A Review of Kernel Methods in Machine Learning

Urvashi Ramdasani
Department of Computer Science and Engineering
Institute of Technology, Nirma University
Ahmedabad, India
18bce247@nirmauni.ac.in

***Abstract*—Machine Learning (ML) is an efficient technology for pattern recognition and analysis. Pattern recognition involves recognizing patterns from strings, images, etc. Pattern analysis and recognition have found great applications in the areas of Natural Language Processing, Optical Character Recognition, Computer Vision, Autonomous Path Navigation and Speech Recognition. Kernel methods are a category of ML algorithms extensively used for pattern analysis. These methods use one or more kernel functions for analysis. These methods are used in specific applications. A variety of kernel functions and methods available which have found great use in the areas of medicine, information processing, image processing, etc. In this paper, a review of kernel functions and methods along with their applications has been presented.**

***Index Terms*—Algorithm, Kernel, Kernel Functions, Kernel Methods, Machine Learning**

I. Introduction

Machine Learning (ML) is an application of Artificial Intelligence (AI) which gives computers the ability to predict values or classify things on their own without being explicitly programmed. We develop algorithms that run on input data and produce a model. After running an algorithm on input data, the state of the output is saved as a model. This model improves its performance with experience. A model represents what is learned by an algorithm. It is further used for prediction and is deployed on an application. ML algorithms are further classified as supervised, unsupervised, semi-supervised, and reinforcement learning algorithms. In supervised learning problems, we have input and output variables and we need to find a mapping between input and output variables. Supervised learning algorithms are of two types: Regression and Classification.

In the case of regression, our model predicts a continuous value for a given set of inputs, based on the relationship between the input and output variables. In classification, given an object or a data point, our model or classifier predicts which class the object or the data point belongs to. The classifier finds the probability of a data point belonging to a particular class. The class with the highest probability is predicted as the class of the data point. Based on the number of classes to separate, classification can be binary (with two classes) and multi-class (with more than two classes).

The regression includes linear, multiple, non-linear, and logistic regression. Various classification algorithms are known such as Decision Trees, Naive Bayes, Random Forests, etc. Support Vector Machine is an algorithm that can be used for both classification and regression problems. In this algorithm, we plot the data points in an n-dimensional space. We then find a decision boundary (hyperplane) that separates the classes. A hyperplane is an (n-1) dimensional subspace for n-dimensional space. This boundary can be linear and non-linear. A linear boundary can efficiently separate a linearly separable dataset. However, most of the datasets are not simple, they are not linearly separable. We need to find ways to separate non-linear datasets.

One way to solve this problem is to transform the dataset into another plane such that it becomes linearly separable. To transform the dataset, we need a mapping function that maps the dataset to higher dimensions. By plotting the transformed dataset and changing its orientation, we can then linearly separate the classes. However, in the case of large datasets, it becomes infeasible to map all the data points to a higher dimension. It may take a large amount of time and memory to complete execution. Hence this solution, for real-time large datasets, is not feasible to apply.

Instead of transforming data to higher dimensions, we can use kernels to modify the data without transforming them into another plane. Each data point is represented as a vector in an n-dimensional space. A kernel function considers as if the vectors are transformed into another plane and then performs modification. It does not change the dimension, it computes in such a way that there is no need to store the transformed vectors. In this way, computations are faster and cheaper. This technique is known as the kernel trick.

Various authors have extensively studied kernel methods. Authors in [1] have proposed a method based on the cross-breed genetic algorithm for choosing kernel function and its parameters for Support Vector Machines. In this paper, different kernel methods are described in depth along with their applications.

A. Organization

The rest of the paper is organized as follows. Section II gives a brief introduction to the kernel theory. Section III describes various Kernel Functions available. Section IV discusses various Kernel Methods. Section V shows the applications of Kernel Methods proposed by different authors. Finally, the paper is concluded in Section VI.

II. Kernel Theory

A kernel k has an associated feature mapping ϕ. Let X be an input space and F be a feature space. F should be a vector space on which the dot product is defined. It is also known as Hilbert Space. Let an input $x \in X$ is present which is to be mapped to feature space F. Then the formula for k is shown below. It gives the similarity between the vectors x and y in F.

$$k(x,y) = \phi(x)^{\rm T}\phi(y) \tag{1}$$

For a function to become a kernel function, it must satisfy the Mercer's condition. First, there must exist a feature space F on which the dot product is defined. Second, the function should be a positive definite function, i.e., Equation 2 must hold. In Equation 2, f is any square-integrable function.

$$\int dx \int dz f(x)k(x,z)f(z) > 0 \tag{2}$$

A. Kernel Algebra

Let k_1 and k_2 be two kernel functions. Then the following conditions hold true.

1) The sum of k_1 and k_2 is also a kernel function.

$$k(x,y) = k_1(x,y) + k_2(x,y) \tag{3}$$

2) A scalar α multiplied with k_1 (scalar product) is also a kernel. function

$$k(x,y) = \alpha k_1(x,y) \tag{4}$$

3) The direct product of k_1 and k_2 is also a kernel function.

$$k(x,y) = k_1(x,y)k_2(x,y) \tag{5}$$

III. Types of Kernel Functions

The simplest kernel is a linear kernel. It is used when the dataset is linearly separable. It is also used when there are a large number of features present in the dataset. One such example is text classification [2]. Given two vectors x_1 and x_2, then the linear kernel is defined as the dot product of x_1 and x_2. Other types of most commonly used kernels are the Polynomial kernel, Gaussian kernel, Exponential kernel, Laplacian kernel, Hyperbolic or Sigmoid kernel, Anova radial basis kernel, etc. The equations for these kernels are shown in I. This list is not an exhaustive list. There are various kernel functions proposed by different authors [3].

In the polynomial kernel, d is the degree of the polynomial that we want to map to. A radial basis function kernel or RBF kernel is one of the popular kernel functions. It involves an exponent term, and hence it will give a very strong non-linear hyperplane. It can classify complex datasets well. The variations of RBF are Gaussian, Exponential, Laplacian, and Anova radial basis kernels. Some properties of these kernels are discussed here. In the Gaussian kernel, the hyper-parameter gamma plays a very important role in the performance of the Gaussian kernel. Laplacian kernel is less prone to modifications. Sigmoid kernel, as the name suggests, is widely used in neural networks. Anova radial basis, Laplacian, and Gaussian kernels perform well in multidimensional regression problems.

TABLE I
Kernel Equations of Common Kernel Functions

Method	**Equation**
Linear	$K(x_1,x_2) = {x_1}^{\rm T}.x_2 + c$
Polynomial	$K(x_1,x_2) = (\alpha {x_1}^{\rm T}.x_2 + c)^d$
Gaussian	$K(x_1,x_2) = exp(-\gamma\|\|x_{\rm i} - x_{\rm j}\|\|^2)$
Exponential	$K(x_1,x_2) = exp\left(-\frac{\|\|x_1 - x_2\|\|}{2\sigma^2}\right)$
Laplacian	$K(x_1,x_2) = exp\left(-\frac{\|\|x_1 - x_2\|\|}{\sigma}\right)$
Hyperbolic or Sigmoid	$K(x_1,x_2) = tanh(\alpha {x_1}^{\rm T}x_2 + c)$
Anova radial basis	$K(x_1,x_2) = \sum_{k=1}^{n} exp(-\sigma({x_1}^k - {x_2}^k)^2)^d$

IV. Types of Kernel Based Methods

This section briefly describes the different Kernel Methods available in Machine Learning. These methods use one or more of the above-discussed kernel functions, and hence the name kernel methods. Kernel Methods are algorithms widely used for pattern analysis. Pattern analysis is done to find relations in the dataset. These relations can be a correlation, classification, ranking, clustering, principal components, etc. The Kernel Methods available in Machine Learning are Principal Component Analysis (PCA), Spectral Clustering, Support Vector Machines (SVM), Canonical Correlation Analysis, Kernel Perceptron, Gaussian Process, and Ridge Regression. Classical machine learning algorithms use a single kernel, but multiple kernels can be used without the loss of information [4].

A. Principal Component Analysis (PCA)

PCA is a statistical approach for feature extraction. It combines some input variables and eliminates some other variables such that the important parts (principal components) of all variables are retained. The resultant variables are independent.

Hence it is used in Exploratory Data Analysis (EDA). This is usually required in the case of linear regression. Since some variables are eliminated, it is used in dimensionality reduction. However, PCA is a linear method, it can only be applied to linear-separable datasets.

Kernel Principal Component Analysis (KPCA) is a variation of PCA. It uses kernel function in the component analysis. It can also be applied to non-linear complex datasets. Similar to SVM, it also transforms the dataset to a higher dimension so that it can be linearly separated. When the dataset becomes linearly separable, PCA can be applied for reducing the variables. KPCA is used to remove noise from images. The authors in [5] have shown a comparative study on classical and kernel PCA. They have compared running time and variance of classical PCA and KPCA by using different datasets such as iris, moon, circle, and swiss roll datasets. In KPCA, different kernels such as RBF, sigmoid, polynomial, and Laplace are compared with each other.

B. Spectral Clustering

Clustering is an unsupervised learning algorithm in which data points with a similar pattern are grouped. Depending on the algorithm chosen, there may be different clusters of data points. Spectral Clustering is a method in which data points are treated as the nodes of a graph. So, the clustering now becomes a graph partitioning problem. This algorithm uses the Gaussian kernel function to compute a fully connected graph from data points. The data points are projected to a lower dimension space to account for the fact that the data points from the same cluster may be far away. These points become close after reducing dimension. After this, similar data points are clustered together from the rest of the data points. Spectral Clustering has applications in the areas of statistics, ML, EDA, and Computer Vision. It has been used in Salient Object Detection [6] and Data Summarization [7].

The authors in [8] have focused on scalability and robustness of Spectral Clustering for large datasets. Different variants of Spectral Clustering have been proposed that have promising applications. One such variant is presented in [9] using fuzzy C-Means Clustering. Other variants are proposed by authors in [10], [11], and [12].

C. Support Vector Machines (SVM)

As discussed earlier, SVMs involve finding a hyperplane which linearly separates two or more classes. A normal SVM can only separate a linearly-separable dataset. A kernel SVM uses kernel trick to find a hyperplane separating the classes. The decision boundary for a linear case is

$$\sum_{i=1}^{n} y_{\mathrm{i}} \alpha_{\mathrm{i}} x_{\mathrm{i}} \cdot x + b \tag{6}$$

For a non-linear case, we can take the dual of the original optimization problem and find the decision rule for classification. Thus, the decision rule now becomes

$$\sum_{i=1}^{n} y_{\mathrm{i}} \alpha_{\mathrm{i}} k(x_{\mathrm{i}}, x) + b \tag{7}$$

Depending on the dataset chosen, different kernel functions have different accuracies. For example, the authors in [13] have tested RBF, linear, and polynomial of degree 3 kernels for crop classification using SAR images. The RBF kernel achieved the highest accuracy followed by the polynomial kernel of degree 3. The linear kernel in this case had the least accuracy of 79.12 percent. SVMs are widely used for classification problems. Hence they have many applications in Biology. In [14], SVMs are used for structural classification of protein sequences. Other applications in Biology include identification of glucose-binding pockets in Human Serum Albumin [15], Hand Arthritis stage detection [16], classification of Parkinson's disease and essential tremor [17] and much more. It is also used in English and Tamil character recognition [18], stenography detection, and speech recognition [19].

D. Canonical Correlation Analysis

CCA is also known as Canonical Variates Analysis. It is a study of the linear relations between two variables. Given two vectors x and y, CCA seeks to find vectors a and b such that linear variates $a^T x$ and $b^T y$ have a maximum correlation. This problem can be mathematically stated as

$$(a', b') = argmaxcorr(a^{\mathrm{T}} x, b^{\mathrm{T}} y) \tag{8}$$

It has been used in the classification of LANDSAT images, detecting risk factors leading to breast cancer, score analysis, etc. However, in some cases, linear variates may not be adequate for finding associations. Kernel Canonical Correlation Analysis (KCCA) is a generalized version of CCA where kernel functions (such as Gaussian) are used again to find non-linear associations between variables. While applying KCCA, one unsolved challenge is the choice of kernel and its parameters. Although KCCA is found to be statistically consistent, it is still a wide area of research.

E. Kernel Perceptron

Kernel perceptron is a variant of the perceptron learning algorithm. In perceptron learning, a single layer of perceptrons can perform binary linear classification. Multiple layers can be used for multiclass classification. It is also known as error-driven learning. In this algorithm, weights corresponding to a feature is updated whenever an incorrect prediction has been made by the model. The hyper-parameters for this algorithm are the learning rate, number of epochs, and activation function.

Kernel perceptron uses kernel functions to learn non-linear models. It allows the perceptron model to capture non-linear behavior. We need to choose a kernel and apply the kernel to the perceptron. The updated algorithm for kernelized perceptron is shown in Algorithm 1. In the algorithm, a is a mistake counter vector. It is incremented every time a wrong prediction is made. $K(x_i, x_j)$ is a kernel function chosen for training. It is also a hyper-parameter. b is the bias vector. In this algorithm, the weight vector is eliminated. Kernel perceptron has found applications in face recognition. In [20], RBF network is used for partial face recognition.

Algorithm 1 Kernalized Perceptron Algorithm

```
Input: $D$, $epochs$
Output: $a$, $b$
$a \leftarrow 0$, $b \leftarrow 0$
for $i = 1 ... epochs$ do
    for $(x, y) \in D$ do
        $\hat{y} \leftarrow \sum_{j=1}^{m} a_j y_i K(x_i, x_j)$
        if $\hat{y} \neq y_j$ then
            $a_i \leftarrow a_i + y_i$
            $b \leftarrow b + y$
        end if
    end for
end for
```

F. Gaussian Process

Gaussian process is an algorithm for regression and probabilistic classification. A stochastic process is a collection of random variables, either indexed by time or space. GP is a stochastic process such that every linear combination of these random variables follows the normal or Gaussian distribution. A Gaussian distribution in terms of mean and covariance matrix is defined as

$$p(x; \mu, \sum) = \frac{1}{(2\pi)^{n/2} |\sum|^{1/2}} exp(-\frac{1}{2}(x - \mu)^T \sum{}^{-1}(x - \mu)) \tag{9}$$

In 9, $\sum$ is the covariance matrix and μ is the mean of the distribution. Mean is usually kept 0 since it centers the distribution around $y - axis$. For non-zero mean, the distribution only changes its position with respect to $y - axis$. If X is a random variable that is normally distributed, then the covariance matrix defines the shape of the distribution. We generate the covariance matrix by evaluating the kernel function or the covariance function. The covariance functions largely affect the predictive power of GPs. One of the applications of GPs is Bayesian Global Optimization. It involves GPs to predict the validation error at any hyper-parameter setting for an ML algorithm. It is then used to tune these hyper-parameters.

G. Ridge Regression

A parsimonious model is the model that has the right number of predictors to define a model well. Other models may contain more or fewer predictor variables. Sometimes, the number of predictors may exceed the number of observations. In that case, there may be collinearity among some of these variables. To avoid all these problems, ridge regression is used. In ridge regression, the variables are penalized if they become very large. Hence all the variables are present but they have smaller coefficients. It uses the L2 regularization technique to shrink the parameters. This leads to reduced model complexity and prevents multicollinearity. Regularization is used to achieve bias-variance trade-off and prevent overfitting. The optimization problem for ridge regression is

$$\hat{f(x)} = min_{\mathrm{f}} \sum_{i=1}^{n} (y_{\mathrm{i}} - f(x_{\mathrm{i}}))^2 + \lambda ||f||^2 \tag{10}$$

Kernel Ridge Regression (KRR) combines the dual of ridge regression and kernel trick. The equation for KRR then becomes

$$\hat{f(x)} = \sum_{i=1}^{n} \alpha_{\mathrm{i}} k(x, x_{\mathrm{i}}) \tag{11}$$

KRR is not much different from normal ridge regression. KRR simply has higher computational efficiency. It has applications in forecasting.

V. Applications of Kernel Methods

Kernel methods have become very popular due to their ability to reduce the number of computations to transform the dataset to a higher dimension. They are also memory efficient since no extra space is required to store the transformed vectors. According to [21], the applications of kernel methods include bioinformatics, handwriting recognition [22], 3D construction [23], and information extraction [24]. Authors in [25] have proposed an approach for analysis of neuroimages using Bayesian kernel methods. In [26], the authors have demonstrated network anomaly detection using multiple kernels. Kernel methods are also used in object classification. The authors in [27] have used kernel methods for object classification in Synthetic Aperture Radar (SAR) images. In [28], the authors have used the direct kernels method with SVM in an Intrusion Detection System (IDS).

VI. Conclusion

In this paper, some kernel methods in machine learning are discussed. Linear separation becomes difficult when the complexity of datasets increases. To tackle this problem, kernel methods are used. They are computationally faster and cheaper. Kernel methods use kernel functions to find the hyperplane separating the classes. Kernel methods have various applications in the field of medical science, text classification, network anomaly detection, etc.

VII. Acknowledgement

I express my sincere gratitude Prof. Dhaval Pujara, Head of Department - Department of Electronics and Communication Engineering, Institute of Technology, Nirma University for sharing insights on research methodology and helped us in attaining our goals in a systematic method.

References

[1] H. J. Liu, Y. N. Wang, and X. F. Lu, "A method to choose kernel function and its parameters for support vector machines," in *Fourth International Conference on Machine Learning and Cybernetics*, 18-21 August 2005.

[2] L. A. Trindade, H. Wang, W. Blackburn, and N. Rooney, "Text classification using word sequence kernel methods," in *2011 International Conference on Machine Learning and Cybernetics*, 10-13 July 2011.

[3] T. Hofmann, B. Sch, and A. Smola, "A review of kernel methods in machine learning," 01 2006.

[4] R. Zhang and X. Duang, "A new compositional kernel method for multiple kernels," *2010 International Conference On Computer Design And Applications (ICCDA 2010)*, vol. 1, pp. 27–30.

[5] K. Ezukwoke and S. Zareian, "Kernel methods for principal component analysis (pca) a comparative study of classical and kernel pca," 12 2019.

[6] X. Hu, W. Yang, X. Wang, and Q. Liao, "Salient object detection via spectral clustering," in *2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference*, pp. 165–169, 2016.

[7] N. Sapkota, A. Alsadoon, P. W. C. Prasad, A. Elchouemi, and A. K. Singh, "Data summarization using clustering and classification: Spectral clustering combined with k-means using nfph," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pp. 146–151, 2019.

[8] D. Huang, C. Wang, J. Wu, J. Lai, and C. Kwoh, "Ultra-scalable spectral clustering and ensemble clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 6, pp. 1212–1226, 2020.

[9] A. Ben Ayed, M. Ben Halima, and A. M. Alimi, "Adaptive fuzzy exponent cluster ensemble system based feature selection and spectral clustering," in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–6, 2017.

[10] F. Nie, Z. Zeng, I. W. Tsang, D. Xu, and C. Zhang, "Spectral embedded clustering: A framework for in-sample and out-of-sample spectral clustering," *IEEE Transactions on Neural Networks*, vol. 22, no. 11, pp. 1796–1808, 2011.

[11] K. Güzel and O. Kurşun, "Improving spectral clustering using path-based connectivity," in *2015 23nd Signal Processing and Communications Applications Conference (SIU)*, pp. 2110–2113, 2015.

[12] W. Zhu, F. Nie, and X. Li, "Fast spectral clustering with efficient large graph construction," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2492–2496, 2017.

[13] B. Yekkehkhany, A. Safari, S. Homayouni, and M. Hasanlou, "A comparison study of different kernel functions for svm-based classification of multi-temporal polarimetry sar data," *The 1st ISPRS International Conference on Geospatial Information Research*, 15-17 November 2014.

[14] C. Chrysostomou and H. Seker, "Structural classification of protein sequences based on signal processing and support vector machines," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3088–3091, 2016.

[15] P. Ranganarayanan, N. Thanigesan, V. Ananth, V. K. Jayaraman, and V. Ramakrishnan, "Identification of glucose-binding pockets in human serum albumin using support vector machine and molecular dynamics simulations," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 13, no. 1, pp. 148–157, 2016.

[16] F. Akhbardeh, F. Vasefi, N. MacKinnon, M. Amini, A. Akhbardeh, and K. Tavakolian, "Classification and assessment of hand arthritis stage using support vector machine," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 4080–4083, 2019.

[17] D. Surangsrirat, C. Thanawattano, R. Pongthornseri, S. Dumnin, C. Anan, and R. Bhidayasiri, "Support vector machine classification of parkinson's disease and essential tremor subjects based on temporal fluctuation," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 6389–6392, 2016.

[18] R. Ramanathan, S. Ponmathavan, N. Valliappan, L. Thaneshwaran, A. S. Nair, and K. P. Soman, "Optical character recognition for english and tamil using support vector machines," in *2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies*, pp. 610–612, 2009.

[19] K. Aida-zade, A. Xocayev, and S. Rustamov, "Speech recognition using support vector machines," in *2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT)*, pp. 1–4, 2016.

[20] K. Sato, S. Shah, and J. K. Aggarwal, "Partial face recognition using radial basis function networks," in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 288–293, 1998.

[21] "Kernel method." https://en.wikipedia.org/wiki/Kernel_method. Online; Accessed: November 2020.

[22] D. Yang and L. Jin, "Kernel modified quadratic discriminant function for online handwritten chinese characters recognition," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 1, pp. 38–42, 2007.

[23] H. J. Parmar and S. Ramakrishnan, "Semi-automatic 3d construction of liver using single view ct images," in *2012 38th Annual Northeast Bioengineering Conference (NEBEC)*, pp. 157–158, 2012.

[24] Xiaofeng Zhang, Zhiqiang Gao, and Man Zhu, "Kernel methods and its application in relation extraction," in *2011 International Conference on Computer Science and Service System (CSSS)*, pp. 1362–1365, 2011.

[25] A. S. Lukic, M. N. Wernick, D. G. Tzikas, X. Chen, A. Likas, N. P. Galatsanos, Y. Yang, F. Zhao, and S. C. Strother, "Bayesian kernel methods for analysis of functional neuroimages," *IEEE Transactions on Medical Imaging*, vol. 26, no. 12, pp. 1613–1624, December 2007.

[26] G. Song, X. Jin, G. Chen, and Y. Nie, "Multiple kernel learning method for network anomaly detection," *Proceedings of 2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering, ISKE 2010*, 11 2010.

[27] P. D. Jordhana and K. Soundararajan, "Kernel methods and machine learning techniques for man-made object classification in sar images," *2014 International Conference on Information Communication and Embedded Systems*, 2014.

[28] A. G. Gedam and S. G. Shikalpure, "Direct kernel method for machine learning with support vector machine," *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies*, pp. 1772–1775, 2017.