PFG 2010 / 5, 0371–0384
Stuttgart, Oktober 2010

Article

Identifying Architectural Style in 3D City Models with Support Vector Machines

Christoph Römer & Lutz Plümer, Bonn

Keywords: Support Vector Machines, City Model, Data Mining, Pattern Recognition, CityGML

Summary: The purpose of the study presented here is to enrich low resolution (LOD1) 3D city models with semantic information about the architectural style of buildings. The focus is on identifying Wilhelminian-style buildings - terraced houses with three to four floors, high storeys, richly decorated facades and an architectural style typical for the turn of the 20th century. We use this example for the evaluation of the potential of Support Vector Machines (SVMs) for data mining in 3D city models. Identification of Wilhelminian-style buildings with low level data is demanding since most distinguishing characteristics are on higher levels of detail. However, Wilhelminian-style buildings tend to form specific ensembles of buildings with similar shape, thus context information is significant. After careful pre-processing, feature extraction, and feature weighting, a Support Vector Machine with a radial basis function as kernel was trained. In contrast to other classifiers which minimize the empirical, Support Vector Machines are based on structural risk minimization. This approach turned out to be especially useful for a task where the respective classes are highly inhomogeneous and a high percentage of the training data is mislabeled. In order to reduce the effect of the latter, an outlier detection was applied as part of the pre-processing. Above, a clustering algorithm was used to cope with hidden structures in the training data. Despite difficult boundary conditions, the SVM classifier was able to detect Wilhelminian-style houses with high accuracy. This demonstrates the high potential of Support Vector Machines for data mining in 3D City Models. However, intelligent pre-processing and kernel parameter optimization are necessary.

Zusammenfassung: *Erkennung des Gebäude-Architekturstils in 3D Stadtmodellen mit Support Vektor Maschinen*. Das Ziel der hier präsentierten Studie ist es, 3D Stadtmodelle mit geringer Auflösung (LOD1) mit semantischen Informationen über den Architekturstil der Gebäude anzureichern. Der Fokus liegt dabei auf Gründerzeithäusern – Reihenhäuser mit drei bis vier Stockwerken, hohen Räumen, reichhaltig verzierten Fassaden und einem Architekturstil, welcher für die Wende zum 20ten Jahrhundert typisch ist. Das Beispiel wird für die Evaluierung des Potentials der Support Vektor Maschinen für Data-Mining in 3D Stadtmodellen benutzt. Die Identifizierung von Gründerzeithäusern in niedrig aufgelösten Daten ist anspruchsvoll, da die charakteristischsten Merkmale dieser Gebäude erst in einem höheren Detailgrad beobachtbar sind. Allerdings lassen sich Gründerzeithäuser oftmals zu spezifischen Gruppen von Häusern mit ähnlicher Form zuordnen, wodurch Kontextinformationen wichtig werden. Nach einer Vorverarbeitung der vorhandenen Daten, einer Merkmalsextraktion und einer Merkmalsgewichtung wird eine Support Vektor Maschine (SVM) mit radialen Basisfunktionen als Kernel trainiert. Im Gegensatz zu anderen Klassifikatoren, welche das empirische Risiko der Fehlklassifikation minimieren, basieren SVMs auf dem Prinzip der strukturellen Risikominimierung. Dieses Vorgehen ist besonders nützlich für Aufgaben, wo die Klassen besonders inhomogen und die Trainingsdaten zu einem hohen Prozentsatz falsch vorklassifiziert sind. Um letzteren Effekt zu reduzieren, wurde eine Ausreißersuche bei der Vorverarbeitung der Daten angewendet. Zusätzlich wurde eine Cluster-Methode genutzt, um unbekannte Zusammenhänge in den Trainingsdaten auszunutzen. Trotz der schwierigeren Randbedingungen war der Klassifikator in der Lage, Gründerzeithäuser in dem Stadtmodell mit hoher Genauigkeit zu prädizieren. Dies zeigt, dass Support Vektor Maschinen ein großes Potential haben für Data-Mining in 3D Stadtmodellen. Eine geschickte Datenvorverarbeitung und Optimierung der Kernelparameter sind jedoch immer noch notwendig.

DOI: 10.1127/1432-8364/2010/0063 1432-8364/10/0063 $ 3.50

1 Introduction

Three dimensional city models are available for an increasing number of cities. Not only the volume of data increases, but also the development of consistent geospatial data modelling methods (Kolbe & Gröger 2003), such as CityGML, and of models suitable for representing buildings in different levels of detail is progressing fast.

However, nowadays most city models are only given in the level of detail 1 (LOD1). In LOD1 each building is represented by an abstract solid which is derived from the height and the cadastre ground plan of the building. These models make it difficult to detect contentual connections, even if these could substantially support the user when analyzing and pre-processing the dataset.

For instance, a user who intends to move from one town to another and has a specific style of life intends to query a 3D database in order to find areas of a certain architectural style in his new neighborhood. He is able to point on streets in his old neighborhood which are dominated by his preferred architectural style, say Wilhelminian-style buildings. However, he is not willing to study the technical details of the database and to translate his internal concept into an SQL query. Instead he wants to approach the database by saying: "I look for a neighborhood similar to the one where I live now, which is located there". A system which supports this kind of queries is the vision behind our research.

The aim of this study was to design a classifier which identifies areas and streets in cities, based on LOD1 3D city model datasets, which are dominated by a specific architectural style. This information may be used to enrich the LOD1 solids and help the user to interpret the abstract solids or may be used to visually enhance the LOD1 models with automatically generated textures (Krückhans & Schmittwilken 2009) which fit the specific building type. This is important as LOD1 city models have been derived for large regions from airborne laserscans and cadastre coordinates, whereas higher resolutions are more difficult to obtain. We restrict to the dichotomous case and leave multi-class labeling for further discussion.

The classifier has to face two main problems. Firstly, an inhomogeneous dataset can be expected because houses of the same architectural style may differ significantly with regard to the features in the given level of detail. Hence, there may be not one prototype for an architectural style, but an unknown number of seperate clusters. Secondly, we do not assume that a set of correctly labeled buildings is given. Instead streets dominated by a specific building style are available, where nevertheless buildings with a different architecture appear. A rather high propability of mislabeling in the training dataset can be expected.

Additionally, Wilhelminian-style houses are an especially challenging scenario. Their most distinguishing characteristics (high storeys, richly decorated facades) are not represented in the LOD1 model. Hence, the classifier has to detect hidden patterns which are unknown a-priori. This is the reason why the identification of Wilhelminian-style buildings in LOD1 data poses a demanding for data mining.

Data Mining is the implementation of specific algorithms and learning methods to automatically extract patterns from data (Fayyad et al. 1996). The classifier has to be able to learn what is specific for a given class from a given set of samples, a capability which is named generalization. Generalization reduces the risk of overfitting and therefore of wrongly predicting the affiliation of unseen data to a certain class. This is achieved by finding a model which is not overly dependent on single examples and therefore resistant against outliers and noise.

A classification method which has received increasing attention in recent years is the Support Vector Machine (Vapnik 1998, Schölkopf & Smola 2002, Heinert 2010). In contrast to other classifiers who aim at minimizing the empirical risk with regard to the training data, SVMs are based on the principle of structural risk minimization. This leads to a superior generalization capability (Gunn 1998).

As with other supervised learning methods, the quality of the training dataset significantly influences the quality of the class prediction. Hence, the amount of mislabeled examples in the training dataset has to be reduced. To achieve this, two methods are combined. The

LOF outlier detection (Breuning et al. 2000) is eligible for finding outliers in datasets where no information about the underlying distributions is given. Even if this algorithm can find outliers, the example training set still contains clusters of mislabeled buildings. Therefore, a combination of a clustering algorithm with a SVM-classifier is used to pre-process the training dataset and to remove training examples where the probability of a correct prelabeling is low.

After an overview over the concept of the Support Vector Machines is given in Section 2, the main contribution of this paper is presented in Section 3, which describes how the SVMs are applied to LOD1 city models in order to classify buildings and how the training dataset is pre-processed in order to reduce the number of misclassified training examples.

2 Support Vector Machines

The following Section introduces the concept of the Support Vector Machines. Support Vector Machines is a supervised learning method for binary classification problems. Based on a set of labeled training samples SVMs learn a model which enables the classifier to differ between two classes and predict the class of unseen samples.

Starting with the general problem of binary classification the important principle of structural risk minimization is presented as well as the separating hyperplane on which the class indicator function is based, and subsequently linear and nonlinear Support Vector Machines are introduced. Section 2.4 discusses the case of not linearly separable classes.

2.1 *Binary Classification*

The starting point is a given dataset of training samples $(\mathbf{x}_1, y_1) \ldots (\mathbf{x}_m, y_m)$, with $\mathbf{x}_i$ denoting an n-dimensional feature vector and with $y_i \in \{\pm 1\}$ indicating the class labels of the samples $\mathbf{x}_i$. The dataset is generated indepedent and identically distributed by the underlying, unknown propability distribution $P(\mathbf{x}, y) = P(\mathbf{x})P(y \mid \mathbf{x})$.

We want to generalize the discrete training samples in a way which allows to predict the class label y of any unknown sample $\mathbf{x}$ generated by the same propability distribution $P(\mathbf{x}, y)$ as the training samples. Therefore we need an indicator function $f : \rightarrow \{\pm 1\}$ which approximates the unknown $P(\mathbf{x}, y)$.

At first appearance an appropriate classifier is given by an indicator function which minimizes the expected risk of misclassification, i.e. where the difference between the predicted and the given class label of the training samples is minimal. This approach is called empirical risk minimiziation.

However, this approach risks overfitting. In an attempt to reduce the empirical risk as much as possible, the classifer may use a model with too many degrees of freedom and therefore may adapt to noise and outliers instead of the underlying patterns we truly search.

A basic insight of statistical learning theory (Vapnik 1998) states that in order to achieve good classification performance one has to restrict the set of possible indicator functions. This is done by a function which estimates and penalizes the complexity of the model. The model complexity is measured with the VC-Dimension (Vapnik & Chervonenkis 1971). Identification of the appropriate VC-Dimension with regard to a given problem is the basic idea of structural risk minimization. Support Vector Machines are based on this principle. They balance the risk of overfitting with the minimization of the empirical risk.

2.2 *Linear Support Vector Machines*

We start with the basic case of linear support vector machines and assume that the training data is linearly separable. The set of indicator functions is thus restricted to separating hyperplanes of the form

$$\langle \omega, \mathbf{x} \rangle + b = \sum_{i=1}^{n} \omega_i \mathbf{x}_i + b = 0 \ .$$

The distance between the origin and the hyperplane is given by $b \in R$, while $\omega \in R^n$ denotes the normal vector (weight vector) of the hyperplane. The hyperplane divides the feature space into two regions, one for the class {+1} and one for the class {–1}. New, unseen points are classified depending on which side of the hyperplane they are.

We now search the hyperplane which results in no misclassification and where the distance between the hyperplane and the nearest points in the training dataset is maximal. Among all separating hyperplanes (there are infinitely many) this *maximal margin* hyperplane proves to minimize the structural risk (Section 2.1) and to achieve the best prediction accuracy. If we fix the minimal distance of the nearest point in the dataset to the hyperplane to one, we get a so called canonical hyperplane and the distance of any point in the dataset to the hyperplane is given by

$$d(\boldsymbol{\omega}, b; \mathbf{x}) = \frac{\left|\langle \omega, x \rangle + b\right|}{\|\omega\|}.$$

If we would not use canonical hyperplanes $\boldsymbol{\omega}$ and b could always be multiplied by the same non zero constant. Fixing the distance to any scalar abolishes this degree of freedom, which would prevent the construction of a unique maximal margin hyperplane.
Hence, we have to minimize

$$h(\boldsymbol{\omega}) = \frac{1}{2}\|\boldsymbol{\omega}\|^2 \qquad (1)$$

to get the optimal hyperplane with the maximal distance to the nearest point of the training dataset.

The separating hyperplane will only then be optimal, if the risk of misclassification is minimized at the same time. This is achieved by the constraint:

$$y_i\left(\langle \boldsymbol{\omega}, \mathbf{x}_i \rangle + b\right) \geq 1, i = 1 \ldots m \qquad (2)$$

Finding the optimal hyperplane therefore is identical to solving equation (1) subject to the constraint (2). This turns out to be a convex minimization problem which is guaranteed to have one global solution which may be found with well known methods. With regards to performance SVMs tend to solve the following equivalent dual maximization problem (Boyd & Vandenberghe 2004). Using the Karush-Kuhn-Tucker conditions (Rockafellar 1993) we get:

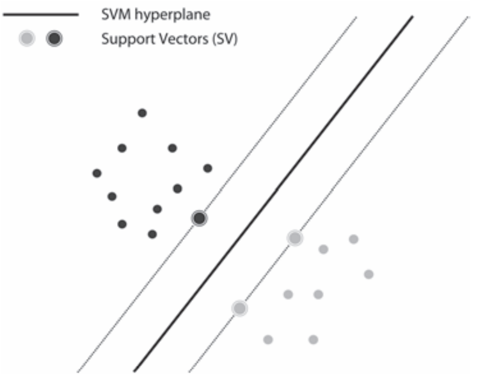

Fig. 1: Separating hyperplane between two classes (grey and black points). The optimal hyperplane is the one with the largest distance to both classes.

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left\langle x_i, x_j \right\rangle, \qquad (3)$$

$$\text{subject to } \alpha_i \geq = 0 \qquad (4)$$

$$\text{and } \sum_{i=1}^{m} \alpha_i y_i = 0. \qquad (5)$$

Solving (3) returns the dual Lagrangian α_i for every $\mathbf{x}_i$. One key ability of the Lagrangian dual variable is that it becomes zero for every $\mathbf{x}_i$ where the constraint does not become zero, i.e. where the distance between the hyperplane and the corresponding class is not minimal. Those points where $\alpha_i \neq 0$ is true are called support vectors. With the help of the Lagrangian dual variable the optimal hyperplane can be calculated again:

$$\hat{\boldsymbol{\omega}} = \sum_{i=1}^{m} \alpha_i y_i x_i, \qquad (6)$$

$$\hat{b} = -\frac{1}{2}\langle \hat{\boldsymbol{\omega}}, \mathbf{x}_a + \mathbf{x}_b \rangle, \qquad (7)$$

where $\mathbf{x}_a$ and $\mathbf{x}_b$ are any support vectors of the two classes $(\mathbf{x}_a, \mathbf{y}_a) \in \{+1\}$ and $(\mathbf{x}_b, \mathbf{y}_b) \in \{-1\}$. The coefficients of ω can be interpreted as the weights of the individual features.

Predicting the class of a new, unseen data point is now possible through the sign of the distance between this point and the hyperplane, which leads to the indicator function

$$f = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + \hat{b}\right). \quad (8)$$

2.3 *Nonlinear Support Vector Machines*

Until now it was assumed that the data is linearly separable, though this indeed is not appropriate for all datasets. As the Cover-Theorem (Cover 1965) states, the propability that classes are linearly separable increases when the features are nonlinearly mapped into a higher dimensional feature space. Mapping the original dataset into a higher dimensional space therefore allows the construction of an optimal separating hyperplane.

This, however, may be rather computational inefficient for large datasets. Hence, kernels (Shawe-Taylor & Cristianini 2004) are used to compute the Support Vector Machine in high dimensional feature spaces without explicitly mapping into these. As the kernel trick states we can replace every dot product with a kernel function $k(\mathbf{x}_1,\mathbf{x}_2)$. Since all feature vectors only occur in dot products, the kernel trick can be applied (Schölkopf & Smola 2002) to map the data into a high dimensional space and simultaneously avoid the drawback of a high runtime.

Performing the mapping with a kernel function $k(\mathbf{x}_1,\mathbf{x}_2)$ results in a modified optimization problem:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j), \quad (9)$$

subject to the constraints (4) and (5).

The most widely used kernel function is the Gaussian radial basis function

$$k(\mathbf{x}_1,\mathbf{x}_2) = \exp\left(-\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|^2}{2\sigma^2}\right), \quad (10)$$

where the parameter σ influences the smoothness of the decision boundary in feature space.

The hyperplane is now given by

$$\langle \hat{\boldsymbol{\omega}}, \mathbf{x} \rangle = \sum_{i=1}^{m} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) \quad (11)$$

and

$$\hat{b} = -\frac{1}{2} \sum_{i=1}^{m} \alpha_i y_i (k(\mathbf{x}_i, \mathbf{x}_a) + k(\mathbf{x}_i, \mathbf{x}_b)), \quad (12)$$

where $\mathbf{x}_a$ and $\mathbf{x}_b$ are any support vectors from both classes. The indicator function is

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i k\langle \mathbf{x}, \mathbf{x}_i \rangle + \hat{b}\right). \quad (13)$$

2.4 *Soft Margin Classifier*

Until now it has been assumed that the training data are free of noise and that both classes do not overlap in feature space. This assumption, again, is not realistic in most practical cases. Above, as discussed in Section 2.1, the classifier has to handle the trade-off between minimizing the empirical risk and calculating a hyperplane which generalizes the data well. Hence, soft constraints are required which allow misclassifications in the training data. In order to handle misclassifications, the slack variable ξ is introduced, which results in a new constraint for (1)

$$y_i (\langle \boldsymbol{\omega}_i, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \; i = 1 \ldots m, \quad (14)$$

$$\xi_i \geq 0, i = 1 \ldots m. \quad (15)$$

With an unbounded slack variable ξ, however, all examples would meet the constraints. Thus the sum of the slack variables has to be minimized.

$$\text{Minimize } h(\boldsymbol{\omega}) = \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{m} \xi_i, \quad (16)$$

subject to constraint (14) and (15), where the penalty term C weights the influence of the slack variables. Interestingly, the Lagrangian dual optimization problem (9) stays the same, apart from a slightly modified constraint $C \geq \alpha_i \geq 0$

3 Predicting Wilhelminian-style Buildings

Our aim has been to learn a SVM model of Wilhelminian-style buildings for LOD1 data. Wilhelminian-style buildings, typical for the late 19th to early 20th century, are mainly terraced houses with three to four floors. The facade is typically richly decorated and has many small oriels. While the first and second floor are mostly high with huge windows the upper floors were mainly inhabited by servants or people from the poorer working class and display small windows and low ceilings.

We used the LIBSVM Tool from (Chang & Lin 2001), which is readily available and provides a stable, performant and convenient SVM implementation on several platforms. Most other algorithms used here are available in the open source data mining system RapidMiner (Mierswa et al. 2006).

The first Section describes how the feature space is constructed from the LOD1 model. In Section 3.2 the training examples for the Wilhelminian-style buildings and for "Others" are chosen. In Section 3.3 the Support Vector Machines method is applied to our classification task. Section 3.4 describes how the results can be improved using an outlier detection and a clustering algorithm and finally in Section 3.5 the results are discussed.

3.1 *Constructing the Features from the LOD1 City Model*

The example dataset is given as a 3D-CityGML model of Bonn in the LOD1 representation, where buildings are reduced to abstract solids without roofs (see Fig. 2). The solids are constructed from cadastre coordinates of the conjugated houses ground plans, whereas the height is extracted from airborne laserscans. Overall, the database used for this experiment has 57,510 buildings. In addition, every building is given with its address.

The SVM classifier expects feature vectors of fixed length which allow a comparison between the buildings. The given cadastre coordinates for the ground plans of the buildings, however, are lists of arbitrary length which are not suitable for a similarity test. As such, the given ground plans are not suitable for our tasks. Therefore, feature construction was necessary.

As every solid can be described by its ground plan and its height, descriptions of both are used as features. Hence, the first feature used is the height of every solid. The ground plan is approximated by its 2D-bounding box. From this bounding box, the width (here the shortest edge of the box), the length (the longest edge of the box) and the slimness as the ratio of width to length is extracted. Another feature is the area of the exact ground plan. The last feature derived from a single building solid is the type of the floor plan. Un-

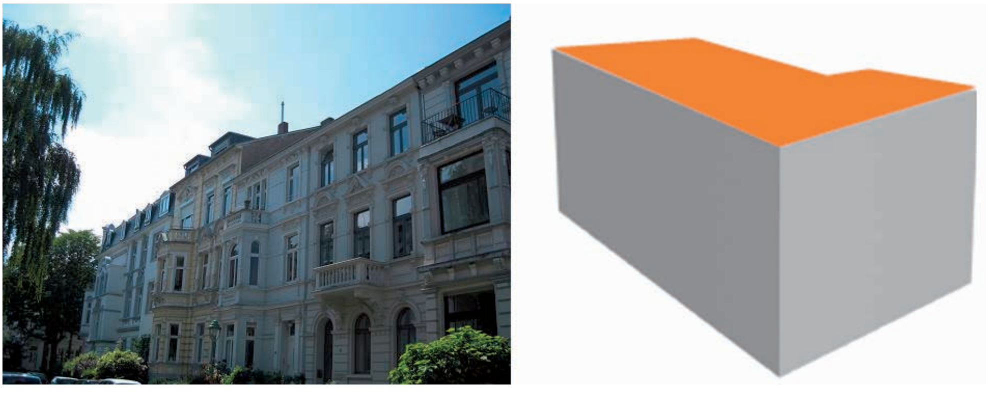

Fig. 2: Typical examples of Wilhelminian-style buildings (left side) compared with a LOD1 solid (Kolbe & Gröger 2003) on the right side.

der the assumption that every angle is rectangular, the ground plan is categorized into rectangles, L-,U-, T-forms and atriums. If none of these forms fit the ground plan sufficiently it is labeled as a complex form.

Most Wilhelminian-style buildings are terraced houses and therefore very similar in their features. All houses are grouped in house-blocks. As a rule, adjacent houses are closer than one meter to its neighbor. Since Wilhelminian-style houses tend to be rather similar within a given neighborhood, context turns out to be rather important. For this reason, the context is represented explicitly for every single building. For every feature, with the exception of the type of the ground plan, the median of its house-block is calculated and added as an additional feature.

To take buildings into account which differ considerably from the other buildings in a house-block, the difference between the median and the single buildings' feature has been added as an additional feature.

All in all, 16 features were derived (see Tab. 1 for a list of all features used). These features build the feature vector **x** of the binary classification task for every building the SVM algorithm is supposed to learn. The rank is based on a weighting which will be discussed in Section 3.3.

Finally, as the features have different scales, all features are normalized to a distribution with mean 0 and standard deviation of 1. This is a standard step in pre-processing and avoids unintended influence of features with a large scale in equation (8).

3.2 *Choosing the Training Examples*

The next step has been to provide a sufficiently large number of training data containing both Wilhelminian-style buildings and "Others". The focus has been on the design of a (semi-) automatic procedure following the lines sketched in the introduction.

For this reason, previous knowledge has been used. Suburbs were selected which are dominated by a certain building type. The buildings of a whole suburb are either assigned

Tab. 1: Features derived from the LOD1 solids, ordered according to their relevance for the classifier calculated by the SVM-Weighting algorithm explained in Section 3.3.

Feature	Rank
$\text{Height}_{\text{Median of the associated house block}}$	1
$\text{Slimness}_{\text{Median of the associated house block}}$	2
$\text{Width}_{\text{Median of the associated house block}}$	3
$\text{Height}_{\text{Difference to the median of the associated house block}}$	4
$\text{Width}_{\text{Difference to the median of the associated house block}}$	5
Height	6
$\text{Length}_{\text{Median of the associated house block}}$	7
Width	8
Slimness	9
Type of the floor plan	10
$\text{Area}_{\text{Median of the associated house block}}$	11
Length	12
$\text{Length}_{\text{Difference to the median of the associated house block}}$	13
$\text{Slimness}_{\text{Difference to the median of the associated house block}}$	14
$\text{Area}_{\text{Difference to the median of the associated house block}}$	15
Area	16

to the class Wilhelminian-style buildings or "Others". The "Bonner-Suedstadt", a suburb dominated by the Wilhelminian-style, is the training example for Wilhelminian-style buildings, where three other suburbs are examples for the class "Others". Those three suburbs are chosen in such a way that one is expected to be easily separable, one with only few buildings of a similar architectural style and one which is expected to be very difficult to separate. The last one was a former "Wilhelminian-style-suburb", before it was destroyed by bomb raids in World War II. As some single buildings and streets still are Wilhelminian-style, the new buildings were constructed with similar height, width and length to fit into the overall picture of the suburb. Putting together this demanding training set was done intentionally to provide a challenging scenario for the learner.

The result has been a training dataset of 812 houses serving as the Wilhelminian-style examples and 1992 houses within the other three suburbs as examples for the class "Others".

3.3 *Learning the Model*

For reasons which become clear later we started with learning a linear SVM as described in Section 2.3. As both classes overlap in their training examples, soft margin SVMs were used, as described in Section 2.4. This means the only parameter which had to be optimized was the penalty term C which weights the slack variable ξ in equation (16) and determines the influence of wrongly classified buildings on the position of the hyperplane.

There is no possibility to estimate the best value for C a-priori. As Fig. 3 shows, the percentage of correctly predicted buildings is very sensitive to minimal changes in the parameter C. Therefore, different values for C had to be tested and the one which minimizes the empirical risk was assumed to be the best. To test the empirical risk a ten-fold cross-validation (Kohavi 1995) was used.

By and large, the accuracy of the *linear* SVM was not satisfactory. However, as a desired side-effect, the linear SVM provides a good weighting of the relative relevance of the distinct features. According to formulas (6) and (7) in Section 2.3 the learned model consists of a normal vector $\boldsymbol{\omega}$ and a bias b. The normal vector $\boldsymbol{\omega}$ specifies the coefficents of the linear equation defining the hyperplane. The absolute value of the components of $\boldsymbol{\omega}$ is a good indicator of the weight of the respective attributes.

The calculated weights have been used to rank the features as was suggested by (Guyon et al. 2002) and to rescale the normalized fea-

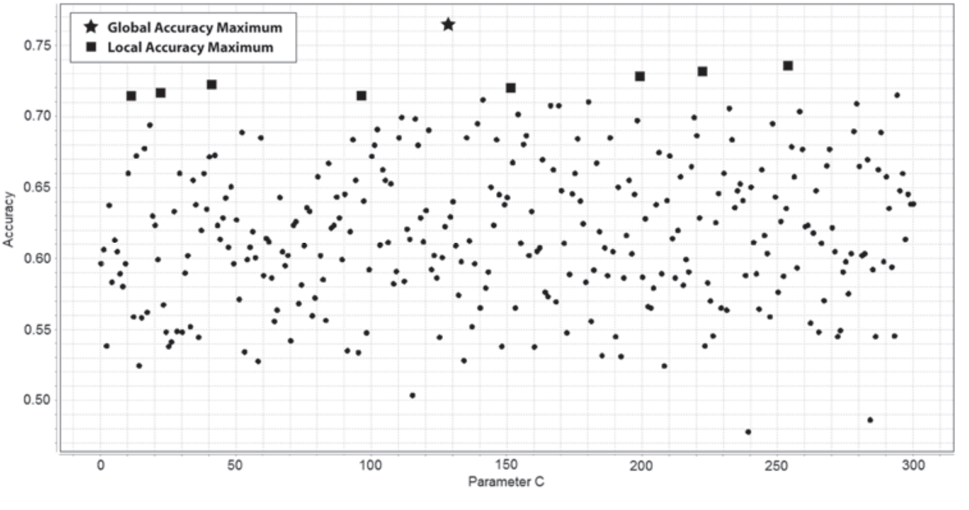

Fig. 3: Prediction accuracy of an unweighted, linear SVM on the training dataset using a ten-fold cross validation with respect to different choices for the parameter C.

tures in order to give more weight to more relevant features. The accuracy was improved by 7 % this way. The drawback here is, however, that we have to learn two linear SVM models and have to estimate the parameter C two times independently: at first for the normalized training dataset and then at second for the rescaled, weighted dataset.

Even if the linear SVM classifer was already able to predict some Wilhelminian-style buildings, it resulted in too many obvious errors (cf. Fig. 4) and gave reason to doubt that the problem can be linearly separated through hyperplanes. As this is a binary classification problem with the task to distinguish between Wilhelminian-style houses and all other buildings in the city, it is easy to imagine that there are always examples of buildings from another architectural style which are both larger and smaller in one feature than the Wilhelminian-style buildings. Hence the possibility that both classes can be distinguished with linear hyperplanes seemed unlikely.

Therefore, a non-linear model based on a radial basis function as described in Section 2.4, formula (10), was learned. If this kernel is applied to the Support Vector Machine, the method can be compared with a radial basis network, though instead of putting the radial basis functions in the center of each class, they are placed at the support vectors which result from the Lagrangian optimization process described in equation (9) (Schölkopf et al. 1996).

This affords a second parameter σ which controls the smoothness of the decision boundary. So two parameters, namely C and σ, had to be learnt in order to specify the optimal model. They are different from the parameters learnt for the linear model.

For the linear SVM the evolutionary algorithm (Goldberg 1989) was used. As we have many local maxima (see Fig. 3) a high number of iterations is suggested. It found the optimal value for the SVM-weighting at $C_W = 66.16$. The coefficients of this SVM were now used to rescale the feature space and subsequently the final SVM-classifier was learned with $C_{SVM} = 145.41$. This parameter combination resulted in 84.45 % correctly classified buildings on the whole training dataset. Please note that C_W is not near the global maximum of Fig. 3. This is due to the fact that we chose a step size of one for the parameters in Fig. 3 while the evolutionary algorithms uses decimal places too. The great difference between the optimal values for C underlines how sensitive the performance is to minimal changes in the parameter C.

The indicator function we derived from the training examples was used to classify all buildings in the city of Bonn. In order to evaluate the results, specific suburbs which are known to consist of both Wilhelminian-style

Fig. 4: Results of the linear Support Vector Machine and the RBF Support Vector Machine. Shown are the 2D-bounding boxes of each building. A grey box was predicted by both classifiers. Orange boxes were identified only by the linear SVM and blue boxes only by the RBF SVM.

houses and other buildings were compared with the classification results.

Fig. 4 shows the exemplary results for the streets 'Rosental' and 'Rosenst.'. Both are not part of the training dataset. The classifier was able to successfully predict the Wilhelminian-style buildings in the 'Rosenst.'. The street 'Rosental' is also consisting of Wilhelminian-style houses, but only the southern buildings were correctly identified, whereas the northern part of the street was wrongly classified as "Others".

In addition, some buildings which are obviously too large for the searched Wilhelminian-style buildings with respect to their ground plan, for instance the one south of 'Rosental', were also wronlgy classified.

If we calculate the RBF-SVM the optimal result was achieved with the parameters C_W = 289.97 for the SVM Weighting and C_{SVM} = 18.46 and σ_{SVM} = 0.05 for the SVM classifier. This resulted in 87.67 % correctly classified buildings on the training dataset.

The use of the Gaussian kernel significantly improved the results of the classifier. The obvious outliers are no longer predicted as part of the Wilhelminian-style and also some additional buildings in the street 'Rosental' are correctly classified. Nevertheless some errors remain. For example the first building in the picture of the street 'Rosental' is clearly no Wilhelminian-style house. Also both the linear and the RBF-SVM wrongly predicted whole streets in other parts of the city as Wilhelminian-style.

3.4 *Pre-processing the Training Dataset: Outlier Detection and Clustering*

It was already mentioned that a relevant number of buildings was mislabeled. As Support Vector Machines is a supervised classifier, this has negative effects on the quality of the classifier.

In order to improve quality, data preprocessing should refine the quality of the training data. We applied two methods, namely identification and elimination of outliers and clustering. Both are closely linked. We start with the discussion of outlier detection.

There are different approaches for outlier detection (Hodge & Austin 2004). Most of them make certain assumptions on the distribution of the data. These assumptions could not be confirmed in the given data. For this reason we applied two more recent methods, namely LOF outlier detection (Breuning et al. 2000) and X-Means (Pelleg & Moore 2000).

Instead of finding global outliers, the Local Outlier Factor (LOF) test is local and depends on how isolated a point is with respect to the neighborhood. The decision of how isolated a point is relies on its local density compared with the local density of its neighbors. From this comparison a degree of being an outlier is derived. The only parameter needed for this approach is the number of nearest points which are used as the neighborhood. As the test compares local densities, it can be interpreted as the decision if a point belongs to a certain cluster of the size of at least a minimal number of points, controled by the parameter min_points. Therefore, we only have to estimate the size of the minimal cluster we wish to keep in the training dataset with min_points. In this experiment, a clustersize of ten was used, though other choices between 10 and 40 led to only very small changes which had no great impact in this dataset. This indicates that a rough estimation is sufficient for this preprocessing step.

The result is a value between zero and one, which indicated the degree of a point being an outlier. Good results were achieved by chosing the mean outlier factor plus three times the standard deviation, i.e. a LOF value of 0.24. This way the more obvious outliers were deleted. This is important as the next step uses a K-Means algorithm which may be influenced by those outliers. Stricter bounds, say the mean plus two times the standard deviation led to similar results. Closer boundaries resulted in too many deleted Wilhelminian-style buildings.

The next step was to identify clusters of buildings which had the wrong label. The training dataset of Wilhelminian-style houses, for instance, contained a cluster of semi-detached houses which are not built in the Wilhelminian-style.

The idea to solve this problem is based on the assumption that a classifier will predict buildings of clusters which are compact in feature space as patterns of the same class. This follows the underlying cluster assumption of semi-supervised learning which states that if points are in the same cluster, they are likely to be of the same class (Chapelle et al. 2006). Therefore, if we have buildings with a different architectural style than the Wilhelminian-style in the training dataset, we should be able to find them with an unsupervised clustering algorithm.

If we now cluster the training dataset for Wilhelminian-style buildings and assume that each cluster represents either one architectural subcategory of Wilhelminian-buildings or of another building style, say semi-detached houses, we could use each cluster as a seperate class and train it versus the class "Others" using an SVM. A high misclassification ratio will now indicate that a significant number of similar buildings lies within the training dataset "Others". For example, there may be about ten times the number of semi-detached houses in the training samples for "Others" as in the training dataset of Wilhelminian-style buildings, hence the classification performance of the cluster containing the semi-detached houses will be relatively poor.

This way, the clusters which are hard to separate from "Others" are deleted from the training dataset of the Wilhelminian-style-class". This works especially well with Support Vector Machines because of its superior generalization ability.

There are several cluster analysis methods available for this step. Well-known algorithms, like DB-SCAN (Ester et al. 1996) are not used, as they are not optimal for clusters with different densities (Ertöz et al. 2003). X-Means (Pelleg & Moore 2000) is an extension of the classical K-Means algorithm where the optimal number of clusters K is estimated through the Bayesian information criterion (Schwarz 1978).

To reduce the influence of unimportant or highly correlated features on clustering, a feature weighting algorithm was applied beforehand. Best results were achieved if the top eight features (half the feature space) were used.

This way, four clusters of Wilhelminian-style-samples were derived. These were trained in the same way as described in Section 3.3 using a multi-class SVM against the class "Others". This resulted in two clusters with correctly classified building ratios of more than 80 % and two classes with ratios below 55 %. The latter were deleted from the training dataset, the remainig two were consolidated. The process from Section 3.3 was repeated on this refined dataset.

The question for a good threshold remains. In our experiment, a threshold of 66 % led to good results. A higher boundary denied too many buildings for the classifier, whereas a lower boundary was not always able to catch all errors.

As a result, the mentioned semi-detached houses were completely removed from the training dataset, together with other wrongly pre-labeled samples. Though this, of course, also resulted in the loss of some correctly pre-

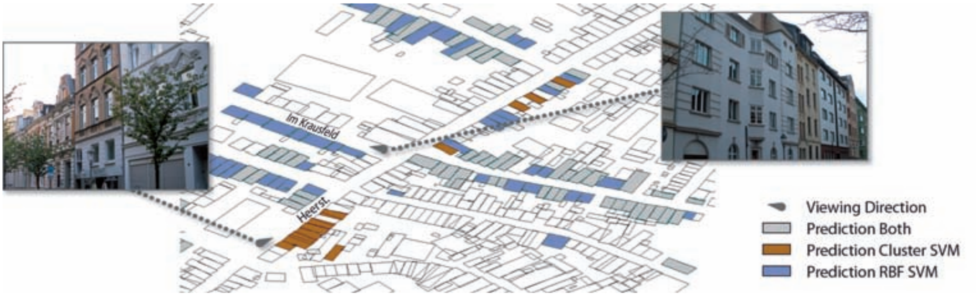

Fig. 5: Comparison of the SVM classifier after the clustering (brown) and without the clustingering, using the RBF-Kernel from Section 3.3 (blue). Buildings identified as Wilhelminian-style buildings by both classifiers have grey bounding boxes.

labeled buildings, especially those Wilhelminian-style houses which have atypical features, like the ones placed at the corners of the street, who have a ground plan which is more rectangular and not typical for the terraced houses of the other Wilhelminian-style examples.

The trained classifier was then again used to identify the Wilhelminian-style buildings in the whole city of Bonn. Fig. 6 shows the results for the same region used to validate the classifiers in Section 3.3. As we see, the street 'Im Krausfeld', which was previously wrongly labeled as Wilhelminian-style houses was now correctly identified as part of the class "Others".

3.5 *Results and Conclusions*

Out of 57,510 buildings on the complete dataset, 2,629 were predicted to be of the Wilhelminian-style and 54,881 were assigned to the class "Others". As there is no pre-labeled dataset of Bonn, validation of the results can only be based on a visual verification. As manually checking all buildings would be a non manageable effort we limit the validation to the areas and streets where large blocks of Wilhelminian-style houses were predicted and check if streets known to be characterized by the Wilhelminian-style not contained in the training examples were identified successfully.

As a general rule the classifier worked well in streets which are dominated by Wilhelminian-style. Problems occurred in areas where single Wilhelminian-style buildings were part of a street dominated by other buildings. Here the classifier was not always capable of predicting the correct house class. The reason for this is the usage and high relevance of context, represented by the median features as discussed in Section 3.1. These features bring about a bias in favour of the dominating architectural type of a specific street.

In this regard the results from the 'Wolfsst.' are worth being mentioned. This street was part of the training examples of the class "Others" because Wilhelminian-style houses are not predominant. Several buildings, however, were correctly predicted as Wilhelminian-style buildings by the classifier. This example shows that the structural risk minimization works well and is capable of handling a certain amount of wrongly pre-labeled training data examples.

When the buildings are very similar in their geometric characteristics misclassifications happened. This is due to the mentioned limitation of LOD1. This limitation proved to be especilly critical when whole streets of the "Wilhelminian-style-suburb" were destroyed in World War II and replaced by more modern houses with similar characteristics in order to not disturb the urban image in these suburbs.

All in all, the classifier was able to predict the architectural style of buildings with a high accuracy in spite of the limitations of LOD1. Using city models of a higher resolution could further improve the accuracy. Especially the roof type and the facades which are denied in LOD1 could be of great relevance for the classifier.

The LOF outlier detection proved to be a good method for finding outliers. Clustering caused an even more reliable class prediction, as it successfully removed wrongly labeled clusters of buildings from the training dataset. Without the clustering, two whole streets where predicted as Wilhelminian-style, even if a visual control of those streets showed that those were not part of this architectural style.

The running example of this study were Wilhelminian-style houses. The methods presented here, however, are not confined to this specific architectural type. The identification of detached houses, terraced houses or semi-detached houses, for example, is feasible as well. Henn (2009) successfully distinguished between those building types with a slightly different approach using SVMs as well. His study, in contrast, used manually pre-labeled data sets.

Support Vector Machines proved to be eligible for a challenging classification task. The principle of structural risk minimization resulted in a successful generalization of the data set and avoided overfitting to a high degree. The information content of the given LOD1 city model is limited and the variety within the two classes Wilhelminian-style and "Others" is large. There were many misclassified individuals in the training dataset. Nevertheless the classification accuracy was amazingly high. This demonstrates the high potential of SVMs for data mining in 3D city models.

Careful pre-processing, namely the construction of features representing context, outlier elimination and cluster analysis, substantially contributed to the achieved accuracy.

References

Boyd, S. & Vanderberghe, L., 2004: Convex Optimization. – Seventh Edition, Cambridge University Press, Cambridge, UK.

Breunig, M.M., Kriegel, H.-P., Ng, R.T. & Sander, J., 2000: Identifying Density-Based Local Outliers. – ACM SIGMOD Conference.

Chang, C.-C. & Lin, C.-J., 2001: LIBSVM: a library for support vector machines. – www.csie.ntu.edu.tw/~cjlin/libsvm.

Chapelle, O., Schölkopf, B. & Zien, A., 2006: Semi-Supervised Learning – MIT Press, Massachusetts, USA.

Cover, T.-M., 1965: Geometric and statistical properties of systems of linear inequalities with applications in pattern recognition. – IEEE Transactions in Electronic Computers **14:** 326–334.

Ertöz, L., Steinbach, M. & Kumar, V., 2003: Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. – Second SIAM International Conference on Data Mining.

Ester, L., Krieger, H.-P., Sander, J. & Xu, X., 1996: A density-based algorithm for discovering clusters in large spatial databases with noise. – 2nd International Conference on Knowledge Discovery and Data Mining.

Fayyad, U., Piatetzky-Shapiro, G. & Smyth, P., 1996: From data mining to knowledge discovery in databases. – AI Magazine **17:** 37–54.

Goldberg, D. E., 1989: Genetic algorithms in search, optimization, and machine learning. – Addison-Wesley, Boston, USA.

Gunn, S.-R., 1998: Support vector machines for classification and regression. – School of Electronics and Computer Science, University of Southampton, Southampton, UK.

Guyon, I., Weston, J., Barnhill, S. & Vapnik, V., 2002: Gene selection for cancer classification using support vector machines. – Machine Learning **46** (1–3): 389–422.

Heinert, M., 2010: Support Vector Machines – Teil 1: Ein theoretischer Überblick. – Zeitschrift für Vermessungswesen **135** (3): 179–189.

Henn, A., 2009: Prototypische Verfeinerung von 3D-Stadtmodellen mit Methoden des Maschinellen Lernens. – Diplomarbeit, Universität Bonn.

Hodge, V.J. & Austin, J., 2004: A Survey of Outlier Detection Methodologies. – Artificial Intelligence Review **22** (2): 85–126.

Koch, K., 1997: Parameterschätzung und Hypothesentests in linearen Modellen. – 3. Aufl., 381 S., Dümmler, Bonn.

Kohavi, R., 1995: A Study of cross-validation and bootstrap for accuracy estimation and model selection. Fourteenth International Joint Conference on Artificial Intelligence **2** (12): 1137–1143.

Kolbe, T. & Gröger, G., 2003: Towards unified 3D city models. – ISPRS Comm. IV Joint Workshop on Challenges in Geospatial Analysis, Integration and Visualization II.

Krückhans, M. & Schmittwilken, J., 2009: Attributierte Grammatiken zur artifiziellen Verfeinerung von 3D-Stadtmodellen. – DGPF-Jahrestagung.

Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. & Euler, T., 2006: YALE: Rapid prototyping for complex data mining tasks. – ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

Pelleg, D. & Moore, A., 2000: X-Means: Extending k-means with efficient estimation of the number of clusters. – 17th International Conference on Machine Learning: 727–734.

Rockafellar, R.T., 1993: Lagrange multipliers and optimality. – SIAM Review **35:** 183–238.

Römer, C., 2009: Support-Vektor-Maschinen für Data Mining in Stadtmodellen. – Diplomarbeit, Universität Bonn.

Schölkopf, B. & Smola, A., 2002: Learning with Kernels – Support Vector Machines, Regularization, Optimization and Beyond. – MIT Press, Cambridge, www.learning-with-kernels.org.

Schölkopf, B., Sung, K., Burges, C., Girosi, F., Niyogi, P., Poggio, T. & Vapnik, V., 1996: Comparing Support Vector Machines with Gaussian kernels to radial basis function classifiers. – A.I. Memo No. 1599, Massachusetts Institute of Technology.

Schwarz, G., 1978: Estimating the dimension of a model. – Annals of Statistics **6:** 461–464.

Shawe-Taylor, J. & Christianini, N., 2004: Kernel Methods for Pattern Analysis. – Cambridge University Press.

Tax, D.M.J. & Duin, R.P.W., 1999: Support Vector Domain Description. – Pattern Recognition Letters **20** (11–13): 1191–1199.

Vapnik, V., 1998: Statistical Learning Theory. – Wiley-Interscience, New York, USA.

Vapnik, V. & Chervonenkis, A., 1971: On the Uniform Convergence of Relative Frequencies of Events to their Probabilities. – Theory of Probability and its Applications **16** (2): 264–280.

Address of the Authors:

Christoph Römer, Prof. Dr. Lutz Plümer, University of Bonn, Institute for Geodesy and Geoinformation, D-53115 Bonn, Tel.: +49-228-73-1759, -1750, Fax: -1753, e-mail: roemer@igg.uni-bonn.de, pluemer@igg.uni-bonn.de.

Manuskript eingereicht: November 2009
Angenommen: Juli 2010