A Feature Selection Strategy for the Relevance Vector Machine

ARMIN SHMILOVICI, DAVID BEN-SHIMON
Department of Information Systems Engineering
Ben-Gurion University of the Negev
P.O.B 653, 84105 Beer-Sheva
ISRAEL
armin@bgu.ac.il http://www.ise.bgu.ac.il/faculty/armin/

Abstract: - The Relevance Vector Machine (RVM) is a generalized linear model that can use kernel functions as basis functions. The typical RVM solution is very sparse. We present a strategy for feature ranking and selection via evaluating the influence of the features on the relevance vectors. This requires a single training of the RVM, thus, it is very efficient. Experiments on a benchmark regression problem provide evidence that it selects high-quality feature sets at a fraction of the costs of classical methods.

Key-Words: - Feature Selection, Relevance Vector Machine, Machine Learning

1 Introduction

Consider is a machine learning method for training a generalized linear model (GLM) such as

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i \cdot k(\mathbf{x}, x_i) + \varepsilon \qquad (1)$$

where $\mathbf{x} = [x_1, \ldots, x_M]^T$, $k(x, x_i)$ is a bivariate kernel function centered on each one of the N training data points, $x_i \in R^M$ is a vector of dimensionality M, $w = [w_1, \ldots, w_N]^T$ is a vector of regression coefficients, and ε is the noise. Sparsity is generally considered a desirable attribute of a machine learning algorithm. In the context of GLM, a sparse solution will have a small number of non-zero coefficients in w. The Relevance Vector Machine (RVM) [1] is a machine learning algorithm for the GLM that will often generate a sparse solution.

Feature selection [2], also known as variable selection, is a process commonly used in machine learning, wherein a subset of the features x_i is selected for the application of a learning algorithm. Feature selection is necessary either because it is computationally expensive to use all available features, or when some features are suspected to be highly correlated or non-informative (noise features). The feature selection process detects the most informative features.

Feature selection methods can be distinguished as either *filter methods* [3] – which use general criteria independent of the learning algorithm to select a feature subset; *wrapper methods* [4] – consider the learning algorithm as a black box used to assist in searching for the best set of features; and *embedded approaches* [7] simultaneously determine features and model structure during the model training phase.

The RVM is a computationally intensive algorithm for large datasets [5]. A naïve implementation of the RVM algorithm [1] has a complexity of $O(N^3)$ where N indicates the number of the examples, therefore when there are many features, an exhaustive search in the space of all possible feature subsets is computationally expensive. In [1] it is suggested to modify the kernel of the RVM with an additional multiplicative parameter for each feature, and to optimize the values of those extra parameters with a gradient descent like algorithm. After convergence, a close to zero parameter would indicate an irrelevant feature. This algorithm requires numerous evaluations of the RVM, which makes it impractical for datasets with many features.

The Support Vector Machine (SVM) [6] is another sparse method for training a GLM. In [7] an embedded method was suggested which exploits the sparsity of the SVM solution and requires a *single* evaluation of the SVM. The heuristics ranks the features according to their average influence evaluated at the support vectors. Features with little effect on the support vector are practically irrelevant.

The RVM has the same functional form as the SVM (though it typically produces sparser solutions [1]). *The contribution in this paper* is in adapting the heuristics of [7] to the RVM. The benefit of the method is that the RVM has to be trained *only once*, then, evaluated on the RVs to detect features with

ISBN: 978-1-61804-162-3

small influence on the solution. After selecting the best subset of features, the RVM may be computed once again for obtaining the best solution. This process is practical even for large datasets with many features.

Our experiments on a benchmark dataset demonstrate that the feature ranking method is indeed viable. The best features were identified and the solution found with the reduced features RVM turned out to have the same or better accuracy and the same or better sparseness than the original solution.

The rest of this paper is as follows: section 2 introduces the RVM algorithm; section 3 presents the feature selection heuristic; section 4 presents some experiments with a benchmark dataset; and section 5 concludes with a discussion.

2 The RVM Algorithm

2.1 The Regression RVM

Consider a dataset of input-target pairs, $\{x_i, t_i\}_{i=1}^N$ where targets are assumed, *jointly* Normal distributed - $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are the unknowns to be determined by the algorithm. Each target t_i is also assumed Normally distributed with mean $y(x_i)$ and uniform variance σ^2 of the noise ε so $p(\mathbf{t} \mid \mathbf{x}) = N(\mathbf{t} \mid \mathbf{y}(\mathbf{x}), \sigma^2)$. Following those assumptions the conditional probability of the targets given the parameters and the data can be expressed as

$$p(\mathbf{t} \mid \mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-\frac{N}{2}} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}\|\right\} \quad (2)$$

where the data is hidden in the *NxN* kernel function matrix $\boldsymbol{\Phi}$ representing all the pairs $\phi_{i,j} = k(x_i, x_j)$, $i, j \in [1, \ldots, N]$. $\boldsymbol{\Phi}$ could be extended to include a possible bias term.

The goal of the RVM is to accurately predict the target function, while retaining as few basis functions as possible in (1). Sparseness is achieved via the framework of sparse Bayesian learning [8] and the introduction of an additional vector of hyper parameters α_i that controls the width of a Normal prior distribution over the precision of each element of w_i.

$$p(w_i \mid \alpha_i) = \sqrt{\frac{\alpha_i}{2\pi}} \exp(-\frac{1}{2}\alpha_j {w_j}^2) \quad (3)$$

The solution is derived via the following iterative type II maximization of the marginal likelihood $p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2)$ with respect to $\boldsymbol{\alpha}$ and σ^2:

$$\alpha_i^{new} = \frac{1 - \alpha_i \Sigma_{ii}}{{\mu_i}^2} \quad (4)$$

$$(\boldsymbol{\sigma}^2)^{new} = \frac{\|\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{\mu}\|^2}{N - \sum_{i=1}^{N}(1 - \alpha_i \Sigma_{ii})} \quad (5)$$

The unknowns $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are computed as

$$\boldsymbol{\Sigma} = (\boldsymbol{\Phi}^{\mathbf{T}}\mathbf{B}\boldsymbol{\Phi} + \mathbf{A})^{-1} \quad (6)$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\Phi}^{\mathbf{T}}\mathbf{B}\mathbf{t} \quad (7)$$

where $\mathbf{B} \equiv \sigma^{-2}\mathbf{I}_{NxN}$. The basic RVM algorithm cycles between (4),(5),(6),(7), reducing the dimensionality of the problem when any α_i larger than a preset threshold. The algorithm stops when the likelihood $p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2)$ ceases to increase. The non-zero elements of $\boldsymbol{w}$ are called Relevance Values, and their corresponding data-points are called Relevance Vectors (RVs) as an analogy to the Support Vector Machine [6].

2.2 The Classification RVM

In the binary classification problem each target t_i is Binary: $t_i \in \{0,1\}$. The model (1) is assumed to be noise-free. That is $\sigma^2 \equiv 0$. The sigmoid function $\rho(y) = 1/(1 + e^{-y})$ is used to generalize the linear model. The main idea of the sigmoid function is to make an approximation of the regression case to the binary classification problem. With the sigmoid link function we can adopt the Bernoulli distribution $p(t \mid x)$ and rewrite the likelihood as:

$$p(t \mid w) = \prod_{i=1}^{N} \rho(\phi_i^T w)^{t_i} \left(1 - \rho(\phi_i^T w)\right)^{1-t_i} \quad (8)$$

In the classification RVM framework, we need to solve two different coupled problems, an optimization problem and a regression RVM problem [1]. The multi-class problem is solved with an assembly of binary classifiers. The classification RVM will not be considered in this paper.

2.3 Attributes of the RVM

The RVM is an approximate Bayesian method, thus it can generate not only predicted values, but also the probability distribution of the values. For further details of the basic RVM and for a discussion of convergence and sparseness look in [1],[9].

The matrix inversion operation in (6), which

requires $O(N^3)$ operations is the computationally intensive part of the algorithm. The matrices $\mathbf{\Phi}$ and $\mathbf{\Sigma}$ are full rank, thus require initially $O(N^2)$ space complexity. Furthermore, it is common that the inversion of a large matrix becomes ill-conditioned after several cycles even for positive definite matrices unless the parameters of the kernel function are optimized. These problems limit the practicality of the basic RVM algorithm for moderately sized problems. Fortunately, practical approaches were developed for reducing the runtime complexity to $O(N^2)$ [5],[10].

An important step in GLM learning is to find a feature space – a projection of the data on highly dimensional space – where the data is linear for regression problems and linearly separable for classification problems. The choice of projection (the kernel function) is important for the accuracy and the convergence of the RVM [9],[11]. Note that the RVM typically produces very sparse solutions compared to the SVM, when its kernel is as the SVM kernel [1].

3 The Feature Selection Heuristic

Motivation for the heuristic: when $f(\boldsymbol{x})$ is the function defined by equation (1) and e_k is the unit vector in direction k, The influence of the feature x_k on any given point $\boldsymbol{x}$ in the input space can be defined as the squared[1] partial derivative of the function $f(\boldsymbol{x})$ with respect to the feature x_k:

$$d_{k,x} = \left(\left\langle \nabla f(x), e_k \right\rangle\right)^2 \tag{9}$$

The purpose of squaring the partial derivative is to disregard the sign of the derivative and to amplify large partial derivatives.

Normally, the SVM solution $y(x) = \Phi_{x,x_i} w$; where Φ_{x,x_i} is the kernel matrix; is determined by a very small set of Relevance Vectors $RV = \{x_i : w_i > 0\}$ that shape the function $y(\boldsymbol{x})$ as a good approximation to the function $f(\boldsymbol{x})$. Therefore, features which have a very small effect on $f(\boldsymbol{x})$ everywhere in the input space (such as noise features) will also have a very small effect on $y(\boldsymbol{x})$. Considering the following approximation to the partial derivative:

$$\frac{\partial}{\partial x_k} f(x) \cong \lim_{\delta \to 0} \frac{f(x + \delta \cdot e_k) - f(x)}{\delta}$$

we obtain the following approximation to equation (9) for a specific Relevance Vector $\boldsymbol{x_i}$:

$$d_{k,x_i} \cong [\lim_{\delta \to 0} \frac{f(x + \delta \cdot e_k) - f(x)}{\delta}]^2 \approx [\frac{(\Phi_{x,x_i + \delta \cdot e_k} - \Phi_{x,x_i}) w}{\delta}]^2 \tag{10}$$

where $\delta = 10^{-4}$ in our case.

Following the idea of [7], since the relevance vectors are sufficient to determine a GLM approximation (1) to the dataset, a good measure of the importance d_k of feature k is its average influence evaluated at the Relevance Vectors:

$$d_k = \frac{1}{|RV|} \sum_{x_i \in RV} \frac{d_{k,x_i}}{\sum_k d_{k,x_i}} \tag{11}$$

$|RV|$ is the number of Relevance Vectors. The numerator in (11) is the importance of a feature k for a specific Relevance Vector x_i. Note that the denominator $\sum_k d_{k,x_i}$ ensures that each Relevance Vector's contribution sums to one, when we have no reason to believe that one relevance vector is more important than the others in terms of reducing the overall approximation error.

After computing the importance measure $\{d_k\}_{k=1....d}$ for each one of the features, we rank them according to their order of magnitude. For example, Fig. 1 illustrates a typical result for the RVM-based feature selection, for the synthetic benchmark dataset *Pumadyn-32fh* which is fairly linear, highly noisy, and with 32 features (further details in the next section). The vertical axis denotes the d_k values and the horizontal axis denotes the features. As can be seen, there are five features with importance values significantly larger than those of the remaining 27 features. Fig. 2 illustrates another example, for the synthetic benchmark dataset *Pumadyn-8nh* which is non-linear, highly noisy, and with 8 features. In this case, two features demonstrate importance values significantly larger than those of the remaining 3 features.

In the typical cases demonstrated in Fig. 1 and Fig. 2, a simple heuristic can be devised that selects the best features *without* further evaluations of the RVM:

- a) Rank the features according to their decreasing importance values;
- b) The "cutoff" feature is the feature that demonstrates a significant drop in its

[1] As detailed in [7], for the case of a linear SVM classifier, this formula is *exact*. For the RVM with a general kernel – this term is only an approximation of the true influence.

ISBN: 978-1-61804-162-3

importance value, e.g. it drops to 20% of the importance of its neighbor. The solution is the sub-set of features before the "cutoff".

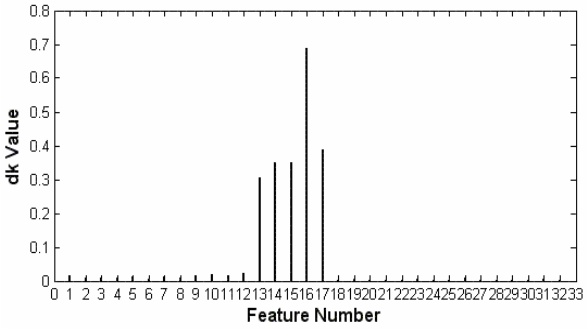

Fig. 1: The resulting dk values for the Pumadyn-32fh dataset. Features 13, 14, 15, 16, 17 are the most important features

In the general case, it may not always be obvious where the "cutoff feature" is; thus, we need to measure the effect of our feature sub-set selection on the training set error. Unfortunately, in that case, we need to *retrain* the RVM with the reduced set of features. The change in the number of features may change the kernel basis functions to the point that the RVM solution is not of maximum accuracy, and the kernel parameters need to be re-optimized. In order to save on the number of RVM trainings, we can use a recursive two phase heuristic like the following:

- *Phase I*: as in the previous heuristics, identify a "candidate cutoff" feature after the first drop to 50% of the feature importance value. Retrain the RVM with the current sub-set of features whose importance is larger than that of the "candidate cutoff". Compute the training set error.
- *Phase II*: add another feature, to the current sub-set of features, and again retrain the RVM using this sub-set of features. Compute the training set error.
- *If* the additional feature reduces the training set error – repeat *Phase II* with an additional feature; *Else* remove the last feature from the current sub-set of features and stop.

According to [7], to select a subset of q features from a set of s given features requires $(q^2+q(2r+1))/2$ RVM evaluations with a wrapper method where the number of removed features is $r=(s-q)$. While a hill climbing method like the second heuristic requires at most $(r+1)$ RVM evaluations and typically less than 10 RVM evaluations. The first heuristic we present requires only two RVM evaluations.

The next section presents some experiments. It turned out that the first heuristic was sufficient for detecting the best sub-set of features.

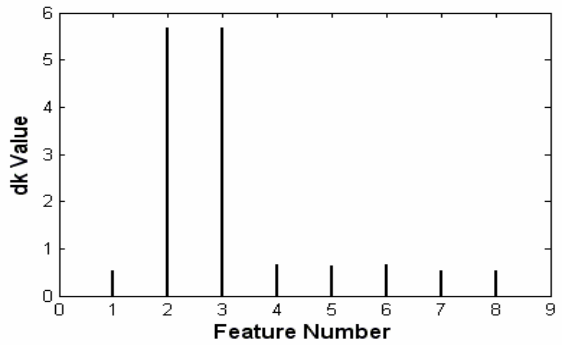

Fig. 2: The resulting *dk* values on the Pumadyn-8nh dataset. Features 2, 3 are the most important features.

4 Comparative Experiments

The purpose of the following experiments is to demonstrate the viability of the feature selection strategy for the RVM on a benchmark dataset. The feature selection algorithm has to demonstrate that it neither deteriorates the accuracy nor the sparseness of the original solution.

Instead of taking a set of unrelated benchmark data sets, we used the *Pumadyn* family of datasets[2]. These are eight synthetic datasets generated from a realistic simulation of the dynamics of the Puma 560 robotic manipulator. The regression problem is to predict the angular acceleration of one of the robotic links. Each dataset has a unique combination of three attributes: dimensionality (8 or 32 attributes), non-linearity (fairly linear or non-linear), and output noise (moderate or high). Table 1 present the details about the datasets and the 75%/25% split between the training set and the test set. We selected this specific set of benchmark datasets in order to study the dependency of the feature selection algorithm on the attributes of the dataset (such as non-linearity).

We used a MATLAB implementation of the RVM from [10]. The Training set was used for the learning phase, and the error was measured on the testing set. The Gaussian kernel function was used in the experiments, and its width parameter was optimized for each training set via cross-validation experiments. We used two measures in these experiments: the number of RVs; and the accuracy (Root Mean Square Error). Each dataset was

[2]www.cs.toronto.edu/~delve/data/pumadyn/desc.htmwww.cs.toronto.edu/~delve/data/pumadyn/desc.html

ISBN: 978-1-61804-162-3

simulated for 10 different repetitions (with permutations of the dataset to avoid dependency on the data order). The mean as well as the standard deviation was computed for each set of 10 different repetitions. The left columns in Table 2 present the number of RVs and the RMSE respectively. The right columns in Table 2 present the comparative results after simulating the RVM on the same datasets with the *reduced feature-set* resulting from the first feature selection algorithm of the previous section**.**

Analysis of the results in Table 2 indicates that:

- The RVM feature selection strategy has obtained a significant feature reduction: 2 features were selected for the datasets with 8 features, and 5 features were selected for the datasets with 32 features.
- The *same* 2 features (respectively 5 features) were selected for each one of the four puma8 datasets (respectively the puma32 datasets). Considering that the different datasets results from simulating the *same* robotic system with a different contamination with noise and non-linearity, the algorithm seems fairly robust to noise and non-linearity, and succeeded to detect the true significant features of the datasets (since some features are dummy features).
- Though it may be a special characteristic of these datasets, it seems that the feature selection did not reduce the accuracy of the solution, moreover, for the puma8 datasets it seems that the accuracy has improved following feature selection.
- The feature selection did not reduce the sparseness of the solution. Moreover, for the puma8 datasets the sparseness has improved significantly after feature selection. Considering that some features contain noise instead of information – this might be expected.
- In most cases, the feature selection seems to reduce the standard deviation of the simulation results. This is a desirable characteristic, since the RVM solution is not unique, and may depend on the randomization of the dataset [1].

These experiments demonstrate that the proposed feature selection heuristics can find the best features irrespective of high dimensionality, nonlinearity or noise.

Table 1. Details of the Pumadyn family of datasets

Name	#Features	Level of noise	Level of non linearity	Training set /test set
puma8fh	8	high	fairly linear	6144/2048
puma8fm	8	moderate	fairly linear	6144/2048
puma8nh	8	high	non-linear	6144/2048
puma8nm	8	moderate	non-linear	6144/2048
puma32fh	32	high	fairly linear	6144/2048
puma32fm	32	moderate	fairly linear	6144/2048
puma32nh	32	high	non-linear	6144/2048
puma32nm	32	moderate	non-linear	6144/2048

Table 2. Comparative Accuracy Before and After Feature Selection

Name	Before Feature Selection			After Feature Selection		
	#Feat.	#RV	RMSE	#Feat.	#RV	RMSE
puma8fh	8	37.4±2.8	3.16±0.04	2	8.3±0.48	3.14±0.027
puma8fm	8	70.2±4.2	1.05±0.012	2	10.5±0.84	1.03±0.012
puma8nh	8	145.1±5	3.28±0.06	2	14.4±0.8	3.15±0.017
puma8nm	8	81.8±3.1	1.26±0.02	2	12±0.8	1.23±0.011
puma32fh	32	79.8±4.8	.02±.0002	5	73.1±4.5	.021±.00001
puma32fm	32	203±22.7	.005±.0001	5	89.3±6.1	.005±.00001
puma32nh	32	9.1±1.4	.033±.0004	5	10.3±3	.033±.00003
puma32nm	32	64.4±8	.027±.0003	5	63.1±14	.027±.00002

ISBN: 978-1-61804-162-3

5 Discussion

Feature selection is an important problem in machine learning. Selecting the optimal subset of features from the space of all possible features is computationally intensive, thus, many heuristics have been proposed to reduce the search. Nevertheless, most heuristics still require several computationally expensive evaluations of the machine learning algorithm for different combinations of features.

The RVM is a kernel method that typically generates very sparse solutions. The RVM is a computationally intensive algorithm for large datasets, therefore, when there are many features in the data, feature selection heuristics that require many evaluations of the RVM are computationally expensive.

In [7] a feature selection heuristics was suggested for the SVM that ranks the features according to their average influence evaluated at the support vectors. Features with little effect on the support vector are practically irrelevant. The RVM has the same functional form as the SVM (though it typically produces sparser solutions). The contribution in this paper is in proving the viability of this idea to the RVM. The benefit of the method is that the RVM has to be trained only once, before evaluated on the RVs to detect features with small influence on the solution. After selecting the best subset of features, the RVM may be computed once more for obtaining the best solution. This process makes it practical even for large datasets with many features. Moreover, the proposed feature selection algorithm may be used as a filter for preprocessing data before activation of a more complex machine-learning method (e.g., Artificial Neural Network).

Our experiments on a benchmark dataset demonstrated that the feature ranking method is indeed viable – the best features were identified and the solution found with the reduced features RVM turned out to have the same or better accuracy and the same or better sparseness than the original solution. Obviously, this point has to be validated also for the classification problems.

Please note that the features' ranks may also be used to improve the numerical stability of the RVM algorithm: The RVM algorithm uses multiple matrix inversion of an expression of the kernel. Scaling each feature value by its rank before computing the kernel matrix is expected to generate a kernel matrix with a smaller condition number, thus, numerically stable.

References:

[1] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine", *J. of Machine Learning Research* Vol. 1, 2001, pp. 211-244.

[2] I. Guyon, S. Gunn, M. Nikravesh, L. Zadeh, (Eds.), *Feature Extraction, Foundations and Applications*, Series Studies in Fuzziness and Soft Computing, Springer, 2006.

[3] Y.E. Chen, C.J. Lin, "Combining SVMs with Various Feature Selection Strategies", *Studies in Fuzziness and Soft Computing*, Vol. 207, Physica-Verlag, Springer, 2006, pp. 711-712.

[4] R. Kohavi, J.H. John, "Wrappers for Feature Subset Selection", *Artificial Intelligence*, Vol. 97, No. 2, 1997, pp. 273-324.

[5] M. Tipping, A. Faul, "Fast Marginal Likelihood Maximization for Sparse Bayesian Models" In C. Bishop, B., Frey, (Eds.): *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics,* January 3–6 2003, Key West, Florida.

[6] N. Cristianini, J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, 2003.

[7] M. Heiler, D. Cremers, C. Schnorr, "Efficient Feature Subset Selection for Support Vector Machines", 2001, *Technical Report 21/2001*, Computer Vision, Graphics, and Pattern Recognition Group, Dept. of Mathematics and Computer Science University of Mannheim.

[8] M.E. Tipping, "Bayesian inference: An introduction to principles and practice in machine learning", In O. Bousquet, U. von Luxburg, and G. Rätsch (Eds.), *Advanced Lectures on Machine Learning*, Springer, 2004, pp. 41–62.

[9] A. Schmolck, R. Everson, "Smooth relevance vector machine: a smoothness prior extension of the RVM*"*, *Machine Learning*, Vol. 62, No. 2, 2007, pp. 107-135.

[10] D. Ben-Shimon, A. Shmilovici, "Accelerating the Relevance Vector Machine via Data Partitioning", *J. of Computing and Decision Sciences*, Vol. 31, No. 1, 2006, pp. 27-41.

[11] D. Ben-Shimon, A. Shmilovici, "Kernels for the Relevance Vector Machine – An Empirical Study", in M. Last, P.S. Szcepaniat, Z. Volkovich, A. Kandel, (Eds.), *Algorithmic Techniques of Data Mining, Advances in Web Intelligence and Data Mining*, Studies in Computational Intelligence Series, Springer, 2006, pp. 253-264.