Instituto Tecnológico
y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

Especialidad en Sistemas Embebidos

A novel SVM voltage supervisor

TRABAJO RECEPCIONAL que para obtener el **DIPLOMA** de
ESPECIALISTA EN SISTEMAS EMBEBIDOS

Presenta: **FRANCISCO JOSÉ RODRÍGUEZ GARCÍA**

Asesor **LUIS RIZO DOMÍNGUEZ**

Tlaquepaque, Jalisco. 24 de agosto de 2021.

A novel SVM voltage supervisor

Francisco J. Rodriguez-Garcia and Luis Rizo-Dominguez
Department of Electronics, Systems and Informatics
ITESO (Instituto Tecnológico y de Estudios Superiores de Occidente)
Tlaquepaque, Jalisco, 45604 MX

***Abstract*—Voltage supervisors ensure that an electronic system is turned off whenever the rail voltage drops below the threshold value. Common implementations rely on hard decisions to assert the system's reset; however, these schemes lack flexibility in configuration. In this paper, we introduce a soft-decision system using a machine learning algorithm called support vector machine (SVM). The proposed monitoring system's software is built on top of *scikit-learn* SVM libraries and experimentation was conducted in the Raspberry Pi 4 platform. Confusion matrix for the SVM model shows that the system will perform well on new and training data. Overall, the resulting system is configurable and, unlike other implementations, it can be trained in online and offline modes.**

***Index Terms*—support vector machine, voltage supervisor, voltage monitoring.**

I. INTRODUCTION

A monitoring voltage device, commonly known as voltage supervisor, is a key element for any embedded system. These devices monitor voltage rails and can warn the processor when power is failing. This mechanism prevents processor brownouts and system failure. Basic supervisors monitor processor's V_{DD} and reset the processor when V_{DD} drops below a voltage threshold—usually 5-20% below the factory-programmed reset threshold (V_{IT-}) [1]—. Fig. 1 shows a very basic voltage supervisor with no hysteresis.

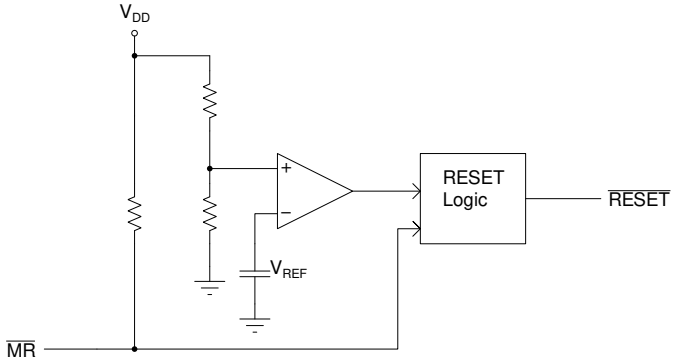

Fig. 1. Basic voltage supervisor functional block diagram.

The circuit operation is straightforward: the device uses a comparator and a reference voltage to turn off the system whenever the voltage rail is out of a fixed voltage limit. However, this circuit topology is not immune to voltage fluctuations and glitches and can eventually trigger false resets. To avoid this issue caused by noisy power supplies and discharging batteries, hysteresis can be added to the voltage supervisor.

In addition to these hardware techniques, embedded system techniques such as the on-off control system with hysteresis are widely employed. Similar to voltage supervisors, an on-off controller turns on the system ($RESET = 0$) when the voltage is lower than the desired value (Fig. 2).

Note how both hardware and software methods make a (hard) decision—assert the reset—based on an immutable threshold voltage or, hysteresis-wise, a limited set of values. Such hard decision schemes might be less reliable compared to soft decision implementations where inputs may take on a whole range of values in between.

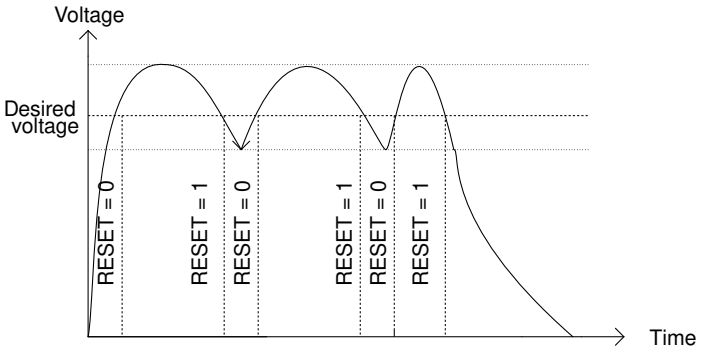

Fig. 2. On-off control system behavior

In this paper, the voltage monitoring problem is tackled using a machine learning (ML) algorithm called support-vector machine (SVM) that analyzes data for classification. SVMs are built on top of statistical learning frameworks proposed by Vladimir Vapnik, described in [2]. The algorithm implementation allows for both online and offline model training to take place. Decisions—whether assert or not the reset—are derived from the trained model. Python scikit-learn ML library was used for the SVM implementation. The Raspberry Pi 4 was selected as the target platform for its cost-computer resources tradeoff.

II. FUNDAMENTALS OF SUPPORT VECTOR MACHINE ALGORITHM

The SVM is a supervised learning algorithm. According to Hsu et al, "The goal of SVM is to produce a model (based on the training data) which predicts the target values of the test data given only the test data attributes" [3, p. 1]. While most learning algorithms learn differences, SVM learns similarities

Identify applicable funding agency here. If none, delete this.

and tries to find the optimal separating line, called *hyperplane*, between two classes. Fig. 3 represents a two-dimensional dataset consisting of circles and crosses.

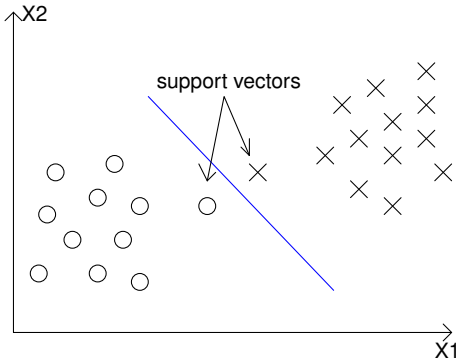

Fig. 3. Two-dimensional data linearly separable by a hyperplane

The hyperplane—blue line—can be used to predict target values. Points close to the hyperplane are called *support vectors* and help define the position of the hyperplane [4]. Notice how data in the above figure can be classified using a single line; however, there are cases where a straight line cannot be used to separate classes. A case in point is voltage classification itself (Figure 4a). For such non-linear cases, it is possible to map the dataset to a higher dimension and find the optimal hyperplane using a mathematical function called kernel function. Fig. 4b represents the one-dimensional voltage dataset after mathematical manipulation using a kernel function.

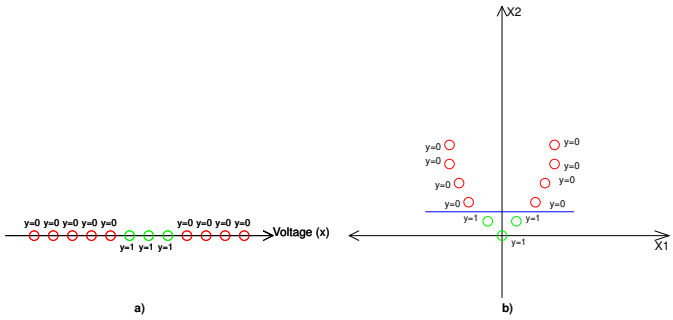

Fig. 4. a) Voltage depicted as a one-dimensional dataset distributed in multiple interleaved regions. b) Voltage dataset after kernel function transformation.

This mathematical transformation is often referred to as *kernel trick*. At the most basic level, kernel functions K map data into a new space and then take the inner product between the new vectors. Common types of kernel functions include:

- Linear Kernel

$$K(x, x') = x \cdot x' \tag{1}$$

where x and x' are vectors in the input space.

- Degree-d polynomial kernel

$$K(x, x') = (x \cdot x' + 1)^d \tag{2}$$

where x and x' are vectors in the input space.

- Gaussian Radial Basis Function (RBF)

$$K(x, x') = exp(\frac{-||x - x'||^2}{2\sigma^2}) \tag{3}$$

where x and x' are vectors in the input space and σ is a free parameter.

Because voltage classification in Figure 4a is nonlinear in nature, the kernel function must be carefully selected. In general, the RBF kernel is the best choice for this scenario. This kernel function maps the original dataset into a higher dimensional space where it is possible to find a linear decision boundary. More in-depth details about the RBF kernel can be found in [5].

III. SCIKIT-LEARN LIBRARY

scikit-learn is a free Python module that integrates a wide range of state-of-the-art ML algorithms [6]. This library provides built-in classes and functions for classification and regression. The following code snippet shows an example of support vector classification (SVC) usage:

```
# This program will output: [1 2 1]

import SVC
import numpy as np
import pickle from sklearn
import svm from sklearn.svm

trained_model = 'svm_model.sav'
X = np.array([[-1,0], [-2,0], [1,0], [2,0]])
y = np.array([1, 1, 2, 2])

# fit the method
clf = svm.SVC(kernel='linear')
clf.fit(X, y)

# save the model
pickle.dump(clf, open(trained_model,'wb'))

# load model and make prediction
loaded_svm_model = pickle.load(open(
    trained_model,'rb'))
print(loaded_svm_model.predict([[-0.8,
    0],[1,0],[-3,0]]))
```

In fact, scikit-learn supports different kernel functions for the decision functions. More specifically, it is possible to train an SVM using the RBF built-in kernel function. Consider the following code snippet:

```
>>> from sklearn import svm
>>> from sklearn.svm import SVC
>>> rbf_svc = svm.SVC(kernel='rbf', gamma=10,
    C=1)
>>> rbf_svc.kernel
'rbf'
```

where kernel, $gamma$ and C are called the *hyper-parameters* of the SVM. Intuitively, $gamma$ is a measure of the decision boundary curvature and C represents the penalization for misclassified data. Although most parameters inside an ML model are tuned directly from data, hyper-parameters

must be selected by the user based on experimentation and intuition [7]. Careful selection of $gamma$ and C parameters for voltage classification will be discussed in more detail in the following section.

IV. CHARACTERIZATION OF HYPER-PARAMETERS

Selecting wrong hyper-parameters might lead to data misclassification. Therefore, tunning C and $gamma$ is crucial for model performance. For comparison and selection purposes, the SVM with RBF model was fit using different sets of $gamma$ and C. Training data (Fig. 5) consisted of voltage values—feature x_1—in the range from $-0.17V$ to $+2.1V$, and separated in two classes, $y = -1$ and $y = 1$, in the intervals $x_1 \notin (1.5V, 1.85V)$ and $x_1 \in (1.5V, 1.85V)$, respectively.

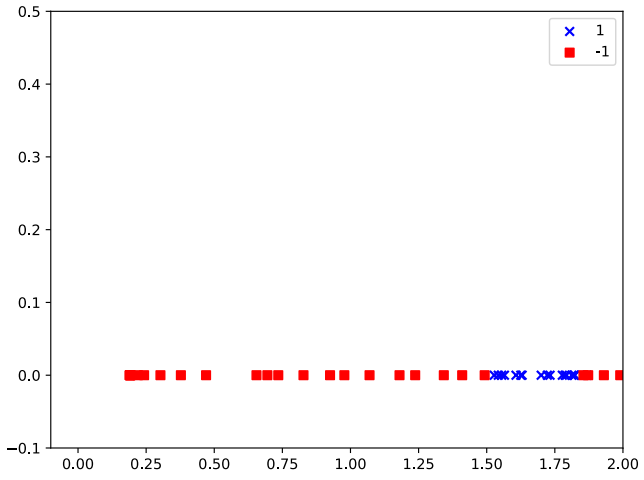

Fig. 5. Voltage dataset for hyper-parameter tunning.

The four charts below (6) show the comparison results. When using low $gamma$ and C, the model fails to classify data at all. SVM with RBF using $gamma = 1.0$ and $C = 1000$ has a better performance but still fails to classify some data points. In general, the higher $gamma$ and C, the better results obtained, with little to no misclassified data points.

scikit-learn provides a built-in function for parameter tunning called GridSearchCV. This function considers all parameter combinations exhaustively and outputs the best C and $gamma$ pair:

```
# This function will output: hyper-parameters
    {'C': 10, 'gamma': 100}
def select_svc_hyper_parameters(X, y):
    cs = [0.001, 0.01, 0.1, 1, 10, 100, 1000,
        10000]
    gammas = [0.001, 0.01, 0.1, 1, 10, 100,
        1000]
    param_grid = {'C': cs, 'gamma' : gammas}
    grid_search = GridSearchCV(svm.SVC(kernel=
        'rbf'), param_grid)
        grid_search.fit(X, y) grid_search.
        best_params_ return
        grid_search.best_params_
```

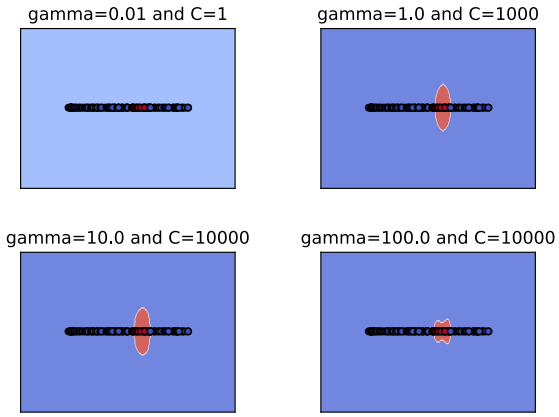

Fig. 6. Classification regions for SVM with RBF using different sets of $gamma$ and C.

V. MONITORING SYSTEM

The monitoring system proposed in this paper consists of two main components: user input interface and classification app. The user input interface is shown in Fig. 7.

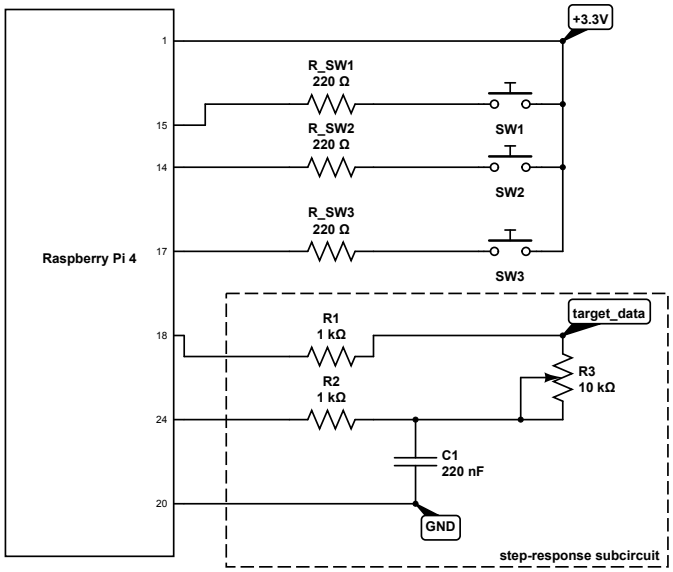

Fig. 7. User input interface.

Considering that the Raspberry Pi 4 cannot read analog signals directly without additional ADC chips, target data is read from $R4$ in the above figure. Potentiometer $R4$ is part of a subcircuit called *step-response*, where the sensed signal is a measure of the time required to charge and discharge capacitor $C1$ via resistors $R1$ and $R2$. Push buttons $SW1$ and $SW2$ are used to decide whether the target data belongs to class $y = -1$ or $y = 1$. Similarly, $SW3$ is used to toggle application states displayed in Fig. 8.

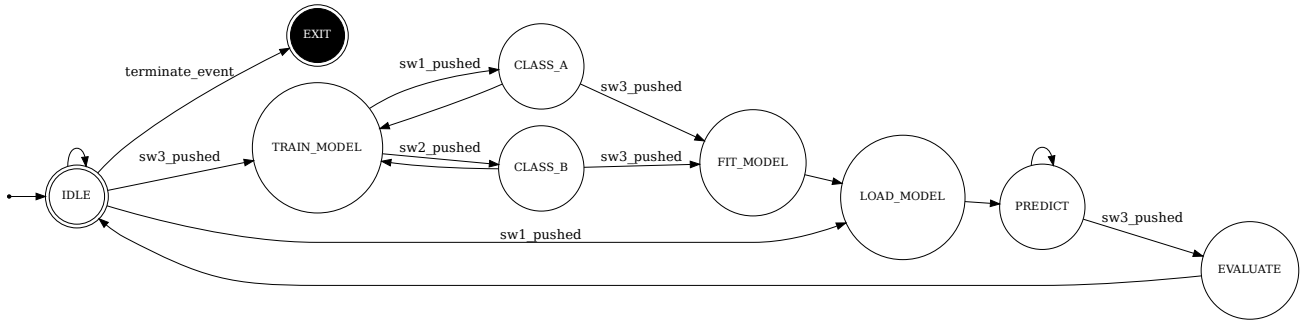

Fig. 8. State machine for classification app

VI. EXPERIMENTAL TESTS AND RESULTS

Training an SVM involves its evaluation. To evaluate the classifier's output quality, scikit-learn provides several tools, including the confusion matrix module. In simple terms, a confusion matrix is a 4-quadrant plot that summarizes the classifier's performance. Every quadrant score ranges from 0 to 1.0, where 1.0 means that the model correctly predicted the positive class–True Positive (TP)–. Fig. 9 shows the confusion matrix for the trained voltage SVM model:

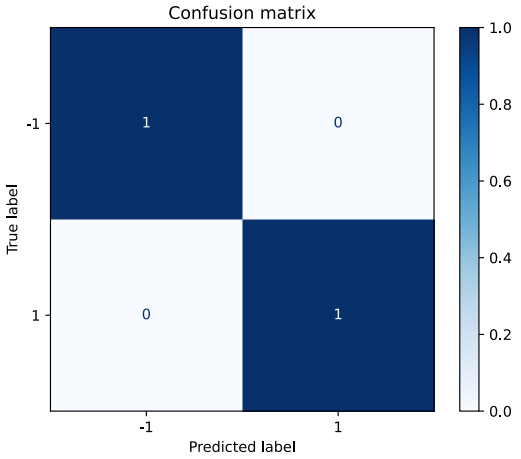

Fig. 9. Confusion matrix for classification system.

It is important to note that the potentiometer $R4$ introduces noise to the samples, hence affecting the training stage. To lessen the noise effects and avoid unwanted reading fluctuations, the following software regularization technique was implemented:

```
diff = abs(analog_in(t) - analog_in(t-1))
if ( diff < 0.2 ):
    print("sample_is_valid")
else:
    print("sample_is_invalid")
```

where $analog\_in(t)$ is the current analog sample, and $analog\_input(t-1)$ is the previous analog reading.

Test set for confusion matrix—created on the fly (during the *PREDICT* state in Fig. 8)—, consisted of data that the model had never seen before. Though this result indicates a 100% accuracy, it should be emphasized that the input data for training and subsequent model evaluation consisted of a small dataset. More exhaustive training—a few thousands of samples—is suggested for future work.

VII. CONCLUSION

SVM can be effectively used to replace hard decision-based voltage supervisors. The classification system proposed in this paper allows both online and offline model training. Decisions—whether assert or not the reset—can be derived from the trained model. Results indicate that the system will perform well on both new and training data.

ACKNOWLEDGMENT

This work has been partially supported by the Mexico federal government through the CONACyT agency.

REFERENCES

[1] C. Tran and A. Sharma, "Voltage Supervisor and Reset ICs: Tips, Tricks and Basics", Texas Instruments, 2019. Accessed on: June 15, 2021. [Online]. Available: https://www.ti.com/lit/eb/slyy167/slyy167.pdf

[2] V. N. Vapnik, Statistical Learning Theory. New York: Wiley, 1998.

[3] A. Chih-Wei Hsu, Chih-Chung Chang, "A Practical Guide to Support Vector Classification", BJU International, pp. 1396–1400, 2008.

[4] M. H. A. Banna et al., "Application of Artificial Intelligence in Predicting Earthquakes: State-of-the-Art and Future Challenges," in IEEE Access, vol. 8, pp. 192880-192923, 2020, doi: 10.1109/ACCESS.2020.3029859.

[5] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines. Cambridge, MA: Cambridge Univ. Press, 2000.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Perrot and É. Duchesnay, "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research, 2011, pp. 2825–2830.

[7] S. Hamida, O. E. Gannour, B. Cherradi, H. Ouajji and A. Raihani, "Optimization of Machine Learning Algorithms Hyper-Parameters for Improving the Prediction of Patients Infected with COVID-19," 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS), 2020, pp. 1-6, doi: 10.1109/ICECOCS50124.2020.9314373.