ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

Algebraic multigrid support vector machines

Ehsan Sadrfaridpour[1], Sandeep Jeereddy[2], Ken Kennedy[2], Andre Luckow[2]
Talayeh Razzaghi[1], Ilya Safro[1]

1- Clemson University, School of Computing, Clemson SC, USA

2- Innovation Lab, BMW Group IT Research Center, Information Management Americas
Greenville SC, USA

Abstract. The support vector machine is a flexible optimization-based technique widely used for classification problems. In practice, its training part becomes computationally expensive on large-scale data sets because of such reasons as the complexity and number of iterations in the parameter fitting methods, underlying optimization solvers, and nonlinearity of kernels. We introduce a fast multilevel framework for solving support vector machine models that is inspired by the algebraic multigrid. Significant improvement in the running has been achieved without any loss in the quality. The proposed technique is highly beneficial on imbalanced sets. We demonstrate computational results on publicly available and industrial data sets.

1 Introduction

Soft margin support vector machine (SVM) is a classification method in which the optimal classifier is achieved through solving a convex quadratic programming (QP) model that typically scales between $\mathcal{O}(n_f n_s^2)$ to $\mathcal{O}(n_f n_s^3)$, where the numbers of features, and samples are denoted by n_f and n_s, respectively. Clearly, this complexity is prohibitive for kernel based SVM applied on big data.

Given n data points $\{x_i\}_{i=1}^n$ in $\mathbb{R}^d$, we define the corresponding labeled pairs (x_i, y_i), where each x_i belongs to the class determined by the given label $y_i \in \{-1, 1\}$. Classes of of points with positive and negative labels are denoted by $\mathbf{C}^+$, and $\mathbf{C}^-$, respectively, where $|\mathbf{C}^+| = n^+$, and $|\mathbf{C}^-| = n^-$. Finding w, and b produces the hyperplane with maximum margin between $\mathbf{C}^+$, and $\mathbf{C}^-$

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ & \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned} \tag{1}$$

The mapping of points to higher dimensional space is done by $\phi : \mathbb{R}^d \to \mathbb{R}^p$ $(d \leq p)$ to make two classes separable by a hyperplane. The term slack variables $\{\xi_i\}_{i=1}^n$ are used to penalize the misclassified points with $C > 0$ that controls the penalization. The weighted SVM (WSVM) addresses imbalanced problems with assigning different weights to classes with parameters C^+ and C^-. The set of slack variables is split into two disjoint sets $\{\xi_i^+\}_{i=1}^{n^+}$, and $\{\xi_i^-\}_{i=1}^{n^-}$, respectively. In WSVM, the objective of (1) is changed to $\frac{1}{2}\|w\|^2 + C^+\sum_{i=1}^{n_+}\xi_i^+ + C^-\sum_{j=1}^{n_-}\xi_j^-$. In all (W)SVM models, we use the Gaussian kernel $\exp(-\gamma\|x_i - x_j\|^2)$. Overall, in WSVM model, three parameters (C^+, C^-, and γ) require tuning which is one of the main reasons of high complexity

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

of underlying QP solvers. Typically, parameter tuning techniques (such as the uniform design) iteratively run the underlying solver many times to find the optimal parameters.

In this paper, we propose a method for efficient and effective solution of (W)SVM. In the heart of this method lies a multilevel algorithmic framework (MAF) inspired by the multiscale optimization strategies [1]. The main objective of MAF is to construct a hierarchy of problems (coarsening), each approximating the original problem but with fewer degrees of freedom. This is achieved by introducing a chain of successive restrictions of the problem domain (in our case Eq. 1) into low-dimensional or small-size domains and solving the problem in them using local processing.

Our contribution We introduce a novel multilevel framework for (W)SVM. The algorithm is based on the algebraic multigrid (AMG) multilevel scheme [1]. We combine the AMG coarsening with the principles of: (a) coarse approximations of the support vectors, and (b) effective model selection parameter tuning through inheriting from the coarse scales. To the best of our knowledge, this is the first AMG-based algorithm for (W)SVM classification. The proposed method can be parallelized as any AMG algorithm, and its superiority is demonstrated on publicly available and industrial datasets of BMW. Our work extends and generalizes previous multilevel approaches such as [2, 3] which results in a better running time and higher quality classifiers.

The major difference between typical computational optimization MAF, and the (W)SVM is the output of the model. In (W)SVM, the main output is the set of the support vectors which is usually much smaller than the total number of data points. We use this observation in our method by redefining the training set during the uncoarsening. In particular, we inherit the support vectors from the coarse scales, add their neighborhoods, and refine the support vectors at each scale. In other words, we improve the separating hyperplane throughout the hierarchy by gradual refinement of the support vectors and model parameters until a global solution at the finest level is reached. *For more details on the algorithm and computational results, we refer to the long version of this paper* [4].

2 Algorithm

Our algorithm belongs to the family of multiscale learning strategies with the following main phases: (a) coarsening; (b) coarsest scale learning; and (c) uncoarsening. In the coarsening process, the original problem is gradually restricted to smaller spaces by creating aggregates of fine points and their fractions (an important feature of AMG), and turning them into points at coarse levels. The main mechanism underlying the coarsening phase is the AMG which successfully helps to identify the interpolation operator for obtaining fine level solution from the coarse aggregates. When a hierarchy of coarse representations is created, and the number of coarse points is sufficiently small, the coarsest scale learning is applied. In this stage, the (W)SVM problem is solved exactly on coarsest aggregates. In the uncoarsening phase, the solution obtained at the coarsest level (i.e., the support vectors and parameters) is gradually projected back to the finest level by interpolation and further local refinement of support vectors and parameters. A critical difference between our approach and [2] is that in our approach the coarse support vectors are, in fact, not real data points prolongated from the finest

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

level. Instead, they are aggregates of full fine-level data points and their fractions.

Framework initialization We initialize MAF with an undirected affinity graph $G = (V, E)$ generated from the training set of (W)SVM. Each point i is associated with node $i \in V$ (same notation is used for points and nodes), and the set E is determined by the approximate k-nearest neighbor (k-NN) graph.

Coarsening Phase The main goal of the coarsening is to create a hierarchy of coarse representations of the original data manifold using the AMG coarsening for the approximated k-NN graph Laplacians. We denote the sequence of K next-coarser graphs by $\{G_i = (V_i, E_i)\}_{i=0}^{K}$, where $G_0 = G$ is the original graph that corresponds to the training set, and K is the number of levels in the hierarchy.

We describe a two-level process of obtaining the coarse graph $G_c = (V_c, E_c)$ and the corresponding coarse training set from the current fine level $G_f = (V_f, E_f)$ and its training set (e.g., the transition from level l to $l + 1$). The process is started with selecting seed nodes (see Algorithm 1 in [4]) that will serve as centers of coarse nodes, called aggregates. Coarse nodes will correspond to the coarse data points at level c.

In the first step, we compute the future-volume ranking [5] for all nodes to determine the decreasing order in which f-level nodes will be tested for declaring them as seeds. Then, all nodes are traversed and declared as C-nodes if at the moment of decision they are not strongly coupled to the current C, namely, if for point $i \in F$, $\sum_{j \in C} w_{ij} / \sum_{j \in V_f} w_{ij}$ is less than or equal to a threshold (in AMG, typically 0.5). In the end, V_f is split into disjoint sets C, and F. The points with larger future-volumes usually have a better chance to be selected to C to serve as centers of future coarse points. Adding more seeds prevents too aggressive coarsening that can lead to "over-compressed" information at the coarse level and low quality classification model.

When the set C is selected, we compute the AMG interpolation matrix $P \in \mathbb{R}^{|V_f| \times |C|}$ with limited interpolation order [1, 4]. The coarse graph Laplacian is computed by $L_c \leftarrow P^T L_f P$, where L_c, and L_f are coarse and fine graph Laplacians, respectively. The volume for the aggregate created from seed $i \in C$ in the coarse graph is calculated by $\sum_j v_j P_{ji}$, i.e., the total volume of all points is preserved at all levels during the coarsening. The corresponding data point is defined as $\sum_j v_j P_{ji} x_j$.

The stopping criteria for the coarsening depends on the available computational resources to learn the classifier fast at the coarsest level (in our experiments, 500 points).

Note: ► One of the major advantages of the proposed coarsening scheme is the natural ability to deal with the imbalanced data. When the coarsening is performed on both classes simultaneously, and in a small class the number of points reaches an allowed minimum, this level is simply copied throughout the rest of levels required to coarsen the big class. Since the number of points at the coarsest level is small, this does not affect the overall complexity of the framework.◄

Coarsest Level and Uncoarsening When both classes are small enough, the training reinforced by the parameter tuning is fast. We use the uniform design (UD) as a model selection technique to tune C^+, C^-, and γ [6]. The tuned parameters are projected from the coarsest level back to next finer level, where they will be refined and projected up again (see Algorithms 2, 3 in [4]). In contrast to MAF, for most computational optimization problems [1] in which each variable should be solved, the solution of (W)SVM consists of the set of support vectors whose size is typically smaller than the

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

Table 1: Quality and running time (in seconds) for (ML)WSVM on data in [7].

Datasets				WSVM					MLWSVM				
Name	r_{imb}	n_f	$\lvert\mathbf{C}^+ \cup \mathbf{C}^-\rvert$	ACC	SN	SP	κ	Time	ACC	SN	SP	κ	Time
Advertisement	0.86	1558	3279	0.92	0.99	0.45	0.67	231	0.83	0.92	0.81	0.86	**213**
Buzz	0.80	77	140707	0.96	0.99	0.81	0.89	26026	0.88	0.97	0.86	0.91	**233**
Clean (Musk)	0.85	166	6598	1.00	1.00	0.98	0.99	82	0.97	0.97	0.97	0.97	**7**
Cod-RNA	0.67	8	59535	0.96	0.96	0.96	0.96	1857	0.94	0.97	0.92	0.95	**102**
Forest	0.98	54	581012	1.00	1.00	0.86	0.92	353210	0.88	0.92	0.88	0.90	**479**
Hypothyroid	0.94	21	3919	0.99	1.00	0.75	0.86	3	0.98	0.83	0.99	0.91	**3**
Letter	0.96	16	20000	1.00	1.00	0.97	0.99	139	0.98	1.00	0.97	0.99	**12**
Nursery	0.67	8	12960	1.00	1.00	1.00	1.00	192	1.00	1.00	1.00	1.00	**2**
Ringnorm	0.50	20	7400	0.98	0.99	0.98	0.98	26	0.98	0.98	0.98	0.98	**2**
Twonorm	0.50	20	7400	0.98	0.98	0.99	0.98	28	0.98	0.98	0.97	0.98	**1**

Table 2: Evaluation of regular and multilevel WSVM for DS1 set of BMW benchmark.

Class	Size in	Size in	WSVM on DS1		MLWSVM on DS1		MLWSVM on DS2		
number	DS1	DS2	ACC	κ	ACC	κ	ACC	κ	Time (in sec.)
Class 1	6867	204497	0.87	0.90	0.79	0.79	0.80	0.79	1123
Class 2	373	9892	0.99	0.36	0.90	0.69	0.63	0.69	200
Class 3	5350	91952	0.96	0.92	0.91	0.91	0.83	0.82	135
Class 4	278	9339	0.99	0.42	0.87	0.57	0.77	0.71	52
Class 5	2167	57478	0.93	0.62	0.63	0.69	0.62	0.66	53

number of points. Thus, the main time-consuming "operation" of the uncoarsening is to project back and refine the set of coarse support vectors. This can be done very fast if we do not take into account all points at each level for the training. Instead, at each level, we define a new training set that includes only points from fine aggregates of the respective coarse level support vectors.

The parameter tuning using UD or other similar methods is a computationally expensive part of (W)SVM training which takes most of the time for large-scale data sets. The MAF is ideally suitable for applying these methods at coarse levels only when the data is small. When parameters are inherited in the transition from level $i+1$ to i, we either use them in the model if the size of the training set is already large or refine them by running substantially less iterations of a parameter fitting method without any significant loss in the running time. This gives us an effective and efficient practical parameter tuning technique that can be applied on hard and big data. The framework works in a similar way for both regular SVM and WSVM. The WSVM shows better performance for classification of the small class when the data is imbalanced.

3 Computational Results

Our framework is implemented in C++, and PETSc [8]. Current implementation is not parallel. Based on the experience with similar multilevel approaches [1], we anticipate the total complexity and performance of parallel version will be comparable to those of parallel AMG. Small-scale (W)SVM models in the refinement, are solved using LibSVM 3.20 [9] and the approximate k-NN graphs are constructed using FLANN [10].

To evaluate our algorithms, we compute sensitivity (SN), specificity (SP), G-mean (κ), and accuracy (ACC) on the UCI collection [7] and real-world industrial decision

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

system of BMW. All computational results are averages over 20 executions with different random seeds, and reordered data. *Omitted observations:* (1) We are mostly interested in imbalanced problems, so the results of worse κ-quality (ML)SVM are not discussed; (2) We do not discuss faster linear solvers because of their significantly worse κ-quality. However, we note that our MAF can include them in the refinement stage as well.

In Table 1 (section "Datasets"), we present an information about the size of the data and its split into majority and minority classes. The notation r_{imb}, and n_f correspond to the imbalance factor, and the number of features, respectively. Performance measures of regular and multilevel WSVM are presented in sections WSVM, and MLWSVM of Table 1, respectively. Our main performance measure is κ since we are dealing with the imbalanced classification. We observed one significant improvement in the quality of κ in Advertisement data set. *In general, on these and several other data sets, no significant difference in the quality of κ between the proposed fast ML(W)SVM, and the full-time (W)SVM has been observed.* The running time (in seconds) for both WSVM and MLWSVM is presented in columns "Time" in Table 1. The running time includes calculation of the approximated k-NN graphs and UD (model selection) for parameter tuning. *We demonstrate that the proposed fast AMG framework justifies the idea of multilevel algorithms for (W)SVM, and clearly exhibits superior running time.*

In much harder data from BMW, there are 5 labeled classes of plain text customer satisfaction surveys represented in normalized tf-idf form using the uni-, and bi-grams with $n_f \approx 200.000$. The dimensionality is further reduced to 100 using SVD. *We note that we did not observe any change in the quality of the results for full, and reduced dimensional data except the increased running time for full dimensionality. While the multilevel (W)SVM framework running time is not fast but still realistic, the regular (W)SVM cannot be executed on such data at all without introducing significant changes such as high-performance parallelization or switching to linearized SVM of bad quality.*

The size of both DS1 and DS2 data sets is presented in columns 2-3, Table 2. Different classes (1-5) correspond to different major product problems addressed in the customer satisfaction surveys. For the evaluation of DS1 we focus only on the quality of the classifier because all running times are fast for this small dataset and mostly depend on the hardware, while for the DS2 set the running time is reported. While there is no loss in quality on both DS1, and DS2, the running time of MLWSVM on DS2 is substantially better than that of the regular WSVM which is measured in days, so it is comparable to the difference in running time of the Forest data set.

Does AMG help? Aggregation of points and their fractions is the main feature of AMG that helps to approximate the geometry of original data. In [4], we show the comparison of κ and running time for different orders of interpolation (the number of non-zeros in rows of matrix P). It is easy to see that for several datasets the quality of classifiers is improving with increasing the interpolation order.

4 Conclusions

We presented a new algorithmic framework for fast (W)SVM models. The framework belongs to the family of multiscale algorithms in which the problem is solved

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

at multiple scales of coarseness, and gradually combined into one global solution for the original problem. We introduced the flexibility of the AMG coarsening and reinforced it with local learning of the support vectors and model selection parameters. This opens a number of interesting research directions to pursue such as combining multiple local hyperplanes into one global at the refinement, and designing effective parameter inheritance schemes. The implementation of our algorithms is available at `https://github.com/esadr/mlsvm`.

References

[1] A. Brandt and D. Ron. Chapter 1 : Multigrid solvers and multilevel optimization strategies. In J. Cong and J. R. Shinnerl, editors, *Multilevel Optimization and VLSICAD*. Kluwer, 2003.

[2] Talayeh Razzaghi and Ilya Safro. Scalable multilevel support vector machines. In *International Conference on Computational Science (ICCS), Procedia Computer Science*, volume 51, pages 2683–2687. Elsevier, 2015.

[3] Haw-ren Fang, Sophia Sakellaridi, and Yousef Saad. Multilevel manifold learning with application to spectral clustering. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 419–428. ACM, 2010.

[4] Ehsan Sadrfaridpour, Sandeep Jeereddy, Ken Kennedy, Andre Luckow, Talayeh Razzaghi, and Ilya Safro. Algebraic multigrid support vector machines. *arXiv:1611.05487*, 2016.

[5] Ilya Safro, Dorit Ron, and Achi Brandt. Graph minimum linear arrangement by multilevel weighted edge contractions. *Journal of Algorithms*, 60(1):24–41, 2006.

[6] C.M. Huang, Y.J. Lee, D.K.J. Lin, and S.Y. Huang. Model selection for support vector machines via uniform design. *Computational Statistics & Data Analysis*, 52(1):335–346, 2007.

[7] M. Lichman. UCI machine learning repository, 2013.

[8] S Balay, S Abhyankar, M Adams, J Brown, P Brune, K Buschelman, V Eijkhout, W Gropp, D Kaushik, M Knepley, et al. Petsc users manual revision 3.5. Technical report, Technical report, Argonne National Laboratory (ANL), 2014.

[9] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[10] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.