Comparative Analysis of Support Vector Machine and Multilayer Perceptron in High Energy Particle Source Classification

Stephanie Jury and Linh Pham

Motivation and Problem Description

Over the past decade, advances in astroparticle physics have led to the formulation of profound questions about the extremes of the Universe. One area of research addressing such questions is the use of ground-telescopes in detection of particle showers, cascades released by gamma rays generated by cosmic objects and phenomena. The principal challenge is in discriminating between the weak signals generated by primary gamma sources and the dominating secondary photons from hadronic showers in the atmosphere. These are additionally modulated by noise detected by the hardware [1]. Fortunately, however, images generated by the gamma and hadron signals display discriminating characteristics, making it possible to classify the signals using neural computing.

Given the proven success of classifying high-energy particles in collision experiments in the literature [2] we have chosen to compare the effectiveness of support vector machines (SVM) and multilayer perceptron (MLP) as binary classifiers, applied to the problem of accurate gamma signal detection by the 'Major Atmospheric Gamma Imaging Cherenkov' (MAGIC) telescopes. These are a system of two 17 m diameter, F/1.03 'Imaging Atmospheric Cherenkov Telescopes' (IACT) situated at the Roque de los Muchachos Observatory on La Palma, dedicated to the observation of gamma rays from galactic and extragalactic sources in the very high energy range (30 GeV to 100 TeV).

Dataset Description and Preliminary Analysis (10%)

The MAGIC dataset was generated by a Monte Carlo program, CORSIKA (1998), running with parameters to simulate extensive particle air showers [3]. The dataset contains 19,020 observations, each described by 10 continuous variables, being the characteristic parameters of the ellipsoidal image generated by the modelled shower, Hillas parameters [4]. Also included is the positional distribution of energy depositions across each image, the extent of the cluster in each image plane, and the total sum of depositions in each image. Each candidate has been assigned a binary class by human annotators, labelled either gamma (1, being positive) or hadron (0, being negative). Preliminary analysis of the dataset showed there were no missing values or non-numeric datatypes. All feature variables are continuous. Table 1 shows the summary statistics grouped by class.

Variable	Gamma					Noise				
	Mean	Std. Dev.	Skew	Min.	Max.	Mean	Std. Dev.	Skew	Min.	Max.
1. fLength: major axis of ellipse [mm]	-0.23	0.62	1.24	-0.97	5.17	0.42	1.37	1.28	-1.16	6.63
2. fWidth: minor axis of ellipse [mm]	-0.20	0.49	2.48	-1.21	8.40	0.36	1.48	2.12	-1.21	12.77
3. fSize: 10-log of sum of content of all pixels [#phot]	-0.09	0.98	0.82	-1.75	4.62	0.16	1.02	0.98	-1.87	5.29
4. fConc: ratio of sum two highest pixels over fSize [ratio]	0.02	0.97	0.57	-2.00	2.78	-0.03	1.05	0.37	-2.01	2.80
5. fConc1: ratio of highest pixel over fSize [ratio]	0.00	0.96	0.74	-1.88	4.17	-0.01	1.07	0.61	-1.94	3.88
6. fAsym: distance from highest pixel to center (major axis) [mm]	0.13	0.67	-0.92	-5.83	3.79	-0.24	1.39	-0.53	-7.66	9.79
7. fM3Long: 3rd root of third moment along major axis [mm]	0.14	0.67	0.14	-4.11	4.03	-0.26	1.39	-0.79	-6.71	4.47
8. fM3Trans: 3rd root of third moment along minor axis [mm]	0.00	0.65	0.08	-4.40	4.86	0.01	1.44	0.09	-9.90	8.62
9. fAlpha: angle of major axis with vector to origin [deg]	-0.34	0.82	1.55	-1.06	2.39	0.63	1.00	0.07	-1.06	2.39
10. fDist: distance from origin to center of ellipse [mm]	-0.05	0.94	0.19	-2.52	3.43	0.09	1.10	0.21	-2.58	4.04

Table 1 – Descriptive statistics of feature variables, segmented by class.

All variables were normalised using the z-score before classifiers were applied. These illustrate the class-associated disparities in some of the variable characteristics, for example the mean value of the 'fAsym' variable, providing encouragement that these variables are suitable for classification. Also apparent is the bias towards occurrences of the positive class. The dataset contains 12,332 positive examples and 6,688 negative examples, resulting in a majority positive class bias of 64.8%. Such bias can affect the predictive power of classifiers, as a portion of classification accuracy can be achieved by simply predicting the majority class. The heatmap in Figure 2 demonstrates that fLength, fWidth and fSize, and fConc and fConc1 have correlation coefficients above 0.75, indicating dimensionality reduction may be required.

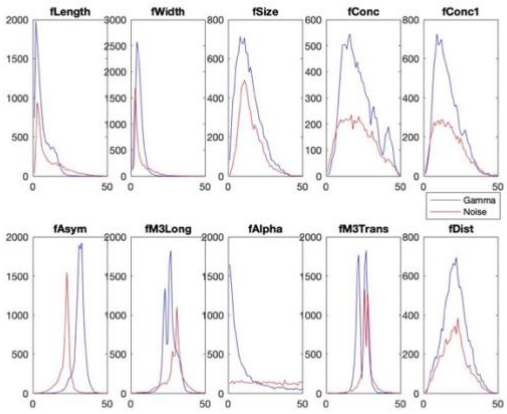


Figure 1 – Normalised feature distributions, segmented by class. (Histogram traces with 50 bins).

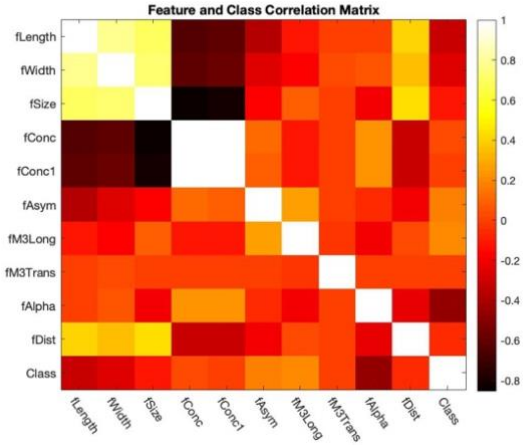


Figure 2 – Correlation plot of features and classes

Summary of Models

Support Vector Machines

SVM is a family of non-probabilistic, linear classifiers. Applied to a binary classification problem, the algorithm searches for an optimal hyperplane which maximises the geometric margin of separation between positive and negative observations. In the case that observations are not linearly separable in the finite-dimensional space of the original problem, the space may be mapped onto a much higher-dimensional space in which separation is achievable. In this case the "kernel trick" can be applied, which avoids the explicit mapping of observations into the higher-dimensional space, instead defining them in terms of a kernel function that computes only the inner (dot) product of pairs of observations in that space.

Advantages

SVM models exhibit good generalization performance without prior domain knowledge. They are effective in high-dimensional spaces and cases where the number of dimensions is greater than the number of samples. Because different kernel functions can be specified for the decision function, they are versatile without the need for assumptions about the form of the kernel. Due to the convex nature of the optimisation problem, they are guaranteed to find the global minimum error.

Disadvantages

SVM models are only directly applicable to binary classification problems and require multi-class tasks to be framed as a sequence of binary tasks. They are deterministic, i.e. they do not provide probability estimates for their predictions. They can be memory intensive in both training and testing when the feature space is large. They are slow to train as the Sequential Minimal Optimization solving algorithm cannot account for prior domain knowledge.

Multilayer Perceptron

MLP is a class of feedforward neural network models, structures that resemble the biological networks in the brain, which map a set of input data onto a set of appropriate outputs [5]. A network's architecture comprises multiple layers of parallel neurons, including an input layer, one or more hidden layers, and an output layer, with a system of synaptic weights and biases creating a fully-connected, directed graph between them. Multiple layers and non-linear activation enable MLP to distinguish data that are not linearly separable in the problem space. One of the most popular methods of training an MLP model is backpropagation, a supervised learning technique that adjusts the weight of neurons in the direction that minimises the gradient of a loss function (gradient descent).

Advantages

Neural networks are universal approximators which can approximate virtually any function. Being adaptive learners, they do not require prior knowledge of variable or class probability distributions, instead learn how to perform a task based on the data provided in training. They exhibit “fault tolerance” due to their distributed, brain-like structure, continue to work in the case of neuron failure or interconnection damage, and can relearn relatively quickly after damage.

Disadvantages

Large neural networks, with large numbers of layers and neurons may not generalise well and be difficult to optimise. In these cases, tuning hyperparameters can become computationally expensive. The optimization problem is not convex thus optimisation may halt at local minima.

Hypothesis

In the literature, the relative performance of SVM and MLP models has been shown to differ according to the task to which they are applied, with MLP more versatile and outperforming SVM in many cases [6] [7]. Studies applying both models to similar problems of astrophysical phenomena and high energy particle detection from image data however, conclude outperformance of tuned SVM [8] [9]. We hypothesize that the best SVM model will achieve greater classification accuracy than the best MLP model, but likely at the expense of computational cost and training time.

Choice of Training and Evaluation Methodology

Before training our models, we randomly reordered observations in the MAGIC dataset to remove any embedded ordering generated by the Monte Carlo simulation. We split the data into a training set (70%) and test set (30%) before carrying out majority class under-sampling, as described above; the test set was left imbalanced. Working with the training set, we investigated multiple model parameters using a grid search for both MLP and SVM models and tested their effect on mean classification accuracy using 3-fold cross validation. The SVM models were trained using the MATLAB 'fitcsvm' function and the MLP models trained using the 'train' function. Due to the computational complexity of the chosen algorithms in this study, higher-fold cross-validation was too expensive to run on our machines. The parameters applied to the held-out test data were those that resulted in the highest mean classification accuracy in validation.

When evaluating the test set performance, as well as looking at mean classification accuracy, we also generated confusion matrices. This enabled us to derive error metrics including the true positive rate (TPR), true negative rate (TNR), positive predictive value (PPV), negative predictive value (NPV) and F1 score. These metrics enable investigation into the components of model accuracy.

Choice of Parameters and Experimental Results

Support Vector Machine

We investigated the classification performance of five different kernel functions, linear, Gaussian, and first, second and third order polynomial. For each kernel function we varied two hyperparameters; box constraint and kernel scale. In our initial investigation, we tested the numeric parameters in the range [1, 10], with step size of one and discovered that, in all cases, the best model performance was achieved at values of one. The grid search was then repeated in the range [0.1, 1], with a step size of 0.05 and a mean classification accuracy heatmap generated, shown in Figure 3.

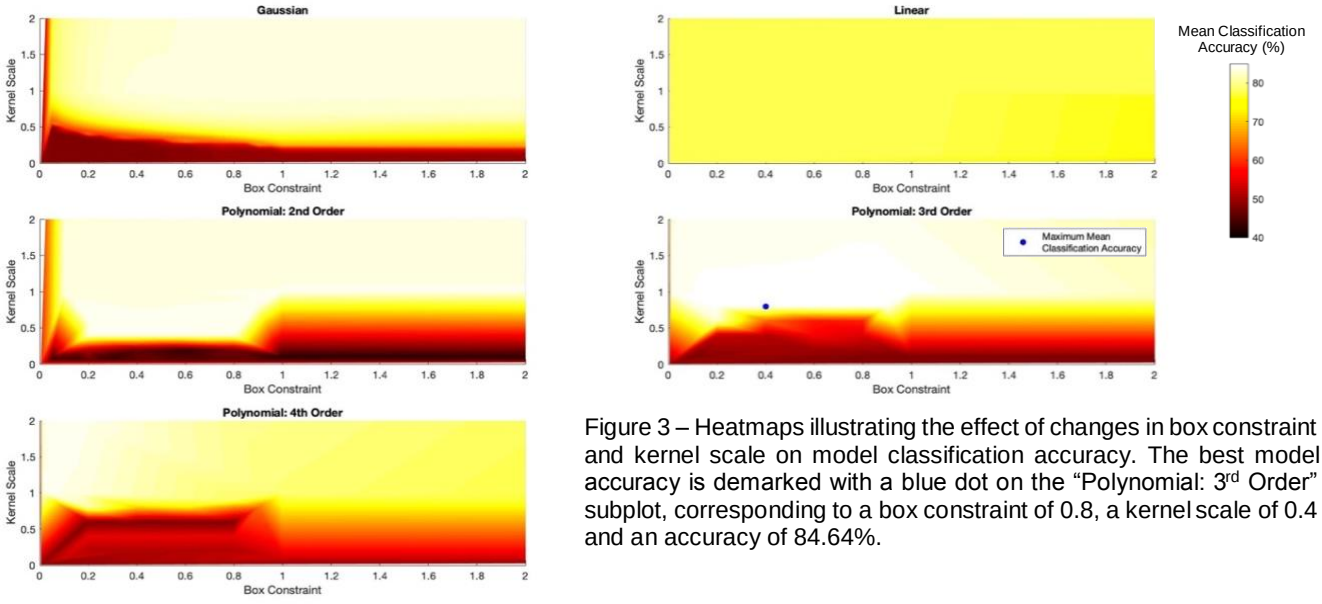
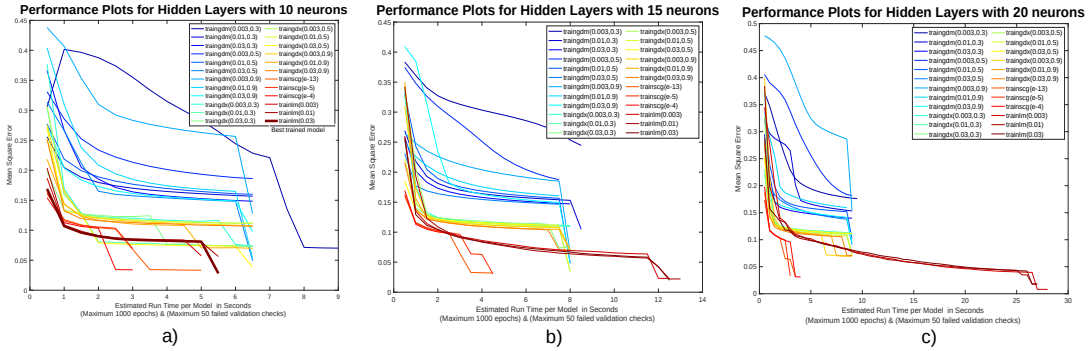


Figure 3 – Heatmaps illustrating the effect of changes in box constraint and kernel scale on model classification accuracy. The best model accuracy is demarked with a blue dot on the "Polynomial: 3rd Order" subplot, corresponding to a box constraint of 0.8, a kernel scale of 0.4 and an accuracy of 84.64%.

Multilayer Perceptron

The network has two hidden layers. A “softmax” output function was chosen to give a probabilistic decision for the binary outputs (gamma and noise). A mean squared error performance function was applied, allowing different training functions to be compared. Random weights and biases were re-initialised every time training occurred in order to avoid the network becoming caught in the same local minima. The initialisation process was the Nguyen-Widrow function, which distributed the active region of each neuron evenly across the input space [10]. Each hidden layer was activated with a hyperbolic tangent sigmoid, which is symmetrical, allowing negative weights to be as meaningful as positive weights [11]. The maximum number of epochs was set at 1,000 as an early stopping condition and the maximum number of failed validation checks to be 50 to improve the model’s generalisability. We investigated four different training functions: Gradient Descent with Momentum (GDM) using ‘traingdm’, Gradient Descent with Momentum and Adaptive Learning Rates (GDX) using ‘traingdx’, Scaled Conjugate Gradient (SCG) using ‘trainscg’ and Levenberg-Marquardt (LM) using ‘trainlm’. For all training functions we tried three different values (10, 15 and 20) for the number of neurons in the hidden layers. For the GDM and GDX, additional variations were applied for two hyperparameters, three values (0.003, 0.01 and 0.03) for the learning rate and three (0.3, 0.5 and 0.9) for momentum. The SCG was tuned according to a variation (10^{-13}, 10^{-5} and 10^{-4}) in the sigma parameter, required to be no larger than 10^{-4}, which adjusts the weight of the conjugate system in the second order information calculation [12]. For the LM, three values satisfying the requirement to be no larger than one, 0.003, 0.01 and 0.3, were tried for the LM regularising parameter, whichß determines the initial size of the regularising factor [5]. For other parameters, the values have been chosen based on the observation that higher values led to low accuracies and much slower training. Figure 4 shows the estimated performance history for each combination of hyperparameters for different numbers of neurons.



a) b) c)

Figure 4. Mean squared error vs. estimated run time for each model, for a different number of neurons in the hidden layer. Each model is represented by a coloured line, identified by the following formats: ‘function_name(learning rate,momentum rate)’ for GDM and GDX, ‘function_name(regularisation_factor)’ for LM and ‘function_name(sigma)’ for SCG.

The best accuracy achieved during validation was 85.53%, using the LM function for 10 neurons and the LM regulariser at 0.03 (Figure 4a). LM and SCG models converged to the lowest minima. When the number of neurons increased, all models required more time to reach the minimum, especially for the LM models (Figure 4c).

Analysis and Critical Evaluation of Results

Model Selection

In the SVM validation phase, the most striking observation was that the linear kernel function returned model accuracies in a very narrow range of 74% to 77%, as shown by the ubiquitous yellow colour of the subplot in Figure 3. As expected, tuning the numeric hyperparameters had little effect on the linear model’s classification performance, as it is limited to generating the best straight-line fit between the classes directly in the problem space. In the Gaussian and polynomial models however, where the observations are mapped to a higher dimensional space, hyperparameter selection had a considerable impact on classification accuracy.

In these models, low box constraint values of below approximately 0.05, resulted in classification performance on par with a random guess. This makes intuitive sense, as small box constraint values define a kernel function with large variance, producing models that consider two points to be similar, even if they lie very far from one another. It follows that setting very large box constraint values results in models which consider proximate points to be dissimilar, which leads to model overfitting as

experienced in our SVM model training using box constraint values of 1 and above. The kernel scale factor enlarges the kernel at the positions of the support vectors, and hence the separation between the two classes [13]. Therefore, with very low scaling, the class-dividing boundary is narrow, and discrimination is difficult. In SVM model training, increasing the scaling factor resulted in better classification accuracy, but with no marginal improvement above approximately 1.5. The best SVM model accuracy in validation, 84.64%, was achieved using a third order polynomial kernel function, with a box constraint of 0.8 and kernel scale of 0.4.

Of all the training functions we applied for the MLP, GDM consistently resulted in the lowest accuracies. With the addition of changing learning rates in GDX, the results improved significantly. This is because changing learning rates allow the learning to be faster when the error function goes downhill and slower when it does not [11]. The two second-order methods - SCG and LM - both performed better than GDX, but SCG converged much faster with more or less similar performance to LM (Figure 4). The SCG conducts the search for the minimum by using the conjugate gradient instead of the gradient descent, making use of the second-order information, which can help increase the speed of convergence if the search is near the minimum. In addition, by using a regularising factor to regulate the definiteness of the Hessian matrix and avoiding time-consuming line-search, the SCG is more likely to converge faster [12]. The LM requires storage of order W^2 where W is the number of weights for the approximation of the Hessian matrix, hence when the data set or the layers get bigger it takes longer to train [5]. However, the LM has the flexibility of behaving as either the Newton method or the steepest descent depending on the LM regularising parameter, hence it can often perform well when the network is not too large. The best MLP model accuracy in validation, 85.53%, was achieved with the LM function for 10 neurons in each hidden layer and an initial regularising parameter of 0.03.

Algorithm Comparison

In testing, the best SVM and MLP models identified in validation achieved classification accuracies of 87.1% and 86.6% respectively. This supports our hypothesis that the SVM model would outperform the MLP model but, surprisingly, for both models the accuracy achieved in testing exceeded the accuracy achieved in validation. This implies that the data randomly selected for the testing was arbitrarily easier to classify. The decomposition of the accuracy scores are shown in Figure 5.

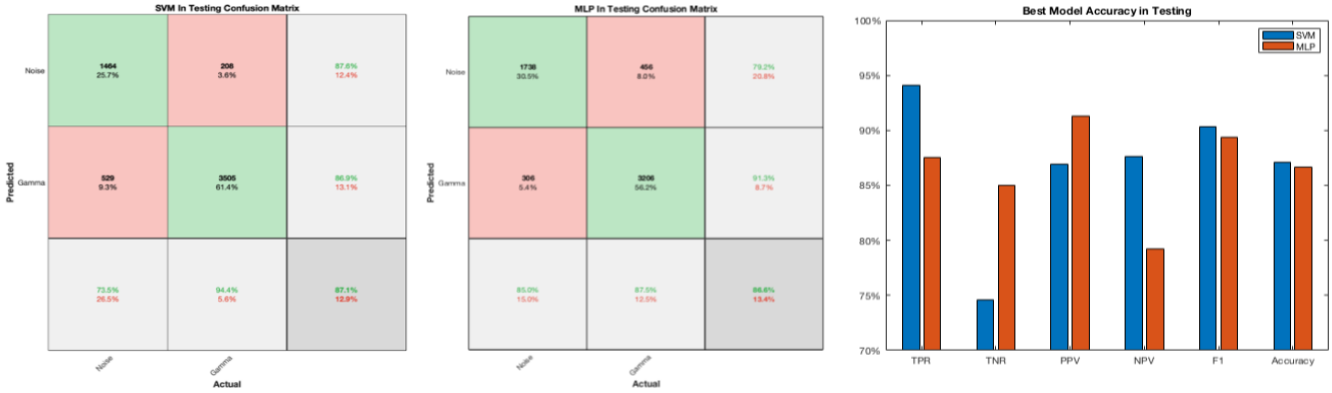


Figure 5 – Confusion matrices and error metrics bar chart of best SVM and MLP model performance in testing.

The confusion matrices illustrate that the SVM model was accurately classifying gamma observations in 94.4% of cases (TPR), but noise observations in only 73.5% of cases (TNR), whereas the MLP was accurately classifying both gamma and noise observations comparably (87.5% and 85.0% respectively). The bias exhibited by the SVM model is likely due to the features of different instances in the majority gamma class being more similar to one another than those of instances in the noise classes. These would more likely be contiguous in the mapped feature space and less likely to fall on the wrong side of the dividing plane than the more disparate noise observations. Taken together however, both models exhibited similar F1 scores, with SVM slightly outperforming MLP, achieving 0.905 compared to 0.861. F1 keeps a balance between PPV and TPR and, as such, remains a good indicator of accuracy in datasets with class bias, meanwhile the ROC curve simply compares TPR and FPR, ignoring the uneven distribution. Hence, we have chosen to use F1 as the measure to compare the two classes of models. In addition, it is preferable to use a model with a high TPR due to the high scientific cost of missing a true discovery, even if this does result in more time wasted investigating false positive instances. Based on these observations, the SVM model was the better model for the gamma detection problem, in line with our initial hypothesis, based on previous studies for similar astrophysical events [8] [9].

Lessons Learned and Further Study

During the process of tuning hyperparameters for both SVM and MLP models we observed that, although SVM had fewer parameters to tune to the features of a given problem, it can still perform on par with and even better than the more complex MLP models. Another lesson was the relationship between model complexity and computational expense. For MLP models, there was a five-fold difference in training time between the LM model with 10 neurons versus the LM model with 20 neurons, and for SVM models, there was a 100-fold difference in training time between the best performing linear and Gaussian kernel functions versus the best performing polynomial functions. In further work we suggest a number of variations to improve the results in this paper. Without the limitation on computing power, we would try higher order for the cross validation, such as 10-fold as opposed to 3-fold, as this would help reduce the effect of model overfitting, although in both our best models performed better in testing than in validation. We would also test the effect of dimensionality reduction on our input vectors, as our data analysis suggests that some input variables have correlations higher than 0.75 (Figure 2). This may help improve classification accuracy and the speed of model convergence [11]. Furthermore, although we already carried out majority-class under-sampling to account for the imbalance in the data, another method that could be used is SMOTE sampling, an oversampling balancing method which creates synthetic data for the minority class.

References

[1] R.K. Bock, W.Wittek, “Multidimensional event classification in images from gamma-ray air showers,” [Online]. Available: http://www.ippp.dur.ac.uk/old/statistics/proceedings/bock.pdf. [Accessed 25 3 2019].

[2] P. Vannerem, K.-R. Muller, B. Scholkopf, A. Smola, S. Soldner-Rembold, “Classifying LEP Data with Support Vector Algorithms,” in *AIHENP99*, Crete, April 1999.

[3] D. Heck, J. Knapp, J.N. Capdevielle, G. Schatz, T. Thouw , “CORSIKA: A Monte Carlo code to simulate extensive air showers,” in *FZKA 6019*, 1998.

[4] M. d. Naurois, “Analysis methods for Atmospheric Cerenkov Telescopes,” in *Towards a Network of Atmospheric Cherenkov Detectors VII*, Palaiseau, 2005.

[5] S. Haykin, Neural Networks and Learning Machines, Pearson, 2009.

[6] E. Frias-Martinez, A. Sanchez, and J. Velez, “Support vector machines versus multi-layer perceptrons for efficient off-line signature recognition,” *Engineering Applications of Artificial Intelligence,* vol. 19, no. 6, p. 693–704, 2006.

[7] S.R. Amendolia, G. Cossu, M.L. Ganadu, B. Golosio, G.L. Masala, G.M. Mura, “A comparative study of K-Nearest Neighbour, Support Vector Machine and Multi-Layer Perceptron for Thalassemia screening,” *Chemometrics and Intelligent Laboratory Systems,* vol. 69, no. 1-2, pp. 13-20, 2003.

[8] M. Qu, F.Y. Shih, J. Jing , “Automatic solar flare detection using MLP, RBF, and SVM,” *Solar Physics,* vol. 217, no. 1, pp. 157-172, October, 2003.

[9] N. Barabino, M. Pallavicini, A. Petrolini, M. Pontil, A. Verri, “Support Vector Machines vs Multi-Layer Perceptron in Particle Identi cation,” in *ESANN*, Bruges (Belgium), 21-23 April 1999.

[10] M.H. Beale, M.T. Hagan, H.B. Demuth, Deep Learning Toolbox: User's Guide, MathWorks, 2018.

[11] Y.A. Lecun, L.Boltou, G.B. Orr, K-R. Muller, “Efficient BackProp,” in *Neural Networks: Tricks of the Trade*, Springer, September, 2012, pp. 53-131.

[12] M. F. Moller, “A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning,” *Neural Networks,* vol. 6, pp. 525-533, 1993.

[13] P. Williams, S. Li, J. Feng, and S. Wu, “Scaling the Kernel Function to Improve Performance of the Support Vector Machine,” [Online]. Available: https://www.dcs.warwick.ac.uk/~feng/papers/Scaling%20the%20Kernel%20Function.pdf. [Accessed 25 3 2019].