Support Vector Machine

Emmanuella Anggi Siallagan (2106678006)

December 22, 2022

Abstract

Vapnik first introduced the concept of Support Vector Machine (SVM) in 1962. The general idea of SVM lies in the concept of support vectors point that builds a linear separating line between the classes. SVM emphasizes the concept of using a margin to separate the different classes on the dataset from each other. This approach then became one of the most renowned machine learning methods. After more than 50 years since SVM was introduced and the rise of many more machine learning techniques, SVM remained one of the most commonly used algorithms. This paper presents a comprehensive study of SVM from the theoretical aspects, benefits, and challenges, former to very recent applications of SVM.

1 Introduction

Support vector machine (SVM) was first presented in 1962. Vapnik mentioned in his book Estimation of Dependencies Based on Empirical Data [1] that the development of SVMs took a 30-year history from 1965 to 1995. He summarized it into three stages: (1) Finding the optimal separating decision boundary (which is called hyperplane), (2) Controlling the capacity in the Hilbert Space, and then finally to the term of (3) SVM. These stages derived to idea of the support-vector networks with combining three solutions: (1) optimal hyperplanes with no errors, (2) convolution on the dot-product that expands the idea of linearity to non-linearity, and the last (3) the idea of soft margins that takes the function on (1) but with the relaxation of some extend of error. It was developed for a binary classification problem around the idea that input vectors that are not linearly separable can be mapped to a higher dimensional feature space. The function needs to be constructed as a linear decision so that it can ensure its high generalization. The restriction of the separable dataset eases the decision result. The use of SVM mostly revolves around the systematic statistical learning theory [2]. It is fully optimized because of the global optimization, is strongly adaptive, and has well-formed generalization. The training process also involves the process of optimizing the convex cost function so that the local minima don't complicate the learning process.

SVM has been implemented in most of the fields that use classification problems. It is a robust technique for observation and small insufficient datasets–especially, in higher dimensional spaces. The support vectors that build the SVM also work well in an unbalanced dataset for binary outcome [3]. SVM has been implemented in a wide range of fields, from image classification to text classification, data mining, and many more. Upon looking at the research database, we will see hundreds–if not thousands–records of implementation of SVM, and so many libraries that provide SVM tools directly. Still, though the concept is very practical, SVM hugely faces the resources issue. It scales with the data points, the more the dataset, the slower the training process [4]. SVM also relies so much on the data selection, hence the cleaning and transformation become one pivotal process for every dataset used. It becomes a bigger challenge if there are cases of incomplete data, missing values, or outliers.

This paper is written as follows: First, in Section 1, we discuss the big idea of SVM. Next in Section 2, we start to elaborate the theoretical aspect of SVM which we reviewed from Vapnik's article [5] with the perspective of advantages and disadvantages. In Section 3, we talk about the advancement with some modifications from the original theory. Section 4 reviews each implementation of a support vector machine in many fields, furthermore in this section, we also discuss the implementation of SVM in deep learning. Last, we give Conclusions of the reviewed papers in Section 5.

2 Underlying Theory of SVM

Vapnik started the original paper of SVM [5] by introducing the idea of pattern recognition which was first presented by Fisher [6] which shows two populations' optimum solution on Bayesian by a quadratic decision function. This is achieved by providing a linear function with n-free parameters. This was obtained with a very small number observation, estimated less than $10n^2$ with $O(n^2)$ non-reliable parameters assumption. Following Fisher with a different angle, Rosenblatt [7] presented a separating hyperplane on the neuron so that the perceptron–connected neurons–implement the linear with a separating surface. However, till that time the algorithms which were introduced didn't minimize the error on the vectors by adjusting the weight from Rosenblatt's angle and the weights of the output presumed to be adaptive.

Tackling these challenges from both papers, Vapnik introduced the Support Vector Machines, formerly introduced as support-vector networks that take the very small number observation problem into the idea of higher dimensional features. The input vectors are taken to a high dimensional feature space Z with non-linear mapping that is defined by the given theoretical definition. The linear space in which the decision surface is defined carefully so that it takes the correct generalization of the input vector. Vapnik introduced the support-vector networks general idea by combining three solutions: (1) optimal hyperplanes with no errors, (2) convolution on the dot-product that expands the idea of linearity to non-linearity, and the last (3) the idea of soft margins that takes the function on (1) but with the relaxation of some extend of error.

Given $\mathbf{w}_0.\mathbf{z} + b_0 = 0$ is the optimal decision boundary (hyperplane) for the feature vector. Vapnik then proved that this optimal plane was supported by the sum of support vectors $\mathbf{w}_0 = \sum_{\text{support vectors}} \alpha_i \mathbf{z}_i$, with the function of the boundary condition is $\boldsymbol{l(z)} = \text{sign}(\sum_{\text{support vectors}} \alpha_i \mathbf{z}_i.\mathbf{z} + b_0)$. Where in the boundary line, the $\mathbf{z}_i.\mathbf{z}$ is a dot-product of the support vector z_i and its vector z from the feature vector. Even though the generalization gives the optimal hyperplane, the technical problem of treating the high dimensions persists.

We will further explain the mathematical theory in SVM in three big concepts: the General Concept, Lagrangian Multipliers, and Soft-SVM.

2.1 General Concept

Given set of labeled training set of:

$$(y_1, \mathbf{x}_1), ..., (y_\ell, \mathbf{x}_\ell) \qquad y_i \in \{-1, 1\}$$

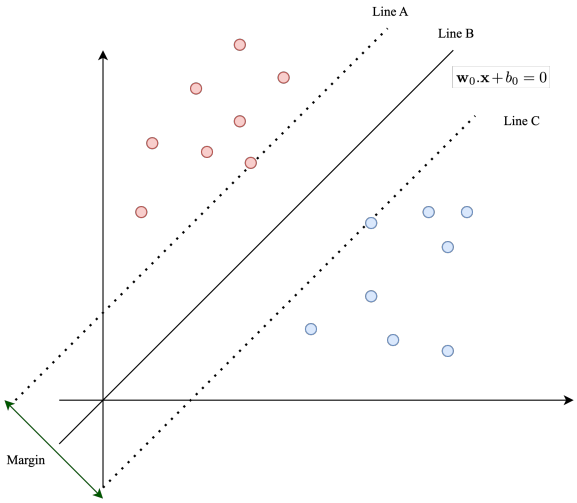

Figure 1: Example of Margin

which is linearly separable with vector $\mathbf{w}$ and scalar $\mathbf{b}$ given the inequality function

$$\begin{cases} \mathbf{w}.\mathbf{x}_i + b \geq 1, & \text{if } y_i = 1 \\ \mathbf{w}.\mathbf{x}_i + b \leq -1, & \text{if } y_i = -1 \end{cases} \tag{1}$$

that maps to the constraint function of:

$$y_i(\mathbf{w}.\mathbf{x}_i + b) \geq 1 \qquad i = 1, .., \ell \tag{2}$$

Hence the hyperplane $\mathbf{w}_0.\mathbf{x}+b_0 = 0$ is the optimal solution that separates the training set with the largest possible margin and its direction of the two classes (given w/—w— and the distance between two planes is maximum of:

$$\rho(\mathbf{w}, b) = \min_{\{\mathbf{x}:y=1\}} \frac{\mathbf{x}.\mathbf{w}}{|\mathbf{w}|} - \max_{\{\mathbf{x}:y=-1\}} \frac{\mathbf{x}.\mathbf{w}}{|\mathbf{w}|}. \tag{3}$$

So that being defined, the optimal hyperplane $(\mathbf{x}_0, b_0)$ in Line B of Figure 1 is the argument that maximizes the distance in Problem 3 is:

$$\rho(\mathbf{w}, b) = \frac{2}{|\mathbf{w}_0|} = \frac{2}{\sqrt{\mathbf{w}_0.\mathbf{w}_0}}. \tag{4}$$

Concluding that this approach to achieving the optimal hyperplane is a quadratic programming problem.

Next, we will get into the Langragian use in SVM which will be needed to determine the optimal hyperplane $\mathbf{w}_0$ can be written as a linear combination of vectors input $\mathbf{w}_0 = \sum_{i=1}^{\ell} y_i \alpha_i^0 \mathbf{x}_i$.

2.2 Lagrangian Multipliers in SVM

Lagrangian is one related theory that builds the SVM. On SVM, Lagrange expresses the dual problem of the SVM origin expression where minimizing the Lagrange multiplier comes simpler than minimizing the original expression. In this subchapter, we will discuss Langragian before discussing the implementation in soft-SVM.

To maximize $\mathbf{\Phi} = \mathbf{w} \cdot \mathbf{w}$, used Lagrangian as the standard optimization theory for solving the minimization-constrained problem:

$$L(\mathbf{w}, b, \Lambda) = \frac{1}{2}\mathbf{w} \cdot \mathbf{w} - \sum_{i=1}^{\ell} \alpha_i [y_i(\mathbf{x}_i.\mathbf{w} + b) - 1] \tag{5}$$

Where the $\Lambda^T = (\alpha_1, ..., \alpha_\ell)$ is the non-negative vectors of Lagrange multipliers that correspond to the initial constraints of Function 2.

The solution to the optimization problem of this form will be defined by the definition point of the Lagrangian function at the $2\ell + 1$-dimensional space of $\mathbf{w}, \Lambda,$ and b The minimal value will be taken w.r.t the parameters $\mathbf{w}$ and b, while the maximal value will be taken from the Langrage, multiplied w.r.t Λ. With the minimum point of $\mathbf{w}$ and b, obtained:

$$\frac{\partial L(\mathbf{w}, b, \Lambda)}{\partial \mathbf{w}}|_{\mathbf{w}=\mathbf{w}_0} = (\mathbf{w}_0 - \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i) = 0, \tag{6}$$

$$\frac{\partial L(\mathbf{w}, b, \Lambda)}{\partial \mathbf{b}}|_{b=b_0} = \sum_{i=1} y_i \alpha_i = 0, \tag{7}$$

From the Equality 6 alone, we got the following derivation:

$$\mathbf{w}_0 = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i \tag{8}$$

This describes that the hyperplane that is the optimal solution is also a linear combination of the vectors input for training. With the $\alpha_i > 0$ to be effective to the result of the sum on the Derivation 8. Exchanging Derivation 8 and 7 to the Lagrangian (Equation 5). Thus, we get:

$$W(\Lambda) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2}\mathbf{w}_0.\mathbf{w}_0 \tag{9}$$

$$W(\Lambda) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j \mathbf{x}_i.\mathbf{x}_j. \tag{10}$$

From the perspective of vector notation we can define Problem 10 with:

$$W(\Lambda) = \Lambda^T \boldsymbol{l} = \frac{1}{2}\Lambda^T \mathbf{D}\Lambda \tag{11}$$

where $\boldsymbol{l}$ is an $\boldsymbol{l}$-dimensional vector and the D is the symmetrical $\ell \times \ell$-matrix which comes from $D_{ij} = y_i y_j \mathbf{x}_i.\mathbf{x}_j$. The maximum point of Equation 11 lies on the Constraint 6 that defined from $\Lambda^T \mathbf{Y} = 0$ of $\mathbf{Y}^T = (y_1, ..., y_\ell)$ with $\Lambda \geq 0$.

From this, we are introduced to the Karush-Kuhn-Tucker theory. Karush-Kuhn-Tucker theory or KKT theory are the conditions where the first derivative tests for some solution needs to be in the nonlinear programming to achieved the optimal; achieved with some regularity conditions satisfied. The point of all Lagrange corresponding multiplier α_i^0 of the constraint (w_0, b_0, Λ_0) will be connected on the equality $\alpha_i[y_i(\mathbf{x}_i.\mathbf{w} + b) - 1] \quad i = 1, ..., \ell$. If α_i is not 0, the only chance of it being achieved is with $y_i(\mathbf{x}_i.\mathbf{w}_0 + b_0) - 1 = 0$. So in the case of α_i is not zero; we see that the inequality only met on equality. Hence we concludes that every support vectors x_i is $y_i(\mathbf{x}_i.\mathbf{w}_0 + b_0) = 1$.

Another view of Kuhn-Tucker on Equation 9 and Equation 10 as the solution of maximum value of $W(\Lambda_0)$ and ρ_0 as the separation distance of:

$$\mathbf{w}_0.\mathbf{w}_0 = \sum_{i=1}^{\ell} \alpha_i^0 y_i \mathbf{x}_i \mathbf{w}_0 = \sum_{i=1}^{\ell} \alpha_i^0 y_i (1 - y_i b_0) = \sum_{i=1}^{\ell} \alpha_i^0. \tag{12}$$

Exchanging Equation above into Equation 9 to get $\boldsymbol{W}(\Lambda_0)$ we get:

$$\boldsymbol{W}(\Lambda_0) = \sum_{i=1}^{\ell} \alpha_i^0 - \frac{1}{2}\mathbf{w}_0.\mathbf{w}_0 = \frac{\mathbf{w}_0.\mathbf{w}_0}{2}.$$

Based on

$$\rho(\mathbf{w}_0, b_0) = \frac{1}{|\mathbf{w}_0|} = \frac{2}{\sqrt{\mathbf{w}_0.\mathbf{w}_0}},$$

where the ρ_0 is the optimal hyperplane. Margin that separates the training set is valid with $\rho < \sqrt{\frac{2}{W_0}}$, we took that $\mathbf{W}(\Lambda_0) = \frac{2}{\rho_0^2}$ where the ρ_0 is the optimal hyperplane. The margin that separates the training set is valid with $\rho < \sqrt{\frac{2}{W_0}}$. When obtained the inequality of $\mathrm{W}(\Lambda_\star) > W_0$ for $\Lambda_\star$ and large constant W_0.

Next, we will discuss when the training set is linearly not separable, Vapnik and Cortes [5] introduce the relaxation of the equation on SVM with the idea of Soft Margin SVM.

2.3 Soft-SVM

Soft-SVM comes as the relaxation of SVM that was discussed before: where it's certain that the hyperplane separates the two classes with zero error. The real dataset will unlikely cover that scenario. Hence, Vapnik and Cortes introduced the idea of Soft-SVM or mentioned as a soft-margin hyperplane.

Assuming that there will be an error that will be tolerated on the hyperplane that separates the two classes, we will use a non-negative variable. This variable is the relaxation of the error points from the dataset, hence what we aim for is the value of this variable as smallest as possible. Therefore, with $\xi_i \geq 0, i = 1, ..., \ell$. We will minimize the sum of it, represented with If the sum of the variables of the error is relatively small, it is safe to take a new subset without the error $(y_i, \mathbf{x}_i), ..., (y_{i_k}, \mathbf{x}_{i_k})$, or formally written as:

$$\Phi(\xi) = \sum \xi_i \sigma.$$

Then our constraints will be:

$$y_i(\mathbf{w} \ . \ \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, ..., \ell, \xi_i \geq 0, \quad i = 1, ..., \ell$$

If the sum of the variables of the error is relatively small, it is safe to take a new subset without the error $(y_i, \mathbf{x}_i), ..., (y_{i_k}, \mathbf{x}_{i_k})$, or formally written as:

$$\frac{1}{2}\mathbf{w}^2 + CF(\sum_{i=1}^{\ell} \xi_i^\sigma) \tag{13}$$

where $F(u)$ is a convex function and C is a constant. Hence the larger C and the smaller σ is, vector w_0 and b_0 is the one that determines how much the number error on the maximal margin hyperplane that separates the training set.

Nevertheless, this approach to minimizing the number of errors is general NP-complete (nondeterministic polynomial-time complete), where the solution of the problem is predictable and can be

calculated in polynomial time with no certain rule (nondeterministic) rule followed for guessing the time. Therefore in avoiding NP-completeness, σ will be considered 1. On the large C, the Problem 13 will then describe getting hyperplane with minimal sum deviations of ξ in the training error and with maximal margin in with the correct points from training. This approach is called the soft-margin hyperplane.

Based on the before-explained Langragian, the optimal hyperplane algorithm with vector $\mathbf{w}$ can be defined as a linear combination of support vectors x_i, where $\mathbf{w}_0$ is the sum of $\alpha_i^0 y_i \mathbf{x}_i$. To find the vector of $\Lambda^T = (\alpha_1, ..., \alpha_\ell)$, the quadratic programming problem on maximizing should be solved with:

$$W(\Lambda, \delta) = \Lambda^T \mathbf{l} - \frac{1}{2}\mathbf{D}\Lambda + \frac{\delta^2}{C} \text{ s. t the constraints } \Lambda\mathbf{Y} = 0, \delta \geq 0, 0 \leq \Lambda \leq \delta 1, \tag{14}$$

Where all the elements used $\mathbf{1}$, Λ, $\mathbf{Y}$, and, $\mathbf{D}$ are the same elements that constructed the hyperplane; δ is scalar and coordinate-wise inequalities represented with $0 \leq \Lambda\delta\mathbf{l}$, that implies that the smallest admissible value of δ on $\delta = \alpha_{\max}(\alpha_1, ..., \alpha_\ell)$.

Consequently to the equation, achieving the soft-margin classifier hyperplane there should be achieved vector Λ that maximizes under the constraint of $\Lambda \geq 0$ and $\Lambda^T\mathbf{Y}= 0$ is:

$$W(\Lambda) = \Lambda^T \mathbf{l} - \frac{1}{2}[\Lambda^T \mathbf{D}\Lambda + \frac{\delta^2_{\max}}{C}]$$

With this equation, the problem between the optimal margin classifier where there exists no error (or as known separately as Hard-SVM) and Soft-SVM is the $\alpha_{\max}$. Soft-SVM becomes a unique construction and can exist for any training set. The unique solution of the construction becomes one powerful benefit of Soft-SVM. On side note, while constructing the Soft-SVM, the convex programming problem should be in the l-dimensional space with the parameters Λ or in the dual programming $l+1$ space with Λ and δ. Vapnik and Cortes solved the dual quadratic programming problem.

2.4 Advantages and Challenges

SVM is a powerful tool in Machine Learning. The use of Support Vectors makes SVM a robust technique of observation–the model focuses on the support vectors that build the hyperplane. It also suits a small number of a dataset with a large number of predictors; comes handier too in high dimensional spaces, where even the number of training sets is less than the dimensions. The support vectors that build the SVM also work well in an unbalanced dataset for binary outcome [3].

From the experiments conducted in 2014 [8], comparing 179 classifier algorithms in 121 datasets, SVM was among the top-performing algorithms in classification problems. Proving that even after more than fifty years, this algorithm is working exceptionally well in the field. However, as believed in the No-Free-Lunch (NFL) theory [9], there is no single algorithm that can solve all the problems, as well as SVM. There are some challenges that SVM faced too, such as:

1. **Algorithmic complexity**. SVM requires a very large amount of computational effort and memory for estimation. This will make the process heavy especially if the training data is very large [10]. When the dataset increases, the learning process comes even slower and takes more resources [11]. This is not suitable when we are pursuing the rise of the big data trend.

2. **Kernel Dependency**. Though we don't explain the kernel further in this paper, the dependency of the SVM model on the use of kernel is one of the weaknesses of SVM. The use of kernel then also depends on the parameters. The process of choosing the parameters is very crucial since this impacts directly on the performance [10]. When the features are greater than the samples

by a lot, then the only way to avoid overfitting is by using the kernel function and regularization terms.

3. **Indirect Estimation**. The SVM model does not provide a direct probability estimation, instead, it is achieved by using five-fold cross-validation. This process is not only expensive but also seemingly like a black box in which the final form of the function or the coefficients of the predictors is not provided.

3 Advancements

For more than 50 years since SVM was first introduced, the ground concept of SVM had built a solid foundation for many advancements to various problems derived from the traditional SVM. In the following section, we reviewed a few of these advancements.

3.1 Multi-class and Multi-task SVM

From the original paper, SVM was initially made for 2-class prediction or binary classification. This does not apply to the rising problems in machine learning. The two approaches of Multi-class SVM, introduced by [12] derived two modifications: one direct optimization for all data and combining a few of the 2-class classifications.

The early approach on multi-class SVM of this one-against-all and one-against-one approach is the one commonly used for the multi-class SVM. The idea of one-against-all is to train the SVM repetitively based on the number of classes; a binary classifier on this will be constructed with two considerations: positive and negative. The one-against-one use approach of sampling two classes and getting $\frac{K(K-1)}{2}$, number of models (K is the number of classes) [13]. Then it uses a voting scenario on all the classes that are unseen in each model sampling. However, this scenario also faced the error prediction problem. To improve this, [14] used a different evolution to compensate for the miss classified predictions on the SVM binary classifier.

3.2 Online SVM

Online data processing is very needed in the era of big data, where the data are flowing very fast with much needed to focus on variety, velocity, volume, value, and veracity. The data comes more dynamic and is a challenge for the model to be updated in real-time to perform better hence online. This means, not only need to be trained online, but the process needs to be fast also. Though very complicated to be implemented on SVM, this approach was proposed [15]. The method uses continuous sampling for training the streaming data and updating the prototypes of the model to the data concept continuously. Another approach more recently uses Representative Prototype Area (RPA) where it retained the representation from the historical data, maintained by Online Incremental Feature Map (OIM) [16]. This proves to be effective to be applied to a million samples from a more enormous dataset. In the application, this will benefit much in many important industries such as financial markets, healthcare, finance, and social engineering [17].

3.3 Distributed SVM (DSVM)

Distributed SVM is an SVM approach that takes sub-regions divisions and trained the SVM separately and is built on connected networks. This approach aimed to get the approximation of the solution from

the connected networks. There are some proposed methods on the DSVM, first using parallel SVM [18]; strong connectivity [19]; wireless sensor network [20]; semi-supervised SVM [21], and resilient DSVM [22]. These approaches diversify in the approach of using the connection. The challenges of this approach mostly focus on transferring the model information between each of the learning nodes. Parallel SVM uses distributed raw data propagation and distributed semiparametric SVM to reduce total information in the sharing network. The newer DSVM method called the SVM classification tree uses fuzzy entropy on finding optimum clusters on the overlapped dataset space [23]. While a very intricate approach, DSVM is a very popular direction on the application [24].

3.4 Large-scale problems

The large-scale problem rises because of the time complexity it faces in solving quadratic programming (QP) problems. When the dataset is large, the instances are also rising and use very big computational resources. This made SVM very hard to be trained on a large dataset. There are some approaches to facing these problems, some of which are by (1) under-sampling and (2) transforming the QP problem. Under-sampling takes a representative sample from the large dataset to be trained with SVM. The challenge lies in the selection of taking a small sample. One proposed solution is by taking the support vector candidates from a convex-hull [25]. The observation to take these sample are by choosing the support vector with considering the position of the convex-hull. One other approach uses the beneficial of opposite class [26]. This approach uses the selection of lowest margin instances for the selection scheme by having the small votes instance selection. If in traditional SVM, the consideration lies on the distance between the margin and the data point, this approach also considers the margin of the opposite class. Second, the approach to tackling the large-scale problem is the transformation of the QP. The approach simplifies the QP to reduce the complexity and hence reduce memory and time use. One common example of this is sequential minimal optimization (SMO) [27]. SMO breaks down the QP problem into sub-problems as small as they can be. This way, the SVM becomes simpler and can be solved analytically and avoids the inner loop on the QP optimization. This approach resulted thousand times faster model.

4 Applications

In the original publications of SVM, Vapnik proves SVM's performance on a digit recognition problem. SVM is renowned for its ability to detect patterns. In this section, we review some of the implementations of SVM in a few other relevant real-world problems and close with the review of what SVM used in modern days of machine learning.

4.1 Image Classification

Image classification is one of the most common problems that is iteratively improved in machine learning. The idea of image classification gives the prediction label to the set of images based on the extracted feature from the images. There are many approaches and modifications to using SVM for classification. The main aspect of using SVM for classification is the kernel use [28]. Other than the kernel use, another integral part is the feature extraction and selection and assessing the measurement of success.

Brajesh Kumar et al. proposed a method by using texture and spectra from the images. These hyper-spectral features work well for the small training dataset. The extraction feature process uses

invariants [29]. Another implementation of using SVM for image classification is using remote sensing [30]. In addition, the approach uses the meta-heuristic algorithm to improve the accuracy: Cuckoo Search and Artificial Bee colony algorithm. This approach has proven to use the search space coherently to improve accuracy. Wen Yang et al. use semi-supervised feature learning which combines SVM and ensemble projection (EP) to classify satellite images on an incomplete dataset. The approaching transfer of the limited data then ensemble to WT sets, the sets then project the image back to the ensemble set, then extract the preliminary feature. On the various WT sets, it ensemble the sampling algorithm. The newer approach utilizes neighbor sampling and then discriminating functions learned from the WT sets [31]. Another common implementation of image classification is on medical images. With SVM, Imène Garali et al. [32] uses Positron Emission Tomography Brain Images on predicting neurodegenerative disease. This approach is applied by calculating the first two derivations of volumes of interest, then intensity mean value is applied as a feature and gives better accuracy. Another exploration of SVM is by implementing only one class to get hyperparameters [33]. This uses a self-adaptive data-shifting approach and creates high-quality pseudo outliers. This will target data on getting the error from outlier and target without having to achieve any new manual hyperparameters. The approach applied to get the classification of gender from facial images.

4.2 Text Classification

In an early paper on text classification (originally called text categorization), text classification with SVM introduced particular properties that related to text such as feature spaces in high dimension, and relevant features with dense and sparse vectors. SVM in text classification succeeds to avoids the failure that used to happen with preceding methods. SVM said to not require any additional tuning on the parameters since the parameter settings from SVM are performing well enough [34]. This is also supported by other paper that shows the SVM classifier reached more than 86.26% and improved significantly compares to other methods in that era [35]. In more recent research, SVM still shows the superiority of a conventional traditional classification algorithm. Another early paper that presents text classification with SVM introduces SVM with transductive method (TSVM) [36]. TSVM pays attention to the test set and focuses on minimizing the misclassification of the set. This approach has proven to work well in a small dataset, with the ability to cut down the need for the training example. One other approach uses the X^2 from statistic learning to improve the feature selection for the SVM. With this, the number of categories becomes small and can be divided into very distinctive and practical to be used [37]. From another perspective, one of the pain points of classification is to have the data labeled correctly. There is yet another approach introduced to tackling this side of concern [38]. The goal is to minimize the effort to label by selecting the crucial data to be labeled. So if needed to be labeled, the manual grader can just focus on these selective data. This is applied by selecting batches of the sample that are considered informative. Next, use the posterior probabilities–taken by the SVM classifier–and labeled manually by the expert. The last one that will be addressed in this paper is the exploration of the Scopus dataset. This applied to the Scopus dataset and performed 9% better than the KNN.

4.3 Data Mining

SVM is also applied not only on direct classification tasks but in the data mining process. R. Burbidge et. al. [39] implement SVM to showcase the performance of the algorithm on structure-activity analysis. The research implemented SVM to predict inhibition from the UCI machine learning repository.

SVM even performs better than the artificial neural network approaches as the benchmark. Shu-Xin Du et. al. [40] build an approach with weighted SVM. Where for every misclassification on each training iteration, given a different penalty. This approach results that with the improvement there is a possibility to decrease the accuracy on a large dataset. Zhong Yin [41] then proposed a data-driven framework for operator functional states assessment. The proposed method combines recursive feature elimination with least square SVM, applying that to both binary and multi-class SVM on the feature selection. Though much research applied in this field, some limitations are still concerning for applying this, such as the selection of kernel and its parameters, the time and memory resources for training and testing, and slow convergence [42].

4.4 SVM and Deep learning

SVM is known more as a conventional machine learning method. Now, the trend has shifted to neural network use. However still, SVM is relevant to this neural network trend. SVM implementations in deep learning are usually used as an alternative to SVMs on multistage processes. In its application, SVM is not only implemented as it is with bit modification to adjusting on the data, SVM is also implemented with deep learning. In Tang [43], propose a method which use deep architectures and puts SVM linear on top of the layers as a replaced to softmax. With this approach, the model is trained to layers of deep neural networks. On top of the layer level, there will be SVM model. SVM then backpropagates the gradients; hence get to learn all of the features results from all the layers. It uses stochastic gradient descent optimization on small minibatches. In another approach, it raises the challenges of the dimensionality which makes many anomaly on many other techniques, however SVM tackle this issue more efficiently. However, Erfani et al. [44] believes that this too became less in the performance when applied to a very large datasets. In other side, deep belief networks (DBN) are working better to a large dataset. Hence they propose a hybrid model. The DBN are trained unsupervised to get the generic features that will be used and one-class SVM will be the dataset that train the feature. In a more recent application, Al-Qatf et al. [45] introduced an approach that combine SVM with sparse autoencoder. This built on self-taught learning framework to lessen the dimensionality use; hence reducing the time resources too. The model built on sparse autoencoder mechanism that build another representation with an unsupervised manner. This approach proven to raise thea accuracy and uses the resources more efficient with lesser time. Last example on the application is using SVM in the pre-processing. SVM identify and localize the feature from the image. Then, these features are trained with ResNet-50. SVM improved the accuracy , as well speed up the pre-processing.

5 Conclusions

We present a survey paper on support vector machines, with more than 40 references. We detailed the underlying theory of SVM with its advantages and disadvantages. We then list three main points of challenges that SVM faces. Then we reviewed the existing advancements of SVM. We also talked about the wide range of implementations from SVM. Furthermore in the section, we also discuss implementations of SVM in the era of deep learning. SVM is proven to be an everlasting method that can still be efficient after decades introduced.

References

[1] V. Vapnik, *Estimation of dependences based on empirical data*. Springer Science & Business Media, 2006.

[2] A. Smola and B. Schölkopf, "From regularization operators to support vector kernels," *Advances in Neural information processing systems*, vol. 10, 1997.

[3] P. Attewell, D. Monaghan, and D. Kwong, *Data mining for the social sciences: An introduction*. Univ of California Press, 2015.

[4] R. Strack, V. Kecman, B. Strack, and Q. Li, "Sphere support vector machines for large classification tasks," *Neurocomput.*, vol. 101, p. 59–67, feb 2013. [Online]. Available: https://doi.org/10.1016/j.neucom.2012.07.025

[5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[6] R. A. FISHER, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x

[7] F. Rosenblatt, *Perceptions and the theory of brain mechanisms*. Spartan books, 1962.

[8] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *The journal of machine learning research*, vol. 15, no. 1, pp. 3133–3181, 2014.

[9] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

[10] G. Horváth, "Neural networks in measurement systems," *Advances in learning theory: methods, models and applications*, pp. 375–402, 2003.

[11] J. Nayak, B. Naik, and P. D. H. Behera, "A comprehensive survey on support vector machine in data mining tasks: Applications challenges," *International Journal of Database Theory and Application*, vol. 8, pp. 169–186, 02 2015.

[12] R. Debnath, N. Takahide, and H. Takahashi, "A decision based one-against-one method for multi-class support vector machine," *Pattern Analysis and Applications*, vol. 7, no. 2, pp. 164–175, 2004.

[13] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network," in *Neurocomputing*. Springer, 1990, pp. 41–50.

[14] A. A. Aburomman and M. B. I. Reaz, "A novel weighted support vector machines multiclass classifier based on differential evolution for intrusion detection systems," *Information Sciences*, vol. 414, pp. 225–246, 2017.

[15] J. Zheng, F. Shen, H. Fan, and J. Zhao, "An online incremental learning support vector machine for large-scale data," *Neural Computing and Applications*, vol. 22, no. 5, pp. 1023–1035, 2013.

[16] X. Wang and Y. Xing, "An online support vector machine for the open-ended environment," *Expert Systems with Applications*, vol. 120, pp. 72–86, 2019.

[17] A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer, *Machine learning for data streams: with practical examples in MOA*. MIT press, 2018.

[18] A. Navia-Vázquez, D. Gutierrez-Gonzalez, E. Parrado-Hernández, and J. Navarro-Abellan, "Distributed support vector machines," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, p. 1091, 2006.

[19] Y. Lu, V. Roychowdhury, and L. Vandenberghe, "Distributed parallel support vector machines in strongly connected networks," *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1167–1178, 2008.

[20] W. Kim, M. S. Stanković, K. H. Johansson, and H. J. Kim, "A distributed support vector machine learning over wireless sensor networks," *IEEE transactions on cybernetics*, vol. 45, no. 11, pp. 2599–2611, 2015.

[21] S. Scardapane, R. Fierimonte, P. Di Lorenzo, M. Panella, and A. Uncini, "Distributed semi-supervised support vector machines," *Neural Networks*, vol. 80, pp. 43–52, 2016.

[22] Z. Yang and W. U. Bajwa, "Rd-svm: A resilient distributed support vector machine," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2444–2448.

[23] P. de Boves Harrington, "Support vector machine classification trees based on fuzzy entropy of classification," *Analytica chimica acta*, vol. 954, pp. 14–21, 2017.

[24] H. Wang, J. Xiong, Z. Yao, M. Lin, and J. Ren, "Research survey on support vector machine," in *10th EAI International Conference on Mobile Multimedia Communications*, 2017, pp. 95–103.

[25] E. Osuna and O. D. Castro, "Convex hull in feature space for support vector machines," in *Ibero-American Conference on Artificial Intelligence*. Springer, 2002, pp. 411–419.

[26] L. Guo and S. Boukir, "Fast data selection for svm training using ensemble margin," *Pattern Recognition Letters*, vol. 51, pp. 112–119, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865514002499

[27] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," 1998.

[28] M. A. Chandra and S. Bedi, "Survey on svm and their application in image classification," *International Journal of Information Technology*, vol. 13, no. 5, pp. 1–11, 2021.

[29] B. Kumar and O. Dikshit, "Spectral–spatial classification of hyperspectral imagery based on moment invariants," *IEEE Journal of selected topics in applied earth observations and remote sensing*, vol. 8, no. 6, pp. 2457–2463, 2015.

[30] S. Goel, M. Gaur, and E. Jain, "Nature inspired algorithms in remote sensing image classification," *Procedia Computer Science*, vol. 57, pp. 377–384, 2015.

[31] W. Yang, X. Yin, and G.-S. Xia, "Learning high-level features for satellite image classification with limited labeled samples," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 8, pp. 4472–4482, 2015.

[32] I. Garali, M. Adel, S. Bourennane, and E. Guedj, "Classification of positron emission tomography brain images using first and second derivative features," in *2016 6th European Workshop on Visual Information Processing (EUVIP)*. IEEE, 2016, pp. 1–5.

[33] S. Wang, Q. Liu, E. Zhu, F. Porikli, and J. Yin, "Hyperparameter selection of one-class support vector machine by self-adaptive data shifting," *Pattern Recognition*, vol. 74, pp. 198–211, 2018.

[34] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*. Springer, 1998, pp. 137–142.

[35] Z. Liu, X. Lv, K. Liu, and S. Shi, "Study on svm compared with the other text classification methods," in *2010 Second international workshop on education technology and computer science*, vol. 1. IEEE, 2010, pp. 219–222.

[36] T. Joachims *et al.*, "Transductive inference for text classification using support vector machines," in *Icml*, vol. 99, 1999, pp. 200–209.

[37] M. Wang, H. Zhang, and R. Ding, "Research of text categorization based on svm," in *Proceedings of the 2011, International Conference on Informatics, Cybernetics, and Computer Engineering (ICCE2011) November 19–20, 2011, Melbourne, Australia*. Springer, 2011, pp. 69–77.

[38] M. Goudjil, M. Koudil, M. Bedda, and N. Ghoggali, "A novel active learning method using svm for text classification," *International Journal of Automation and Computing*, vol. 15, no. 3, pp. 290–298, 2018.

[39] R. Burbidge, M. Trotter, B. Buxton, and S. Holden, "Drug design by machine learning: support vector machines for pharmaceutical data analysis," *Computers & chemistry*, vol. 26, no. 1, pp. 5–14, 2001.

[40] S.-X. Du and S.-T. Chen, "Weighted support vector machine for classification," in *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 4. IEEE, 2005, pp. 3866–3871.

[41] Z. Yin and J. Zhang, "Operator functional state classification using least-square support vector machine based recursive feature elimination technique," *Computer methods and programs in biomedicine*, vol. 113, no. 1, pp. 101–115, 2014.

[42] J. Nayak, B. Naik, and H. S. Behera, "A comprehensive survey on support vector machine in data mining tasks: Applications & challenges," *International journal of database theory and application*, vol. 8, pp. 169–186, 2015.

[43] Y. Tang, "Deep learning using support vector machines," *CoRR, abs/1306.0239*, vol. 2, p. 1, 2013.

[44] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[45] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep learning approach combining sparse autoencoder with svm for network intrusion detection," *Ieee Access*, vol. 6, pp. 52 843–52 856, 2018.