Support Vector Machines in Big Data Classification: A Systematic Literature Review

Mohammad Hassan Almaspoor

Islamic Azad University South Tehran Branch

Ali Safaei (✉ aa.safaei@modares.ac.ir)

Tarbiat Modares University Faculty of Medical Sciences <https://orcid.org/0000-0003-1985-8720>

Afshin Salajegheh

Islamic Azad University South Tehran Branch

Behrouz Minaei-Bidgoli

Iran University of Science and Technology

Research Article

Keywords: Support vector machines, Big data, Online learning, SLR, Classification, Large-scale dataset

Posted Date: August 9th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-663359/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Support Vector Machines in Big Data Classification: A Systematic Literature Review

Mohammad Hassan Almaspoor^a, Ali A. Safaei^b, Afshin Salajegheh^a, Behrouz Minaei-Bidgoli^c

^a Department of Computer Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran

^b Department of Medical Informatics, Faculty of Medical Sciences, Tarbiat Modares University, Tehran, Iran

^c Department of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

Abstract

Classification is one of the most important and widely used issues in machine learning, the purpose of which is to create a rule for grouping data to sets of pre-existing categories is based on a set of training sets. Employed successfully in many scientific and engineering areas, the Support Vector Machine (SVM) is among the most promising methods of classification in machine learning. With the advent of big data, many of the machine learning methods have been challenged by big data characteristics. The standard SVM has been proposed for batch learning in which all data are available at the same time. The SVM has a high time complexity, i.e., increasing the number of training samples will intensify the need for computational resources and memory. Hence, many attempts have been made at SVM compatibility with online learning conditions and use of large-scale data. This paper focuses on the analysis, identification, and classification of existing methods for SVM compatibility with online conditions and large-scale data. These methods might be employed to classify big data and propose research areas for future studies. Considering its advantages, the SVM can be among the first options for compatibility with big data and classification of big data. For this purpose, appropriate techniques should be developed for data preprocessing in order to convert data into an appropriate form for learning. The existing frameworks should also be employed for parallel and distributed processes so that SVMs can be made scalable and properly online to be able to handle big data.

Keywords: Support vector machines, Big data, Online learning, SLR, Classification, Large-scale dataset

1. Introduction

The rapid growth of digital data production and the rapid development of scientific computing, networking, data storage, and data collectors have enabled us to generate large data sets known as big data. The concept of big data is usually characterized by three characteristics: volume, velocity and variety. Big data is expanding rapidly in all areas of science and engineering. Beyond the characteristics of this type of data, the aspects of analyzing this data and extracting new insights are of special importance[1]–[3].

SVMs have notable advantages such as high generalizability, simple presentation through only a few parameters, and a strong theoretical foundation [4]. Nevertheless, the standard SVM algorithm has been proposed for batch learning and is not suitable for online learning. Unlike batch methods that make all training samples available at once, online learning is a classic learning scenario in which training is done via providing one sample at a time[5]. An important advantage of online algorithms is that they allow for additional training whenever data are available without restarting the training process [5]. The time complexity of an SVM ranges between $O(n^2)$ and $O(n^3)$ [6]. Increasing the number of training samples will intensify the need for computational resources and memory. Given the properties of big data generated rapidly in large amounts, it is necessary to develop certain algorithms that can handle these data properties. This paper reviews the previous studies of SVM compatibility with online learning and SVM scalability, which can potentially be used for big data classification. To the best of our knowledge, no systematic literature review (SLR) has been conducted so far. In fact, an SLR identifies, classifies and synthesizes a comparative overview of state-of-the research and transfers knowledge in the research community [7], [8]. This study aimed to systematically identify and classify the existing methods for SVM compatibility with big data properties (*i.e.*, massive amount and rapid generation) in order to present future research areas. The study addresses the following questions:

RQ1) What is the motivation for SVM compatibility with big data properties?

RQ2) What are the existing SVM-based methods and techniques that support online learning?

RQ3) What are the existing SVM-based methods and techniques that support large-scale data?

RQ4) What are the necessary parameters for comparing big data classifications in volume and time complexity?

RQ5) What are the shortcomings and problems of the existing methods and techniques? What are the future research areas?

This paper is an SLR of the latest developments in SVMs for online learning and large-scale data. For this purpose, the existing methods are analyzed, identified, classified, and reviewed. The shortcomings and problems of these methods are then identified to describe a research environment for future works. This SLR provides researchers with a description of machine learning (Table 1 contains some of the most widely used concepts in machine learning) and data analysis. The rest of this paper consists of different sections. The research methodology is discussed in Section 2,

whereas the SVM is reviewed in Section 3. Moreover, Section 4 addresses the research questions, and Section 5 draws a conclusion.

Table 1: Special terms in machine learning

Word	Explain
Supervised learning	A set of machine learning techniques that require sample label
Unsupervised learning	A group of machine learning techniques that do not require labeling of sample
Semi-supervised learning	.A class of machine learning techniques that use labeled and unlabeled examples
Feature	Variable in learning model
Learning	The process of learning data patterns by model
Testing	Model evaluation process
Training set	A set of examples used for training
Testing set	A set of samples used to evaluate a model
Generalization	The ability of the model to predict the labeling of new and unseen samples
Over-fitting	A model that is very relevant to the training suite and may not be suitable for new data
Classification	Establish a rule for classifying data into a set of categories that already exist based on the set of training sets
Clustering	Grouping is a collection of objects into different clusters
Support Vector Machine (SVM)	A supervised classifier that wants to find the best Hyper plane for data separation
Decision Tree	A tree with a set of hierarchical decisions that ultimately determines the final decision
K-Nearest Neighbors (KNN)	A type of sample-based learning, in which predictions are only locally approximated to k nearest neighbors
Bayesian Networks	Is a non-circular oriented graph that shows a set of random variables and how they relate .independently
Artificial Neural Network	A computational model based on a set of nodes called artificial neurons, which model biological brain neurons

2. Research methodology

This study comprises a three-phase process that includes planning the review, conducting the review, and documenting the review. The process was based on the SLR instructions [7], [8]. Figure 1 presents the phases in detail.

Figure 1: Review process

2.1. Planning Review

2.1.1. Identify requirements

This SLR aims to identify, classify, compare, and present instructions for future studies. To the best of our knowledge, no SLR has yet been conducted to review the SVM compatibility with big data. Springer, IEEE, ScienceDirect, and ACM were also searched for a similar SLR through the following strings;

(Online Support Vector Machine OR Online SVM OR Incremental Support Vector Machine OR Incremental SVM OR Large-Scale Support Vector Machine OR Large-Scale SVM OR Distributed Support Vector Machine OR Distributed SVM OR Parallel Support Vector Machine OR Parallel SVM)

AND

(Systematic Literature Review OR Systematic Review OR SLR OR Systematic Mapping OR Research Synthesis)

however, no SLR of SVM compatibility with big data has been found.

2.1.2. Research questions

This subsection presents research questions and their motivations (Table 2), whereas the scopes of research goals are defined through PICOC criteria (population, intervention, comparison, outcomes, and context) according to Table 3.

Table 2 : Research questions

	Research question	Motivation
RQ1	What is the motivation for SVM compatibility with big data properties?	Given the considerable advantages of SVMs matched with big data properties, the advantages can be used for classification and regression in big data.
RQ2	What are the existing SVM-based methods and techniques that support online learning?	Identification, classification, and comparison of methods which can be used for big data classification
RQ3	What are the existing SVM-based methods and techniques that support large-scale data?	Identification, classification, and comparison of methods which can be used for big data classification.
RQ4	What are the necessary parameters for comparing big data classifications in volume and time complexity?	Identification of parameters to compare the existing methods and techniques
RQ5	What are the shortcomings and problems of the existing methods and techniques? What are the future research areas?	Identification of existing gaps to develop the methods which can classify big data

Table 3 : Scope and Goals of the SLR Criteria (PICOC)

Criteria	RQ1	RQ2	RQ3	RQ4	RQ5
Population	Motivation	Comparing parameters	Methods & Techniques	Methods & Techniques	Research Challenges & Future dimensions
Intervention	Characterization, Internal/External validation, Extracting data and synthesis				
Comparison	Comparison study by mapping the primary studies in the field of large-scale and online learning				
Outcome	Classification and comparison of Existing methods and hypotheses (directions) for future research				
Context	A systematic investigation to consolidate the peer reviewed research in use SVMs to big data classification				

109 2.2. Conducting Review

110 In this phase, studies are extracted in accordance with inclusion/exclusion criteria, and the resultant
111 information is synthesized.

112 2.2.1. Selection of studies

113 The primary studies were selected by searching digital databases and using the snowball
114 methodology, which act as supplementary techniques. The digital databases were ScienceDirect,
115 ACM, Springer, and IEEE, and the search words were selected based on the instructions proposed
116 by [8] and research motivations. The search words are presented below. Based on the needs for
117 every digital database, different data entry styles were used. Several search attempts were made in
118 the digital databases to reach the best compromise between recall and precision. The search was
119 conducted on 11/05/2020.

120 ***Online Support Vector Machine OR Online SVM OR Incremental Support Vector Machine OR Incremental***
121 ***SVM OR Fast SVM OR Incremental Learning OR Active Learning OR Online Classification OR Fast Training***

OR Distributed Support Vector Machine OR Distributed SVM OR Parallel Support Vector Machine OR Parallel SVM OR MapReduce SVM OR Large Scale Support Vector Machine OR Large Scale SVM OR Large Scale Learning OR Cluster Based SVM OR Scaling up SVM OR Scalable SVM OR Divide-And-Conquer SVM OR Reduced Support Vector Machine OR Reduced SVM OR Distributed Parallel SVM OR LASVM OR LSSVM

In addition to searching the digital databases, the snowball methodology was used through forward and backward methods in accordance with [9], [10] so that it could act as a supplementary to the search process. For the snowball process, 11 primary studies (table 4) meeting the inclusion/exclusion criteria were used as the primary groups selected through the strategy for searching digital databases.

Figure 2: Search summary and selection process of preliminary studies

Table 4 : Selected primary studies.

ID	Paper ID Full bibliographic reference
1	X. Wang and Y. Xing, "An online support vector machine for the open-ended environment," <i>Expert Syst. Appl.</i> , vol. 120, pp. 72–86, 2019
2	S. Agarwal, V. Vijaya Saradhi, and H. Karnick, "Kernel-based online machine learning and support vector reduction," <i>Neurocomputing</i> , vol. 71, no. 7–9, pp. 1230–1237, 2008
3	Y. Liu, Z. Xu, and C. Li, "Distributed online semi-supervised support vector machine," <i>Inf. Sci. (Ny)</i> , vol. 466, pp. 236–257, 2018
4	Y. Zhang, G. Cao, B. Wang, and X. Li, "A novel ensemble method for k-nearest neighbor," <i>Pattern Recognit.</i> , vol. 85, pp. 13–25, Jan. 2019
5	Y. Ma, Y. He, and Y. Tian, "Online Robust Lagrangian Support Vector Machine against Adversarial Attack," <i>Procedia Comput. Sci.</i> , vol. 139, pp. 173–181, 2018.
6	X. J. Shen, L. Mu, Z. Li, H. X. Wu, J. P. Gou, and X. Chen, "Large-scale support vector machine classification with redundant data reduction," <i>Neurocomputing</i> , vol. 172, pp. 189–197, 2016.
7	B. Xu, S. Shen, F. Shen, and J. Zhao, "Locally linear SVMs based on boundary anchor points encoding," <i>Neural Networks</i> , vol. 117, pp. 274–284, 2019.
8	T. I. Dhamecha, A. Noore, R. Singh, and M. Vatsa, "Between-subclass piece-wise linear solutions in large scale kernel SVM learning," <i>Pattern Recognit.</i> , vol. 95, pp. 173–190, 2019.
9	F. Alamdar, S. Ghane, and A. Amiri, "On-line twin independent support vector machines," <i>Neurocomputing</i> , vol. 186, pp. 8–21, 2016
10	J. Zheng, F. Shen, H. Fan, and J. Zhao, "An online incremental learning support vector machine for large-scale data," <i>Neural Comput. Appl.</i> , vol. 22, no. 5, pp. 1023–1035, 2013.
11	X. J. Shen, L. Mu, Z. Li, H. X. Wu, J. P. Gou, and X. Chen, "Large-scale support vector machine classification with redundant data reduction," <i>Neurocomputing</i> , vol. 172, pp. 189–197, 2016

143 The selection process includes two tasks, the first of which is to completely define
 144 inclusion/exclusion criteria, whereas the second one is to make real use of these criteria for
 145 selecting the primary studies [7]. Table 5 presents the inclusion/exclusion criteria:

Table 5: Inclusion/ Exclusion Criteria

Criteria	
Inclusion	Quality scientific papers that have been reviewed and contain significant content on the compatibility of the support vector machine with large-scale data and online learning.
Exclusion	Articles in languages other than English. Short research articles (less than 4 pages). Research articles that are not a primary research study. Articles that do not explicitly provide an online learning approach for SVM. Articles that do not explicitly provide a way to scale SVM. Any kind of gray literature (books, lectures, posters, prefaces, editorials, education, ...). Types of research theses or PhD, master's and bachelor's theses.

3. Support Vector Machine

The classification problem includes a rule for grouping data into a set of predetermined categories based on the training set. Vapnik introduced the SVM as a kernel-based machine learning model for classification and regression problems. In fact, the SVM is among the most popular and promising classification algorithms [11], [12] based on the VC dimension and developed through the statistical learning theory and the structural risk minimization principle [4], [11]. The success of an SVM lies in its good generalizability and convergence [13]. In addition to SVMs, there are many other good classification techniques such as the KNN algorithm [14], [15], Bayesian networks [16], [17], artificial neural networks[18], [19], and decision trees [20]. The KNN algorithm is very simple to implement but is slow to handle big data and is very sensitive to irrelevant parameters [14], [15]. The decision tree is a classifier that has widely been used. It is faster than other techniques in the training phase but is inflexible in parameter modeling [21], [22]. Neural networks have extensively been used in many applications; however, many factors such as learning algorithms, number of neurons in every layer, number of layers, and data representation should be considered in the development of neural networks [23], [24].

Using a training dataset, the SVM creates an optimal hyperplane which separates two classes with the maximum margin. This process is based on the structural risk minimization principle that reduces the model generalization error by minimizing the mean squared error in the training set. This is the same philosophy used often by empirical methods for risk minimization [25]. The maximum margin between two classes is equal to the minimum VC dimension[4], [26]. If this hyperplane is very appropriate for training data, the model starts learning training data instead of learning generalization, something which reduces the generalizability of the classifier. The SVM aims mainly to separate classes in a training set through a hyperplane that has the maximum margin between classes. In other words, the SVM allows for the maximization of generalizability. The SVM solution is obtained by minimizing the following objective function.

$$\begin{aligned} \text{Min } J(w, b, \xi) &= \frac{1}{2} \|W\|^2 + C \sum_{i=1}^{\ell} \xi_i^p \\ \text{s. t: } y_i(w \cdot x_i + b) &\geq 1 - \xi_i \\ i &= 1, \dots, \ell \end{aligned} \quad)1($$

Where $w \in \mathbb{R}^n$ is an ordinary vector of the plane, and $b \in \mathbb{R}$ shows the offset. Moreover, $\xi_i \geq 0$ is the slack variable that measures the classification error, whereas $C \in \mathbb{R}$ is an error penalty coefficient. Finally, p is usually either 1 or 2. This problem is defined by introducing both coefficients α_i and μ_i in a Lagrangian formulation. The following equation is then minimized if $\mu_i, \alpha_i \geq 0$:

$$L_p = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{\ell} \alpha_i (y_i(w \cdot x_i + b)) - 1 + \xi_i + C \sum_{i=1}^{\ell} \xi_i^p - \sum_{i=1}^{\ell} \mu_i \xi_i \quad)2($$

Under the Karush–Kuhn–Tucker (KKT) conditions, the following equation will then be obtained:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i = 0 \quad)3($$

181 Where $\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i$. Therefore, the estimation function can be defined as below:

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x} \cdot \mathbf{x}_i + b \quad)4($$

182 Usually, $\mathbf{x}_i$ is mapped onto a higher limited-dimension space (a feature space) through a nonlinear
 183 mapping $\Phi(\mathbf{x})$ to improve the SVM distinguishability power. The SVM core is called the kernel
 184 function defined as $K(a, b) = \Phi(a) \cdot \Phi(b)$, in which the boldfaced parameters show either a matrix
 185 or a vector, and the kernel dimensions refer to the feature space dimensions. The famous kernels
 186 are polynomial (with limited dimensions) and Gaussian (with limited dimensions). Equation 4 is
 187 rewritten as below:

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b \quad)5($$

188 After L_P is minimized, some of the α_i parameters (in fact, most of them in practical applications)
 189 are equal to zero. The nonzero ones are called the support vectors (SVs) on which the SVM
 190 solution depends [27].

191

192 **4. Review of research questions**

193 *RQ1) What is the motivation for SVM compatibility with big data properties?*

194 Despite many advantages of SVMs, they have special weaknesses including high complexity,
 195 parameter selection, difficulty of online and multiclass classification, and efficiency in unbalanced
 196 datasets [25], [28]–[32]. The advent of big data has pressured many of the machine learning
 197 methods that have also been challenged by big data properties[33]. The most important challenge
 198 is to encounter the massive amount and high generation speed of data, which cannot be handled
 199 by many learning algorithms. The main flaw of an SVM might be its high computational cost in
 200 large-scale data classification. Since training an SVM needs a quadratic programming problem to
 201 be solved, the training process becomes slow and requires a massive amount of memory. The
 202 traditional SVM acts as a closed system. In other words, the SVM parameters are frozen after the
 203 training process ends. This strategy makes the SVM incompatible with online learning
 204 environments [34]. In some of the real-world problems, data might be made available sequentially
 205 but not completely all at once. Online learning is an important machine learning problem with
 206 theoretical characteristics and an interesting model. The advent of big data and its recent
 207 applications (e.g., bioinformatics and personal medicine), online learning has drawn a great deal
 208 of attention. Although online learning has been successful in many applications, it is necessary to
 209 improve such limitations as the low efficiency of online learning, high-dimensional disaster
 210 problems, instability of test accuracy in online learning, and effects of noise data [35]. Given its
 211 considerable advantages in conventional learning, the SVM is expected to be used as the main
 212 option for the process of online learning in order to achieve the best performance[36].

Figure 3: Scaling methods for SVM

RQ2) What are the existing SVM-based methods and techniques that support large-scale data?

Different strategies have been proposed for SVM training in large-scale datasets so far. They can be categorized as five classes: reduce samples, decomposition, using parallelization, improving solvers, and incremental learning.

Table 6: Number of methods for scaling

Method used	Number of Papers
Reduce samples	36
Decomposition	12
Parallelism	18
Improved Solvers	20
Incremental Learning	-

- Reduce samples

This approach aims to select a subset of training samples to scale down the training data before the process of training the SVM[37]. The training dataset size is decrease in this method by eliminating the samples playing no roles in the definition of the separating hyperplane. Simple random sampling (SRS) is among the first techniques for scaling down the training set by selecting a subset of training samples to train the SVM. The Reduced SVM algorithm was proposed in [37] to use SRS and employ a subset containing 1–10% of training samples. In fact, SRS has a low computational cost and is considered an appropriate selection scheme in terms of several statistical criteria; however, the standard offset of classification accuracy is often large in this technique[38]. SRS was employed in [38]–[40] to select a subset of training samples. Sampling was performed

in [41]–[43] by determining the selection probability for every sample to train the SVM with the selected samples. The probabilities were then updated, and the probability of false classification increased. This process was iterated several times. The Systematic Sampling Reduced SVM algorithm was proposed in [44], [45] to select informative data points to create the reduced set. Unlike the reduced SVM [37] which employs the random selection scheme, the process starts with a small set first in this method. Some of the falsely classified points are then added iteratively to the reduced set based on the linear classifier. This process continues until the validation set accuracy is large enough.

Other methods of dataset reduction use the distance between samples and the optimal hyperplane through such metrics as Euclidean distance[46], Hausdorff distance[47], [48], and Mahalanobis distance[49]. These methods try to select the samples that are closer to the opposite class and probably have higher chances of being selected as the support vector.

Another class includes data size reduction techniques which use active learning[50]–[52]. A set of training samples was employed in [53] to increase the classifier scalability. The samples were selected heuristic.

Other techniques include clustering-based sample reduction. In [54], the core vector machine (CVM) was proposed to select a kernel set by solving a minimum enclosing ball (MEB) problem regarded as a simple version of MEB. Nevertheless, both CVM and MEB provide the appropriate solutions when there is a small number of support vectors in terms of the training set; otherwise, training will be very time-consuming. In [55], an extension of CVM [54] was proposed by keeping the ball radius constant; therefore, it was not necessary to minimize it. The cluster base SVM (CB-SVM) was proposed in [56] to manage massive datasets. In fact, the CB-SVM applies a hierarchical micro-clustering algorithm that scans the entire dataset only once to propose an SVM with high-quality samples having statistical summaries of data. The cluster-SVM algorithm was introduced in [57] by dividing the clustering data into several clusters first and then using the cluster representatives for training the SVM in order to nearly identify SVs and non-SVs. The support cluster machine (CSM) algorithm was proposed in [58] by adopting a compatible kernel-probability product kernel, which can manage similarity not only between clusters in the training phase but also between a cluster and a vector in the test phase. The clustering reduced SVM algorithm was introduced in [59] by developing the SVM model through the RBF. In this method, the clustering algorithm is employed to create the cluster centroids of every class in order to create the reduced SVM working set. Like most of the clustering-based method, the paper reviewed by [60] tried to find the boundary points between two classes which would be the most qualified samples for training. In this method, the dynamically growing self-organization tree algorithm was used for clustering. The MEB was used in [61] to propose a classification method for dividing the training data through the proposed method to use the cluster centroids for classification. After that, classification was performed by using either the clusters whose centroids were SVs or other different classes. Most of the data were eliminate in the second step. A combination of SRVM and K-mode clustering KMO-SVM was proposed in [62] to classify big datasets. The C²LSVM algorithm was proposed in [63] by developing the local SVM algorithm based on cooperative clustering. Other algorithms can be found in [64], [65].

Some techniques use the neighborhood characteristics of SVs to reduce the training dataset size. For instance, the neighborhood entropy was employed in [66]; however, only the models existing around the decision boundary were used in [67]. The same procedure for fuzzy C-mean clustering was employed in [68] to select samples on class distribution. The clustering-based SVM training was used in [69] to eliminate the data that were farther from SVs. The data points existing in the inner layer of a cluster were considered non-SV points and were then eliminated; however, the data points scattered in the external layer were regarded as the SV points and were then retained. In this method, the Fisher ratio and cluster data point distribution were employed to determine the boundary between the cluster data points and the data points scattered in a cluster.

Basic functions were used in [70] to develop a classifier. The independent linear vectors were used in [31] for SVM training; moreover, this method was extended in [71] to use the twin SVM instead of the SVM.

In [72], an algorithm was proposed to identify and remove the unnecessary SVs that would have no effects on the solution. A method was proposed in [73] to select a subset of vectors that would operate directly by creating a vocabulary of vectors; however, this formulation was not convex.

Figure 4: Number of papers submitted for SVM scalability

- Decomposition

The decomposition methods are based on the fact that the training time can be reduced if the active limitations of the quadratic programming (QP) problem are considered [66]. A similar idea of active methods is also employed for optimization in decomposition techniques; however, two sets (a working set and a set of constant variables) are used in the active set method, and optimization is performed only on the working set. In the SVM problem, the working set usually consists of the items which violate the KKT conditions. An advantage of the decomposition method is that it needs memory in proportion to the number of linear training sets [25]. These methods are sensitive to the selection of the working set because only a fraction of variables is considered in every iteration. If these elements are not selected accurately, the process will be time-consuming[29], [74]. The convergence of these methods was also proven[75]. Chunking is among the earliest

decomposition methods [76]. It obtains the maximum margin from a number of samples and then creates a new chunk with the SVs of the previous solution and some of the new samples. The sequential minimum optimization (SMO) was proposed in [77] to convert the main QP problem into a series of smaller QP problems, each of which would optimize only a subset of size 2. This algorithm is faster than the chunking algorithm. Using the boost SMO algorithm[78], the Platt's SMO algorithm can become faster. The LIBSVM[79] is an SMO-based algorithm with advanced improvements in the working set selection mechanism through the second-order information method previously proposed in [80]. The SVM^{Light} is another advanced method of decomposition proposed in [81]. A parallel optimization phase was introduced in [82] by using the diagonal matrix block estimate the original kernel matrix in order to divide the SVM classification into hundreds of sub-problems. A superior recursive and computational mechanism was proposed in [83] as the adaptive partitioning technique which would operate based on the piece-wise linear decision function. This method uses a non-Gaussian criterion for extracting the subclass of data and also a new formulation of the optimization problem obtained from the subclass information. In [84] the Cluster SVM (CSVM) algorithm is introduced which controls the data in the form of division and conquer. This algorithm groups the data into several clusters, in each of which a linear SVM is trained. A local classification was proposed in [85] to achieve efficiency based on coding the boundary anchor points. The LLBAP (Locally Linear SVMs model based on Boundary Anchor Points) divides the linear inseparable data into nearly dividable segments by scanning the boundary points and local coding. A linear SVM is then used in each segment of data to solve the problem. The subclass reduced SVM (SRS-SVM) algorithm was proposed in [86] to select the subclass structure of data to effectively estimate the set of candidate SVs. Since SVs account partially for the training set cardinality, the training set reduces with no change in the decision boundary. This method depends on the input data domain knowledge, *i.e.*, the number of subclasses. The hierarchical SRS-SVM algorithm was proposed to decrease dependency on the domain knowledge and also sensitivity to the subclass parameter. It is a hierarchical and improved model of SRS-SVM. Since both methods divide the original optimization problem into several optimization sub-problems, both can be parallelized. The SVMTroch algorithm [87] used the working set and shrinking to improve the training time in regression problems. Basically considered a stochastic sub-gradient descent optimization algorithm, the Pegasos algorithm [88] employed the decomposition technique to reduce the training time.

- Parallelization

Some other techniques use parallelization or distributed computation to handle large-scale data. It is difficult to perform the parallel and distributed implementation of a QP problem, for there is great dependency between data [30]. Most of the parallel methods of SVM training divide the training set into independent subsets for training the SVM in different processors. A method was proposed in [82] to estimate the SVM kernel matrix through a block of diagonal matrices to convert the original optimization problem into hundreds of sub-problems, which can easily be solved through parallelization methods. The training set was divided into m random subsets which were then trained separately in [89]. The data were divided into several subsets on which the optimization operations were performed in [6]. The results of each subset were then combined and

filtered in a cascade of SVMs. This method can be distributed on multiple processes with the minimum communication overhead because the matrices are small and require low memory. A distributed incremental algorithm was proposed in [90] for nonlinear kernels. In this method, LS-SVM[91] was employed to develop the algorithm. Moreover, a distributed algorithm (as opposed to parallelization) of SVM was proposed in [92]. It was assumed that the training data followed the same distribution and were locally stored in different spots that could be processed. For this purpose, two approaches were employed. The first approach benefited from a distributed naïve chunking technique in which SVs were exchanged, whereas the second approach used a distributed semi-parametric SVM to reduce the interactions between machines for privacy retention. In [93], the distributed parallel SVM (DPSVM) algorithm was proposed in a configurable network environment for distributed data mining. The main idea is that SVs should be exchanged in a strongly connected network so that several servers can work simultaneously with distributed datasets at a limited communication cost but a high speed. A parallel SVM algorithm was proposed in [94]. The authors claimed that it increased the SVM speed unprecedentedly. This algorithm decreased the SVM time complexity to $O(np/m)$ in which p indicates the dimensions of a reduced matrix after factorization p is much smaller than n , whereas m refers to the number of machines used. The distributed parallel SVM algorithm was proposed in [90] to distribute data among different nodes which were prevented from communicating with the central processing unit (due to communication complexity, scalability or privacy). The MapReduce SMO algorithm was proposed in [95]. The MapReduce SMO algorithm divides the training suite into m random subset of same size, and each partition is assigned to a Map task. Each Map function is trained on a partition and its output is a partial weight vector and a value of b . Finally, the algorithm calculates the total weight vector and mean of b . The resource-aware SMO (RASMO) algorithm was introduced in [96] to optimize the SVM in a parallel framework based on the MapReduce model by dividing the training set into smaller partitions and using a cluster. In this algorithm, the load balancing scheme is based on the genetic algorithm designed for the algorithm performance optimization in heterogeneous environments. The paper reviewed by [97] adopted a similar procedure to [96]. The parallel algorithm to SVM was proposed in [98] to divide the dataset into K clusters by using the K -means algorithm and then develop the nonlinear SVM model on local data. After that, it labels the decision trees of SVMs to the terminal nodes. A combination of SVM and decision tree was proposed in the semi-supervised SVM (S^3VM) in [99] for labeled and unlabeled data in a network of interconnected agents, in which data were distributed on machines. In this method, communications are only limited to the neighboring agents, and there is no coordination reference. The distributed gradient descent algorithm and the NEXT framework were employed to find a solution based on sequential convexity of the main problem. A distributed online semi-supervised algorithm was proposed in [100] to use a series of anchor points adaptively through an online strategy. This method benefits from a random sparse mapping to estimate the map of kernel features. This mapping can estimate the model parameters without transferring the original data jointly between neighbors (to respond to privacy protection concerns). This algorithm is efficient in cases where data have limited labels. Other parallel implementations of SVM can be seen in [101]–[104].

• Improving Solvers

Some of the SVM methods aim to accelerate the training process by improving the solver. A few of these techniques improve the SVM training time by losing a level of accuracy[74]. These methods make some changes to the original QP formula. An instance of these methods is the least square SVM (LS-SVM)[91]. This classifier transforms the objective function into a series of linear equations by changing the initial formulation. Introduced in [105], the proximal SVM also acts like the LS-SVM.

In [59], a method was proposed to reduce the number of necessary SVs. The reduction process repeatedly selects and combines the two nearest SVs belonging to the same class. In [106], by adding a new constraint to SVM, the classification sparsity is controlled and a method for solving the formulated problem is proposed. Basically, the proposed approach finds a subspace that can be covered by a small number of vectors. This subspace separates different classes of data linearly. In [107], a new method was developed to solve the linear SVM through the L_2 loss function using the Modified Finite Newton Method, which suits large-scale data mining tasks such as text classification. In [89], a cutting-plane algorithm was proposed to train a linear SVM. The proposed algorithm can perform classification with $O(SN)$, in which S indicates the number of nonzero features. According to the authors, this method operates many times faster than the SVM^{Light} in big data. In [108], a new method called the dual coordinate descent was proposed for the linear SVM with L_1 and L_2 loss functions. The authors claimed that the proposed method was faster than other solvers such as Pegasos, TRON, SVM^{Pref}, and the primal coordinate descent implementation. In [72], an algorithm was proposed to identify and eliminate unnecessary SVs without changing the solution. In [109], a local linear SVM was proposed with smooth bound and bounded curvature to obtain the classifier solution through local coding. In addition, the stochastic gradient descent can be adopted to optimize the model online with the same model convergence guarantee. In [110], the H-SVM algorithm was proposed based on an oblique decision tree, in which the nodes are split based on a linear SVM to reduce both the training and test time (it is necessary for cases purposes such as fraud detection and intrusion detection to reduce the test time). This method can also be parallelized. In [61], the PWL-SVM algorithm was proposed to implement a piecewise SVM method through the piecewise feature mapping. In [111], a method was developed to propose the piecewise-linear structure in a multiclass scenario based on the latent SVM formulation. In [112], a new version of the Frank-Wolf(FW) algorithm was proposed to accelerate the basic FW procedure convergence, in which the formulation focuses on the concavity maximization. Having a hierarchical structure with a linear SVM composite in every node, the HMLSVM(Hierarchical Mixing Linear Support Vector Machines) algorithm was proposed in [113].

The geometric methods of SVM are based on the calculation of the optimal separating hyperplane that finds the nearest points to the convex hull [114]. In addition to heuristic methods such as [115], [116], alpha seeding was proposed in [115], [116] to estimate the initial value of α_i to start the QP problem. In [117], the decision tree was employed to propose an SVM decision boundary approximation method. In addition, the SVM was integrated with the decision tree to develop novel methods in [118], [119].

Figure 5: Type of papers submitted for SVM scalability

- Incremental learning

Laying the foundation for online learning, the incremental learning methods can also lead to learning scalability. Since the incremental learning techniques are employed to apply online incremental learning to the model, it is reviewed in the next subsection.

RQ3) What are the existing SVM-based methods and techniques that support online learning?

Many attempts have so far been proposed to apply online learning and incremental learning to the SVM. They can be categorized as three groups [71], *i.e.* 1) Unbounded: these methods make no attempt to reduce the solution, and the solution size expands by increasing the number of input samples. 2) Amender: these methods try to prevent the problem solution growth by decreasing the current problem dimensions. 3) preventive: these methods seek to prevent the new sample from being added to the problem solution if the new sample has no effect on the solution.

Figure 6: Apply online learning on SVM

- Unbounded methods for online and incremental learning

In [120]–[122], a method was proposed for the SVM compatibility with incremental learning by performing the learning process through data batches based on the batch SVM. In [123], online learning was considered with kernel reproduction in the Hilbert space. The classic SGD was also used in a feature space along with certain techniques to develop a simple but efficient algorithm for classification and regression problems. In [124], a similar algorithm was proposed by using a novel implicit updating technique. An incremental algorithm was also proposed in [5] where the authors claimed that it would accelerate the incremental SVM by 5 to 20 rates. The acceleration was based on the numerical storage operations. In [90], a distributed parallel incremental algorithm was developed based on the LS-SVM for big data classification. In [125], the Lagrangian SVM (LSVM) algorithm was introduced as an improved version of the linear SVM algorithm. The solution was obtained from an iterative scheme with linear convergence by using the previous calculations. The LSVM aims to look for a hyperplane in an $(n+1)$ -dimensional space instead of an n -dimensional space. The SMO was employed in [126] to propose the LSVM algorithm, which is closer to the explicit solution yielded by the SVM. It also provides online learning. In this method, the working set is replaced in the SMO phases to improve speed and accuracy significantly. Moreover, a second-order greedy working set selection strategy is put in every step to increase progress. In [127], an online incremental SVM algorithm was proposed for large-scale data by using two LPs (learning prototypes) and LSVs (learning Support Vectors). The LPs learn the prototype, whereas the LSVs integrate LPs with previous SVs to create a new SVM. An online semi-supervised algorithm was proposed in [100] by using a series of anchor points selected adaptively through an online strategy. This algorithm is efficient in cases with a limited number of labeled data. Based on LSVM, an online robust algorithm was selected in [128] to make a simple change to the kernel matrix. This algorithm is useful when the new data points might be contaminated (*i.e.*, the label is changed by an attacker). Since the online learning model is more sensitive to contamination; therefore, it is affected to a greater extent if it is exposed to a contaminated sample. Therefore, the algorithm is made robust to deal with such cases.

Figure 7: Number of articles submitted for online and incremental learning

- Amender Methods

In [129], a recursive online algorithm was proposed by presenting one sample every time. The learning method is reversible; therefore, unlearning is also possible in this method. The multiple-version of this method was also proposed in [130] to reduce the training time, learn, or unlearn multiple samples. In [131], an incremental SVM was proposed to learn and unlearn a single sample or multiple samples and adapt the current SVM by changing the configurations and kernel parameters. In [132], an incremental algorithm called the I-SVM was proposed to accelerate the training process and reduce the need for storage by throwing away some historical samples. In [133], an online SVM was proposed by reducing the number of previous samples used for prediction to minimize the need for memory. Previous algorithm was expanded in [134], It is also free of the presence of noisy samples and can also be used for batch cases. In [135], the LASVM algorithm was introduced to accelerate the training process by selecting active samples. This method benefits from the pairwise optimization principle for online learning. It also defines two direction operations called the PROCESS and the REPROCESS. The process tries to enter the new sample into the current set of SVs, whereas the reprocess aims to reduce the number of SVs. In online iterations, the algorithm switches between the single executions of these two operations. In [136], an incremental algorithm was proposed for the SVM by dividing the dataset into several segments, each of which was employed to select training samples through the K-means algorithm. After that, every sample was given a weight in the active query based on the sample coefficient and distance from the hyperplane. A criterion is developed to eliminate non-informative training samples incrementally. In [137], a TS-type fuzzy classifier called the ISVM-FC was proposed. This classifier can be employed when data are available sequentially. First, there is no fuzzy rule for learning the structure through the ISVM-FC. The rules are generated with respect to the distribution of training data. The ISVM is employed to regulate the parameters of rules to improve the classifier generalizability. The incremental learning can also be utilized to exclude the previous training data based on their distance from the hyperplane in order to improve the classifier efficiency. In [138], a single-class incremental classifier was proposed. In fact, the single-class classification is among the most challenging areas of machine learning in specific areas such as medical analysis. The proposed algorithm is called the ICOSVM(incremental Covariance- guided One-Class Support Vector Machine) which improves the classifier efficiency by relying on the low-variance direction. After the new sample arrives, previous SVs are controlled in this method, which can be implemented on large-scale stream data.

Table 7: Number of articles for incremental and online learning

Method used	Number of Papers
Unbounded	12
Amender	10
Preventative	6

- Preventive Methods

In [139], the OSVC(online support vector classifier) algorithm was proposed for sequential data. In this method, the SVM is first trained with the existing data to determine the decision boundary. If the new sample is classified correctly when it arrives, it means that it is not an SV; therefore, it will be ignored. If the new sample violates the KKT conditions, it is considered an SV; thus, the decision boundary is updated. In [140], the concept of span of SV was proposed by Vapnik to develop a classifier. In addition to meeting spatial and temporal constraints, it yields acceptable performance. The idea of using the span is that it can directly affect the bounded generalization error. In this method, there is a constraint on the number of SVs. Every sample that is classified wrongly should be verified for inclusion. After that, a previous SV should be excluded. The memory is free of SVs at first. In the SV set, the points are replaced if they have the maximum S-span reduction. In [31], an online algorithm called OISVM(On-line Independent Support Vector Machines) was proposed. It converges nearly on the ideal solution to the SVM. Like the kernel-based algorithms, OISVM generates a hypothesis through the samples which have been observed so far. These samples are called the base. The new samples can be placed in the base only if they are linearly independent of the current base set in the feature space. The method introduced in [71] is the extended version of OISVM that uses the twin SVM for training and improves the algorithm time complexity. In most of the online SVMs, the classifier is retrained with previous SVs and the newly arrived samples. The maintained system usually loses its performance due to the loss of much information on data. To solve this problem, an online algorithm was proposed in [34]. The proposed algorithm includes a representative prototype area (RPA) that represents the previous historical data. In the RPA, every class is retained by an online incremental feature mapping that includes a sample set of stream data automatically.

Figure 8: Type of articles for incremental and online learning

In [35], an accelerative model was proposed for online SVM learning based on the window technology of KKT conditions. It is not an independent online algorithm but can be considered a

facilitator for the other online algorithms. This algorithm develops a working set of SVMs with a fixed window including the samples that violate KKT conditions. As a result, not only does the model create the training samples of the same size, but it is also guaranteed that the samples are useful for updating the hyperplane and can mitigate the noise effect by adjusting the KKT window and penalty coefficients.

Figure 9: Type of articles

RQ4) What are the necessary parameters for comparing data classifications in volume and time complexity?

According to the literature review, the common performance evaluation criteria include accuracy, CPU time, and the number of support vectors described as below.

Classification Accuracy:

The performance calculation criteria are employed to evaluate the classification algorithm performance (Table 8).

Table 8: Confusion matrix

	Actually Positive	Actually Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Moreover, evaluation criteria are defined as sensitivity (SN), specialty (SP), G-mean, and accuracy (ACC) for a binary classification problem.

$$SN = \frac{TP}{TP + FN} \quad SP = \frac{TN}{TN + FP}$$

560

561

$$G - mean = \sqrt{SP * SN}$$

562

563

$$ACC = \frac{TP + TN}{FP + TN + TP + FN}$$

564

565

566 *Number of Support Vectors:*

567 This criterion is the mean number of support vectors employed to develop the hyperplane.

568

569 *CPU Time:*

570 This criterion is the mean time required to create the hyperplane.

571

572 **RQ5) What are the shortcomings and problems of the existing methods and techniques?**

573 **What are the future research areas?**

574 Big data are characterized mainly by volume, velocity, and variety. A characteristic of big data is
575 the massive size of data produced heterogeneously in high dimensions. Information collectors use
576 specific schemes and structures to record data, whereas different applications lead to the
577 presentation of different data. In such conditions, heterogeneous features and data of different
578 dimensions will result in different representations. This problem is a serious challenge to the
579 collection and combination of data from different sources [141], [142]. The next problem is the
580 presence of independent sources with decentralized distributions and control, something which is
581 a major challenge to the applications of big data. Every source is independently able to generate
582 and collect information without relying on a central control framework. In addition, the complexity
583 of data relationships will intensify if big data expand. Therefore, it is necessary to consider the
584 complicated relationships of data (nonlinear and many-to-many) along with the ongoing changes
585 in order to extract useful models from big datasets [141]. Machine learning provides great potential
586 and constitutes a necessary component for big data analysis [143]. The advent of big data has
587 created many opportunities for machine learning. At the same time, big data features have
588 challenged machine learning. The challenges posed to machine learning by big data include high
589 dimensions, model scalability, distributed computation, data stream, compatibility, *etc.*

590 In such conditions, classification algorithms must change effectively to encounter and classify big
591 data. The first problem is the variety of data. Most of the classification algorithms can work with
592 a specific type of data, whereas big data might include structured, semi-structured, and

unstructured data. Hence, data should be transformed into an appropriate form for learning in an initial preprocessing step. For this purpose, it is necessary to perform the right preprocessing operations on data with respect to the input data. These operations might include feature selection, sample reduction, noise elimination, and data annotation. Therefore, the existing methods of data preprocessing should be evaluated with respect to their applications and then adopted to meet the need for desirable changes. For instance, the annotation of unstructured data can be employed in the classification process.

The next problem includes massive size of data and high-speed data generation. In other words, data are continuously generated at a high rate. Therefore, the proposed learning algorithms should be both scalable and compatible with online learning conditions. Processing such massive data would require the use of parallel or distributed computation, for which MapReduce[144] and Spark[145] frameworks can be used. Google MapReduce provides an efficient and effective framework for parallel processing of big data. Moreover, the Google file system (GFS) [146] can be used along with this framework to provide distributed, reliable, and efficient data storage for big datasets. Furthermore, MapReduce can be employed to provide an abstraction of problems such as distributed programming and support effective and efficient parallelization. MapReduce is implemented by considering certain problems such as load balancing, network throughput, and error tolerance. The Apache Hadoop[147] project is among the most popular and widely used open-source implementations of MapReduce programmed by Java. The MapReduce model includes two separate steps called Map and Reduce. In fact, Map is an initial conversion step in which the input records are processed in parallel manner, whereas Reduce is a summarization step in which all of the related records are processed together [97].

Spark is a cluster computation technology implemented for rapid computation. Based on MapReduce, Spark develops this model to support all different types of computations. Spark is mainly characterized by the in-memory cluster processing which accelerates the processing speed. In fact, Spark supports a wide range of applications including batch applications, iterative algorithms, interactively query, and data stream. Moreover, Spark mitigates the difficulty of managing different maintenance tools. MLLIB[148] is one of the components of Spark, which is an open source distributed library for machine learning. This library provides functions for a wide range of learning applications and includes basic statistical primitives, optimization and linear algebra. MLLIB, along with Spark, supports several languages including Java, Scala, Python, and R, and provides high-level APIs.

An important feature of Spark is the reuse of a working set of data among several parallel operations, something which is also necessary in many machine learning algorithms and also interactive tools of data analysis. In addition to supporting such applications, Spark maintains scalability and error tolerance of MapReduce. For this purpose, Spark introduces an abstraction called the resilient distributed datasets (RDD), which is a read-only set of objects partitioned among a set of machines and can be recreated if the partitions are destroyed. In machine learning applications, Spark can operate 10 to 100 times faster than Hadoop.

The next problem is the large number of samples, all of which might not be necessary for the learning process. It might also be impossible to process all samples. Hence, dataset reduction

methods such as selection of samples and preventive online learning methods can be useful. Given the high speed of data generation, using unlimited and corrective online learning methods will result in high costs of time and space; therefore, it appears to be more appropriate to use preventive online learning methods.

5. Conclusion

This study is an SLR of support vector machines in the era of big data. It includes three phases of planning, implementation, and documentation. In fact, an SLR identifies, classifies and synthesizes a comparative overview of state-of-the research and transfers knowledge in the research community. This study aimed to systematically identify and classify the existing methods of SVM by considering two features of big data, *i.e.*, massive size and high speed of generation, and presenting the future research areas.

Considering its advantages, the SVM can be among the first options for compatibility with big data and classification of big data. For this purpose, appropriate techniques should be developed for data preprocessing in order to covert data into an appropriate form for learning. The existing frameworks should also be employed for parallel and distributed processes so that SVMs can be made scalable and properly online to be able to handle big data. Therefore, data reduction methods such as sample selection as well as preventive online learning methods can be useful. Due to the high speed of data production, the use of unlimited online learning methods and modifiers will have a high time and space cost, so it seems that the use of preventive online learning methods is more appropriate for this purpose.

Declaration

Funding: No fund is granted to this research.

Conflicts of interest/Competing interests: No conflict of interests.

Availability of data and material: No data available to be published.

Authors' contributions: All authors have equal contributions.

Research is Compliance with Ethical Standards

References

- 668 [1] H.-J. Y. Choong Ho Lee, "Medical big data: promise and challenges," *Kidney Res. Clin. Pract.*, vol.
669 36, no. 1, pp. 3–11, Apr. 2017, doi: /10.23876/j.krcp.2017.36.1.3 Kidney.
- 670 [2] J. Luo, M. Wu, D. Gopukumar, and Y. Zhao, "Big Data Application in Biomedical Research and Health
671 Care: A Literature Review," *Biomed. Inform. Insights*, vol. 8, p. BII.S31559, Jan. 2016, doi:
672 10.4137/bii.s31559.
- 673 [3] X. Wu, X. Zhu, W. Gong-Qing, and W. Ding, "Data Mining with Big Data," *IEEE Trans. Knowl. Data*
674 *Eng.*, vol. 26, no. 1, pp. 97–107, 2014, doi: 10.1109/TKDE.2013.109.
- 675 [4] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Min. Knowl.*
676 *Discov.*, vol. 2, no. 2, pp. 121–167, 1998, doi: 10.1023/A:1009715923555.
- 677 [5] P. Laskov, C. Gehl, S. Krüger, and K. R. Müller, "Incremental support vector learning: Analysis,
678 implementation and applications," *J. Mach. Learn. Res.*, vol. 7, pp. 1909–1936, 2006.
- 679 [6] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, and V. Vapnik, "Parallel Support Vector Machines :
680 The Cascade SVM," *Adv. Neural Inf. Process. Syst.*, pp. 521–528, 2005, [Online]. Available:
681 http://machinelearning.wustl.edu/mlpapers/paper_files/NIPS2005_190.pdf.
- 682 [7] "Kitchenham, B. and Charters, S. (2007) Guidelines for Performing Systematic Literature Reviews
683 in Software Engineering. Evidence Based Software Engineering Technical Report. - References -
684 Scientific Research Publishing."
685 [https://www.scirp.org/\(S\(351jmbntvnsjt1aadkposzje\)\)/reference/ReferencesPapers.aspx?Refere](https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?Refere)
686 [ncelID=1738514](https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?Refere) (accessed Dec. 23, 2020).
- 687 [8] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the
688 systematic literature review process within the software engineering domain," *J. Syst. Softw.*, vol.
689 80, no. 4, pp. 571–583, Apr. 2007, doi: 10.1016/j.jss.2006.07.009.
- 690 [9] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software
691 engineering," *ACM Int. Conf. Proceeding Ser.*, 2014, doi: 10.1145/2601248.2601268.
- 692 [10] D. Badampudi, C. Wohlin, and K. Petersen, "Experiences from using snowballing and database
693 searches in systematic literature studies," in *ACM International Conference Proceeding Series*, Apr.
694 2015, vol. 27-29-April-2015, pp. 1–10, doi: 10.1145/2745802.2745818.
- 695 [11] V. N. Vapnik, *The nature of statistical learning theory*. Springer, 2000.
- 696 [12] Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern
697 classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 5, pp. 905–910, 2007, doi:
698 10.1109/TPAMI.2007.1068.
- 699 [13] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-*
700 *based Learning Methods*. Cambridge University Press, 2000.
- 701 [14] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *Am.*
702 *Stat.*, vol. 46, no. 3, pp. 175–185, 1992, doi: 10.1080/00031305.1992.10475879.
- 703 [15] Y. Zhang, G. Cao, B. Wang, and X. Li, "A novel ensemble method for k-nearest neighbor," *Pattern*
704 *Recognit.*, vol. 85, pp. 13–25, Jan. 2019, doi: 10.1016/j.patcog.2018.08.003.
- 705 [16] B. G. Marcot and T. D. Penman, "Advances in Bayesian network modelling: Integration of modelling
706 technologies," *Environmental Modelling and Software*, vol. 111. Elsevier Ltd, pp. 386–393, Jan. 01,

- 2019, doi: 10.1016/j.envsoft.2018.09.016.
- [17] B. Drury, J. Valverde-Rebaza, M. F. Moura, and A. de Andrade Lopes, "A survey of the applications of Bayesian networks in agriculture," *Eng. Appl. Artif. Intell.*, vol. 65, pp. 29–42, Oct. 2017, doi: 10.1016/j.engappai.2017.07.003.
- [18] J. Du, C. Zhai, and Y. Wan, "Radial basis probabilistic neural networks committee for palmprint recognition," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2007, vol. 4492 LNCS, no. PART 2, pp. 819–824, doi: 10.1007/978-3-540-72393-6_98.
- [19] "Neural Networks for Pattern Recognition | Guide books." <https://dl.acm.org/doi/book/10.5555/525960> (accessed Dec. 29, 2020).
- [20] M. Paliwoda, "Decision Trees Learning System," in *Intelligent Information Systems 2002*, Physica-Verlag HD, 2002, pp. 77–90.
- [21] A. Trabelsi, Z. Elouedi, and E. Lefevre, "Decision tree classifiers for evidential attribute values and class labels," *Fuzzy Sets Syst.*, vol. 366, pp. 46–62, Jul. 2019, doi: 10.1016/j.fss.2018.11.006.
- [22] M. Fratello and R. Tagliaferri, "Decision trees and random forests," in *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, vol. 1–3, Elsevier, 2018, pp. 374–383.
- [23] D. S. Huang and J. X. Du, "A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks," *IEEE Trans. Neural Networks*, vol. 19, no. 12, pp. 2099–2115, 2008, doi: 10.1109/TNN.2008.2004370.
- [24] J. X. Du, D. S. Huang, G. J. Zhang, and Z. F. Wang, "A novel full structure optimization algorithm for radial basis probabilistic neural networks," *Neurocomputing*, vol. 70, no. 1–3, pp. 592–596, Dec. 2006, doi: 10.1016/j.neucom.2006.05.003.
- [25] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, no. xxxx, 2020, doi: 10.1016/j.neucom.2019.10.118.
- [26] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer New York, 2000.
- [27] I. Steinwart, "Sparseness of support vector machines," *J. Mach. Learn. Res.*, vol. 4, no. 6, pp. 1071–1105, 2004, doi: 10.1162/1532443041827925.
- [28] X. Li, J. Cervantes, and W. Yu, "A novel SVM classification method for large data sets," *Proc. - 2010 IEEE Int. Conf. Granul. Comput. GrC 2010*, pp. 297–302, 2010, doi: 10.1109/GrC.2010.46.
- [29] C. W. Hsu and C. J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 415–425, Mar. 2002, doi: 10.1109/72.991427.
- [30] R. F. W. Pratama, S. W. Purnami, and S. P. Rahayu, "Boosting Support Vector Machines for Imbalanced Microarray Data," in *Procedia Computer Science*, Jan. 2018, vol. 144, pp. 174–183, doi: 10.1016/j.procs.2018.10.517.
- [31] F. Orabona, C. Castellini, B. Caputo, L. Jie, and G. Sandini, "On-line independent support vector machines," *Pattern Recognit.*, vol. 43, no. 4, pp. 1402–1412, 2010, doi: 10.1016/j.patcog.2009.09.021.

- 746 [32] A. Rojas-Dominguez, L. C. Padierna, J. M. Carpio Valadez, H. J. Puga-Soberanes, and H. J. Fraire,
747 "Optimal Hyper-Parameter Tuning of SVM Classifiers with Application to Medical Diagnosis," *IEEE*
748 *Access*, vol. 6, pp. 7164–7176, Dec. 2017, doi: 10.1109/ACCESS.2017.2779794.
- 749 [33] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Neurocomputing Machine learning on big data :
750 Opportunities and challenges," vol. 237, no. January, pp. 350–361, 2017, doi:
751 10.1016/j.neucom.2017.01.026.
- 752 [34] X. Wang and Y. Xing, "An online support vector machine for the open-ended environment," *Expert*
753 *Syst. Appl.*, vol. 120, pp. 72–86, 2019, doi: 10.1016/j.eswa.2018.10.027.
- 754 [35] H. Guo, A. Zhang, and W. Wang, "An accelerator for online SVM based on the fixed-size KKT
755 window," *Eng. Appl. Artif. Intell.*, vol. 92, no. April, p. 103637, 2020, doi:
756 10.1016/j.engappai.2020.103637.
- 757 [36] X. Zhou, X. Zhang, and B. Wang, *Online Support Vector Machine: A Survey*. 2016.
- 758 [37] Y.-J. Lee and O. L. Mangasarian, "RSVM: Reduced Support Vector Machines," pp. 1–17, 2001, doi:
759 10.1137/1.9781611972719.13.
- 760 [38] X. Li, J. Cervantes, and W. Yu, "Fast classification for large data sets via random selection clustering
761 and Support Vector Machines," *Intell. Data Anal.*, vol. 16, no. 6, pp. 897–914, Jan. 2012, doi:
762 10.3233/IDA-2012-00558.
- 763 [39] Y. J. Lee and S. Y. Huang, "Reduced support vector machines: A statistical theory," *IEEE Trans.*
764 *Neural Networks*, vol. 18, no. 1, pp. 1–13, Jan. 2007, doi: 10.1109/TNN.2006.883722.
- 765 [40] F. Zhu, J. Yang, N. Ye, C. Gao, G. Li, and T. Yin, "Neighbors' distribution property and sample
766 reduction for support vector machines," *Appl. Soft Comput. J.*, vol. 16, pp. 201–209, Mar. 2014,
767 doi: 10.1016/j.asoc.2013.12.009.
- 768 [41] B. Gärtner and E. Welzl, "A simple sampling lemma: Analysis and applications in geometric
769 optimization," *Discret. Comput. Geom.*, vol. 25, no. 4, pp. 569–590, Apr. 2001, doi:
770 10.1007/s00454-001-0006-2.
- 771 [42] G. Loosli, S. Canu, and L. Bottou, "Training Invariant Support Vector Machines using Selective
772 Sampling," *Large Scale Kernel Mach.*, pp. 301–320, 2007, [Online]. Available:
773 <http://leon.bottou.org/papers/loosli-canu-bottou-2006>.
- 774 [43] J. L. Balcázar, Y. Dai, J. Tanaka, and O. Watanabe, "Provably fast training algorithms for support
775 vector machines," *Theory Comput. Syst.*, vol. 42, no. 4, pp. 568–595, May 2008, doi:
776 10.1007/s00224-007-9094-6.
- 777 [44] C. C. Chang and Y. J. Lee, "Generating the reduced set by systematic sampling," *Lect. Notes Comput.*
778 *Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3177, pp. 720–725,
779 2004, doi: 10.1007/978-3-540-28651-6_107.
- 780 [45] L. I. J. Chien, C. C. Chang, and Y. J. Lee, "Variant methods of reduced set selection for reduced
781 support vector machines," *J. Inf. Sci. Eng.*, vol. 26, no. 1, pp. 183–196, 2010, doi:
782 10.6688/JISE.2010.26.1.13.
- 783 [46] Y. G. Liu, Q. Chen, and R. Z. Yu, "Extract candidates of support vector from training set," in
784 *International Conference on Machine Learning and Cybernetics*, Jan. 2003, vol. 5, pp. 3199–3202,

785 doi: 10.1109/icmlc.2003.1260130.

786 [47] D. Wang, D. S. Yeung, and E. C. C. Tsang, "Weighted Mahalanobis distance kernels for support
787 vector machines," *IEEE Trans. Neural Networks*, vol. 18, no. 5, pp. 1453–1462, Sep. 2007, doi:
788 10.1109/TNN.2007.895909.

789 [48] D. Wang and L. Shi, "Selecting valuable training samples for SVMs via data structure analysis,"
790 *Neurocomputing*, vol. 71, no. 13–15, pp. 2772–2781, Aug. 2008, doi:
791 10.1016/j.neucom.2007.09.008.

792 [49] S. Abe and T. Inoue, "Fast training of support vector machines by extracting boundary data," in
793 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and*
794 *Lecture Notes in Bioinformatics)*, 2001, vol. 2130, pp. 308–313, doi: 10.1007/3-540-44668-0_44.

795 [50] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines," *Mach. Learn.*
796 *Work. Then Conf.*, pp. 839–846, 2000, [Online]. Available:
797 <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.31.6090&rep=rep1&type=pdf>.

798 [51] S. Tong, K. Cs, and S. Edu, "Support vector machine active learning with applications to text
799 classification," *Support vector Mach. Act. Learn. with Appl. to text Classif.*, vol. 2, no. 1, pp. 45–66,
800 2002, doi: 10.1162/153244302760185243.

801 [52] M. Li and I. K. Sethi, "Confidence-based active learning," *IEEE Trans. Pattern Anal. Mach. Intell.*,
802 vol. 28, no. 8, pp. 1251–1261, 2006, doi: 10.1109/TPAMI.2006.156.

803 [53] A. Bordes, "Fast Kernel Classifiers with Online and Active Learning," vol. 6, pp. 1579–1619, 2005.

804 [54] I. W. Tsang, J. T. Kwok, and P. M. Cheung, "Core vector machines: Fast SVM training on very large
805 data sets," *J. Mach. Learn. Res.*, vol. 6, pp. 363–392, 2005.

806 [55] I. W. Tsang, A. Kocsor, and J. T. Kwok, "Simpler core vector machines with enclosing balls," *ACM*
807 *Int. Conf. Proceeding Ser.*, vol. 227, pp. 911–918, 2007, doi: 10.1145/1273496.1273611.

808 [56] H. Yu, J. Yang, and J. Han, "Classifying large data sets using SVMs with hierarchical clusters," *Proc.*
809 *ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 306–315, 2003, doi: 10.1145/956750.956786.

810 [57] D. Boley and D. Cao, "Training support vector machine using adaptive clustering," *SIAM Proc. Ser.*,
811 pp. 126–137, 2004, doi: 10.1137/1.9781611972740.12.

812 [58] B. Li, M. Chi, J. Fan, and X. Xue, "Support cluster machine," in *ACM International Conference*
813 *Proceeding Series*, 2007, vol. 227, pp. 505–512, doi: 10.1145/1273496.1273560.

814 [59] D. D. Nguyen and T. Ho, "An efficient method for simplifying support vector machines," *ICML 2005*
815 *- Proc. 22nd Int. Conf. Mach. Learn.*, pp. 617–624, 2005, doi: 10.1145/1102351.1102429.

816 [60] M. Awad, L. Khan, F. Bastani, and I. L. Yen, "An effective support vector machines (SVMs)
817 performance using hierarchical clustering," *Proc. - Int. Conf. Tools with Artif. Intell. ICTAI*, no. Ictai,
818 pp. 663–667, 2004, doi: 10.1109/ICTAI.2004.26.

819 [61] X. Huang, S. Mehrkanoon, and J. A. K. Suykens, "Support vector machines with piecewise linear
820 feature mapping," *Neurocomputing*, vol. 117, pp. 118–127, 2013, doi:
821 10.1016/j.neucom.2013.01.023.

822 [62] J. M. Zain, "An alternative algorithm for classification large categorical dataset: k-mode clustering

reduced support vector machine," *serisc.org*, Accessed: Dec. 16, 2020. [Online]. Available: https://www.academia.edu/714205/An_alternative_algorithm_for_classification_large_categorical_dataset_k_mode_clustering_reduced_support_vector_machine.

[63] C. Yin, Y. Zhu, S. Mu, and S. Tian, "Local support vector machine based on cooperative clustering for very large-scale dataset," *Proc. - Int. Conf. Nat. Comput.*, no. Icnc, pp. 88–92, 2012, doi: 10.1109/ICNC.2012.6234598.

[64] J. Cervantes, X. Li, and W. Yu, "Support vector machine classification based on fuzzy clustering for large data sets," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Nov. 2006, vol. 4293 LNAI, pp. 572–582, doi: 10.1007/11925231_54.

[65] R. Collobert and S. Bengio, "SVM-Torch: Support Vector Machines for large-scale regression problems," *J. Mach. Learn. Res.*, vol. 1, no. 2, pp. 143–160, Mar. 2001, doi: 10.1162/15324430152733142.

[66] R. Wang and S. Kwong, "Sample selection based on maximum entropy for support vector machines," in *2010 International Conference on Machine Learning and Cybernetics, ICMLC 2010*, 2010, vol. 3, pp. 1390–1395, doi: 10.1109/ICMLC.2010.5580848.

[67] H. Shin and S. Cho, "Neighborhood property-based pattern selection for support vector machines," *Neural Comput.*, vol. 19, no. 3, pp. 816–855, Mar. 2007, doi: 10.1162/neco.2007.19.3.816.

[68] X. Jiantao, H. Mingyi, W. Yuying, and F. Yan, "A fast training algorithm for support vector machine via boundary sample selection," in *Proceedings of 2003 International Conference on Neural Networks and Signal Processing, ICNNSP'03*, 2003, vol. 1, pp. 20–22, doi: 10.1109/ICNNSP.2003.1279203.

[69] X. J. Shen, L. Mu, Z. Li, H. X. Wu, J. P. Gou, and X. Chen, "Large-scale support vector machine classification with redundant data reduction," *Neurocomputing*, vol. 172, pp. 189–197, 2016, doi: 10.1016/j.neucom.2014.10.102.

[70] S. S. Keerthi, O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," *J. Mach. Learn. Res.*, vol. 7, pp. 1493–1515, 2006.

[71] F. Alamdar, S. Ghane, and A. Amiri, "On-line twin independent support vector machines," *Neurocomputing*, vol. 186, pp. 8–21, 2016, doi: 10.1016/j.neucom.2015.12.062.

[72] T. Downs, K. Gates, and A. Masters, "Exact Simplification of Support Vector Solutions," *J. Mach. Learn. Res.*, vol. 2, no. Dec, pp. 293–297, 2001.

[73] M. Wu, B. Schölkopf, and G. Bakir, "A direct method for building sparse kernel learning algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 603–624, 2006, doi: 10.1016/j.jbiomech.2005.01.007.

[74] G. Wang, "A survey on training algorithms for support vector machine classifiers," in *Proceedings - 4th International Conference on Networked Computing and Advanced Information Management, NCM 2008*, 2008, vol. 1, pp. 123–128, doi: 10.1109/NCM.2008.103.

[75] N. List and H. U. Simon, "A general convergence theorem for the decomposition method," in *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, 2004, vol. 3120, pp. 363–377, doi: 10.1007/978-3-540-27819-1_25.

- [76] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "Training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, 1992, pp. 144–152, doi: 10.1145/130385.130401.
- [77] Z. Q. Zeng, H. Bin Yu, H. R. Xu, Y. Q. Xie, and J. Gao, "Fast training Support Vector Machines using parallel Sequential Minimal Optimization," *Proc. 2008 3rd Int. Conf. Intell. Syst. Knowl. Eng. ISKE 2008*, pp. 997–1001, 2008, doi: 10.1109/ISKE.2008.4731075.
- [78] D. Pavlov, J. Mao, and B. Dom, "Scaling-up support vector machines using boosting algorithm," *Proc. - Int. Conf. Pattern Recognit.*, vol. 15, no. 2, pp. 219–222, 2000, doi: 10.1109/icpr.2000.906052.
- [79] C. C. Chang and C. J. Lin, "LIBSVM: A Library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, Apr. 2011, doi: 10.1145/1961189.1961199.
- [80] R.-E. Fan, P.-H. Chen, and C.-J. Lin, "Working Set Selection Using Second Order Information for Training Support Vector Machines," 2005. doi: 10.5555/1046920.1194907.
- [81] "Making large-scale support vector machine learning practical | Advances in kernel methods." <https://dl.acm.org/doi/10.5555/299094.299104> (accessed Dec. 16, 2020).
- [82] J. X. Dong, A. Krzyzak, and C. Y. Suen, "Fast SVM training algorithm with decomposition on very large data sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 603–618, Apr. 2005, doi: 10.1109/TPAMI.2005.77.
- [83] S. W. Kim and B. J. Oommen, "Enhancing prototype reduction schemes with recursion: A method applicable for 'large' data sets," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 34, no. 3, pp. 1384–1397, Jun. 2004, doi: 10.1109/TSMCB.2004.824524.
- [84] Q. Gu and J. Han, "Clustered support vector machines," *J. Mach. Learn. Res.*, vol. 31, pp. 307–315, 2013.
- [85] B. Xu, S. Shen, F. Shen, and J. Zhao, "Locally linear SVMs based on boundary anchor points encoding," *Neural Networks*, vol. 117, pp. 274–284, 2019, doi: 10.1016/j.neunet.2019.05.023.
- [86] T. I. Dhamecha, A. Noore, R. Singh, and M. Vatsa, "Between-subclass piece-wise linear solutions in large scale kernel SVM learning," *Pattern Recognit.*, vol. 95, pp. 173–190, 2019, doi: 10.1016/j.patcog.2019.04.012.
- [87] "SVM Torch." <http://bengio.abracadoudou.com/SVMTorch.html> (accessed Dec. 16, 2020).
- [88] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for SVM," *Math. Program.*, vol. 127, no. 1, pp. 3–30, 2011, doi: 10.1007/s10107-010-0420-4.
- [89] T. Joachims, "Training linear SVMs in linear time," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vol. 2006, pp. 217–226, 2006, doi: 10.1145/1150402.1150429.
- [90] T. N. Do and F. Poulet, "Classifying one billion data with a new distributed SVM algorithm," *Proc. 4th IEEE Int. Conf. Res. Innov. Vis. Futur. RIVF'06*, pp. 59–66, 2006, doi: 10.1109/RIVF.2006.1696420.
- [91] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999, doi: 10.1023/A:1018628609742.

- 901 [92] A. Navia-Vázquez, D. Gutiérrez-González, E. Parrado-Hernández, and J. J. Navarro-Abellán,
902 "Distributed support vector machines," *IEEE Trans. Neural Networks*, vol. 17, no. 4, pp. 1091–1097,
903 2006, doi: 10.1109/TNN.2006.875968.
- 904 [93] Y. Yumao Lu, V. Roychowdhury, and L. Vandenberghe, "Distributed Parallel Support Vector
905 Machines in Strongly Connected Networks," *IEEE Trans. Neural Networks*, vol. 19, no. 7, pp. 1167–
906 1178, Jul. 2008, doi: 10.1109/TNN.2007.2000061.
- 907 [94] E. Y. Chang *et al.*, "PSVM: Parallelizing support vector machines on distributed computers," *Adv.*
908 *Neural Inf. Process. Syst. 20 - Proc. 2007 Conf.*, no. 2, pp. 1–8, 2009, doi: 10.1007/978-3-642-20429-
909 6_10.
- 910 [95] N. K. Alham, M. Li, Y. Liu, and S. Hammoud, "A MapReduce-based distributed SVM algorithm for
911 automatic image annotation," *Comput. Math. with Appl.*, vol. 62, no. 7, pp. 2801–2811, 2011, doi:
912 10.1016/j.camwa.2011.07.046.
- 913 [96] W. Guo, N. K. Alham, Y. Liu, M. Li, and M. Qi, "A Resource Aware MapReduce Based Parallel SVM
914 for Large Scale Image Classifications," *Neural Process. Lett.*, vol. 44, no. 1, pp. 161–184, 2016, doi:
915 10.1007/s11063-015-9472-z.
- 916 [97] Z. H. You, J. Z. Yu, L. Zhu, S. Li, and Z. K. Wen, "A MapReduce based parallel SVM for large-scale
917 predicting protein-protein interactions," *Neurocomputing*, vol. 145, pp. 37–43, 2014, doi:
918 10.1016/j.neucom.2014.05.072.
- 919 [98] T. N. Do and F. Poulet, "Parallel learning of local SVM algorithms for classifying large datasets," in
920 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and*
921 *Lecture Notes in Bioinformatics)*, 2017, vol. 10140 LNCS, pp. 67–93, doi: 10.1007/978-3-662-54173-
922 9_4.
- 923 [99] S. Scardapane, R. Fierimonte, P. Di Lorenzo, M. Panella, and A. Uncini, "Distributed semi-supervised
924 support vector machines," *Neural Networks*, vol. 80, pp. 43–52, 2016, doi:
925 10.1016/j.neunet.2016.04.007.
- 926 [100] Y. Liu, Z. Xu, and C. Li, "Distributed online semi-supervised support vector machine," *Inf. Sci. (Ny).*,
927 vol. 466, pp. 236–257, 2018, doi: 10.1016/j.ins.2018.07.045.
- 928 [101] G. Zanghirati and L. Zanni, "A parallel solver for large quadratic programs in training support vector
929 machines," *Parallel Comput.*, vol. 29, no. 4, pp. 535–551, Apr. 2003, doi: 10.1016/S0167-
930 8191(03)00021-8.
- 931 [102] T. Eitrich and B. Lang, "On the optimal working set size in serial and parallel support vector machine
932 learning with the decomposition algorithm," *Conf. Res. Pract. Inf. Technol. Ser.*, vol. 61, pp. 121–
933 128, 2006.
- 934 [103] T. Serafini, L. Zanni, and G. Zanghirati, "Some improvements to a parallel decomposition technique
935 for training support vector machines," in *Lecture Notes in Computer Science (including subseries*
936 *Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2005, vol. 3666 LNCS,
937 pp. 9–17, doi: 10.1007/11557265_7.
- 938 [104] S. Qiu and T. Lane, "Parallel computation of RBF kernels for support vector classifiers," *Proc. 2005*
939 *SIAM Int. Conf. Data Mining, SDM 2005*, pp. 334–345, 2005, doi: 10.1137/1.9781611972757.30.
- 940 [105] G. Fung and O. L. Mangasarian, "Proximal support vector machine classifiers," in *Proceedings of*

- 941 *the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*,
942 2001, pp. 77–86, doi: 10.1145/502512.502527.
- 943 [106] M. Wu, B. Schölkopf, and G. Bakir, “Building sparse large margin classifiers,” *ICML 2005 - Proc. 22nd*
944 *Int. Conf. Mach. Learn.*, pp. 1001–1008, 2005, doi: 10.1145/1102351.1102477.
- 945 [107] S. S. Keerthi and D. Decoste, “A modified finite Newton method for fast solution of large scale
946 linear SVMs,” *J. Mach. Learn. Res.*, vol. 6, pp. 341–361, 2005.
- 947 [108] C. J. Hsieh, K. W. Chang, C. J. Lin, S. S. Keerthi, and S. Sundararajan, “A dual coordinate descent
948 method for large-scale linear SVM,” in *Proceedings of the 25th International Conference on*
949 *Machine Learning*, 2008, pp. 408–415, doi: 10.1145/1390156.1390208.
- 950 [109] L. Ladický and P. H. S. Torr, “Locally linear support vector machines,” *Proc. 28th Int. Conf. Mach.*
951 *Learn. ICML 2011*, pp. 985–992, 2011.
- 952 [110] I. Rodriguez-Lujan, C. Santa Cruz, and R. Huerta, “Hierarchical linear support vector machine,”
953 *Pattern Recognit.*, vol. 45, no. 12, pp. 4414–4427, 2012, doi: 10.1016/j.patcog.2012.06.002.
- 954 [111] M. Fornoni, B. Caputo, and F. Orabona, “Multiclass latent locally linear support vector machines,”
955 *J. Mach. Learn. Res.*, vol. 29, pp. 229–244, 2013.
- 956 [112] R. Nanculef, E. Frandi, C. Sartori, and H. Allende, “A novel Frank-Wolfe algorithm. Analysis and
957 applications to large-scale SVM training,” *Inf. Sci. (Ny)*, vol. 285, no. 1, pp. 66–99, 2014, doi:
958 10.1016/j.ins.2014.03.059.
- 959 [113] D. Wang, X. Zhang, M. Fan, and X. Ye, “Hierarchical mixing linear support vector machines for
960 nonlinear classification,” *Pattern Recognit.*, vol. 59, pp. 255–267, 2016, doi:
961 10.1016/j.patcog.2016.02.018.
- 962 [114] M. H. Yang and N. Ahuja, “Geometric approach to train support vector machines,” in *Proceedings*
963 *of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2000, vol.
964 1, pp. 430–437, doi: 10.1109/cvpr.2000.855851.
- 965 [115] D. Feng, W. Shi, H. Guo, and L. Chen, “A new alpha seeding method for support vector machine
966 training,” in *Lecture Notes in Computer Science*, 2005, vol. 3610, no. PART I, pp. 679–682, doi:
967 10.1007/11539087_87.
- 968 [116] D. DeCoste and K. Wagstaff, “Alpha seeding for support vector machines,” in *Proceeding of the*
969 *Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp.
970 345–349, doi: 10.1145/347090.347165.
- 971 [117] M. Arun Kumar and M. Gopal, “A hybrid SVM based decision tree,” *Pattern Recognit.*, vol. 43, no.
972 12, pp. 3977–3987, Dec. 2010, doi: 10.1016/j.patcog.2010.06.010.
- 973 [118] A. A. Aburomman and M. Bin Ibne Reaz, “A novel weighted support vector machines multiclass
974 classifier based on differential evolution for intrusion detection systems,” *Inf. Sci. (Ny)*, vol. 414,
975 pp. 225–246, Nov. 2017, doi: 10.1016/j.ins.2017.06.007.
- 976 [119] N. Settu and M. Rajasekhara Babu, “Enhancing the performance of decision tree using NSUM
977 technique for diabetes patients,” in *SpringerBriefs in Applied Sciences and Technology*, Springer
978 Verlag, 2019, pp. 13–20.
- 979 [120] C. Domeniconi and D. Gunopulos, “Incremental support vector machine construction,” *Proc. - IEEE*

980 *Int. Conf. Data Mining, ICDM*, pp. 589–592, 2001, doi: 10.1109/icdm.2001.989572.

981 [121] P. Mitra, C. A. Murthy, and S. K. Pal, “Data condensation in large databases by incremental learning
982 with support vector machines,” *Proc. - Int. Conf. Pattern Recognit.*, vol. 15, no. 2, pp. 708–711,
983 2000, doi: 10.1109/icpr.2000.906173.

984 [122] S. Rüping, “Incremental learning with support vector machines,” *Proc. - IEEE Int. Conf. Data Mining,*
985 *ICDM*, pp. 641–642, 2001, doi: 10.1109/icdm.2001.989589.

986 [123] J. Kivinen, A. J. Smola, and R. C. Williamson, “Online learning with kernels,” *IEEE Trans. Signal*
987 *Process.*, vol. 52, no. 8, pp. 2165–2176, 2004, doi: 10.1109/TSP.2004.830991.

988 [124] L. Cheng, S. V. N. Vishwanathan, D. Schuurmans, S. Wang, and T. Caelli, “Implicit online learning
989 with kernels,” *Adv. Neural Inf. Process. Syst.*, pp. 249–256, 2007, doi:
990 10.7551/mitpress/7503.003.0036.

991 [125] O. L. Mangasarian and D. R. Musicant, “Lagrangian Support Vector Machines,” *J. Mach. Learn. Res.*,
992 vol. 1, no. 3, pp. 161–177, 2001, doi: 10.1162/15324430152748218.

993 [126] T. Glasmachers and C. Igel, “Second-order SMO improves SVM online and active learning,” *Neural*
994 *Comput.*, vol. 20, no. 2, pp. 374–382, 2008, doi: 10.1162/neco.2007.10-06-354.

995 [127] J. Zheng, F. Shen, H. Fan, and J. Zhao, “An online incremental learning support vector machine for
996 large-scale data,” *Neural Comput. Appl.*, vol. 22, no. 5, pp. 1023–1035, 2013, doi: 10.1007/s00521-
997 011-0793-1.

998 [128] Y. Ma, Y. He, and Y. Tian, “Online Robust Lagrangian Support Vector Machine against Adversarial
999 Attack,” *Procedia Comput. Sci.*, vol. 139, pp. 173–181, 2018, doi: 10.1016/j.procs.2018.10.239.

1000 [129] G. Cauwenberghs and T. Poggio, “Incremental and decremental support vector machine learning,”
1001 *Adv. Neural Inf. Process. Syst.*, 2001.

1002 [130] M. Karasuyama and I. Takeuchi, “Multiple Incremental Decremental Learning of Support Vector
1003 Machines,” *IEEE Trans. Neural Networks*, vol. 21, no. 7, pp. 1048–1059, Jul. 2010, doi:
1004 10.1109/TNN.2010.2048039.

1005 [131] C. P. Diehl and G. Cauwenberghs, “SVM Incremental Learning, Adaptation and Optimization,” *Proc.*
1006 *Int. Jt. Conf. Neural Networks*, vol. 4, no. x, pp. 2685–2690, 2003.

1007 [132] J. L. An, Z. O. Wang, and Z. P. Ma, “An incremental learning algorithm for support vector machine,”
1008 *Int. Conf. Mach. Learn. Cybern.*, vol. 2, no. November, pp. 1153–1156, 2003, doi:
1009 10.1109/icmlc.2003.1259659.

1010 [133] K. Crammer, J. Kandola, and Y. Singer, “Online classification on a budget,” *Adv. Neural Inf. Process.*
1011 *Syst.*, 2004.

1012 [134] J. Weston, A. Bordes, and L. Bottou, “Online (and Offline) on an even tighter budget,” *AISTATS 2005*
1013 *- Proc. 10th Int. Work. Artif. Intell. Stat.*, no. May, pp. 413–420, 2005.

1014 [135] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, “Fast kernel classification with online and active
1015 learning,” *J. Mach. Learn. Res.*, vol. 15, pp. 1579–1619, 2003.

1016 [136] S. Cheng and F. Y. Shih, “An improved incremental training algorithm for support vector machines
1017 using active query,” *Pattern Recognit.*, vol. 40, no. 3, pp. 964–971, 2007, doi:

1018 10.1016/j.patcog.2006.06.016.

1019 [137] W. Y. Cheng and C. F. Juang, "An incremental support vector machine-trained TS-type fuzzy system
1020 for online classification problems," *Fuzzy Sets Syst.*, vol. 163, no. 1, pp. 24–44, 2011, doi:
1021 10.1016/j.fss.2010.08.006.

1022 [138] T. Kefi-Fatteh, R. Ksantini, M. B. Kaâniche, and A. Bouhoula, "A novel incremental one-class support
1023 vector machine based on low variance direction," *Pattern Recognit.*, vol. 91, pp. 308–321, 2019,
1024 doi: 10.1016/j.patcog.2019.02.027.

1025 [139] K. W. Lau and Q. H. Wu, "Online training of support vector classifier," *Pattern Recognit.*, vol. 36,
1026 no. 8, pp. 1913–1920, 2003, doi: 10.1016/S0031-3203(03)00038-4.

1027 [140] S. Agarwal, V. Vijaya Saradhi, and H. Karnick, "Kernel-based online machine learning and support
1028 vector reduction," *Neurocomputing*, vol. 71, no. 7–9, pp. 1230–1237, 2008, doi:
1029 10.1016/j.neucom.2007.11.023.

1030 [141] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data Mining with Big Data," *Knowl. Data Eng. IEEE Trans.*,
1031 vol. 26, no. 1, pp. 97–107, 2014, doi: 10.1109/TKDE.2013.109.

1032 [142] H. Kashyap, H. A. Ahmed, N. Hoque, S. Roy, and D. K. Bhattacharyya, "Big Data Analytics in
1033 Bioinformatics: A Machine Learning Perspective," vol. 13, no. 9, pp. 1–20, 2015, doi:
1034 10.1080/10714413.2017.1344512.

1035 [143] C.-W. Tsai, C.-F. Lai, H.-C. Chao, and A. V. Vasilakos, "Big data analytics: a survey," *J. Big Data*, vol.
1036 2, no. 1, p. 21, Dec. 2015, doi: 10.1186/s40537-015-0030-3.

1037 [144] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun.*
1038 *ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008, doi: 10.1145/1327452.1327492.

1039 [145] E. E. Drakonaki and G. M. Allen, "Spark: Cluster Computing with Working Sets Matei," *Skeletal*
1040 *Radiol.*, vol. 39, no. 4, pp. 391–396, 2010, doi: 10.1007/s00256-009-0861-0.

1041 [146] S. Wadkar and M. Siddalingaiah, "Apache Ambari," in *Pro Apache Hadoop*, Berkeley, CA: Apress,
1042 2014, pp. 399–401.

1043 [147] "Apache Hadoop." <https://hadoop.apache.org/> (accessed May 04, 2021).

1044 [148] X. Meng *et al.*, "MLlib: Machine Learning in Apache Spark," vol. 17, pp. 1–7, 2015, [Online].
1045 Available: <http://arxiv.org/abs/1505.06807>.

1046