3rd International Conference on Computing: Communication, Networks and Security (IC3NS-2018)
Volume: 4 Issue: 3
ISSN: 2454-4248
71 – 75

Application of Machine Learning Algorithms for Profile Reconstruction of IPM Device

Alok Jadhav* Omkar Thawakar* Charul Rathore**
*SGGSIET, Nanded, India
**Mody University of Science and Technology, Laxmangarh, India

Abstract: Measured IPM profiles can be significantly distorted due to the displacement of residual ions or electrons by interaction with beam fields for high brightness or high energy beams [1, 2, 3]. It is thus difficult to deduce the characteristics of the actual beam from the measurements. In this project different Machine Learning Regression Algorithms are applied to reconstruct the actual beam profile from the measurement data.

*****

1 Introduction

Ionization Profile Monitors (IPM) are used for nondestructive transverse beam profile measure- ments at many accelerator facilities. The prin- ciple of operation is the following; the primary beam ionizes the residual gas and the ionized particles (ions or electrons) are extracted via electric fields and sometimes in conjunction with magnetic fields to confine the movement of ion- ized particles in the plane transverse to the elec- tric field. The profile of the extracted parti- cles reflects the transverse profile of the primary beam with the assumption that ionized parti- cles are created at rest and the effect of induced fields by the primary beam on ionized particles can be neglected.

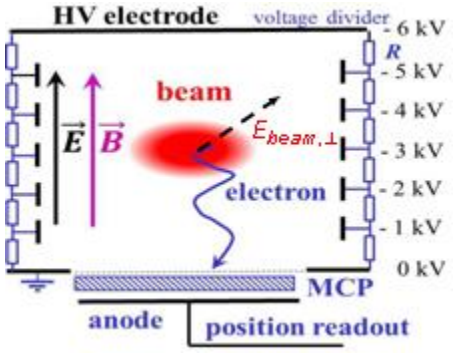

Fig. 1: *Operating principle of the IPM.*

Figure 1 shows the typical components present in an IPM where both the electric and magnetic fields are utilized to confine the ion- ized particles [6]. The support electrodes/rods between the top and bottom electrodes are used to reduce the fringe fields and improve field ho- mogeneity. The field homogeneity is important in order to avoid any distortion in the measured profile and therefore static EM simulations for the full geometry are usually performed. IPMs are often used for nondestructive measurements in low-pressure conditions such as storage rings and hence they usually have to be equipped with a high amplification multichannel plate (MCP) for obtaining sufficient signal to noise ratio. The outputs of MCPs are connected to data acquisi- tion system directly or via phosphor screens and an optical system.

The distortion in measured IPM profiles due to beam space charge is well known and typ- ically magnetic fields are used to confine the generated ionized particles around the point of generation. However, in some cases required magnetic field strengths are prohibitory for ex- tremely dense beams as discussed below for our target case. First major distortion for an IPM with magnetic fields was seen for LHC IPMs at high energies, where the beam profile was signif- icantly broader compared to wire scanner mea- surements [1]. The first solution envisaged to solve the issue was to raise the magnetic fields in the IPM to 1 T [2]. This field strength should be viewed in perspective of the strength of LHC main dipoles which is 8.5 T at top energies and was considered impractical. Recently reliable IPM simulation tools have been developed as a joint effort between several labs [5]. Availabil- ity of reliable description of IPM system includ- ing the profile distorting effects such as space charge and initial velocity distribution of ionized particles transforms the problem of IPM profile correction into a ”supervised learning” problem. In a supervised learning problem, the input and output of an unknown system are provided and the system is approximated by a variety of ma- chine learning algorithms [7]. Here we will apply several Machine Learning (ML) regression algo- rithms to reconstruct actual beam profile from the measured distorted profile. In the next sec- tion, the simulation tool, as well as the beam and device parameters used to train the ML re- gression models, are discussed. Following that, the ML algorithms, their respective parameters, training, and validation are presented and the results are summarized.

2 Machine Learning Regression

The ability to find the underlying process model and its features from data are referred to as ma- chine learning.

This can be used to predict the system behavior or make decisions. The field is evolved primarily from pattern recognition and Artificial Intelligence. Though Machine Learn- ing (ML) algorithms have a variety of applica- tions such regression, classification, dimension- ality reduction, clustering, density estimation, we focus on regression/prediction in our con- text. The choice of ML algorithm is problem specific, and typical deciding factors are bias- variance tradeoff, training time, and dimension- ality of the problem (i.e.the number of inputs and outputs), and any prior information about the features of the underlying model.

Each Machine Learning (ML) regression algo- rithms involve three important functions:

"Machine Learning = Representation + Eval- uation + Optimization"

A) Mathematical learning model (Represen- tation)/Decision function: It is an approxima- tion of the underlying function between inputs and outputs which is later used for predictions of the unknown input dataset.

B) Loss function/Error function (Evaluation): Function indicating the $L : y \quad y \quad R_+$ penalty for an incorrect prediction [19]. It is used for finding error percentage between predicted out- put and true output. Evaluation is essentially how you judge or prefer one model vs. another.

C) Optimization Function(Optimization): it is used to search optimal parameters for mathe- matical learner model. In most algorithms "gra- dient descent method" is used as optimization function. This is how you search the space of represented models to obtain better evaluations. Four typical ML regression methods with in- creasing level of complexity are applied to the problem as discussed below:

1) Linear Regression: It is a linear approach for modeling the relationship between a scalar dependent variable y and one or more explana- tory variables denoted X. The Mathematical learning model (Representation) of linear re- gression can be represented by,

$$y_{pred} = WX + b \qquad (1)$$

where W is an array of coefficients and X(X_1,X_2. X_n) is a $M \quad N$ matrix where M is the number of data sets, N is the number of features of each data set. b is bias, predicted output is y_{pred}.Predicted output y_{pred} is an ar- ray of length M. whereas the equation of loss function of linear regression is [12],

$$min \qquad (y_{real} - y_{pred})^2 \qquad (2)$$

where y_{real} is true output values for specific in- put value X.

2) Ridge Regression: To avoid mentioned shortcomings of linear regression, a regular- ization term is introduced in Ridge regres- sion model [12]. Ridge regression uses simi- lar Mathematical learning model (Representa- tion) as that of linear regression with a regular- ization/penalty term for the magnitude of es- timated weights. The error function of linear regression is:

$$min \rightarrow (y_{real} - y_{pred})^2 + \lambda(W)^2 \qquad (3)$$

where regularization term is $\lambda(W)^2$. Which means that weight of coefficient should be as minimum as possible. If an input attribute has a small effect on improving the error function it is shut down by the penalty term.

Kernel Trick: Kernels are used to transform feature space into higher Dimensional feature space and learn a linear classifier/model in that transformed space. We use kernel functions to do this transformation implicitly.

Kernel Ridge Regression(KRR): Kernel Ridge Regression (KRR) differs from Ridge Re- gression (RR) in its form of Mathematical learn- ing model (Representation). Input values are transformed into some higher dimensional fea- ture space $X \rightarrow \Phi(X)$.

$$y_{pred} = W\,\Phi(X) + b \qquad (4)$$

It is useful for problems when the features are different or larger than the number of test data points. The transformation is made computa- tionally feasible by choice of appropriate kernels and the "kernel trick" [13]. Most common ker- nels are a Radial basis function (RBF) kernel, polynomial Kernel (Poly) and linear Kernel as shown in Table 1 [12, 13]. X and X_i are two points from the dataset. In the dual space, the

Tab. 1: *some of the kernels are shown*

RBF	$(exp(-\gamma(X - X_i)^2))$
Poly	$(XX_i + b)^d$
Linear	(XX_i)

decision function is given by,

$$y_{pred} = -\frac{\alpha}{\lambda} K(X, X_i) \qquad (5)$$

$$K(X, X_i) = \Phi(X) \cdot \Phi(X_i)$$

where K is the considered kernel and α is an array of Lagrange multipliers. KRR uses a sim- ilar loss function as ridge regression and ker- nel trick which has its origins in

Support Vector Machines (SVM). That way, it can be seen as an intermediate between Ridge regression and SVM. Kernel ridge regression requires less time to train compared to SVM [14].

3) Support Vector Machines Regression (SVM Regression): In a nutshell, SVM regression dif- fers from Kernel Ridge Regression (KRR) in its loss function which allows it to select a subset of input data set (called support vectors) which are used in the decision function. Generally, it takes longer to train the SVM model compared to KRR but results in a compact decision func- tion.

The loss function of SVM Regression is,

$$min[1/2(W)^2 + [C \sum (\xi^* + \xi)]] \quad (6)$$

subject to:

$$[y_{real} - (W, \Phi(X)) - b] \leq E + \xi \quad (7)$$

$$[(W, \Phi(X)) + b - y_{real}] \leq E + \xi^* \quad (8)$$

$$\xi, \xi^* \geq 0 \quad (9)$$

Mathematical learning model (Representa- tion)of SVM regression in dual space is:

$$\sum (\alpha - \alpha^*) K(X, X_i) + b \quad (10)$$

where ξ and ξ^* are slack variables to cope with constraints of the optimization. The constant $C > 0$ controls the regularization in SVR while α and α^* are Lagrange multipliers [14].

3 IPM and Machine Learning Regression

Tab. 2: *Parameters used for the simulation. σ_x, σ_y, σ_l and the bunch population have been var- ied within the specified intervals.*

Particle type	Protons
Energy/u	6.5 TeV
Bunch population N_p	1.1 to 1.7 × 10^{11}
Bunch length σ_l (4σ)	0.9 ns to 1.2 ns
Bunch width σ_x (1σ)	0.29 mm to 0.37 mm
Bunch height σ_y (1σ)	0.4 mm to 0.6 mm
Electrode distance	85 mm
Applied voltage	4 kV
Magnetic field	0.2 T

3.1 Simulation Data Details

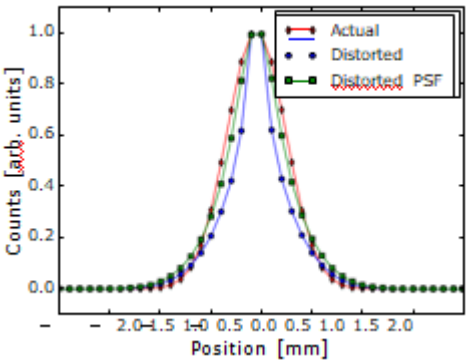

Fig. 2: *Simulation of profile distortion due to space charge using Virtual IPM.*

The Virtual-IPM simulation program has been used in order to generate training data. Table 2 shows the parameters which have been used for the simulations. σ_x, σ_y, σ_l and the bunch population N_p have been varied to cover the relevant operational region. One million particles were used in each simulation provid- ing rather smooth profiles. The bin size is 0.1 mm and Gaussian point spread function with σ = 0.125 mm was used to represent the effect of optical acquisition system in the IPM system. Figure 2 shows the actual primary beam profile, the distorted profile due to space charge and the profile read at the end of the acquisition system after application of point spread function of the optics. The input parameters to simulation were N_p = 1.7 × 10^{11}, σ_l = 0.9 ns, σ_x = 0.29 mm, and σ_y = 0.4 mm. There is a 20 % increase in the second central moment of the measured pro- file (with PSF) with respect to actual (initial) profile. Further details of virtual IPM can be found here [4].

Training

The training of ML regression algorithms is per- formed with three distinct parameter sources, measured profile (100 points in each profile), particle number N_p (1 point) and bunch length σ_l (1 point) as inputs forming an array of 102 points for each training sample while the out- put is actual width denoted by $\sigma_{x,a}$ to differ- entiate from predicted width $\sigma_{x,p}$. $\sigma_{y,a}$ is not used for the training process since, in any ex- perimental usage, it will not be available as an input to the trained network for prediction of $\sigma_{x,p}$. 375 training samples were generated with a parameter scan in N_p, $\sigma_{x,a}$ and $\sigma_{y,a}$ (5 pa- rameters each) and σ_l (3 parameters). Table 2 shows the parameter range over which training data

was generated. Validation data was gener- ated at a spacing 1%, 25% and 50% off the train- ing data sites in each parameter space forming 372 validation samples. In addition to that 0.5% Gaussian white noise (relative to the maximum value in each parameter space) was added to the profiles to depict ADC/camera noise on the measured profile as well as measurement un- certainty on N_p and σ_l. More details can be found in [16] The training data is used to opti- mize the loss function and obtain the parameters of regression algorithms. sklearn library with python interface was used to define and train the ML regression algorithms discussed above. For optimization of parameters, we used grid search method by using GridSearchCV library from sklearn [14]. GridSearchCV does an ex- haustive search over defined parameters with a scoring method of 'mean square error'.

Results and Discussion

For every ML Algorithm, the error% between re- constructed width and true width of the profile is calculated for the whole validation data and plotted as a histogram. All results are generated by adding Gaussian white noise=0.5% with re- spect to source input parameter.

Linear Regression: Figure 3 shows the his- togram of error% in prediction by linear regres- sion. where μ = 0.144 and σ = 0.860

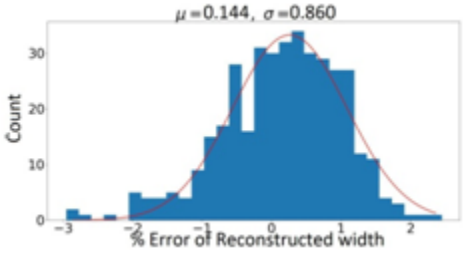

Fig. 3: *Histogram showing the percentage prediction error given by linear regression algorithm.*

Ridge Regression: In Ridge Regression after optimization using Gridsearch method [14] we obtained λ = 0.0189. Figure 4 shows the his- togram of error% between reconstructed width and true width of the profile.

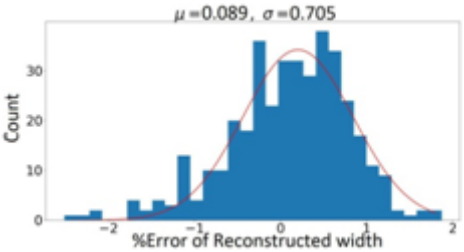

Fig. 4: *Histogram showing the percentage prediction error given by Ridge Regression Algorithm*

Kernel Ridge Regression:After training, dif- ferent results are obtained with different ker- nels . Optimum parameters of KRR using Gridsearch were, kernel="RBF", λ = 0.00106, γ = 0.1. Figure 5 shows histogram of error% between predicted width and true width of the profile.

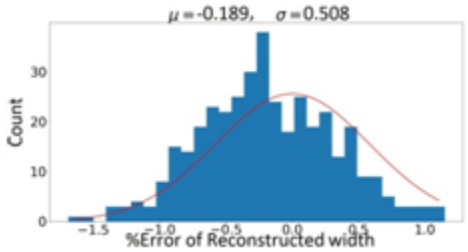

Fig. 5: *Histogram showing the percentage prediction error given by KRR Algorithm using RBF Kernel*

Support Vector Machines Regression: The optimized parameters for SVM Regression for using Gridsearch method were: kernel="RBF", E = 0.024, α = 0.1, C = 1000. Histogram in Fig. 6 shows error% between Predicted width and true width of profile. The value of deviation

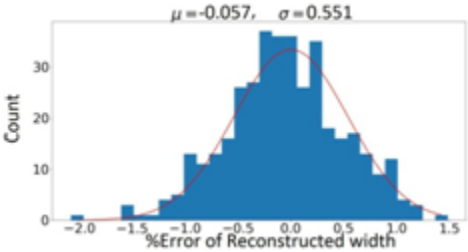

Fig. 6: *Histogram showing the percentage prediction error given by SVM using RBF Kernel*

is not significantly different for the algorithms discussed above while minor improvements with increasing complexity of the algorithm is observ- able.

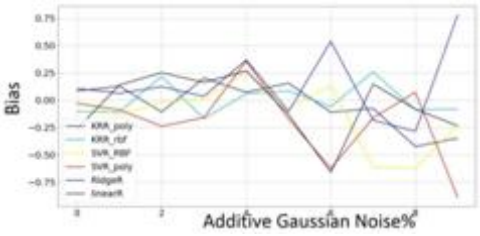

Fig. 7: *Evolution of bias of predictions with respect to noise in training and validation data*

Gaussian white noise was added to both train- ing and validation data in the range of σ_{noise} = 0 % to 10 % relative to the maximum value of each source parameter. For each set of noisy

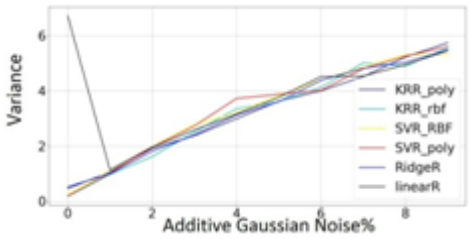

Fig. 8: *Evolution of std. deviation of predictions with respect to noise in training and validation data*

training data, training of all the ML algorithms was performed and the bias and variance of pre- diction error over the validation data were plot- ted against the added % noise. Figures 7 and 8 shows plot of bias and variance against % noise respectively. The bias of the prediction error of almost all algorithms oscillates around zero even with increased noise while the variance of prediction error is in a linear relationship with noise amplitude.

Conclusion

The reconstruction of distorted profiles for Gaussian beams was performed using machine learning regression algorithms and reconstruc- tion errors below 1 % were obtained even with the inclusion of measurement uncertainties. Re- sults show that compared to other machine learning regression algorithms, both SVM Re- gression and Kernel Ridge Regression with RBF kernel gives better prediction results. This might be due to the fact that squared exponen- tial kernel defines a function space that is a lot larger than that of the linear kernel or the poly- nomial kernel.

References

[1]. M. Sapinski et al., The first experience with LHC beam gas ionization monitor, Pro- ceedings of IBIC 2012.
[2]. M. Patecki et al., Electron tracking sim- ulations in the presence of the beam and external fields, Proceedings of IPAC 2013, Shanghai, China.
[3]. D. Vilsmeier et al., Investigation of the ef- fect of beam space charge on electron tra- jectories in Ionization profile monitors, HB 2014.
[4]. D. Vilsmeier et al., A modular application for IPM simulations, these proceedings.
[5]. M. Sapinski et al., Ionization profile mon- itor simulations - status and future plans, Proceedings of IBIC 2016.
[6]. P. Forck, Lecture notes in beam instrumen- tation, JUAS, 2017.
[7]. K. P. Murphy, Machine Learning: A Prob- abilistic Perspective, The MIT Press, 2012.
[8]. Sebastian Raschka,Python Machine Learn- ing, ISBN 978-1-78355-513-0, (2015).
[9]. Bishop, Christopher (2006). Pattern recog- nition and machine learning. Berlin: Springer. ISBN 0-387-31073-8.
[10]. Ryan Tibshirani, Modern regression 1: Ridge regression Data Mining: 36-462/36- 662 (2013). retrieved from: http://www.stat.cmu.edu/ ryantibs/ datamining/lectures/16-modr1.pdf
[11]. Stefan Feuerriegel, Regularization Methods Business Analytics Practice Winter Term (2015/16). retrieved from: https://www.is.uni-freiburg.de/resour ces/business-analytics/XX Regulariza tion.pdf
[12]. P.Paisitkriangkrai, Linear Regression and Support Vector Regression (2012). re- trieved from: http://cs.adelaide.edu.au/ chhshen/ teaching/ML SVR.pdf
[13]. Justin Domke, Statistical Machine Learn- ing Kernel Methods and SVMs, UMass (2011). retrieved from: https://people.cs.umass.edu/ domke/ courses/sml2011/07kernels.pdf
[14]. Scikit-Learn Documentation 0.19.0(2017). retrieved from: http://scikit-learn.org/stable/ modules/svm.html.
[15]. David A. Freedman, Statistical Models: Cambridge University Press.p.26. ISBN: 9780521112437, (2009).
[16]. R. Singh et al., Proceedings of IBIC (2017).
[17]. Wikipedia:Support Vector Machines Defi- nition, retrieved from : https://en.wikipedia.org/wiki/Suppor t vector machine JCGM 200:2008 International Vocabulary of Metrology Basic And General Concepts And Associated Terms (VIM)(2008). re- trieved from: http://www.bipm.org/utils/common/ documents/jcgm/JCGM 200 2008.pdf
[18]. Mehryar Mohri,Introduction to Machine Learning, Courant Institute and Google Research,retrieved from: http://www.cs.nyu.edu/ mohri/mlu/ mlu lecture 1.pdf