Predicting Spatial Data with RBF Networks

Tianming Hu*and Sam Yuan Sung
Dept. of Computer Science, National University of Singapore
Singapore 117543

Abstract: Spatial prediction needs to account for spatial information, which makes conventional radial basis function (RBF) networks inappropriate, for they assume independent and identical distribution. In this paper, we fuse spatial information at different layers of RBF. Experiments show fusion at hidden layer gives the best result and suggest that the optimal value is around one for the coefficient, which is used in the linear combination at the output layer.

1 Introduction

Spatial data distinguish themselves from conventional data in that associated with each object(site), the attributes under consideration include not only non-spatial normal attributes, but also spatial attributes which are unique or emphasized and describe the object's spatial information such as location and shape in the 2-D (dimensional) spatial (physical) space.

Independent and identical distribution (iid) is a fundamental assumption often made in data sampling but it is no longer valid for spatial data. In practice, almost every site is related to its neighbors. For example, houses in nearby neighborhoods tend to have similar prices which are affected by one another. Spatial data also exhibit two unique characteristics: spatial trend (large scale variance) and spatial dependence (small scale variance) [1]. Spatial dependence, also called spatial autocorrelation, has two types: positive and negative. Positive correlation means nearby sites tend to have similar characteristics and thus exhibit spatial continuity. In remote sensing images, close pixels usually belong to the same landcover type: soil, forest, etc.

In this paper, we use radial basis function (RBF) network for predicting spatial lattice data. The focus is on how to incorporate spatial autocorrelation into the framework of RBF. In contrast to raw input fusion, we push spatial autocorrelation further into RBF by fusing the output from hidden and output layers. Experimental evaluation shows our extension gives better results than input fusion.

The rest of the paper is organized as follows. A formal problem formulation is given in Section 2, followed by an introduction to related work. Section 3 first reviews RBF for regression, then presents our extension of fusing data at various levels to incorporate spatial information. Experimental evaluation is reported in Section 4 where various fusions are compared and the effect of spatial autocorrelation coefficient is investigated. Section 5 concludes this paper with a summary and discussion on future work.

2 Problem Background

2.1 Problem Formulation

Here we consider lattice data whose site index is countable. Let S denote the set of locations, e.g., the set of triple (index, latitude, longitude). The spatial prediction (regression) problem can be formulated as follows. We are given 1) A spatial framework of n sites,$S = \{s_i\}_{i=1}^n$, with a neighbor relation $N \subseteq S \times S$. Sites s_i and s_j are neighbors iff $(s_i, s_j) \in N, i \neq j$. Let $N(s_i) \equiv \{s_j : (s_i, s_j) \in N\}$ denote the neighborhood of s_i. 2) Associated with each s_i, there is a d-D feature vector of explanatory attributes $\mathbf{x}_i \equiv \mathbf{x}(s_i) \in \Re^d$ and a dependent variable $y_i \equiv y(s_i) \in \Re$ to be predicted. Let $\mathbf{y} \equiv [y_1, ..., y_n]^T$. We need to find a prediction function $f : \Re^d \rightarrow \Re$. Let

*Corresponding author. Email: hutianmi@comp.nus.edu.sg

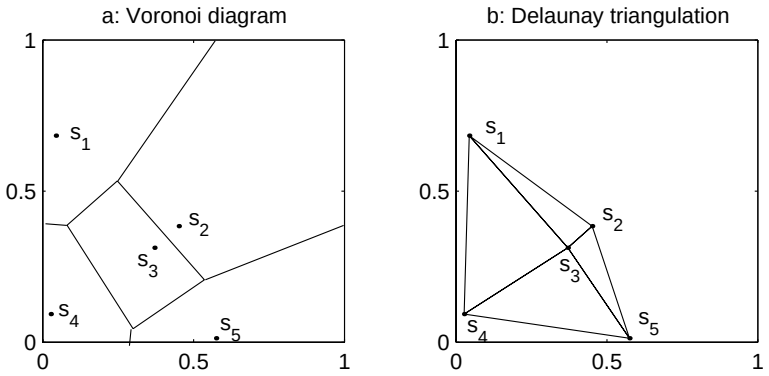

Figure 1: (a): Voronoi diagram. (b): Delaunay triangulation

$\hat{y}_i \equiv f(\mathbf{x}_i)$, $\hat{\mathbf{y}} \equiv [\hat{y}_1, ..., \hat{y}_n]^T$. The objective is to maximize similarity between $\mathbf{y}$ and $\hat{\mathbf{y}}$, e.g., mean squared error (MSE) $\|\mathbf{y} - \hat{\mathbf{y}}\|^2/n$. So far, all items mentioned above are common among normal regression problems, except the additional input, the spatial framework. Besides, the unique characteristics of spatial data lend themselves to the constraint of spatial autocorrelation, that is, y_i is not only affected by $\mathbf{x}_i$, but also by $\mathbf{x}_j$ and y_j of its neighbors $N(s_i)$.

We assume that N is given by a row-normalized contiguity matrix W, where $W(i,j) = 1/|N(s_i)|$ ($|N(s_i)|$ is the number of s_i's neighbors) iff $(s_i, s_j) \in N$ and $W(i,j) = 0$ otherwise. The contiguity matrix W can be computed from sites' latitude-longitude pairs. Two sites are neighbors if they are natural neighbor in Voronoi diagram (Fig. 1(a)) or equivalently, they are linked in the dual Delaunay triangulation (Fig. 1(b)). As shown in Eq. (1), from Voronoi diagram or Delaunay triangulation, the symmetric binary contiguity matrix W_b can be constructed, where $W_b(i,j) = 1$ iff $(s_i, s_j) \in N$ and $W_b(i,j) = 0$ otherwise. The row-normalized contiguity matrix W is obtained from W_b by dividing each element with the sum of its row. Consequently, W is also symmetric in terms of positive/zero. For example, site s_1 in Fig. 1 has three neighbors s_2, s_3 and s_4, so the nonzero elements in the first row of W_b and their counterparts in W are $W_b(1,j) = 1$, and $W(1,j) = 1/3, j = 2, 3, 4$, respectively.

$$W_b = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \overset{\text{normalize}}{\longrightarrow} W = \begin{pmatrix} 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \end{pmatrix} \quad (1)$$

2.2 Related Work

As in classical pattern recognition or data mining, the work on spatial data can be divided into several categories based on the goal: classification, clustering, regression (prediction), etc. A survey from a database perspective is provided in [2] and a comprehensive collection of statistics work is offered in [1]. Several methods exist for taking into account the spatial information: 1) Adding spatial information into dataset [3, 4]. 2) Modifying existing algorithms, e.g., allowing an object assigned to a class if and only if this class already contains its neighbor [5]. 3) Selecting a model that encompasses spatial information [6]. This can be achieved by modifying a criterion function that includes spatial constraints, which mainly comes from the image analysis where Markov random field is intensively used [7]. Another method, where our approach falls, is to directly modify the structure of the model.

Compared to a lot of work in spatial contextual classification [7, 8, 9, 10], spatial prediction receives less attention, not to mention the application of RBF-like local expert network methods. In [11], different machine learning algorithms are applied to non-stationary spatial data analysis: using spatial coordinates to predict the rainfall. Local models, like local version of support vector regression and mixture of experts, which take into account local variability of the data, are found to be better than their global counterparts that are trained

globally on the whole dataset. In [12], RBF coupled map lattice is used as the spatial temporal predictor to model the chaotic dynamic of radar echoes from a sea surface, and to detect embedded targets. The input is fused by weighted averaging each site and its neighbors.

3 RBF for Spatial Prediction

3.1 Conventional RBF

Conventional RBF for prediction has been studied extensively in the literature [13, 14, 15]. It can be described mathematically as a linear combination of nonlinear radially symmetric basis functions, as shown in Eq. (2), where the basis function $\phi_m(z)$ often takes the popular Gaussian kernel in Eq. (3). It is proven in [16] that, given a sufficiently large number M of Gaussian kernels and the freedom to adjust center μ_m and width h_m separately for each kernel, RBF can achieve arbitrarily small error.

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m \left(\frac{|\mathbf{x} - \mu_m|}{h_m} \right) \tag{2}$$

$$\phi_m(z) = \exp(-z^2) \tag{3}$$

In fact, the choice of basis function is less crucial compared to number of centers M and width h_m. M is a hyper-parameter which determines the network structure and its estimation is costly. We select M by trial and error based on a range of values determined by the cross validation. At each iteration the input vector that results in lowering the network error the most, is used to create a hidden neuron (kernel) and it is removed from the training set [17]. This efficient process is repeated until the validation error begins increasing. Once M is determined, centers μ_m can be chosen with k-means algorithm [18].

As for width, too small width would cause underlapping and entail a large number of kernels that lead to overfitting. On the other hand, too large width would cause overlapping and cannot give satisfactory performance. We try three ways to set constant width for all kernels: 1) The average of distance to 10th nearest neighbor (in the input vector space), which is suggested in [19]. 2) The maximum distance between centers divided by $2M$, which is used in [12]. 3) The value h that, for density estimation, minimizes the MSE between the density and the approximation [20]. It has the form in Eq. (4), where $\sigma^2 = \text{trace}(\Sigma)/d$ and Σ is the sample covariance matrix.

$$h = \sigma n^{\frac{-1}{d+4}} \left(\frac{4}{d+2} \right)^{\frac{1}{d+4}} \tag{4}$$

Once the estimation of parameters for radial basis layer is finished, the remaining task of estimating output layer weights $\mathbf{w} = [w_0, ..., w_M]^T$ is essentially a linear regression problem in Eq. (5), where i-th row of matrix Φ is the radial basis output vector for i-th input. It can be solved analytically in Eq. (6) to minimize MSE, where Φ^+ denotes pseudo-inverse $(\Phi^T\Phi)^{-1}\Phi^T$.

$$\mathbf{y} = \Phi\mathbf{w} \tag{5}$$

$$\hat{\mathbf{w}} = \Phi^+\mathbf{y} \tag{6}$$

3.2 Incorporating Spatial Information

Spatial information can be incorporated into RBF at least at three levels: input fusion, hidden fusion and output fusion. Input fusion is tried in [12] for regular lattice data and we adapt it for irregular lattice data. Besides, we push spatial information further into RBF by fusing the output from hidden and output layers.

3.2.1 Input Fusion

Input fusion replaces each input with the weighted average of its neighbors and feeds the new input to a conventional RBF. In [12], the weighting coefficient for each neighbor can be computed for spatial regular lattice data. However, the data used in our experiments are measurement for irregular lattice sites (e.g., counties) where neither the number nor the relative position of neighbors is fixed. We first average all neighbors with $W\mathbf{y}$, then by treating the result $\bar{y}_i$ (i-th element of $W\mathbf{y}$) as the only virtual neighbor for each site s_i, we can compute the correlation coefficient β between y_i and $\bar{y}_i$ in Eq. (7). Similarly, the new fused input vector $\dot{\mathbf{x}}$ can be constructed by fusing the original input $\mathbf{x}_i$ with the average of its neighbors $\bar{\mathbf{x}}_i$ weighted by ρ, as shown in Eq. (8), where $X = [\mathbf{x}_1, ..., \mathbf{x}_n]$, $\bar{\mathbf{x}}_i$ is the i-th column of XW^T, ρ is the correlation coefficient linking $\mathbf{x}_i$ and its virtual neighbor $\bar{\mathbf{x}}_i$ and set equal to β in this case.

$$\beta = \frac{\mathrm{Cov}(y, \bar{y})}{\sigma_y \sigma_{\bar{y}}} \tag{7}$$

$$\dot{\mathbf{x}}_i \equiv \frac{\mathbf{x}_i + \rho \bar{\mathbf{x}}_i}{1 + \rho} \tag{8}$$

3.2.2 Hidden Fusion

Hidden fusion refers to incorporating spatial autocorrelation into the output Φ from hidden radial basis layer by modifying the linear combination in Eq. (5). We develop two modifications: hidden fusion 1 in Eq. (9) and hidden fusion 2 in Eq. (11), with their MSE solution in Eq. (10) and (12), respectively. ρ is set equal to β in Eq. (7) and kept fixed in solving for MSE solution. Intuitively, Eq. (9) can be interpreted as y is a linear combination of the prediction by its own attributes and by its neighbors. Eq. (11) is obtained from Eq. (9) by replacing $\Phi\mathbf{w}$ on its right-hand side with $\mathbf{y}$, i.e., the prediction replaced by the true value.

$$\mathbf{y} = \Phi\mathbf{w} + \rho W \Phi\mathbf{w} \tag{9}$$

$$\hat{\mathbf{w}} = (\Phi + \rho W \Phi)^{+}\mathbf{y} \tag{10}$$

$$\mathbf{y} = \Phi\mathbf{w} + \rho W \mathbf{y} \tag{11}$$

$$\hat{\mathbf{w}} = ((I - \rho W)^{-1}\Phi)^{+}\mathbf{y} \tag{12}$$

3.2.3 Output Fusion

Output fusion is just opposite input fusion. Instead of substituting the input with the weighted average of neighbors, we can train a conventional RBF on the original input as usual and then fuse the output with the average of neighbors. It is similar to the post-processing in spatial contextual classification after pixel-wise classification is finished. Formally, the new prediction $\dot{\hat{\mathbf{y}}}$ by output fusion is given in Eq. (13), where ρ is set equal to β in Eq. (7) and kept fixed, $\hat{\mathbf{y}} = \Phi\hat{\mathbf{w}}$ denotes the prediction by a conventional RBF and $\hat{\mathbf{w}}$ is given in Eq. (6).

$$\dot{\hat{\mathbf{y}}} \equiv \frac{\hat{\mathbf{y}} + \rho W \hat{\mathbf{y}}}{1 + \rho} \tag{13}$$

4 Experimental Evaluation

We evaluate various fusions on two real datasets, a small crime dataset and a large election dataset, both available at [21]. In the crime dataset, household income and housing values in 49 neighborhoods in Columbus Ohio, USA, are treated as explanatory attributes to predict crime rate, which is shown in Fig. 2(a). In the election dataset, income, home ownership and population with college degrees in 3107 counties are used to predict the voting rate for 1980 USA presidential election. We can see spatial continuity exists that neighboring sites tend to have similar values. Besides, spatial trend is also obvious. In Fig. 2(a), high (low) crime sites are located in central (outside) area.

Table 1: MSE of conventional RBF and various fusions.

	RBF	IF	HF1	HF2	OF
crime	114	92	92	84	105
election	0.0057	0.0059	0.0053	0.0051	0.0057

Table 2: Spatial autocorrelation coefficient β of $\mathbf{y}$ and various $\hat{\mathbf{y}}$.

	true	RBF	IF	HF1	HF2	OF
crime	0.76	0.51	0.86	0.82	0.88	0.84
election	0.76	0.69	0.83	0.87	0.93	0.90

4.1 Fusion Comparison

Experiments show width in Eq. (4) always gives the best or comparable to best results, so we only report its results. The numbers of centers, 5 for crime data and 100 for election data, are obtained with cross validation on conventional RBF and they are also applied in other fusions. For each dataset, there are two sets of centers, one for input fusion and the other for hidden/output fusion and conventional RBF.

In principle, for the test set, we must use the data for the same area but in a different year, which are unfortunately unavailable. Neither can we use cross validation by partitioning the training set into N subsets, for one site's neighbor, which is needed in various fusions, may be in another subset. Thus we can only compare various models on the same training set. For fair comparison, we generate 10 sets of centers using k-means algorithm with random initialization and early stop. The average results are reported in Table 1, where RBF, IF, HF1, HF2, and OF stand for conventional RBF, input fusion, hidden fusion 1 in Eq. (9), hidden fusion 2 in Eq. (11) and output fusion, respectively. Compared to conventional RBF, incorporating spatial autocorrelation by fusion at different levels generally reduces MSE with varying success. Fusing output from hidden layer gives better results than those of fusing data at two ends: raw input and final output. HF2 achieves the most significant MSE reduction on both datasets.

4.2 Effect of Coefficient ρ

Since we use $\rho = \beta$, the autocorrelation coefficient of the true value $\mathbf{y}$ in fusion, it is interesting to check the autocorrelation coefficient for various prediction $\hat{\mathbf{y}}$. The new autocorrelation is still obtained with Eq. (7) where $\mathbf{y}$ is replaced by $\hat{\mathbf{y}}$ and the results are listed in Table 2. Compared to the spatial autocorrelation of the true value, the prediction by conventional RBF, without any fusion, yields a lower autocorrelation. On the other hand, all fusions lead to a higher autocorrelation in their prediction.

Because the highest autocorrelation is achieved by HF2, which also achieves the lowest MSE, a natural question arises if performance of HF2 can be improved further by varying coefficient ρ in Eq. (11), especially by increasing it. In contrast to multi-layer perceptron which requires the costly error back-propagation, the major advantage of RBF is in its quick training. In particular, the parameters of linear output layer can be solved analytically to minimize MSE, which is only feasible with a fixed ρ. Otherwise, ρ also needs to be estimated jointly with $\mathbf{w}$ using techniques such as Monte Carlo sampling that is computationally expensive. So it is crucial to see if we can find an optimal value for ρ.

We try a wide range $[0, 2]$ for ρ and illustrate the results in Fig. 2(b-f) for crime and election data, respectively. Note that when $\rho = 0$ in Eq. (11), HF2 is reduced to conventional RBF. Generally, ignoring ($\rho = 0$) and over-emphasizing ($\rho = 2$) spatial autocorrelation both lead to poor results. The former loses the spatial continuity by allowing very different sites close to one another, e.g., high and low crime sites are mixed together in the rightmost part in Fig. 2(b). The latter usually outputs blurred result, e.g., all sites in Fig. 2(d) receive moderate values. As shown in Fig. 2(e) and (f), the lowest MSE is achieved around $\rho = 1$ for both

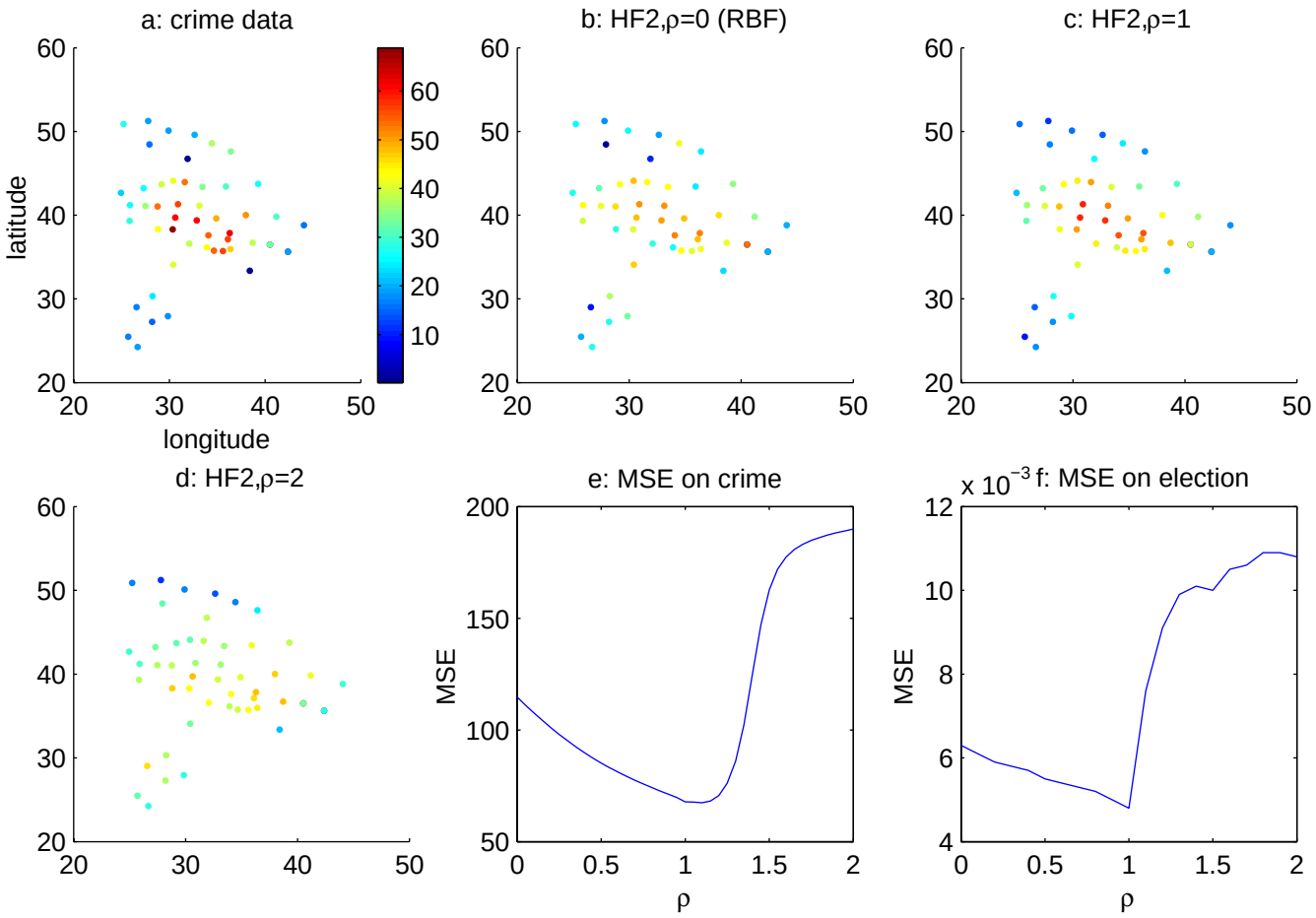

Figure 2: Crime data (a) and its prediction by HF2 with various ρ (b-d). MSE vs ρ on crime data (e) and on election data (f).

datasets.

5 Conclusion

Like other machine learning methods, conventional RBF for prediction assumes iid and ignores spatial information. In this paper, we investigated various possibilities of incorporating spatial autocorrelation into RBF at input, hidden and output levels by fusing data belonging to the same neighborhood in the spatial space. Experiments on two real datasets show hidden fusion, HF2, always gives the best results over conventional RBF and other fusions. However, like total ignorance of spatial autocorrelation in conventional RBF, over-emphasizing it also leads to poor results. Experiments suggest that the optimal value is around $\rho = 1$ for the coefficient, which is used to linearly combine each site with its neighbors at the output layer.

There are other candidate places where spatial information can be pushed into RBF. For instance, the center selection, which is achieved with k-means in our paper, plays a vital role in prediction performance and different clustering techniques apparently would give different results. However, they are all performed in the attribute space and no spatial information is taken into account. It is reasonable that we hope data belong to the same center are also close in the spatial space, provided spatial continuity exits. A more ambitious requirement is that the center label can tell more about the dependent variable. This can be done by optimizing mutual information $I((Y, S), M)$ or conditional entropy $H(Y, S|M)$, where M denotes the unknown center label whose distribution needs to be estimated, Y denotes the dependent variable and S denotes the spatial location. To make computation feasible, Y needs to be discretized and S needs to be clustered, which poses additional challenges.

References

[1] N. A. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, revised edition, 1993.

[2] M. Ester, H.P. Kriegel, and J. Sander. Spatial data mining: A database approach. In *Proceedings of 5th Symposium on Spatial Databases*, pages 47–66, 1997.

[3] A. K. Jain and F. Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

[4] M. A. Oliver and R. Webster. A geostatistical basis for spatial weighting in multivariate classification. *Mathematical Geology*, 21:15–35, 1989.

[5] P. Legendre. Constrained clustering. In *Developments in Numerical Ecology*, pages 289–307, 1987. NATO ASI Series G 14.

[6] C. Ambroise and G. Govaert. Convergence of an EM-type algorithm for spatial clustering. *Pattern Recognition Letters*, 19(10):919 – 927, 1998.

[7] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[8] A. H. Solberg, T. Taxt, and A. K. Jain. A markov random field model for classification of multisource satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 34(1):100–113, 1996.

[9] S. Shekhar, P. Schrater, W. R. Raju, W. Wu, and S. Chawla. Spatial contextual classification and prediction models for mining geospatial data. *IEEE Transactions on Multimedia*, 4(2):174–188, 2002.

[10] L. Hermes and J. M. Buhmann. Contextual classification by entropy-based polygonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 442–447, 2001.

[11] N. Gilardi and S. Bengio. Local machine learning models for spatial data analysis. *Journal of Geographic Information and Decision Analysis*, 4(1):11–28, 2000.

[12] H. Leung, G. Hennessey, and A. Drosopoulos. Signal detection using the radial basis function coupled map lattice. *IEEE Transactions on Neural Networks*, 11(5):1133 –1151, 2000.

[13] M. J. D. Powell. Radial basis functions for multivariable interpolation: A review. In *Algorithms for Approximation*, pages 143–167. Oxford: Clarendon Press, 1987.

[14] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[15] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[16] E. J. Hartman, J. D. Keller, and J. M. Kowalski. Layered neural networks with gaussian hidden units as universal approximations. *Neural Computation*, 2(2):210–215, 1990.

[17] S. Chen, C. F. N. Cowan, and P. M. Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2(2):302–309, 1991.

[18] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

[19] D. J. Hand. *Discrimination and Classification*. John Wiley & Sons, 1981.

[20] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[21] J. P. LeSage. *MATLAB Toolbox for Spatial Econometrics*. http://www.spatial-econometrics.com, 1999.