FACILITATING EFFICIENT DATA ANALYSIS OF REMOTELY SENSED IMAGES USING STANDARDS-BASED PARAMETER SWEEP MODELS

Shahbaz Memon[1,2]*, Gabriele Cavallaro*[1]*, Morris Riedel*[1,2]*,Helmut Neukirchen*[2]

[1] Jülich Supercomputing Centre, Forschungszentrum Jülich, Jülich, Germany
[2] School of Engineering and Natural Sciences, University of Iceland, Reykjavik, Iceland

ABSTRACT

Classification of remote sensing images often use Support Vector Machines (SVMs) that require an n-fold cross-validation phase in order to do model selection. This phase is characterized by sweeping through a wide set of parameter combinations of SVM kernel and cost parameters. As a consequence this process is computationally expensive but represents a principled way of tuning a model for better accuracy and to prevent overfitting together with regularization that is in SVMs inherently solved in the optimization. Since the cross-validation technique is done in a principled way also known as 'gridsearch', we aim at supporting remote sensing scientists in two ways. Firstly by reducing the time-to-solution of the cross-validation by applying state-of-the-art parallel processing methods because the sweep of parameters and cross-validation runs itself can be nicely parallelized. Secondly by reducing manual labour by automating the parallel submission processes since manually performing cross-validation is very time consuming, unintuitive, and error-prone especially in large-scale cluster or supercomputing environments (e.g., batch job scripts, node/core/task parameters, etc.).

Index Terms— Remote sensing, Support Vector Machine (SVM), cross-validation, High-Performance Computing (HPC), Parameter Sweep, Middleware.

1. INTRODUCTION

Remote sensing image datasets are an important source of information for many interdisciplinary applications addressing specific topics such as global and local climate change studies, ecological and environmental monitoring, or urban planning. These datasets can be complex (i.e., with high spectral, spatial, radiometric and temporal resolutions) and not reliable (e.g., equipment failure, noise), thus they can not be directly used by the applications. A powerful and automatic processing scheme for extracting reliable and valuable information must usually include feature engineering approaches (e.g., spatial information enhancement [1]) and data mining methods (e.g., classification including validation and regularization techniques). The classification of remote sensing images is the essential technique [2] used for extracting information. A relevant example is the separation of different types of land-cover classes. But the implicit complexity and dimensionality of sensed images are responsible for extensive limitations in classification. For instance problems arise when the classification methods require fast and highly scalable solutions for real-time applications (e.g., earthquake scenarios or glacial surges). Selected developments in High-Performance Computing (HPC) allow the classification algorithms to scale to large datasets [3]) while yielding high accuracy and results in a reasonable time.

Among the widely used remote sensing classifiers, Support Vector Machines (SVMs) [4] have often been found to be more effective in terms of classification accuracies and stability of parameter settings. However, SVMs are very demanding with respect to the processing time, e.g., in tuning the hyperplane parameters with cross-validation in order to perform model selection. The cross-validation phase is laborious if done manually by remote sensing scientists, as it requires re-runs of SVM optimization corresponding to a wide set of parameter combinations. Without any automation tool this phase takes a considerable amount of time and is also error-prone especially when performed on HPC machines with even more low-level technical parameters (e.g. number of cores, number of tasks per core, number of nodes, memory) that are typically machine-specific. To simplify the enhancement and usability of a parallelized cross-validation phase, we are proposing the adoption of a standards-based HPC middleware that handles an automated parameter sweep model which may consist of complex parametric representations, in a single n-fold cross-validation computational job.

2. BACKGROUND AND RELATED WORK

We shortly introduce the two basic concepts SVM and middleware that have been combined in this paper.

Thanks to Jülich Supercomputing Centre (JSC) for funding. This work was partly supported by NordForsk as part of the Nordic Center of Excellence (NCoE) eSTICC (eScience Tools for Investigating Climate Change at High Northern Latitudes). Correspondence: m.memon@fz-juelich.de

2.1. Support Vector Machines and piSVM

SVMs are one of the most powerful classification techniques today. The general idea of SVMs lies in separating training samples which belong to different classes by tracing maximum margin hyperplanes in the space where the samples are mapped [4]. Hence, SVMs only demand training samples close to the class boundary, and it is thus capable of handling high dimensional data even if only a small number of training samples is available. SVMs were originally introduced to solve linear classification problems. In order to generalize them to non-linear decision functions, i.e., more complex classes that are not linearly separable in the original feature space, the so-called *kernel trick* can be applied [5]. The sensitivity to the choice of the kernel and the cost parameters can be considered as the most important disadvantages of SVM.

We surveyed related work in [3] and have shown that despite the availability of many SVM parallelization strategies, only a very limited set of stable and scalable implementations is available as open source software. We improved a version of piSVM 1.2 [6] that was identified in [3] as a stable implementation since it is based on the libSVM library. We optimized it using better parallel processing techniques such as collective operations of the mature HPC standard Message Passing Interface (MPI). This implementation offers significant speed-ups for the cross validation, training and testing steps while maintaining the same accuracy as achieved when performing the classification with serial algorithms.

2.2. Standards-based Middleware and UNICORE

We use the middleware approach to abstract from low-level HPC machine details to make it easier for non-experts to submit and monitor parallel remote sensing classification jobs. To avoid vendor-locks, we rely on middleware based on standards for parallel job management and monitoring as well as data transfer and management functions [7] such as OGSA-BES [8], JSDL [9] and its extensions [10, 11]. The elements of the SVM cross-validation step can be captured through JSDL's parameter sweep extension [10] for building job executions of parametric nature. JSDL allows any part of job request to be parametrized, in particular application arguments. UNICORE [12] is a middleware which offers a seamless layer of abstraction to access different kinds of HPC environments and implements those standards. UNICORE supports this extension on its client and server tiers [13].

Required parameter sweep functionality is also implemented by gLite and gEclipse. But gLite's Workload Management System uses a proprietary approach called JDL to allow parametric job requests. gEclipse is a client-side application, though it supports the JSDL standard but not the full suite of extensions for HPC environments. We therefore have chosen the UNICORE middleware.

3. EXPERIMENTAL ANALYSIS

We validate our approach by measuring the performance of our parallel SVM implementation and the parameter sweep functionality using UNICORE with a remote sensing dataset.

3.1. Remote Sensing Dataset

The Indian Pines hyperspectral dataset [14] was acquired in June 1992 by the AVIRIS sensor over an agricultural site composed of fields with regular geometry and with a variety of crops. This data set represents a very challenging land-cover classification problem dominated by similar spectral classes and mixed pixels. The scene is made up of 1417×617 pixels (with spatial resolution 20 m) and 30 features, which were obtained with the methods described in [3].

3.2. Experimental Setup

For evaluating the performance of our parallel piSVM implementation, we compared it to the serial SVM implementation in MATLAB running on a laptop computer having one Intel Core i7-4710HQ 2.5 GHz CPU and 16 GB of RAM. The piSVM was executed on the JURECA [15] cluster where each compute node has two Intel Xeon E5-2680 v3 Haswell 12 core processors with 2.5 GHz and 128 to 512 GB of RAM. We deployed the piSVM application and the UNICORE server on the JURECA HPC cluster.

The evaluation was performed in two modes, with and without UNICORE middleware adoption. Fig. 1 depicts the steps required in the manual and in our automated UNICORE workflow methodology.

For evaluation of our application of parameter sweeps for the automated cross-validation, we have implemented a workflow shown in Fig. 1(b) while Fig. 1(a) shows the manual labour process by scientists when running a parallel cross-validation job. The workflow runs tThe whole cross-validation process as a single parametric job on piSVM application parameters and performs model selection by picking the best parameters according to estimated accuracies. The concerned application parameters for our case study are C and G with C being the cost as regularization parameter, and G the parameter of the choosen RBF kernel.

The manual workflow shown in Fig. 1(a) is rather low-level and thus advanced HPC system knowledge is required by a user, e.g. to access and monitor jobs. In the course of job management, the user prepares the environment on the cluster to create a job directory for each cross-validation parameter combination (Step 1). The required data then has to be supplied explicitly to the job environment (Step 2). The user then creates a job script which is hard-coded to the respective HPC machine batch system (Step 3) that is SLURM in our case. Once the job script is prepared, the user submits individual jobs separately or by means of a wrapper script (Step 4) that in turn requires sound UNIX knowledge. All

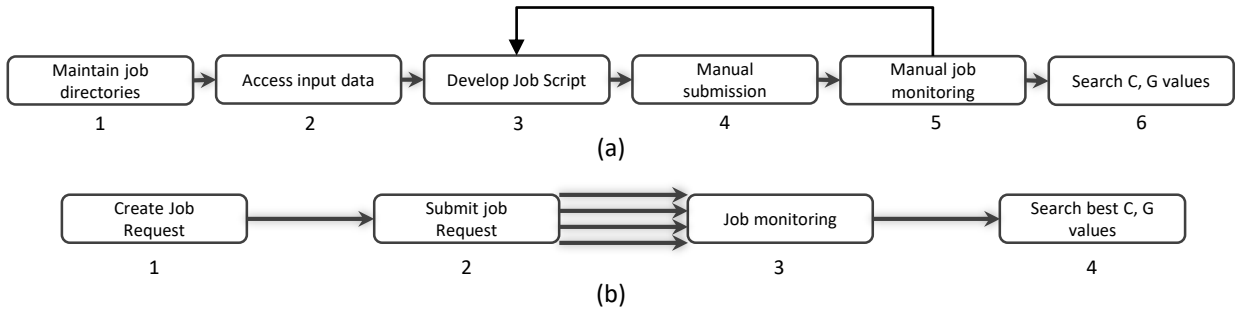

Fig. 1. Flowchart of the (a) manual and the (b) automatic method.

Table 1. Serial 10-fold cross-validation (MATLAB)

G / C	1	10	100	1000	10000
2	48.90 (18.8)	65.01 (19.6)	73.21 (20.1)	75.6 (22.5)	74.42 (21.2)
4	57.53 (16.8)	70.74 (13.9)	75.94 (13.5)	76.04 (14.0)	74.06 (15.6)
8	64.18 (18.3)	74.45 (15.0)	**77.00 (14.4)**	75.78 (14.7)	74.58 (14.9)
16	68.37 (23.2)	76.20 (21.9)	76.51 (20.7)	75.32 (19.6)	74.72 (19.7)
32	70.17 (34.5)	75.48 (34.8)	74.88 (34.1)	74.08 (34.0)	73.84 (38.8)

Table 2. Parallel 10-fold cross-validation (piSVM, 24 cores)

G / C	1	10	100	1000	10000
2	49.02 (7.3)	65.12 (8.6)	73.17 (13.5)	75.76 (22.5)	74.44 (33.0)
4	57.59 (7.4)	70.88 (8.9)	75.87 (11.6)	76.02 (14.7)	74.06 (17.9)
8	64.17 (7.9)	74.53 (9.3)	**77.02 (10.4)**	75.79 (11.3)	74.42 (12.2)
16	68.58 (9.8)	76.07 (10.6)	76.4 (10.9)	75.26 (11.2)	74.53 (11.3)
32	70.12 (13.9)	75.38 (14.3)	74.69 (14.6)	73.91 (14.6)	73.73 (14.6)

the jobs have to be monitored individually (Step 5). After the jobs are finished, then the user searches for the best C and G according to the accuracy somewhere in job output logs (Step 6). As part of cross-validation, many parameter combinations need to be tried, hence Steps 3, 4 and 5 are iterative; if there is no sophisticated script to automatically deal with parameter combinations, then individual job scripts and jobs have to be created for each parameter setting. All in all this process is error-prone and time consuming.

In the automated workflow, depicted in Fig. 1(b), the user creates a job request which conforms to the JSDL [9] [11] and Parameter Sweep [10] specification (Step 1). The job request is formulated as an XML instance in which the user can specify what parameters shall be iterated together with a pointer to the input data source. Once the job request is formalized, the next step is to execute the workflow. In this step, a remote request will be sent to a target server which interfaces the backend cluster, which is in our case JURECA (Step 2). The server validates the job requirements and then performs the resolution of the parameters to be processed before execution. During execution, the server will generate the required number of jobs; in our case, the parameter sweep equals the cartesian product of five C and five G values (=25 jobs). The input data set will also get transparently downloaded from the data source. Furthermore, the server monitors all the generated sweep jobs (Step 3), but these sweep jobs are hidden from the user, only one job is visible to her: the master job representing the sweep. From the result, the best C and G values can be obtained (Step 4).

3.3. Experimental Results

Tables 1 and 2 show the accuracies and the computation times (in minutes shown as value in parentheses) of cross-validation for the serial and the parallel SVM implementation, respectively. When comparing the tables, a significant speed-up was obtained for all parameter combinations while maintaining the accuracies. The best accuracy is marked in bold and indicates the optimal C and G parameter combination which is used in the training phase.

As can be seen, the cross-validation in the serial case is computationally intensive. The reason is that the training-validation is performed 10 times for each of the 25 combinations of the C and G parameters. The total processing time is 534.6 minutes. Because each partition set is independent, the cross-validation performed in parallel can achieve a significant speed up, thus reducing the overall processing time to 322.25 minutes using 24 cores. The biggest impact is shown when performing parallel and scalable cross-validation over the so-called 'gridsearch' as each step in the grid can be also performed in parallel. As a consequence, we implement a two-level parallelization of the cross-validation phase compared to serial MATLAB runs.

It should be noted that Table 2 shows the performance of all parameter combinations without any overhead of UNICORE middleware. In our experience, the use of UNICORE brings additional delay of approximately 2 minutes for completing the whole sweep. Thus, for every single sweep iteration, a delay of few seconds is introduced, which we think is not critical when keeping in mind that otherwise the whole process of cross-validation would involve multiple and time consuming and and error-prone manual user interactions which are now avoided.

In the manual sequence, the user has to engage in multiple steps, e.g. creating batch system specific job scripts (for all the parameters combinations), data management and transfer, and job monitoring. Debugging of failed sweeps may be very cumbersome. The automatic sequence is more high-level and prevents user from manually interacting with individual runs, except for creating and submitting the initial job request.

4. CONCLUSIONS

We conclude that one can obtain significant speed-ups of an automated cross-validation phase used in classification of remote sensing images by applying a parallel and scalable SVM approach. We further conclude that using a standard-based middleware that implements the concept of parameter sweeps for cross-validation runs significantly increases the productivity of scientists when using HPC machines for the analysis. The middleware approach allows not only reduces error-prone and time-consuming and tedious manual labour but also supports the re-use of the workflow on different HPC machines using other standards-based middleware.

In order to automate also other data analysis steps, we intend to extend our cross-validation workflow to include model generation and prediction phases which will directly use best parameters resulted from it.

The described approach has applications in many other remote sensing application areas. For our work, it is promising to apply it to determine calving fronts of glaciers [16] where we are already applying a UNICORE workflow to couple a continuum ice sheet model and a discrete element calving model [17].

5. REFERENCES

[1] G. Cavallaro, M. Dalla Mura, J. A. Benediktsson, and A. Plaza, "Remote Sensing Image Classification Using Attribute Filters Defined over the Tree of Shapes," *IEEE Transactions on Geoscience and Remote Sensing*, 2016.

[2] A. K. Maini and V. Agrawal, *Satellite Technology: Principles and Applications*, John Wiley & Sons, 3rd edition, 2014.

[3] G. Cavallaro, M. Riedel, M. Richerzhagen, J.A. Benediktsson, and A. Plaza, "On understanding big data impacts in remotely sensed image classification using support vector machine methods," *IEEE journal of selected topics in applied earth observations and remote sensing*, vol. 8, no. 10, pp. 4634–4646, 2015.

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20(3), pp. 273–297, 1995.

[5] B. Schölkopf and A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, 2002.

[6] D. Brugger, "piSVM," website, 2014, `http://pisvm.sourceforge.net`.

[7] M. Shahbaz Memon, A. Shiraz Memon, M. Riedel, B. Schuller, D. Mallmann, B. Tweddell, A. Streit, S. van de Berghe, D. Snelling, V. Li, M. Marzolla, and P. Andreetto, "Enhanced resource management capabilities using standardized job management and data access interfaces within UNICORE Grids," in *International Conference on Parallel and Distributed Systems*. 2007, IEEE.

[8] I. Foster et al., "OGSA Basic Execution Service (BES), Version 1.0," Open Grid Forum GFD-R.108, Nov 2008.

[9] A. Anjomshoaa et al., "Job Submission Description Language (JSDL) Specification, Version 1.0," Open Grid Forum GFD-R.136, July 2008.

[10] M. Drescher et al., "JSDL Parameter Sweep Job Extension," Open Grid Forum GFD-R-P.149, May 2009.

[11] A. Savva, "JSDL SPMD Application Extension, Version 1.0," Open Grid Forum GFD-R-P.115, August 2007.

[12] A. Streit et al., "Unicore 6 recent and future advancements," *Annals of Telecommunications*, vol. 65, no. 11-12, pp. 757–762, 2010.

[13] Shahbaz Memon, S. Holl, B. Schuller, M. Riedel, and A. Grimshaw, "Enhancing the Performance of Scientific Workflow Execution in e-Science Environments by Harnessing the Standards Based Parameter Sweep Model," in *Proceedings of the Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery (XSEDE '13)*. 2013, ACM.

[14] M.F. Baumgardner, L.L. Biehl, and D.A. Landgrebe, "220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3," 2015, DOI:10.4231/R7RX991C.

[15] Jülich Supercomputing Centre, "JURECA," website, 2016, `http://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JURECA/JURECA_node.html`.

[16] J. A. Åström, D. Vallot, M. Schäfer, E.Z. Welty, S. O'Neel, T.C. Bartholomaus, Yan Liu, T.I. Riikilä, T. Zwinger, J. Timonen, and J.C. Moore, "Termini of calving glaciers as self-organized critical systems," *Nature Geoscience*, vol. 7, pp. 874–878, 2014.

[17] Shahbaz Memon, D. Vallot, T. Zwinger, and H. Neukirchen, "Coupling of a continuum ice sheet model and a discrete element calving model using a scientific workflow system," in *Geophysical Research Abstracts, Volume 19 – EGU General Assembly 2017*. 2017, Copernicus Publications, submitted.