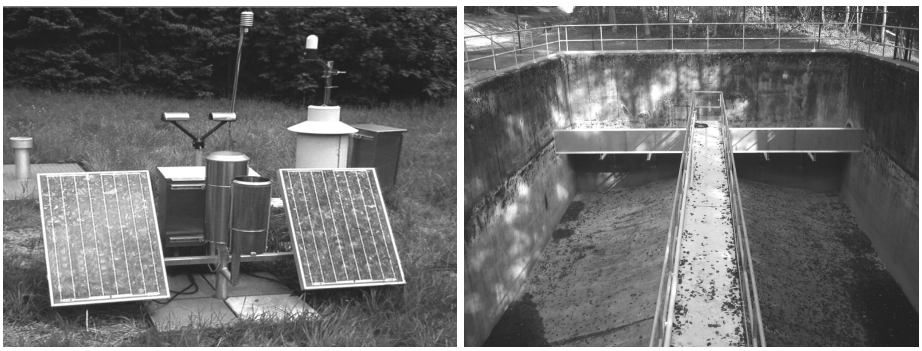

Figure 1: Left: rain gauge (pluviometer). Right: stormwater tank.

For this reason, it would be an advantage to have some standard repertoire of methods which can be easily adapted to new problems. In this paper we use Support Vector Machines (SVM) [2] for Support Vector Regression as a state-of-the-art method from machine learning and apply them to the stormwater problem. SVMs are known to be a strong method for classification and regression. However, it has to be noted that because of the time series structure of the data consecutive records are not independent from each other, as it is the case in normal regression. Therefore we investigate a generic preprocessing operator to embed time series data and to generate new input features for the SVM model. In addition, we apply the sequential parameter optimization toolbox (SPOT) [3] and a genetic algorithm (GA) to the preprocessed data, to find good hyperparameter settings for both preprocessing and SVM parameters. We analyze the robustness of our method against overfitting and oversearching of hyperparameters.

Previous work in stormwater prediction has been done by Hilmer [4], Konen *et al.* [5], Bartz-Beielstein *et al.* [6] and Flasch *et al.* [7]. A conclusion of these previous publications is that good results can be obtained with specialized models (which are 'hand-crafted' and carefully adapted to the stormwater problem). A first step towards a more generic model based on Support Vector Regression has recently been presented by Koch *et al.* [8]. It has been shown, that superior results can be achieved, if hyperparameters are tuned and time series preprocessing is taken into account. Therefore, we point out the main hypotheses of this paper:

H1 It is possible to move away from domain-specific models for time series prediction without loss in accuracy by applying modern machine learning algorithms and modern parameter tuning methods on data augmented through generic time series preprocessing operators.

H2 Parameter tuning for stormwater prediction leads to oversearching, yielding too optimistic results on the dataset during tuning.

Hypothesis **H2** puts emphasis on the fact that a distinction between validation set (used during tuning) and test set (used for evaluation) is essential to correctly quantify the benefits from parameter tuning in data mining. The oversearching issue is prevalent in data mining since the output function to tune shows often a high variance when the data used for training or tuning are changed. This effect is also shown for other benchmark problems in a work of Konen *et al.* [9] in the same book.

2 Methods

2.1 Stormwater Tank Data

Time series data for this case study are collected from a real stormwater tank in Germany and consists of 30,000 data records, ranging from April to August 2006. Rainfall data are measured in three-minute intervals by a pluviometer as shown in Fig. 1. All models described in this paper were trained on a 5,000 record time window (Set 2, Fig. 2) in order to predict another 5,000 record time window for testing (Set 4, Tab. 1).

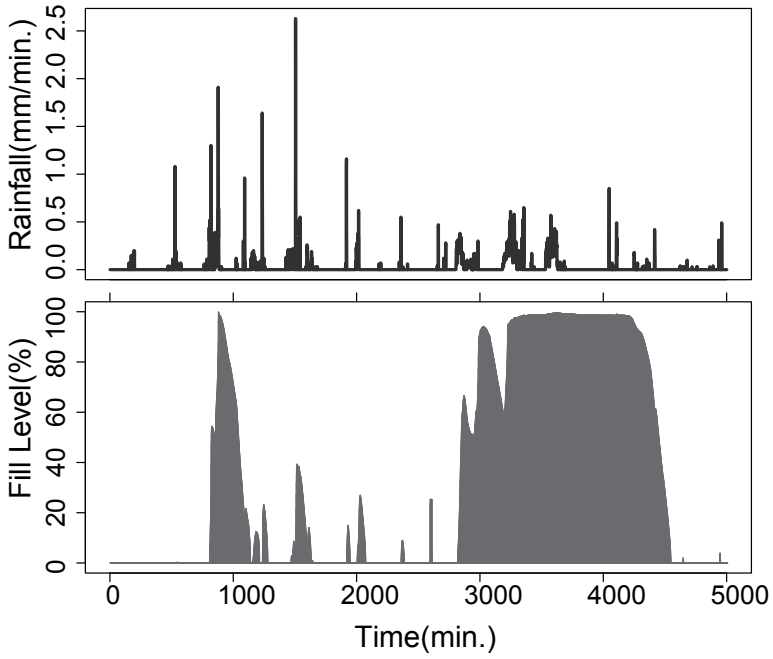

Figure 2: Training set showing rainfall and fill level time series data of the stormwater tank.

Table 1: Real-world time series data from a stormwater tank in Germany.

Set	Start Date	End Date
Set 1	2006-04-28 01:05:59	2006-05-15 09:40:59
Set 2	2006-05-15 09:40:59	2006-06-01 18:20:59
Set 3	2006-06-19 03:01:00	2006-07-06 11:41:00
Set 4	2006-07-23 20:21:00	2006-08-10 05:01:00

2.2 Evaluation of Models

The prediction error on the datasets is taken as objective function for SPOT and for the GA. For comparing models, we calculate the root mean squared error (RMSE) as a quality measure:

$$RMSE = \sqrt{(mean((Y_{predicted} - Y_{true})^2))} \tag{1}$$

We also incorporate the Theil's U index of inequality [10], where the RMSE of the trained model is compared to the RMSE of a naïve predictor. Here we are using the mean of the

training data target as a naïve predictor:

$$U = \frac{RMSE(model)}{RMSE(naive)} \tag{2}$$

U values greater than 1 indicate models that perform worse than the naïve predictor, while values smaller than 1 indicate models that perform better than the naïve predictor.

2.3 Sequential Parameter Optimization Toolbox

The main purpose of SPOT is to determine improved parameter settings for search and optimization algorithms and to analyze and understand their performance.

During the first stage of experimentation, SPOT treats an algorithm A as a black box. A set of input variables $\vec{x}$, is passed to A. Each run of the algorithm produces some output $\vec{y}$. SPOT tries to determine a functional relationship F between $\vec{x}$ and $\vec{y}$ for a given problem formulated by an objective function $f : \vec{u} \rightarrow \vec{v}$. Since experiments are run on computers, pseudorandom numbers are taken into consideration if:

- the underlying objective function f is stochastically disturbed, e.g., measurement errors or noise occur, and/or
- the algorithm A uses some stochastic elements, e.g., mutation in evolution strategies.

SPOT employs a sequentially improved model to estimate the relationship between algorithm input variables and its output. This serves two primary goals. One is to enable determining good parameter settings, thus SPOT may be used as a tuner. Secondly, variable interactions can be revealed for helping in understanding how the tested algorithm works when confronted with a specific problem or how changes in the problem influence the algorithm's performance. Concerning the model, SPOT allows for insertion of virtually any available model. However, regression and Kriging models or a combination thereof are most frequently used. The Kriging predictor applied in this study uses a regression constant λ which is added to the diagonal of the correlation matrix. Maximum likelihood estimation was used to determine the regression constant λ [11, 12].

2.4 The INT2 Model for Predictive Control of Stormwater Tanks

In previous work [6, 5], the stormwater tank problem was investigated with different modeling approaches, among them FIR, NARX, ESN, a dynamical system based on ordinary differential equations (ODE) and a dynamical system based on integral equations (INT2). All models in these former works were systematically optimized using SPO [3]. Among these models the INT2 approach turned out to be the best one [6]. The INT2 model is an analytical regression model based on integral equations. Disadvantages of the INT2 model are that it is a special-purpose model only designed for stormwater prediction and that it is practically expensive to obtain an optimal parameter configuration: the parameterization example presented in [6] contains 9 tunable parameters which must be set. In this paper we compare hand-tuned INT2 parameters with the best parameter configuration found by SPOT in former study [6].

2.5 Support Vector Regression

Support Vector Machines have been successfully applied to regression problems by Drucker *et al.* [2], Müller *et al.* [13], Mattera and Haykin [14], and Chan and Lin [15]. In these studies the method has been shown to be superior to many other methods especially when the dimensionality of the feature space is very large.

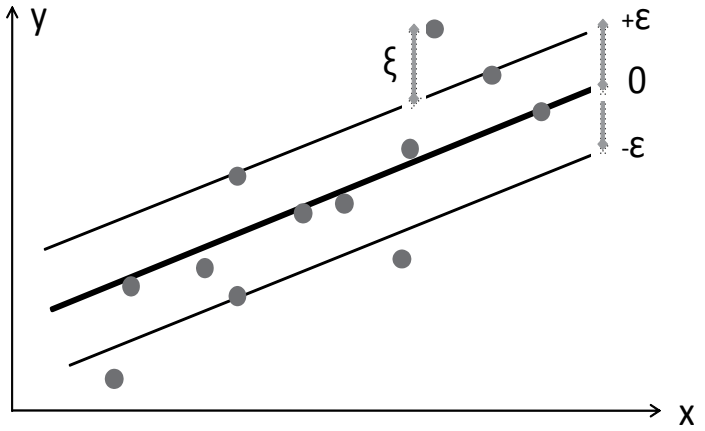

Figure 3: Example for Support Vector Regression. A tube with radius ϵ is learned to represent the real target function. Possible outliers are being regularized considering a positive slack variable ξ.

The idea of Support Vector Regression (SVR) is to transform the input space of the training data usually to a higher-order space using a non-linear mapping, e.g. a radial basis function. In this higher-order space a linear function is learned, which has at most ϵ deviation from the real values in general and at most ξ deviation for certain outliers. An example for Support Vector Regression and its parameters ϵ and ξ is depicted in Fig. 3. The transformation to a higher-order space by a non-linear kernel function with parameter γ is not shown here. For a more detailed description of SVR we refer to Smola and Schölkopf [16].

3 Preprocessing for Stormwater Prediction Models

Time series prediction models can benefit from preprocessing operators which generate new features based on the input data. There are two possibilities to integrate such operators into the SVM modeling process:

- Integration into the SVR kernel function, i.e. replacing standard kernel functions by kernel functions that incorporate preprocessing operators
- Direct preprocessing of the data, i.e. by augmenting the input feature set with results of preprocessing

In this work we choose the second approach, because the effect of this integration into a model is easier to analyze. In a first step, we compare the effects of applying different types of time series preprocessing operators on SVM model accuracy. Details on preprocessing

operators as the *embedding* operator and *leaky rain* as an integral operator suited for time series analysis are given in Sec. 3.2 and Sec. 3.3 respectively. All preliminary models built on the following varieties of input features were used in the following experimental setups to make them comparable:

E1 Predicting fill levels based only on current rainfall

E2 Predicting fill levels with embedding of rainfall

E3 Predicting fill levels with embedding of leaky rain and rainfall

E4 Predicting fill levels with embedding of leaky rain only

E5 Predicting fill levels with embedding of multiple leaky rain kernel functions

In our experiments we used the radial basis SVM kernel from the *e1071* SVM-implementation in R, since we achieved best results with this kernel choice. Other SVR hyperparameters were obtained by SPO, namely parameters γ, ϵ and ξ, to make our results comparable to each other. All models created for optimization of the preprocessing were trained on set 2 and evaluated on set 4 (see Tab. 1). As objective function for SPOT tuning we used the prediction error on the test set, which is topic of criticism and will be treated in the discussion section of this work. In the following subsections, we describe the setup of the preprocessing operators used for these different model variants in more detail.

3.1 E1: Predicting Fill Levels without Preprocessing

The most simple approach is to predict fill levels solely based on the current rainfall, taken as only input feature for the SVM. Regardless of which SVR hyperparameter configuration was used, the obtained RMSE for this model on the test set couldn't get about 45.8% better than a naïve prediction.[1] Although the SVM prediction is more accurate than the naïve one, it can be seen in Fig. 4 that the length of high fill level periods are frequently underestimated throughout the whole test period. For this reason, this very simple model is not competitive to models like the INT2 by Konen *et al.* [5].

3.2 E2: Embedding of Rainfall

One major difference of time series problems in comparison to standard regression problems is that timeseries usually have certain dependencies of successive records. This has not been taken into consideration in our last model, which might be a main reason for its poor accuracy. Therefore usually an embedding of the input data [17] is conducted. Here, the fill level of stormwater overflow tanks $l(t)$ can be represented by a function F on past input features, more precisely by the rainfall $r(t)$ up to $r(t-W)$, where t indicates time and $W \in \mathbb{N}^+$ is the embedding dimension:

$$l(t) := F(r(t), r(t-1), r(t-2), ..., r(t-W)) \tag{3}$$

[1] Naïve prediction means predicting the mean value of the training set.

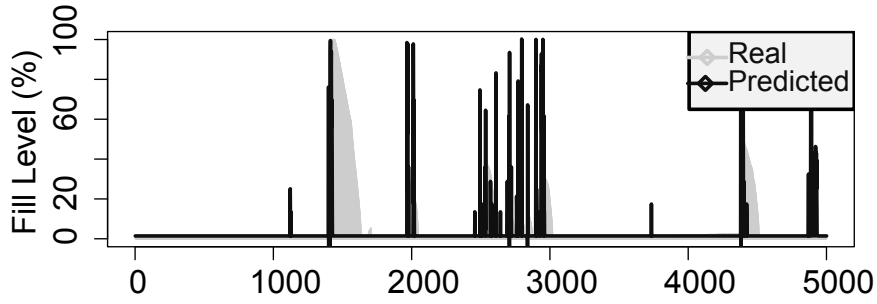

Figure 4: Plot of predicted fill levels using only rain data without any preprocessing.

Table 2: Best SPOT parameter configuration for rainfall embedding. The region of interest (ROI) bounds have been refined after preliminary runs with SPO.

Parameter	Best Value found	ROI
Embedding W_{rain}	43	$[2, 60]$
SVM γ	0.0116667	$[0.005, 0.3]$
SVM χ	1.25	$[0, 10]$
SVM ϵ	0.0116667	$[0.005, 0.3]$

The influence of past data points on the current data point is generally unknown, but might be detected by the SVM model, if we augment each record $r(t)$ with its predecessors $r(t-1), ..., r(t-W_{rain})$. Since the embedding parameter might have a crucial meaning for our model quality, we add this parameter to the SPOT tuning. This setup led us to a tuning of either the embedding dimension W_{rain} and of the SVM parameters γ, χ and ϵ. SPOT tuned parameter values which are presented in Tab. 2.

With an RMSE of 14.98 (cf. Tab. 4), the SVM model has gained accuracy by using an embedding of past rainfall in comparison to just using the current rainfall data point.

3.3 E3: Leaky Rain and Rainfall Embedding

In the design of the previous model, the rainfall at time $t-40$ has been considered to be of the same importance as the rainfall at time $t-5$. This is of course not true, because loosely speaking, the rainfall from two days in the past should not have the same impact on the fill level as the rainfall from the last 20 minutes. Or more precisely, the rain is a measurable quantity which drains into the soil and then – depending on the consistence of the soil – flows into the stormwater tanks with a certain delay. Therefore the rainfall could be summed up and this integrated quantity could than be used as a new feature of the input data. How can it be expressed that, given $w_2 > w_1$, rainfall from time $t-w_2$ has less influence than $t-w_1$ on $l(t)$? We define the preprocessing operator *leaky rain*

$$\sum_{i=0}^{T} \left(\lambda^i \cdot r(t-i)\right) \tag{4}$$

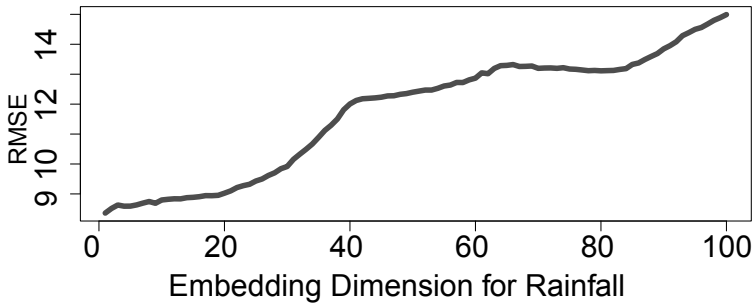

Figure 5: Progressively increasing the rainfall embedding dimension W_{rain}, while keeping leaky rain embedding dimension W constant at 43.

where $\lambda \in [0, 1]$, T is the length of the integration and t is the current time. The formula can be efficiently computed by Fast Fourier Transformation (FFT) even for large datasets.

The summarized results in Tab. 4 show that there is a vast reduction of prediction error by using this more sophisticated input feature: the RMSE with leaky rain embedding decreases from 14.98 to 10.14.

3.4 E4: Embedding of Leaky Rain Only

Surprisingly the rainfall embedding is no longer advantageous when leaky-rain embedding is taken into account. This fact can be clearly deduced from Fig. 5, where the RMSE is nearly monotonously increasing with higher rainfall embedding dimensions, given a constant leaky rain embedding dimension of $W = 43$. Besides that, the modeling process is slower when higher embedding dimensions are used, because the SVM has to cope with more input dimensions and much more data.

Learning from this observation, we omit the rainfall embedding W_{rain} and concentrate on optimizing the embedding dimension for the leaky rain embedding. Therefore, we performed a similar embedding dimension experiment for leaky rain. Results are shown in Fig. 6. The plot shows clearly that there is an almost monotonous decrease of prediction error when using embeddings up to 40 dimensions. The prediction error increases again for more than 40 dimensions. This effect could be explained by two processes. First, the infomation gain decreases with increasing embedding dimension, because the influence of rainfall on the target decreases with increasing time lag. This may lead to a model with worse generalization capabilities. Secondly, the model is fitted to a higher number of input dimensions, including disturbing factors as noise, measurement errors, etc. Summarized, these factors may lead to more complex models which are prone to overfitting.

When tuning the parameters, SPOT delivered an embedding dimension of 43 which is not the global optimum, but is very close to it. Possible improvements to this result can be achieved by increasing the function evaluations in the SPOT settings or incorporating a local search strategy on the SPOT parameters for fine-tuning.

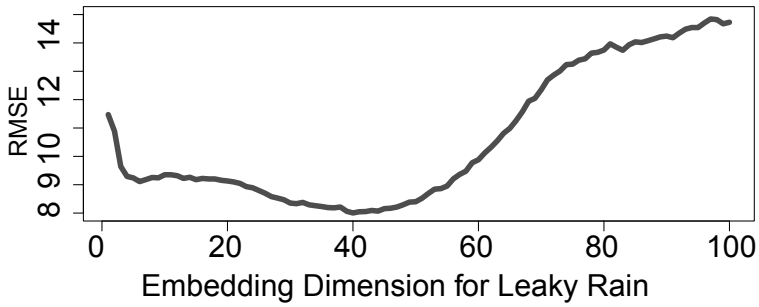

Figure 6: Progressively increasing the leaky rain embedding dimension W. The optimal embedding dimension is at $W = 40$ which was almost found by SPO. No rainfall embedding was used here.

3.5 E5: Two Leaky Rain Functions

Leaky rain has shown to be an adequate preprocessing function to simplify learning of the target function. However, the true function might be more complex due to factors not incorporated in our simple leaky rain function. Therefore, we employed a more complex preprocessing by using two leaky rain functions at the same time (Fig. 7). We tuned the parameters λ, T, and W independently for each of the two functions by SPOT leading to a new parameter configuration as shown in Tab. 3.

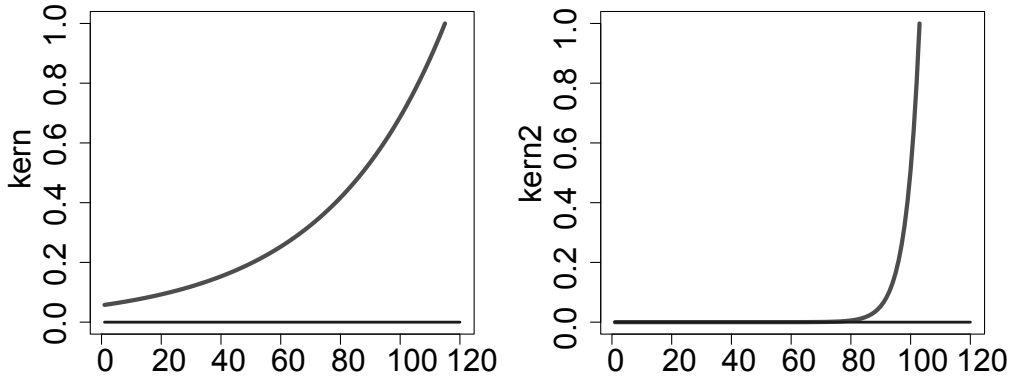

Figure 7: Left and right curve: the two leaky rain kernel functions obtained by SPO. Bottom lines: zero level.

After integrating the two kernel functions in our model, the RMSE slightly improves from 8.09 to 7.80 compared with a single leaky rain function, outperforming the INT2 model again. Note that one leaky kernel function uses a larger λ value and a shorter embedding dimension (less than half size of the first embedding dimension), so that both functions seem to support each other (Fig. 7).

Table 3: Best SPOT parameter configuration for all tuned parameters evaluated on set 4. The region of interest (ROI) bounds are the final values after preliminary runs.

Type	Parameter	Best found	ROI	Remark
Embedding	W_1	39	$[2, 60]$	embed. dimension 1
	W_2	16	$[2, 60]$	embed. dimension 2
	T_1	114	$[50, 120]$	leaky window size 1
	T_2	102	$[50, 120]$	leaky window size 2
	λ_1	0.0250092	$[0.00001, 0.3]$	leaky decay 1
	λ_2	0.225002	$[0.00001, 0.3]$	leaky decay 2
SVM	γ	0.0116667	$[0.005, 0.3]$	RBF kernel width
	ϵ	0.0116667	$[0.005, 0.3]$	ϵ-insensitive loss fct.
	χ	1.25	$[0, 10]$	penalty term

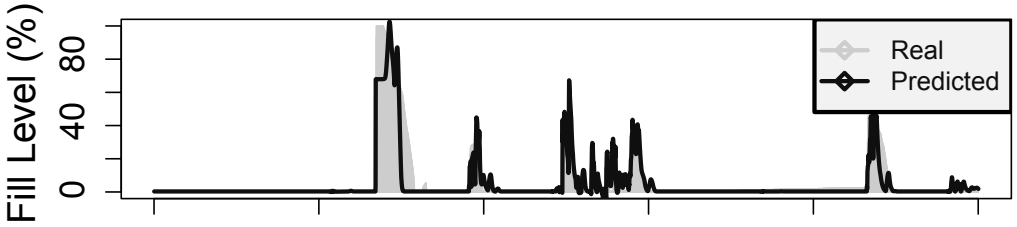

Figure 8: Prediction gathered by SVM with two leaky rain embeddings optimized separately by SPOT.

3.6 Summary of Preprocessing

A comparison of the results of all models obtained so far can be found in Tab. 4. It can be concluded that the model with the most complex preprocessing (two different leaky rain embeddings) leads to the best prediction accuracy. The worst results are obtained when no prepocessing is employed, indicating that no satisfying model for the stormwater problem can be found without using a special preprocessing. It seems that rainfall itself does not contain enough accessible information to do accurate predictions of the fill level. As soon as leaky rain is added as a feature for the SVM model, prediction accuracy increases considerably.

Table 4: Precision comparison of all models on set 4

Model type	RMSE	Theil's U
Naïve prediction	33.34	1.00
E1: Only Rainfall	18.07	0.54
E2: Rainfall Emb. w/o Leaky	14.98	0.45
E3: Rainfall and Leaky Rain	10.14	0.30
INT2	9.00	0.27
E4: Leaky Rain Embedding	8.09	0.24
E5: Two Leaky Embeddings	7.80	0.23

Table 5: Results of SPOT tuning on the stormwater problem. In each row 1-3 of the table, SPOT tunes the RMSE on validation set 1,3,4 leading to different SPOT-tuned parameter configurations. These configurations were applied to the test sets (columns) to make the results comparable. Each experiment was repeated five times with different seeds and we show the mean RMSE; bold-faced numbers are best values on the test set and the numbers in brackets indicate standard deviations.

		Test		
		Set 1	Set 3	Set 4
Validation	Set 1	**9.11** (0.56)	16.40 (6.42)	12.88 (5.50)
	Set 3	10.82 (1.55)	**12.78**(0.34)	12.36 (3.46)
	Set 4	10.45 (0.28)	12.93 (0.35)	**7.69** (0.48)
	S_t	10.64	14.67	12.62
	V_t	16.7%	14.7%	64.1%

4 Discussion

4.1 T1: Parameter Tuning by SPOT

In our last models we used the prediction error gathered on the test dataset as the objective function value for the hyperparameter tuning with SPOT. In the real world this value is unknown and when available during optimization it gives the tuned model an unfair advantage. In order to perform a fair comparison and to show the benefits of parameter tuning in a more realistic setting, we should use a different objective function. Otherwise the test set error might be too optimistic since the model has been tuned and tested on the same set. In Tab. 5, we present the mean results of five SPOT runs for the SVR model to determine optimal parameter settings which are then alternately evaluated on the sets 1,3,4. Again, dataset 2 has been used for training. The best configuration found by SPOT is then applied in turn to the other sets (columns) resulting in 3 RMSE values for each parameter configuration. We used the Matlab implementation of SPO[2] allowing a total budget of 200 SVM models to be built and a maximum number of 500 samples in the metamodel.

A strong indication of oversearching is when best values are often present in the diagonal of the table. It can be seen that this is the case for all validation sets of Tab. 5. Besides this, standard deviations of the offdiagonal values are also larger than the values on the diagonal.

We quantify the oversearching effect by evaluating the following formula: let R_{vt} denote the RMSE for row v and column t of Tab. 5. We define

$$V_t = \frac{S_t - R_{tt}}{R_{tt}} \quad \text{with} \quad S_t = \frac{1}{3}\left(\sum_{v=1}^{4} R_{vt} - R_{tt}\right) \tag{5}$$

With S_t we evaluate the mean off-diagonal RMSE for the columns $t = \{1, 2, 3\}$ which is an indicator of the true strength of the tuned model on independent test data. The diagonal elements R_{tt} are considerably lower in each column of Tab. 5. In case of no oversearching,

[2]The Sequential Parameter Optimization Toolbox for Matlab can be downloaded at `http://www.gm.fh-koeln.de/~bartz/experimentalresearch/`

a value of V_t close to zero would be expected, whereas values larger than zero indicate oversearching.

In summary, a systematic tuning is beneficial but the tuned RMSE is often subject to oversearching effects. E.g. in our case the RMSE on a certain test set was on average 32% higher[3] when the tuned model had not seen the test data before (the realistic case) as compared to the lower value when the test data were used during tuning (T=V).

4.2 T2: Feature Selection by Genetic Algorithms

A Genetic Algorithm (GA) is used to determine good feature subsets for the SVM regressor. We rely here on the GA approach because it has some advantages compared to other feature selection methods: iterative search algorithms can be used to determine feature subsets, where more features are added or eliminated to build the final feature set (Feature Forward Selection and Feature Backward Elimination). Unfortunately these methods often get stuck in locally optimal feature subsets where they finally converge. GAs offer the possibility to escape from such local optima and find the global optimum given enough iterations.

Experimental Setup In our experimental analysis we started five GA runs, each with a population size of 100, elitist selection strategy (e.g. the best 20% of total population were definitely survivors) and termination after 100 generations. GA parameters where chosen by means of preliminary runs. Each GA individual has N genes, each of which representing whether a certain feature should be included in the model or not. The basis input feature set consisted of all features drawn from a sample SPOT-tuned configuration set as described in Sec. 4.1. Here, the gene length N equals the sum of the embedding dimensions for the two leaky rain functions, ranging from 55 to 92. The candidate solution is mapped to a feature vector which is passed to the feature selection preprocessing script before the SVM model is built. This process has an overall runtime of about 17 hours on a 2.4 GHz Intel Xeon CPU.

Results In each objective function, the RMSE was calculated on the validation sets as defined in Sec. 2.1. This resulted in different feature vectors, which were evaluated again on each validation set. The number of selected features ranges from a minimal feature set of 5 (mean value of GA runs when set 1 was used for evaluation) up to a maximum feature set of 50 (mean value of 5 GA runs). The number of features only varies slightly for runs of the same configuration, but usually differs for different configurations.

The evaluation is presented in Tab. 6. Again it has to be noted that all configurations seem to suffer from oversearching, when the validation set V (the set on which the GA was performed) is equal to the test set T: the diagonal in Tab. 6 shows always the seemingly best values. Compared with the results gathered by the SPOT tuning (Tab. 5), GA feature selection leads to a slightly better predictive performance if we look at S_t, the mean off-diagonal RMSE.

Even when feature selection does not produce much better results than SPOT tuning alone, it has an obvious positive effect on the RMSE ranges: the standard deviations of the

[3] average of all V_t in Tab. 5

Table 6: Mean results of five runs using feature selection by genetic algorithms. SVM and preprocessing parameters were obtained using the SPOT configurations 1,3,4 (see Sec. 4.1). The table shows the RMSE values for feature subsets on the validation sets leading to different feature configurations (rows). These configurations were evaluated on the test sets (columns); bold-faced numbers are best values on the test set and the numbers in brackets indicate standard deviations over five runs.

		Test		
		Set 1	Set 3	Set 4
Validation	Set 1	**9.36 (0.11)**	15.48 (0.90)	11.44 (0.91)
	Set 3	10.80 (0.19)	**12.11(0.59)**	7.78 (0.30)
	Set 4	10.99 (0.07)	13.04 (0.04)	**7.36 (0.03)**
S_t		10.90	14.26	9.61
V_t		16.40%	17.75%	30.57%

configurations are considerably smaller with feature selection than without, leading to better generalizing models. Also the variance between the three off-diagonal RMSEs is lower than the high off-diagonal variance observed in experiment **T1**. A reason for this might be the complexity decrease of the models due to the lower number of input features. In addition to this, the runtime for model-building is also reduced, although the GA runtime has should be considered of course.

5 Conclusion and Outlook

In this work we analyzed different predictive models based on Support Vector Machines for a practical application named stormwater prediction. Summarizing the results obtained on real world test data our models are in most cases better than the best-known special-purpose model INT2 under the assumption that i.) preprocessing of the data and ii.) tuning of SVM and preprocessing parameters is conducted. This can be seen as a confirmation of our hypothesis **H1**. This might have a great impact for applications which need a lot of similar models to be built since with our approach most of the time-consuming work of defining and tuning domain-specific models can be replaced by automatic processes.

Our results have also shown that one has to be careful when optimizing data mining models by means of parameter tuning, e.g., SPOT and GA: parameter tuning will often lead to oversearching and to too optimistic error estimates on the datasets used for tuning (as measured by V_t in Tab. 5 and Tab. 6), which was the statement of our hypothesis **H2**. Therefore the distinction between validation datasets (used for tuning) and independent test sets is essential to obtain a realistic estimate on the improvement reached by tuning. Nevertheless, our results have shown that tuning leads to better models as measured by independent test set RMSE. Also feature selection led to more stable results in our case study, which indicates better generalizing models. In a nutshell, feature selection and SPOT tuning can help to improve results, but must always be validated on different test sets to detect possible overfitting and oversearching effects.

In future work we plan to extend and validate our study on other datasets, first by applying our methodology to different stormwater tanks and more comprehensive data (time periods

stretching over several years). Besides that, we want to compare our models with special time series regression frameworks, e.g., Gait-CAD [18], or with software as ClearVu Analytics [19] and MLR [20].

Acknowledgements

This work has been supported by the Bundesministerium für Bildung und Forschung (BMBF) under the grants FIWA (AiF FKZ 17N2309, "Ingenieurnachwuchs") and SOMA (AiF FKZ 17N1009, "Ingenieurnachwuchs") and by the Cologne University of Applied Sciences under the research focus grant COSA. We are grateful to Prof. Dr. Michael Bongards and his research group for discussions and for the stormwater tank data.

References

[1] Brockwell, P.; Davis, R.: *Time series: theory and methods*. Springer Verlag. 2009.

[2] Drucker, H.; Burges, C.; Kaufman, L.; Smola, A.; Vapnik, V.: Support vector regression machines. *Advances in neural information processing systems* (1997), S. 155–161.

[3] Bartz-Beielstein, T.: *Experimental Research in Evolutionary Computation—The New Experimentalism*. Natural Computing Series. Berlin, Heidelberg, New York: Springer. 2006.

[4] Hilmer, T.: *Water in Society – Integrated Optimisation of Sewerage Systems and Wastewater Treatment Plants with Computational Intelligence Tools*. Dissertation, Open Universiteit Nederland, Heerlen. 2008.

[5] Konen, W.; Zimmer, T.; Bartz-Beielstein, T.: Optimierte Modellierung von Füllständen in Regenüberlaufbecken mittels CI-basierter Parameterselektion. *at – Automatisierungstechnik* 57 (2009) 3, S. 155–166.

[6] Bartz-Beielstein, T.; Zimmer, T.; Konen, W.: Parameterselektion für komplexe Modellierungsaufgaben der Wasserwirtschaft – Moderne CI-Verfahren zur Zeitreihenanalyse. In: *Proc. 18th Workshop Computational Intelligence* (Mikut, R.; Reischl, M., Hg.), S. 136–150. Universitätsverlag, Karlsruhe. 2008.

[7] Flasch, O.; Bartz-Beielstein, T.; Koch, P.; Konen, W.: Genetic Programming Applied to Predictive Control in Environmental Engineering. In: *Proceedings 19. Workshop Computational Intelligence* (Hoffmann, F.; Hüllermeier, E., Hg.), S. 101–113. Karlsruhe: KIT Scientific Publishing. 2009.

[8] Koch, P.; Konen, W.; Flasch, O.; Bartz-Beielstein, T.: Optimizing Support Vector Machines for Stormwater Prediction. In: *Proceedings of Workshop on Experimental Methods for the Assessment of Computational Systems joint to PPSN2010* (Bartz-Beielstein, T.; Chiarandini, M.; Paquete, L.; Preuss, M., Hg.), Nr. TR10-2-007. TU Dortmund. 2010.

[9] Konen, W.; Koch, P.; Flasch, O.; Bartz-Beielstein, T.: Parameter-Tuned Data Mining: A General Framework. In: *Proceedings 20. Workshop Computational Intelligence* (Hoffmann, F.; Hüllermeier, E., Hg.). Karlsruhe: KIT Scientific Publishing. 2010.

[10] Theil, H.: Economic Forecasts and Policy. *Bayesian Analysis, Journal of the American Statistical Associa-Amsterdam: North-Holland* (1961), S. 776–800.

[11] Forrester, A.; Sobester, A.; Keane, A.: *Engineering Design via Surrogate Modelling*. Wiley. 2008.

[12] Bartz-Beielstein, T.: SPOT: An R Package For Automatic and Interactive Tuning of Optimization Algorithms by Sequential Parameter Optimization. Techn. Ber. arXiv:1006.4645. CIOP TECHNICAL REPORT 05-10. COLOGNE UNIVERSITY OF APPLIED SCIENCES. Comments: Article can be downloaded from: http://arxiv.org/abs/1006.4645. Related software can be downloaded from http://cran.r-project.org/web/packages/SPOT/index.html. 2010.

[13] Müller, K.; Smola, A.; Rätsch, G.; Schölkopf, B.; Kohlmorgen, J.; Vapnik, V.: Predicting time series with support vector machines. *Artificial Neural Networks–ICANN'97* (1997), S. 999–1004.

[14] Mattera, D.; Haykin, S.: Support vector machines for dynamic reconstruction of a chaotic system. In: *Advances in kernel methods*, S. 211–241. MIT Press. 1999.

[15] Chang, C.; Lin, C.: IJCNN 2001 challenge: Generalization ability and text decoding. In: *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on Neural Networks*, Bd. 2. 2001.

[16] Smola, A.; Schölkopf, B.: A tutorial on support vector regression. *Statistics and Computing* 14 (2004) 3, S. 199–222.

[17] Kantz, H.; Schreiber, T.: *Nonlinear time series analysis*. Cambridge Univ. Press. 2004.

[18] Mikut, R.; Burmeister, O.; Reischl, M.; Loose, T.: Die MATLAB-Toolbox GaitCAD. In: *Proceedings 16. Workshop Computational Intelligence* (Mikut, R.; Reischl, M., Hg.), S. 114–124. Karlsruhe: Universitätsverlag, Karlsruhe. 2006.

[19] Bäck, T.; Krause, P.: ClearVu Analytics. `http://divis-gmbh.de/ClearVu`. accessed 21.09.2010.

[20] Bischl, B.: The `mlr` package: Machine Learning in R. `http://mlr.r-forge.r-project.org`. accessed 25.09.2010.