Edge-Based Online Signature Identification: A TinyML Approach with ESP32 Microcontroller

1st Hatem Zehir
L.E.R.I.C.A Laboratory
Badji Mokhtar Annaba University
Annaba, Algeria
zehir.hatem@univ-annaba.org

2nd Toufik Hafs
L.E.R.I.C.A Laboratory
Badji Mokhtar Annaba University
Annaba, Algeria
hafstoufik@gmail.com

3rd Sara Daas
L.E.R.I.C.A Laboratory
Badji Mokhtar Annaba University
Annaba, Algeria
sara.daas@yahoo.com

***Abstract*—This paper presents an edge-based online signature identification system using Tiny Machine Learning (TinyML) and an ESP32 microcontroller. The proposed system utilizes the SVC2004-task1 dataset and applies Gaussian filtering, position normalization, size normalization, and length normalization as preprocessing techniques. A deep learning model with three fully connected layers is trained using Edge Impulse framework, and then deployed on the ESP32 board after quantizing the model into 8-bit. The system achieves a high identification accuracy of 94.94%, demonstrating the feasibility and effectiveness of using TinyML and edge computing for online signature identification, which has potential applications in security and identification systems.**

***Index Terms*—online signature, biometrics, behavioural biometrics, ESP32, embedded systems, TinyML**

I. Introduction

Signature verification is one of the most used biometric modalities. It has been used for many centuries for authenticating documents and securing financial transactions. Signature recognition can be divided into two categories: online [1]–[3], which are acquired using an electronic device such as a tablet or a smartphone, and they are generally stored in an electronic format; and offline [4]–[6], which are signatures created by pen, and stored as physical copies, such as on a legal document.

And since the world is heading towards digitization today, the interest in systems based on online signatures is increasing, and it has become an integral part of many real-world applications such as national identity cards and bank transactions. However, identifying an individual from his signature could be a complicated process for several reasons. Firstly, the signatures vary greatly even between the same individual, as it is impossible to replicate the same signature twice exactly. Secondly, signatures can be easily forged if the counterfeiter is given enough time to train on how to mimic them. Lastly, the proposed system must be able to handle various types of input data, as the acquisition device may differ from one application to another.

The edge-based approach means that the identification system runs locally on the microcontroller without relying on cloud services or any other powerful computer, this will result in reduced response time and costs.

Many signature verification and identification systems were proposed in the literature, Hafs et al. [7] proposed an authentication approach based on the empirical mode decomposition (EMD) [8] and the Mellin transform, their system achieved a very promising equal error rate (EER) of 2.13%. In another paper, Manjunatha et al. [9] achieved a minimum EER of 3.67% when testing their offline signature verification approach on the MCYT database [10], in their approach they tried various and multiple user-dependent features combinations. In recent years, the use of deep learning for signature verification has gained an increase in popularity, Lai et al. [11] proposed a novel descriptor, which they called the length-normalized path signature (LNPS), and conducted experiments on the SVC2004 and MCYT-100 databases, to train the system they have used a GRU based neural network architecture, and they were able to achieve an EER of 2.37%. Although deep learning models can achieve high accuracies, they require heavy computational resources, which makes them difficult to deploy on devices with limited power.

The main contribution of this paper is the development of an online signature identification technique based on machine learning. By surveying the literature, we have found that most of the published papers focused on authentication, which has the goal of verifying if a given signature is genuine or belongs to an imposter. However, the study of biometric modalities is not limited to authentication only, it extends to identification, which aims to assign a given signature to a particular individual, and to the best of our knowledge, there are only a few researchers who exploited using online signature in an identification system. Additionally, while many papers have studied the use of deep learning and neural networks for signature verification [12], [13] but as far as we are aware, none of them have deployed deep learning models into a computational resource-constrained device such as a microcontroller. This paper aims to fill this gap by introducing a novel tinyML [14], [15] approach for online signature identification, that is not heavy in terms of memory and computational needs and can be easily deployed on a microcontroller. The proposed system was successfully deployed and run on the ESP32 board.

The rest of this paper is organized as follows: In Section II, we present our proposed approach for online signature identification using a tinyML on an ESP32 board. In Section

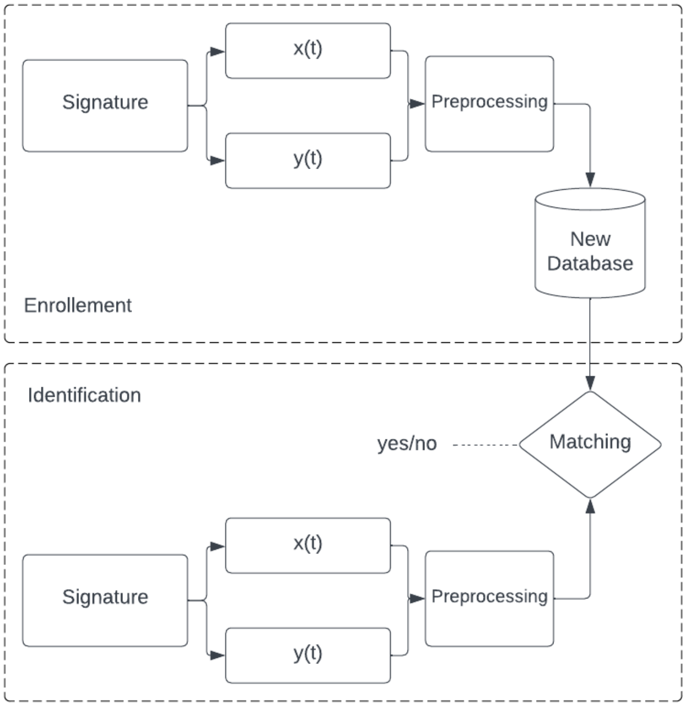


Fig. 1: Block diagram illustrating the proposed system.

III, we evaluate the performance of our approach and give a discussion of our contributions. Finally, Section IV presents the conclusion and the discussion of the potential avenues for future research.

II. Proposed System

The proposed system is shown in Figure 1. We start by applying a Gaussian filter to both of the x and y coordinates of the online signature. This step is then followed by three normalization stages: position normalization, size normalization, and length normalization. The aim of these normalizations is to remove any variation from the signal that can affect the performance of the system. After that, a new database is created using the resulted x and y coordinates, this database will be used to train the model. Once the training is done, the model will be quantized in order to reduce its size, so it can be deployed into the ESP32.

A. Database used

In order to evaluate our system, we have used the publicly available SVC2004 database for task 1 [16]. It contains data collected from 40 different users, each of which provided 20 genuine signatures. To evaluate our system we have used the signatures of the first 20 users, and the skilled forgeries signatures are ignored and not used as they are not needed.

During the collection of the signatures, the participants were advised to not use their original signatures, and develop and train on new ones instead. The data were collected using a WACOM Intuos tablet with 100 Hz as a sampling frequency. Although most of the participants are Chinese, many of them provided English signatures.

B. Gaussian Filter

The Gaussian filter [17] is a smoothing filter, it is used to remove high-frequency noises from unidimensional signals. We applied the filter to both coordinates according to equations (1) and (2).

$$x(t) = \sum_{i=-2\sigma}^{2\sigma} M_i * x(t+i) \tag{1}$$

$$y(t) = \sum_{i=-2\sigma}^{2\sigma} M_i * y(t+i) \tag{2}$$

Where Mi is given by:

$$M_i = \frac{e^{-\frac{i^2}{2\sigma^2}}}{\sum_{j=-2\sigma}^{2\sigma} e^{-\frac{i^2}{2\sigma^2}}} \tag{3}$$

The Gaussian filter will preserve the low-frequency content of the signal and remove any unwanted oscillation without affecting the general shape of the original signature.

C. Postion Normalization

In this step, we will calculate the centre of gravity of the signature, this is done by calculating the mean as follows:

$$G_x = \frac{1}{N} \sum_{i=1}^{N} X_i \tag{4}$$

$$G_y = \frac{1}{N} \sum_{i=1}^{N} Y_i \tag{5}$$

After that, we will subtract the centre of gravity from the input signal as shown by (6) and (7), which will result in an aligned signature centred around the origin.

$$C_x = x(t) - G_x \tag{6}$$

$$C_y = y(t) - G_y \tag{7}$$

D. Size Normalization

This phase consists of normalizing the amplitude of both of x and y coordinates within the range of [0, 1], according to the following equations:

$$x_{rescaled} = \frac{x - min_x}{max_x - min_x} \tag{8}$$

$$y_{rescaled} = \frac{y - min_y}{max_y - min_y} \tag{9}$$

We did size normalization in order to make sure that the signal has consistent and comparable characteristics.

E. Length Normalization

In this step, we resize the x and y coordinates signals using the linear interpolation [18] in order to create 2 new vectors (Nx and Ny) with the size of 32 samples each.

This type of normalization ensures that signatures will be of the same length, so they can be accurately compared and processed. This will later lead as well to improve the overall performance of the identification system.

F. Architecture of the deep learning model

To create our model we have used the well-known Keras and TensorFlow libraries, the architecture of the model consists of a sequential input layer, three dense layers with 32, 24, and 20 hidden units respectively, where the 20 units in the last layer represent the number of the total output classes. In order to prevent overfitting and help the model generalize better to new data, in the first two dense layers, we have used an L1 activity regularization of value 0.00001 with ReLU as an activation function, while in the third and last dense layer, we have used the softmax activation function. The rest of the training parameters are as follows, the used optimizer is Adam, the learning rate was 0.001, the batch size was 32, the categorical cross-entropy loss function was used, and the model was trained for 100 epochs.

G. Deployment on the ESP32 board

Deploying deep learning models on low-power and resource-constrained embedded systems is an emerging research field. These devices require efficient processed and lightweight models. In this section, we are going to describe the deployment of the model described in the previous section on an ESP32 board.

The ESP32 board [19] is a powerful microcontroller with a dual-core LX6 microprocessor that has a frequency that can go up to 240 MHz, 520 KB SRAM, and 4 MB of flash memory. The powerful processing capabilities of the ESP32 make it suitable for TinyML applications.

1) 8-Bit Quantization: Machine and Deep learning models require a lot of memory resources, so they cannot be easily deployed on edge devices such as microcontrollers. In order to reduce the model size, we must optimize it through a process called quantization. The quantization consists of rounding the values to the nearest multiple, which is defined by a fixed step, the new values are represented with a smaller number of bits. The weights and biases of the model are represented as 32 bits floating point numbers, we have used 8-bit quantization, which allowed us to reduce the memory requirements.

2) Deployment: To prepare the model for deployment on the ESP32 board, we have used the Edge Impulse [20] framework. Edge Impulse is a platform that offers the possibility of building, training, testing, and deploying machine and deep learning models on resource-constrained microcontrollers. It can generate optimized libraries for the target device. In our case, after the model was successfully trained we used the Edge Impulse platform to export it as an Arduino library and imported it into the Arduino IDE. We then used the Arduino IDE to develop a new program that can classify the online signature from the test set as discussed in section III. The output of the model can be visualized on the serial monitor as shown by figure 3.

III. Results and Discussion

To evaluate the performance of our proposed system, we utilized the widely used SVC2004 task1 database, which contains a collection of online signatures that comprise both genuine and forged samples. However, in our study, we limited our focus to only the genuine signatures of the first 20 users in the database, while discarding the rest. restricting our analysis to only the genuine signatures of the first 20 users in the database ensured that we had an adequate representation of a diverse group of users. This approach allowed us to obtain a more realistic evaluation of our system's accuracy, given the substantial variations in writing styles among different individuals.

In this paper, we have presented an edge-based approach for identifying individuals from their online signatures, which we have successfully deployed on an ESP32 board. Our proposed system consists of multiple pre-processing techniques, which include the application of a Gaussian filter, position normalization, size normalization, and length normalization to the raw online signature data. By using these techniques, we aimed to enhance the quality of the data inputted into our classification model and improve its ability to distinguish between different signatures accurately.

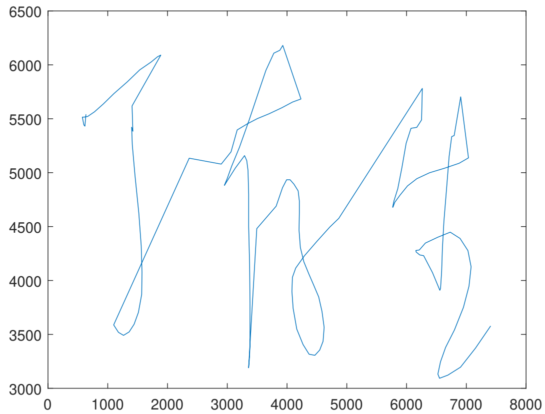
Our system's classification model comprises a sequential input layer and three dense layers, with 32, 24, and 20 hidden units, respectively. The model's final output layer consists of 20 units representing the output classes. The system's performance was evaluated using accuracy as a metric, with the system achieving a high accuracy rate of 94.94%.

The system was trained on a more powerful machine, however, to deploy the system on the ESP32 board, we had to optimize the model by performing an 8-Bit quantization. The quantized system achieved similar results in terms of accuracy and inference time as the unoptimized model and better performance in terms of flash and RAM usage as shown in table I. This indicates that it is feasible to use a microcontroller such as the ESP32 for online signature identification using a TinyML model.

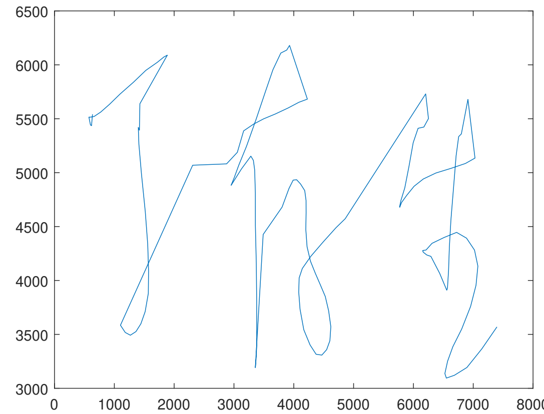
TABLE I: A performance comparison between the quantized and unoptimized models.

	Inference time (ms)	Model Size (KB)	RAM Usage (KB)	Accuracy
Quantized (int8)	4	36	24.71	94.94%
Unoptimized (float32)	3	95	24.86	94.94%

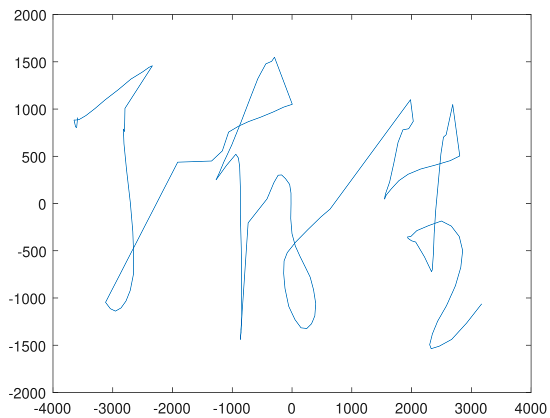
As presented in Table I, the quantized system's performance was better than the unoptimized system. The quantization process did not affect the overall accuracy of the model and, more importantly, led to reduced flash and RAM usage. This



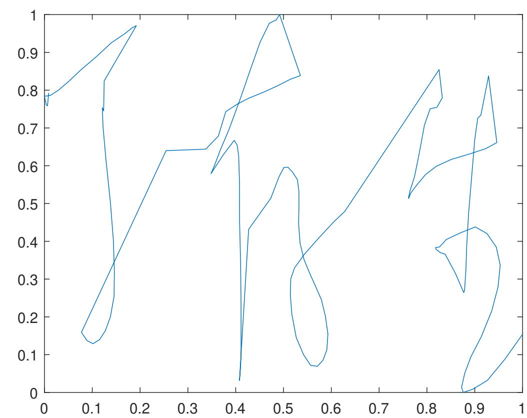
(a) Original signature



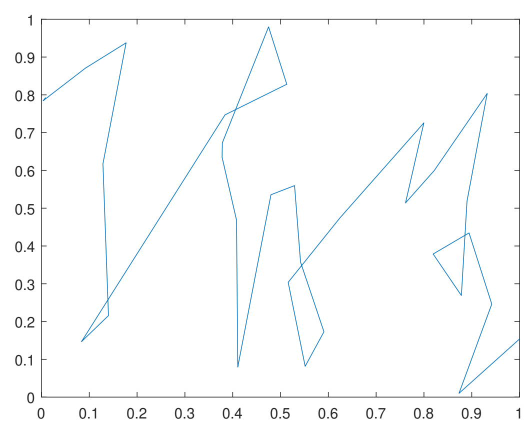
(b) Signature after applying the Gaussian filter



(c) Signature after Postion Normalization



(d) Signature after Size Normalization



(e) Signature after Length Normalization

Fig. 2: Representation of a randomly selected online signature from the SVC2004 database at each stage of the preprocessing phase.

indicates that quantization is a useful and valuable technique for converting and deploying machine and deep learning models into embedded systems with limited computational power.

Microcontrollers such as the ESP32 have limited resources in terms of memory and processing power, which highlights the importance of optimizing the memory usage and processing time of the model. The quantization technique allowed

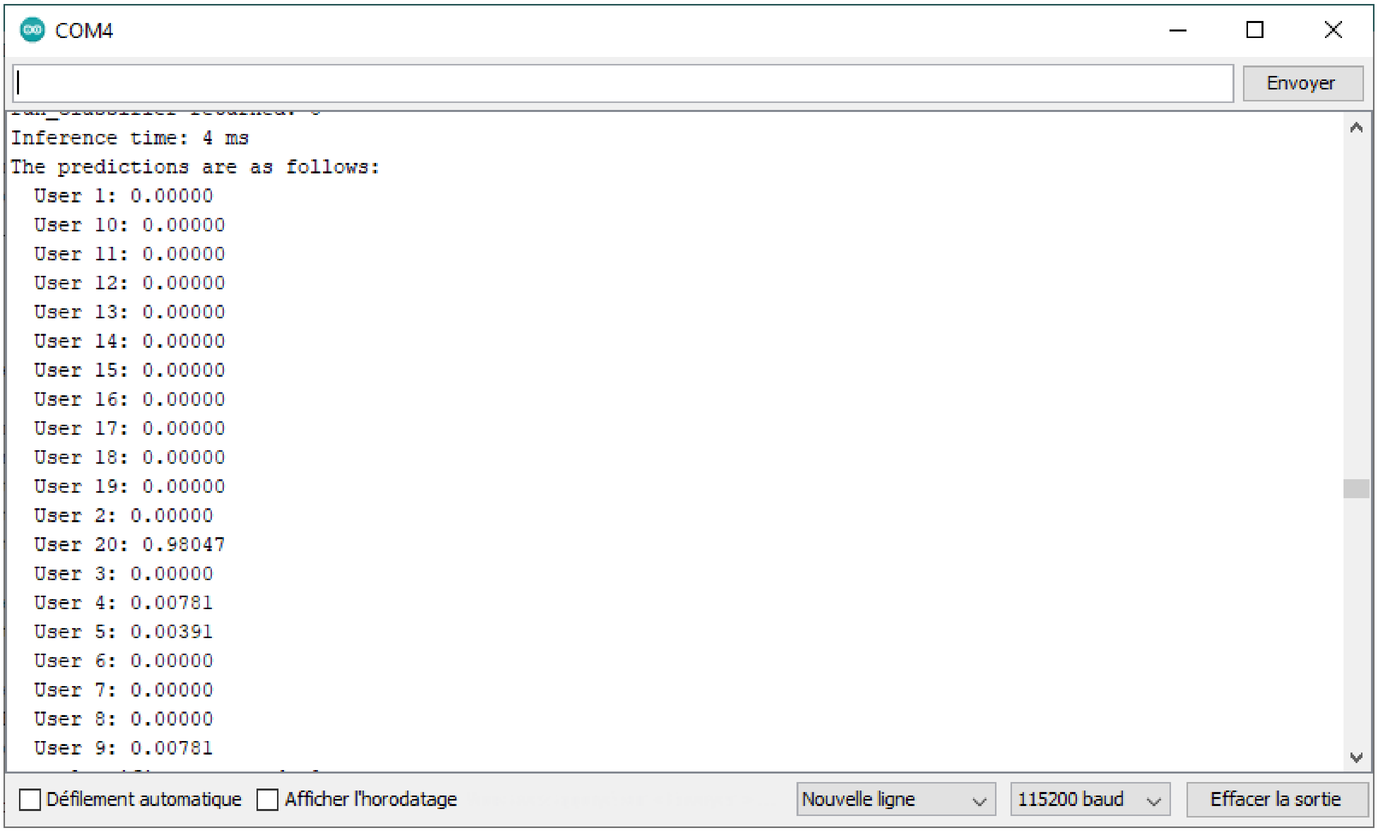


Fig. 3: Visualization of a classification result on the serial monitor of the Arduino IDE, demonstrating successful prediction of the true class of a signature belonging to the 20th user with a high level of confidence, specifically 98.05%.

us to reduce the memory requirements of the model without affecting its performance, making it a critical technique for deploying machine learning models on embedded systems.

One of the drawbacks of our study is the use of a single database for the evaluation of the system, which does not give us a complete idea of how the model may generalize to new data or real-world scenarios.

IV. Conclusion

In conclusion, this paper proposed an innovative and practical solution for online signature identification on edge devices using TinyML and deployed on an ESP32 microcontroller. The pre-processing techniques applied, including Gaussian filtering and multiple normalization techniques, were effective in enhancing the accuracy of the model. We showed that the model can achieve an accuracy rate of 94.94% even when running on constrained devices, such as the ESP32 microcontroller.

The success of our proposed system demonstrates the potential of deploying machine and deep learning models on edge devices, especially for tasks requiring real-time processing. The proposed system can be used in various applications, including access control and financial transactions, where identifying individuals from their signatures is a critical requirement.

In future work, we aim to improve the system's accuracy and robustness by combining online signature with another biometric modality such as fingerprint, which could increase the accuracy of the system even further. Additionally, we could investigate the effectiveness of other pre-processing techniques and architectures to further optimize the system's performance.

Overall, the proposed system is a promising step towards improving online signature identification on edge devices, and we hope that this work will inspire further research in this area.

References

[1] T. Hafs, L. Bennacer, M. Boughazi, and A. Nait-Ali, "Empirical mode decomposition for online handwritten signature verification," *IET Biometrics*, vol. 5, no. 3, pp. 190–199, 2016.

[2] K. Manjunatha, S. Manjunath, D. S. Guru, and M. Somashekara, "Online signature verification based on writer dependent features and classifiers," *Pattern Recognition Letters*, vol. 80, pp. 129–136, 2016.

[3] A. Sharma and S. Sundaram, "An enhanced contextual dtw based system for online signature verification using vector quantization," *Pattern Recognition Letters*, vol. 84, pp. 22–28, 2016.

[4] E. Alajrami, B. A. Ashqar, B. S. Abu-Nasser, A. J. Khalil, M. M. Musleh, A. M. Barhoom, and S. S. Abu-Naser, "Handwritten signature verification using deep learning," 2020.

[5] H.-H. Kao and C.-Y. Wen, "An offline signature verification and forgery detection method based on a single known sample and an explainable deep learning approach," *Applied Sciences*, vol. 10, no. 11, p. 3716, 2020.

[6] P. Maergner, V. Pondenkandath, M. Alberti, M. Liwicki, K. Riesen, R. Ingold, and A. Fischer, "Combining graph edit distance and triplet networks for offline signature verification," *Pattern Recognition Letters*, vol. 125, pp. 527–533, 2019.

[7] T. Hafs, L. Bennacer, A. Nait-Ali, M. Boughazi, and A. Bouzid-Daho, "Online signature verification approach using mellin transform and empirical mode decomposition," in *2017 2nd International Conference on Bio-engineering for Smart Technologies (BioSMART)*, pp. 1–6, 2017.

[8] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[9] K. Manjunatha, H. Annapurna, and D. Guru, "Offline signature verification: An approach based on user-dependent features and classifiers," in *Data Analytics and Learning: Proceedings of DAL 2018*, pp. 235–243, Springer, 2019.

[10] J. Ortega-Garcia, J. Fierrez-Aguilar, D. Simon, J. Gonzalez, M. Faundez-Zanuy, V. Espinosa, A. Satue, I. Hernaez, J.-J. Igarza, C. Vivaracho, *et al.*, "Mcyt baseline corpus: a bimodal biometric database," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 150, no. 6, pp. 395–401, 2003.

[11] S. Lai, L. Jin, and W. Yang, "Online signature verification using recurrent neural network and length-normalized path signature descriptor," in *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, vol. 1, pp. 400–405, IEEE, 2017.

[12] C. S. Vorugunti, V. Pulabaigari, R. K. S. S. Gorthi, and P. Mukherjee, "Osvfusenet: Online signature verification by feature fusion and depth-wise separable convolution based deep learning," *Neurocomputing*, vol. 409, pp. 157–172, 2020.

[13] A. Hefny and M. Moustafa, "Online signature verification using deep learning and feature representation using legendre polynomial coefficients," in *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2019) 4*, pp. 689–697, Springer, 2020.

[14] P. Warden and D. Situnayake, *Tinyml: Machine learning with tensorflow lite on arduino and ultra-low-power microcontrollers*. O'Reilly Media, 2019.

[15] D. Situnayake and J. Plunkett, "Ai at the edge: Solving real world problems with embedded machine learning," 2022.

[16] D.-Y. Yeung, H. Chang, Y. Xiong, S. George, R. Kashi, T. Matsumoto, and G. Rigoll, "Svc2004: First international signature verification competition," in *Biometric Authentication: First International Conference, ICBA 2004, Hong Kong, China, July 15-17, 2004. Proceedings*, pp. 16–22, Springer, 2004.

[17] G. Deng and L. Cahill, "An adaptive gaussian filter for noise reduction and edge detection," in *1993 IEEE conference record nuclear science symposium and medical imaging conference*, pp. 1615–1619, IEEE, 1993.

[18] H. Akima, "A new method of interpolation and smooth curve fitting based on local procedures," *Journal of the ACM (JACM)*, vol. 17, no. 4, pp. 589–602, 1970.

[19] M. Babiuch, P. Foltỳnek, and P. Smutnỳ, "Using the esp32 microcontroller for data processing," in *2019 20th International Carpathian Control Conference (ICCC)*, pp. 1–6, IEEE, 2019.

[20] S. Hymel, C. Banbury, D. Situnayake, A. Elium, C. Ward, M. Kelcey, M. Baaijens, M. Majchrzycki, J. Plunkett, D. Tischler, A. Grande, L. Moreau, D. Maslov, A. Beavis, J. Jongboom, and V. Janapa Reddi, "Edge impulse: An mlops platform for tiny machine learning," 11 2022.