Ensemble Learning with SVM for High-Dimensional Gene Expression Data

Thanh-Nghi Do[1,2(✉)] and Minh-Thu Tran-Nguyen[1,2]

[1] College of Information Technology, Can Tho University, Campus II, 3/2 street, Can Tho City, Ninh Kieu District, Vietnam
{dtnghi,tnmthu}@ctu.edu.vn
[2] UMI UMMISCO 209 (IRD/UPMC), Paris, France

Abstract. The gene expression classification is the most important study in cancer diagnosis and drug discovery. Nevertheless, this task is very complicated to achieve accurate results because datasets have a very large number of dimensions and very few datapoints. In this paper, we propose the new ensemble learning algorithms with support vector machines (SVM) for efficiently handling the gene expression classification task. The Sherman-Morrison-Woodbury formula is used in the Newton SVM (NSVM) algorithm proposed by Mangasarian to make an extension of Newton SVM for dealing with datasets having a very large number of dimensions. Followed which, the ensemble learning trains the new extended Newton SVM for classifying gene expression datasets with simultaneously large number of datapoints and dimensions. The numerical test results on high-dimensional gene expression datasets show that our ensemble learning algorithms of Newton SVM are significantly faster and/or more accurate than the highly efficient standard SVM algorithm LibSVM.

Keywords: Gene expression classification · Newton Support Vector Machines · Ensemble learning

1 Introduction

The gene expression classification is the key study in cancer diagnosis and drug discovery. Gene expression datasets typically have a very few datapoints in a high-dimensional input space (genes). Therefore, the classification task is very complicated to achieve accurate results due to the curse of dimensionality phenomena. It is well-known as one of top 10 challenging problems in data mining research [25]. There have been many researches to adapt learning models for classification to these data [9,11,13,14,19,20]. Support vector machines (SVM [23]) achieves the most accurate classification results.

The SVM algorithm is motivated by statistical learning theory. SVM algorithms use the idea of kernel substitution [1] for dealing with classification, regression and novelty detection tasks. Successful applications of SVMs have been reported for various fields such as face identification, text categorization and bio-informatics [12]. In spite of the prominent properties of SVM algorithms, they are not favorable to handle the challenge of large datasets. The training task of SVM leads to resolve the quadratic programming (QP), so

N. Thai-Nghe et al. (Eds.): ISDS 2023, CCIS 1950, pp. 29–40, 2024.
https://doi.org/10.1007/978-981-99-7666-9_3

that the computational cost of an SVM approach [17] is at least square of the number of training datapoints and the memory requirement. This makes SVM impractical for large datasets. The effective heuristics for scaling-up SVM learning task are to divide the original QP into series of small problems [2, 17], incremental learning [21] updating solutions in growing training set, parallel and distributed learning [18] on PC network or choosing interested datapoints subset [22] (active set) and boosting with SVM [7].

Our research purpose aims to scaling up SVM algorithms for dealing with high dimensional gene expression datasets (with simultaneously large number of datapoints and very high-dimensional input space). We propose a new extended SVM algorithm that is derived from the finite Newton method proposed by Mangasarian [16] for classification. The Newton SVM (NSVM) only requires solutions of linear equations instead of QP. If the dimensional input space is small enough (less than 10^3), even if there are millions datapoints, the NSVM algorithm is able to classify them in minutes on a personal computer (PC). For handling gene expression datasets with a very large number of dimensions and few training data, we propose to the new extended NSVM formulation using the Sherman-Morrison-Woodbury formula [10]. Followed which, we propose ensemble learning algorithms [3, 4]) using the new extended NSVM that are able to handle gene expression datasets with simultaneously large number of datapoints and dimensions. Numerical test results on high-dimensional gene expression datasets [15] have shown that our ensemble learning algorithms with the new extended NSVM are fast and accurate compared with LibSVM [5].

The paper is organized as follows. Section 2 briefly presents the NSVM algorithm for classification tasks. In Sect. 3, we describe how to extend the NSVM to classify gene expression datasets. The experimental results are presented in Sect. 4. We then conclude in Sect. 5.

2 Newton Support Vector Machines

We start with a simple task for the linear binary classification, as shown in Fig. 1, with m datapoints x_i ($i = 1, \ldots, m$) in the n-dimensional input space R^n, having corresponding classes $y_i = \pm 1$. The SVM learning algorithm proposed by [23] aims to find the best separating hyper-plane (denoted by the normal vector $w \in R^n$ and the bias $b \in R$), i.e. furthest from both class $+1$ and class -1. This goal is accomplished through the maximization of the distance or margin between the supporting planes for each class ($x.w - b = +1$ for class $+1$, $x.w - b = -1$ for class -1). The margin between these supporting planes is $2/\|w\|$ (where $\|w\|$ is the 2-norm of the vector w). Any point x_i falling on the wrong side of its supporting plane is considered to be an error, denoted by z_i ($z_i \geq 0$). Therefore, SVM learning algorithms have to simultaneously maximize the margin and minimize errors. The standard SVMs pursue these goals with the quadratic programming of (1).

$$\begin{aligned} \min\ \Psi(w,b,z) &= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} z_i \\ s.t.: y_i(w.x_i - b) + z_i &\geq 1 \\ z_i &\geq 0 \end{aligned} \tag{1}$$

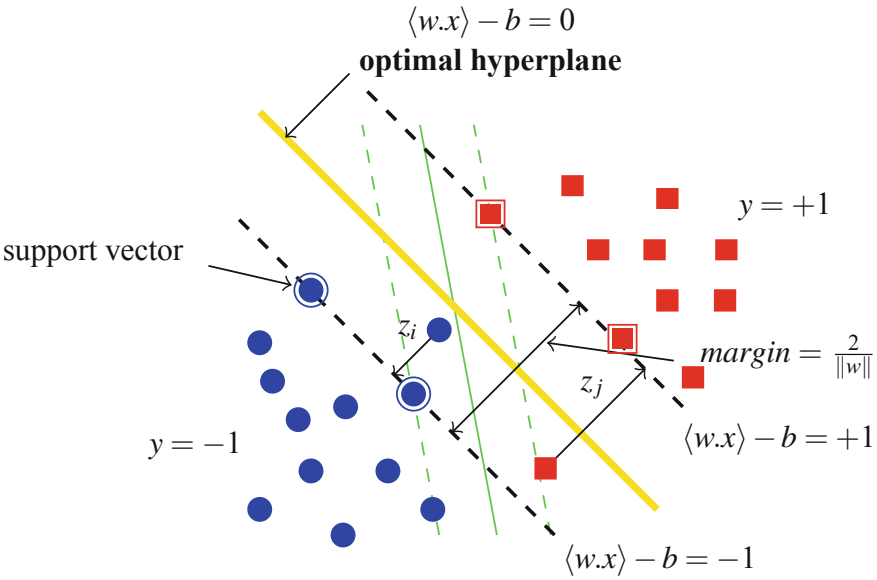

Fig. 1. Linear separation of the datapoints into two classes

where the positive constant C is used to tune errors and margin size.

The plane (w,b) is obtained by solving the quadratic programming (1). Then, the classification function of a new datapoint x based on the plane is:

$$predict(x) = sign(w.x - b) \tag{2}$$

SVM algorithms can use some other classification functions, including a polynomial function of degree d, a RBF (Radial Basis Function) or a sigmoid function. For changing from a linear to non-linear classifier, one must only substitute a kernel evaluation in (1) and (2) instead of the original dot product. More details about SVM and others kernel-based learning methods can be found in [6].

Unfortunately, Platt illustrates in [17] that the computational cost requirements of SVM solutions in (1) are at least $O(m^2)$, where m is the number of training datapoints, making classical SVM intractable for large datasets.

The Newton-SVM (NSVM) proposed by Mangasarian [16] reformulates the SVM problem (1). The new SVM formula achieves:

- the maximization of the margin by $min(1/2)\| w,b \|^2$
- the minimization of errors by $min(c/2)\|z\|^2$

Substituting for $z = [e - D(Aw - eb)]_+$ (where $(x)_+$ replaces negative components of a vector x by zeros, e is the column vector of 1) into the objective function Ψ of the quadratic programming (1) yields an unconstrained problem (3):

$$min\ \Psi(w,b) = (c/2)\| [e - D(Aw - eb)]_+ \|^2 + (1/2)\| w,b \|^2 \quad (3)$$

By setting $[w_1, w_2, \ldots w_n, b]^T$ to u and $[A \quad -e]$ to E, then the unconstrained problem (3) is rewritten by (4):

$$min\ \Psi(u) = (c/2)\| (e - DEu)_+ \|^2 + (1/2)u^T u \quad (4)$$

Mangasarian [16] has proposed the finite stepless Newton method for solving the strongly convex unconstrained minimization problem (4).

Algorithm 1: Newton SVM learning algorithm

input :
Training dataset represented by A and D
Constant $c > 0$ for tuning errors, margin size

output:
(w,b)

Training:
begin
1. Create matrix $E = [A \quad -e]$
2. Starting with $u_0 \in R^{n+1}$ and $i = 0$
3. **repeat**
3.1. The gradient of Ψ at u_i is:

$$\bigtriangledown\Psi(u_i) = c(-DE)^T (e - DEu_i)_+ + u_i \quad (5)$$

3.2. The generalized Hessian of Ψ at u_i is:

$$\partial^2\Psi(u_i) = c(-DE)^T diag([e - DEu_i]_*)(-DE) + I \quad (6)$$

with $diag([e - DEu_i]_*)$ denotes the $(n+1) \times (n+1)$ diagonal matrix whose j^{th} diagonal entry is sub-gradient of the step function $(e - DEu_i)_+$ and I is the identity matrix of size $(n+1) \times (n+1)$.
3.3. Updating

$$u_{i+1} = u_i - \partial^2\Psi(u_i)^{-1} \bigtriangledown\Psi(u_i) \quad (7)$$

3.4. Increment $i = i + 1$
until $\bigtriangledown\Psi(u_i) < tol$;
4. Optimal plane (w,b): $(w_1,\ w_2,\ \ldots,\ w_n,\ b)$ via u_i
end

Mangasarian has illustrated that the sequence u_i of Algorithm 1 terminates at the global minimum solution (with a number of iterations varying between 5 and 8). The NSVM algorithm requires thus only solutions of linear equations (7) of $(n+1)$ variables $(w_1,\ w_2,\ \ldots,\ w_n,\ b)$ instead of the quadratic programming (1). If the dimensional input space is small enough (less than 10^3), even if there are millions datapoints, these algorithms are able to classify them in some minutes on a PC (being much faster than the standard LibSVM [5] while giving competitive test correctness).

3 Newton Support Vector Machines for Gene Expression Datasets

3.1 Newton Support Vector Machines for Classifying Large Number of Dimensions

Gene expression classification tasks handle datasets with a very large number of dimensions (many ten thousands of dimensions) and few training datapoints (hundreds of datapoints). Thus, the $(n+1)\times(n+1)$ matrix $\partial^2\Psi(u_i)$ of (6) in the NSVM algorithm is too large and the inverse matrix $\partial^2\Psi(u_i)^{-1}$ in (7) has a high computational cost. Therefore, the original NSVM algorithm is not suited for classifying gene expression datasets.

To overcome these problems, we propose to extend the NSVM algorithm by applying the Sherman-Morrison-Woodbury formula [10] to $\partial^2\Psi(u_i)^{-1}$. Thus, it leads to obtain a new dual algorithm that only depends the inverse matrix of $(m)\times(m)$ (where m datapoints $\ll n$ dimensions).

$$(A+UV^T)^{-1} = A^{-1} - A^{-1}U(I+V^TA^{-1}U)^{-1}V^TA^{-1} \tag{8}$$

By setting $Q = diag(\sqrt{c[e-DEu_i]_*})$ and $P = Q(-DE)$, we can re-write the inverse matrix $\partial^2\Psi(u_i)^{-1}$ in Eq. (7) as follows:

$$\partial^2\Psi(u_i)^{-1} = (I+P^TP)^{-1} \tag{9}$$

Thus, the Sherman-Morrison-Woodbury formula (8) is applied to the right part of (9), the inverse matrix $\partial^2 f(u_i)^{-1}$ is as (10):

$$\Rightarrow \partial^2\Psi(u_i)^{-1} = I - P^T(I+PP^T)^{-1}P \tag{10}$$

The new extended NSVM formula uses $\partial^2\Psi(u_i)^{-1}$ formula in Eq. (10) to update u_i. And then, the algorithmic complexity of the new extended NSVM only depends on the inversion of the $(m)\times(m)$ matrix $(I+PP^T)$. Therefore, this new extended NSVM formulation described in Algorithm 2 can handle datasets with very large number of dimensions and few training data because the cost of storage and computation scale on the number of training datapoints.

3.2 Ensemble Learning with Newton Support Vector Machines for Classifying Large Amounts of High Dimensional Datapoints

For classification of massive datasets with simultaneously large number (at least 10^4) of datapoints and dimensions, there are at least two problems: the learning time increases dramatically with the training data size and the memory requirement increases according to data size. The NSVM algorithms need to store and invert a matrix with size $(m)\times(m)$ (the new extended NSVM) or $(n+1)\times(n+1)$ (the original NSVM). This requires too much main memory and very high computational time.

For scaling-up the NSVM to large datasets, we propose to apply the ensemble approach, including Bagging [4] and Arc-x4 [3] to NSVM algorithms. The proposed

Algorithm 2: Extended Newton SVM algorithm for large number of dimensions

input :
Training dataset represented by A and D
Constant $c > 0$ for tuning errors, margin size
output:
(w, b)

Training:
begin
1. Create matrix $E = [A \quad -e]$
2. Starting with $u_0 \in R^{n+1}$ and $i = 0$
3. **repeat**
3.1. The gradient of Ψ at u_i is:

$$\bigtriangledown \Psi(u_i) = c(-DE)^T (e - DEu_i)_+ + u_i \tag{11}$$

3.2. The generalized Hessian of Ψ at u_i is:

$$\partial^2 \Psi(u_i) = I - P^T (I + PP^T)^{-1} P \tag{12}$$

with $Q = diag(\sqrt{c[e - DEu_i]_*})$ and $P = Q(-DE)$ and I is the identity matrix of size $(n+1) \times (n+1)$.
3.3. Updating

$$u_{i+1} = u_i - \partial^2 \Psi(u_i)^{-1} \bigtriangledown \Psi(u_i) \tag{13}$$

3.4. Increment $i = i + 1$
until $\bigtriangledown \Psi(u_i) < tol$;
4. Optimal plane (w, b): $(w_1, w_2, \ldots, w_n, b)$ via u_i
end

ensemble learning brings out two advantages. The first one is to be able to overcome the large scale problem and the second one is to maintain the classification accuracy.

The ensemble learning with NSVM is described as in Algorithm 3. Ensemble learning algorithms call repeatedly NSVM learning algorithms *NumIt* times to classify datasets. Here, NSVM algorithms are used to train weak classifiers in ensemble learning algorithms.

With large number of datapoints in dimensional input space being small enough, the original NSVM described in Algorithm 1 is called in ensemble learning algorithms for training weak classifiers.

For dealing with a very large number of dimensions and few datapoints or simultaneously large number of datapoints and dimensions, ensemble algorithms use the new extended NSVM described in Algorithm 2 to train weak classifiers.

At each training step for a weak classifier, ensemble learning algorithms sample a subset of datapoints from the training dataset according to the distribution weights over the training datapoints. For the Arc-x4 mode, it needs increasing the weights of incorrectly classified datapoints in last iterations so that the next weak learner is forced to focus on the hard datapoints in the training dataset.

Algorithm 3: Ensemble learning with Newton SVM for large amounts of high dimensional datapoints

input :
Training dataset with m datapoints:
$\{x_i, y_i\}_{i=1,m}$, $x_i \in R^n$ and $y_i \in \pm 1$
Constant $c > 0$ for tuning errors, margin size
Number of iterations *NumIt*

output:
(w, b)

Training:
begin
1. Initial distribution of m datapoints: $p_1(i) = 1/m$
2. **for** $t \leftarrow 1$ **to** *NumIt* **do**
2.1. Sampling S_t of datapoints using p_t
2.2. Learning $NSVM_t$ from S_t: $h_t = NSVM_t(S_t, c)$
2.3. **if** *Arc-x4* **then**
2.3.1. Computing predicting error for each datapoint x_i with previous classifiers h_t:
$\varepsilon_i = \sum_t h_t(x_i) \neq y_i$
2.3.2. Updating distribution of m datapoints:
for $i \leftarrow 1$ **to** m **do**
$p_{t+1}(i) = \frac{1+\varepsilon_i{}^4}{Z_t}$ (where Z_t is the normalization factor)
end
end
end
3. Optimal plane (w, b) is obtained by aggregating models h_t
end

4 Evaluation

Table 1. Description of Gene expression datasets

ID	Datasets	Classes	Points	Dimensions	Protocol
1	ALL-AML Leukemia	2	72	7129	38 trn - 34 tst
2	Breast Cancer	2	97	24481	78 trn - 19 tst
3	Ovarian Cancer	2	253	15154	leave-1-out
4	Lung Cancer	2	181	12533	32 trn - 149 tst
5	Prostate Cancer	2	136	12600	102 trn - 34 tst
6	Ovarian Cancer NCI-QStar	2	216	373410	leave-1-out
7	Translation Initiation Sites	2	13375	927	3-fold

We are interested in the evaluation of the performances of our ensemble learning algorithms with NSVM in terms of the learning time, the accuracy on large datasets. To pursue this goal, we implement ensemble learning algorithms with NSVM in C/C++, using the high performance linear algebra library, ATLAS/Lapack [8,24]. We also use the highly efficient standard SVM algorithm LibSVM [5] in the performance evaluation of ensemble learning algorithms, including Arc-x4-NSVM and Bag-NSVM. All tests were run under a machine Linux Fedora 32, Intel(R) Core i7-4790 CPU, 3.6 GHz, 32 GB RAM.

The experiment uses 7 high-dimensional gene expression datasets [15] described in Table 1. All datasets have large number of dimensions. Especially, Translation Initiation Sites dataset has simultaneously large number of datapoints and dimensions. The test protocols are presented in the last column of Table 1. With datasets having training set (trn) and testing set (tst) available, we used the training data to tune the parameters of the algorithms for obtaining a good accuracy in the learning phase. Arc-x4-NSVM and Bag-NSVM train 50 weak NSVM classifiers. For LibSVM, NSVM, we tuned the positive constant c for trade-off of errors and the margin size. Then the obtained model is evaluated on the test set. If the training set and testing set are not available then we used cross-validation protocols to evaluate the performance. With datasets having less than three hundred datapoints, the test protocol is leave-one-out cross-validation (loo). It involves using a single datapoint from the dataset as the testing data and the remaining datapoints as the training data. This is repeated so many that each datapoint in the dataset is used once as the testing data. With dataset having more than three hundred datapoints, 3-fold cross-validation is used to evaluate the performance. The dataset is partitioned into 3 folds. A single fold is retained as the validation set, and the remaining 2 folds are used as training data. The cross-validation process is then repeated 3 times (folds). The results from the 3 folds are then averaged to produce the final result.

We obtain classification results in terms of the training time showed in Table 2 and Fig. 2, in terms of the classification correctness showed in Table 3 and Fig. 3. Best results and second ones are in bold and italic.

Table 2. Classification results in terms of training time (sec)

ID	Datasets	Training time (sec)		
		LibSVM	Arc-x4-NSVM	Bag-NSVM
1	ALL-AML Leukemia	9.14	*5.01*	**1.82**
2	Breast Cancer	269.66	*75.43*	**8.16**
3	Ovarian Cancer	403.60	*10.13*	**8.11**
4	Lung Cancer	20.80	*3.51*	**0.47**
5	Prostate Cancer	1.6	*0.7*	**0.51**
6	Ovarian Cancer NCI-QStar	158.95	*20.13*	**9.46**
7	Translation Initiation Sites	314.00	*50.27*	**35.72**

Table 3. Classification results in terms of accuracy (%)

ID	Datasets	Accuracy(%)		
		LibSVM	Arc-x4-NSVM	Bag-NSVM
1	ALL-AML Leukemia	**97.06**	**97.06**	**97.06**
2	Breast Cancer	*73.68*	**84.21**	*73.68*
3	Ovarian Cancer	**100**	**100**	**100**
4	Lung Cancer	**98.66**	*98.00*	97.32
5	Prostate Cancer	73.53	**97.06**	**97.06**
6	Ovarian Cancer NCI-QStar	**97.69**	*97.22*	*97.22*
7	Translation Initiation Sites	**92.41**	*92.08*	91.41

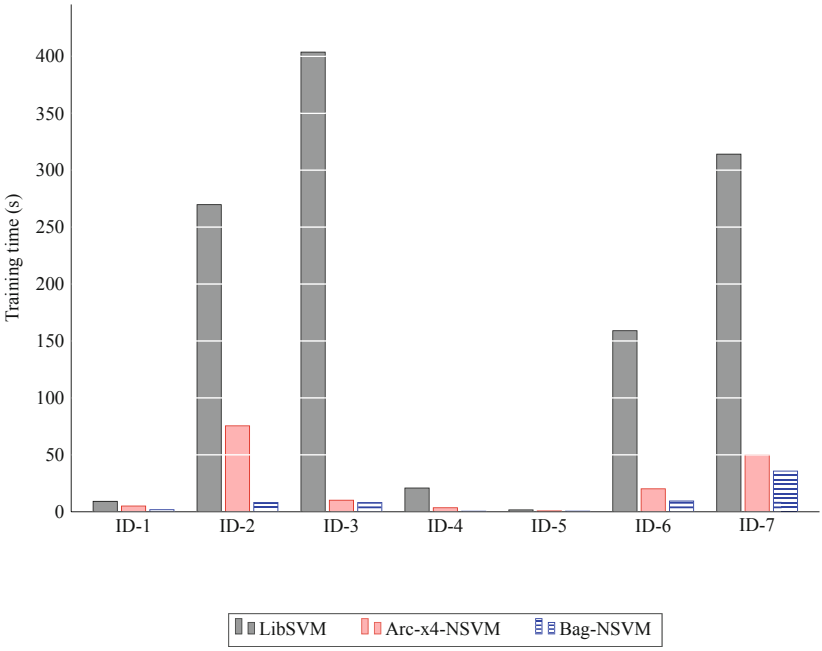

Fig. 2. Comparison of training time (sec)

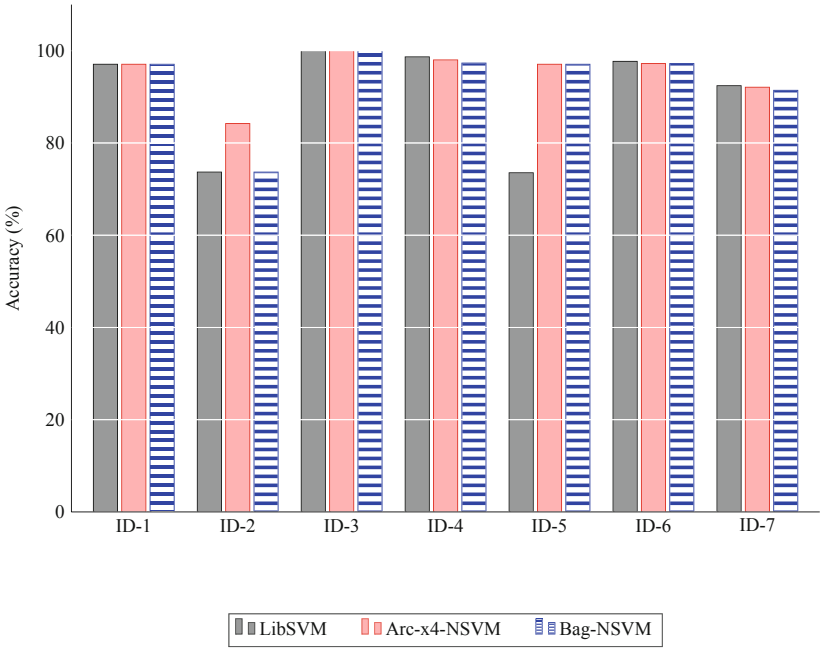

Fig. 3. Comparison of accuracy (%)

The average training time of LibSVM, Arc-x4-NSVM and Bag-SVM are 168.25 sec, 23.60 sec and 9.18 sec, respectively. The comparison in terms of the training time shows that LibSVM is 7.13 times and 18.33 times slower than our Arc-x4-NSVM and Bag-NSVM. Our ensemble learning algorithms with NSVM are always faster than LibSVM for performing all datasets.

In terms of the classification correctness, our Arc-x4-NSVM and Bag-NSVM give very competitive accuracy compared to LibSVM. LibSVM, Arc-x4-NSVM and Bag-NSVM achieve the average accuracy of 90.43%, 95.09% and 93.39%. Arc-x4-NSVM has 2 wins, 2 ties, 3 losses, against LibSVM. Bag-NSVM has 1 win, 3 ties, 3 losses, versus LibSVM.

These results show that our ensemble learning algorithm with NSVM are favorable to deal with very high-dimensional gene expression datasets but also with simultaneously very large number of datapoints and dimensions, e.g. the Translation Initiation Sites dataset. They can learn accurate classification models in short training time.

5 Conclusion and Future Works

We have presented ensemble learning algorithms with NSVM for dealing with high-dimensional gene expression datasets. The Sherman-Morrison-Woodbury formula [10] is used in NSVM to make the extended NSVM for very large number of dimensions and few training datapoints. The ensemble learning [3,4]) trains the new extended

NSVM to classify gene expression datasets with simultaneously large number of datapoints and dimensions. The numerical test results on high-dimensional gene expression datasets [15] show that our Arc-x4-NSVM, Bag-NSVM improve the training speed while achieving good accuracy compared with LibSVM.

In the future, we intend to provide more empirical tests on the large benchmark of gene expression datasets. A forthcoming improvement will be to extend these algorithms for multi-class classification problems.

Acknowledgments. This work has received support from the College of Information Technology, Can Tho University. We would like to thank very much the Big Data and Mobile Computing Laboratory.

References

1. Bennett, K., Campbell, C.: Support vector machines: hype or hallelujah ?. In: SIGKDD Explorations, pp. 1–13 (2000)
2. Boser, B., Guyon, I., Vapnik, V.: An training algorithm for optimal margin classifiers. In: proc. of 5th ACM Annual Workshop on Computational Learning Theory of 5th ACM Annual Workshop on Computational Learning Theory, pp. 144–152. ACM (1992)
3. Breiman, L.: Arcing classifiers. Annals Stat. **26**(3), 801–849 (1998)
4. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 1–27 (2011), software available at http://www.csie.ntu.edu.tw/cjlin/libsvm
6. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press (2000)
7. Do, T., Poulet, F.: Towards high dimensional data mining with boosting of PSVM and visualization tools. In: ICEIS 2004, Proceedings of the 6th International Conference on Enterprise Information Systems, Porto, Portugal, 14–17 April 2004, pp. 36–41 (2004)
8. Dongarra, J., Pozo, R., Walker, D.: LAPACK++: a design overview of object-oriented extensions for high performance linear algebra. In: Proceedings of Supercomputing, pp. 162–171 (1993)
9. Furey, T.S., Cristianini, N., Duffy, N., Bednarski, D.W., Schummer, M., Haussler, D.: Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinform. **16**(10), 906–914 (2000)
10. Golub, G., Loan, C.V.: Matrix Computations, 3rd edn. John Hopkins University Press, Baltimore, Maryland (1996)
11. Golub, T.R., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science **286**(5439), 531–537 (1999)
12. Guyon, I.: Web page on svm applications (1999). http://www.clopinet.com/isabelle/Projects/SVM/app-list.html
13. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. SSS, Springer, New York (2009). https://doi.org/10.1007/978-0-387-84858-7
14. Huynh, P., Nguyen, V.H., Do, T.: Novel hybrid DCNN-SVM model for classifying rna-sequencing gene expression data. J. Inf. Telecommun. **3**(4), 533–547 (2019)
15. Jinyan, L., Huiqing, L.: Kent ridge bio-medical dataset repository. Nanyang Technological University, School of Computer Engineering (2004)
16. Mangasarian, O.: A finite newton method for classification problems. Technical Report 01–11, Data Mining Institute, Computer Sciences Department, University of Wisconsin (2001)

17. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, pp. 185–208 (1999)
18. Poulet, F., Do, T.N.: Mining very large datasets with support vector machine algorithms. In: Camp, V.O., Filipe, J., Hammoudi, S., Piattini, M. (eds.) Enterprise Information Systems, pp. 177–184 (2004)
19. Shinmura, S.: High-dimensional Microarray Data Analysis. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-5998-9
20. Statnikov, A.R., Wang, L., Aliferis, C.F.: A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. BMC Bioinform. **9** (2008)
21. Syed, N., Liu, H., Sung, K.: Incremental learning with support vector machines. In: Proceedings of the ACM SIGKDD International Conference on KDD. ACM (1999)
22. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. In: proc. of the 17th International Conference on Machine Learning, pp. 999–1006. ACM (2000)
23. Vapnik, V.: The Nature of Statistical Learning Theory. Springer-Verlag (1995). https://doi.org/10.1007/978-1-4757-3264-1
24. Whaley, R.C., Dongarra, J.: Automatically tuned linear algebra software. In: Ninth SIAM Conference on Parallel Processing for Scientific Computing (1999), cD-ROM Proceedings
25. Yang, Q., Wu, X.: 10 challenging problems in data mining research. Int. J. Inf. Technol. Decis. Mak. **5**(4), 597–604 (2006)