Deep Learning Algorithms for Complex Pattern Recognition in Ultrasonic Sensors Arrays

Deep Learning Algorithms for Complex Pattern Recognition in Ultrasonic Sensors Arrays

Vittorio Mazzia[1,2], Angelo Tartaglia[1], Marcello Chiaberge[1] and Dario Gandini[1]

Politecnico di Torino - Dept. of Electronic and Telecommunications Engineering (DET)
[1]PIC4SeR - Politecnico Interdepartmental Centre for Service Robotic
[2]SmartData@PoliTo - Big Data and Data Science Laboratory
Turin - Italy

Abstract. Nowadays, applications of ultrasonic proximity sensors are limited to a post-processing of the acquired signals with a pipeline of filters and threshold comparators. This article proposes two different and novel processing methodologies, based on machine learning algorithms, that outperform classical approaches. Indeed, noisy signals and presence of thin or soundproofing objects are likely sources of false positive detections that can make traditional approaches useless and unreliable. In order to take advantage of correlations among the data, multiple parallel signals, coming from a cluster of ultrasonic sensors, have been exploited, producing a number of different features that allowed to achieve more accurate and precise predictions for object detection. Firstly, model-based learning as well as instance-based learning systems have been investigated for an independent time correlation analysis of the different signals. Particular attention has been given to the training and testing of the deep fully connected network that showed, since the beginning, more promising results. In the second part, a recurrent neural network, based on long short term memory cells, has been devised. As a result of its intrinsic nature, time correlations between successive samples are not more overlooked, further improving the overall prediction capability of the system. Finally, cutting edge training methodologies and strategies to find the different hyperparameters have been adopted in order to obtain the best results and performance from the available data.

Keywords: deep learning, ultrasound sensors, industrial security

1 Introduction

The usage of ultrasonic sensors is widely diffused in many and different fields. In [1], this type of sensors are used to measure the distance of an object. It is a very old technology [2], but still new patents [3] and papers [4] are continuously disclosed in a wide range of fields: from distance security measurement, automotive field to robotics applications. Other newest applications use ultrasonic sensors for image reconstruction of organs like liver [5] or heart [6]. These last techniques are also associated with machine and deep learning analysis for post-processing

of the captured images. Other machine learning related works with ultrasonic sensors can be found in [7] [8] [9].

This paper proposes an alternative methodology, based on machine learning algorithm, to manage and extract results from an ensemble of sensors in order to detect the presence of possible obstacles. Indeed, instead of using a single transducer, correlations between different signals open the possibility to detect targets that are not recognized by a classical methodology. In fact, it is possible to demonstrate that a redundancy of sensors makes the entire system more robust and in addition it opens the possibility to drastically reduce the number of false positive detections. So, based on this new type of data and based on modern technique of machine learning, this paper shows that is possible to design a more accurate and precise system than ones offered by tradition approaches.

2 Case Study

Industrial security is continuously evolving enclosing techniques and methods that take advantage of latest researches. Often classical processing of data is not sufficient to reach the desired level of security for workers and this paper deals with this scenario. The proposed research can be a valuable support for classical methodologies, making a more strong and robust overall system. Possible applications of the enhanced system include obstacle avoidance for automatic guided vehicles (AGV), stacker cranes and open computer numerical control (CNC) machines. The devised test-bed is a replica of the worst condition found in a hypothetical real industrial case and it was used to generate the data set and later for the tests of the final models.

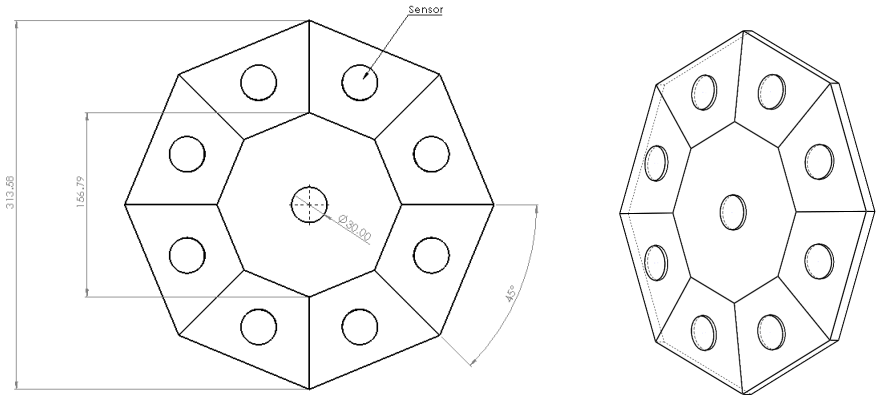

Fig. 1. On the left the scheme of the panel with sensors, on the right the 3D view of overall system.

2.1 Dataset acquisition bench

The system that has been considered for data acquisition is composed of an array of nine coaxial ultrasonic sensors equally spaced on an octagonal board (fig. 1). The test bench has been used to detect the presence of a thin rod posed in front of the cluster. The target has been coated with soundproofing material in order to test the system in critical condition reducing the amount of reflected waves. The target has been placed in different positions, orientations and distances trying to emulate the real conditions and to produce a sufficiently large and comprehensive data set for the training of the different models. Figure 1 shows the sensor cluster and the overall system setting. Lateral borders are slightly tilted in order to further open the field of view of the array of sensors. The described testing area has been used for the generation of a very large data set composed by the multiple signals acquired from the nine different sensors. Signals have been recorded as raw data, only discarding the first noisy part of the acquisitions. Indeed, having used coaxial type of sensors (emitter and receiver are on the same axis), the excitation phase produces, for a short period, saturated signals on all transducers that can be easily omitted. Moreover, it is pretty straightforward to automatize this simple procedure removing always the first part of the incoming signals.

3 Dataset visualization

Firstly, the recorded raw data have been analyzed and visualized with different solutions. Using, principal component analysis (PCA), it has been possible to

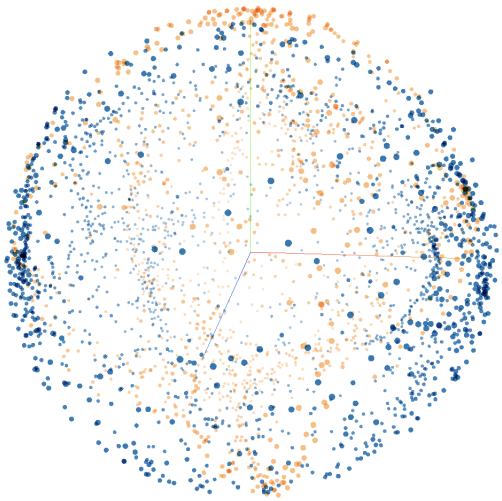

Fig. 2. Projection of the nine signals using PCA. The data is normalized by shifting each point by the centroid and making it unit norm.

plot on a tridimensional graph the different acquisition points. In fig. 2, blue points are samples characterized by the presence of the target and orange points when the object is not present in the field of view. Reflections made by the thin rod are so subtle that they can be easily lost in the background noise. However, even if data points are not well separated, it is interesting to notice how the two classes clustered in specific regions of the space making it clear how correlations between different sensors are a key aspect to exploit for the detection of the target. As already introduced, the main motivation for the use

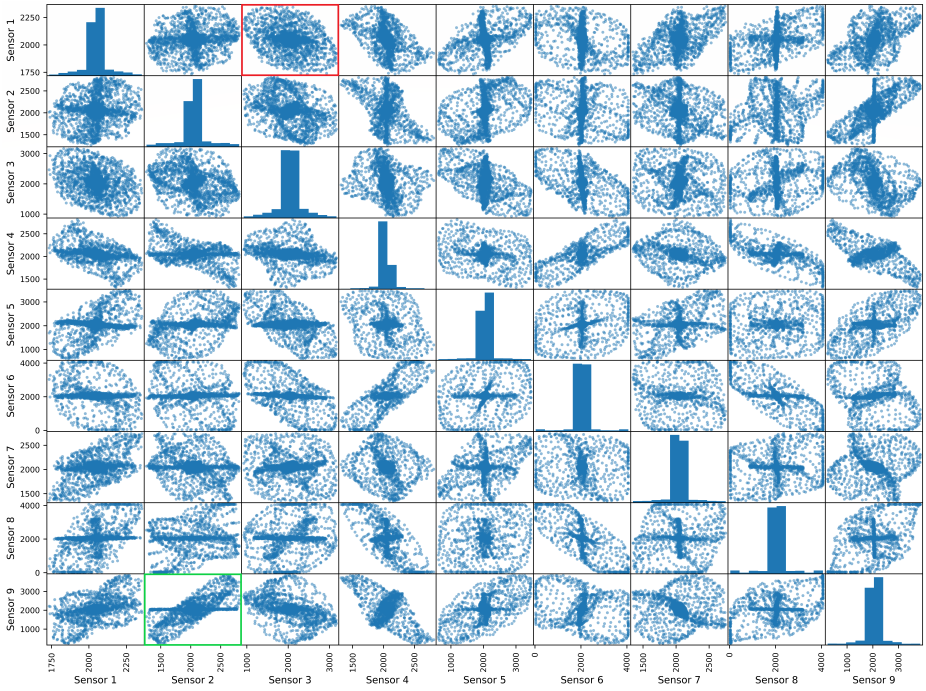

Fig. 3. Correlation matrix of the signals coming from the nine sensors.

of an ensemble of sensors is that it opens the possibility to exploit correlations between the acquired parallel signals (figure 3). A first high-level analysis can be done considering the following considerations: if the correlation between two sensors reaches high values, the points on the graph will be roughly disposed on a straight line and there can be found an example of this in the correlation graph between sensors 9 and 2 (green square); instead when the graph shows a cloud-pattern disposition it indicates a low value of correlation. The graph between sensors 1 and 3 (red square) can be used as example for this case. From those correlations the network is capable to understand information not explicitly given by the sensor itself.

4 Preliminary analysis

First of all, the acquired raw dataset has been analyzed with a simple classical methodology that can be easily improved with more elaborated processing [7]. However, already in this preliminary part, it has been taken into account the advantage of working with a redundancy of sensors. Indeed, it is obvious from the following results that correlations between the sensors can largely improve the detection accuracy. The nine signals produced by one of the acquisitions

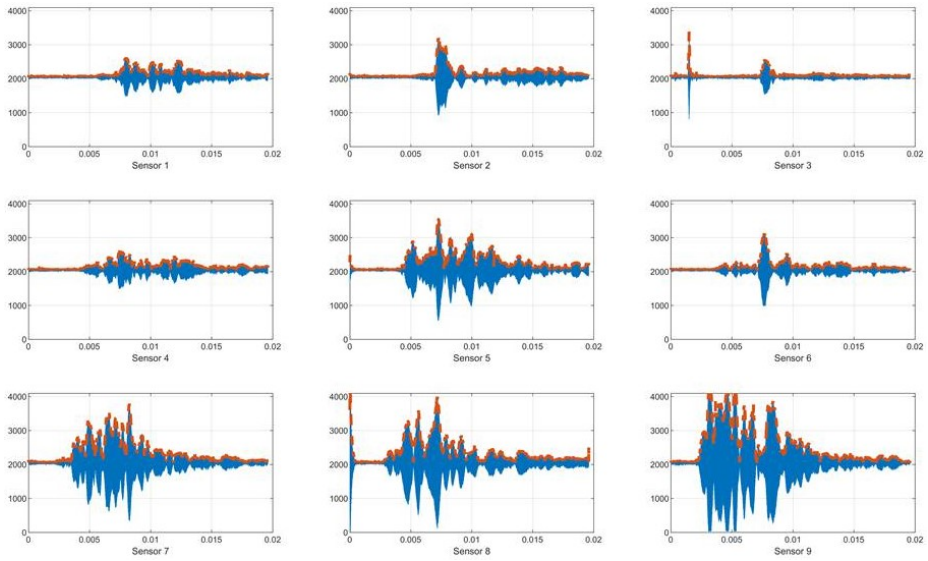

Fig. 4. Waveforms of the nine sensors of a sample acquisition. Envelops are highlighted in orange on each sensors graph.

are represented in 4. The detection is easily recognizable with the second sensor response, but it is lost in the background noise in almost all remaining sensors. The envelope curve of each acquisition is highlighted in orange on each sensor signal. Moreover, it is clear how the acquired waves are very dissimilar due to the different location of the sensors on the board.

Firstly, it is possible to perform a simple detection independently on each sensor, simply by comparing the resulting envelops with a manually tuned threshold T_a. However, due to the critical setting of the designed test bench, it is possible to demonstrate that with this simple strategy each sensor produces a huge number of false positive detections.

On the other hand, it is possible to implement a more elaborated processing, exploiting the concurrent signals coming from the nine sensors. As it is represented in figure 5, it is pretty straightforward to construct a cumulative function

which takes into account how many sensors are above the imposed threshold T_a. At this point, a second threshold T_b has to be set, in order to determine how

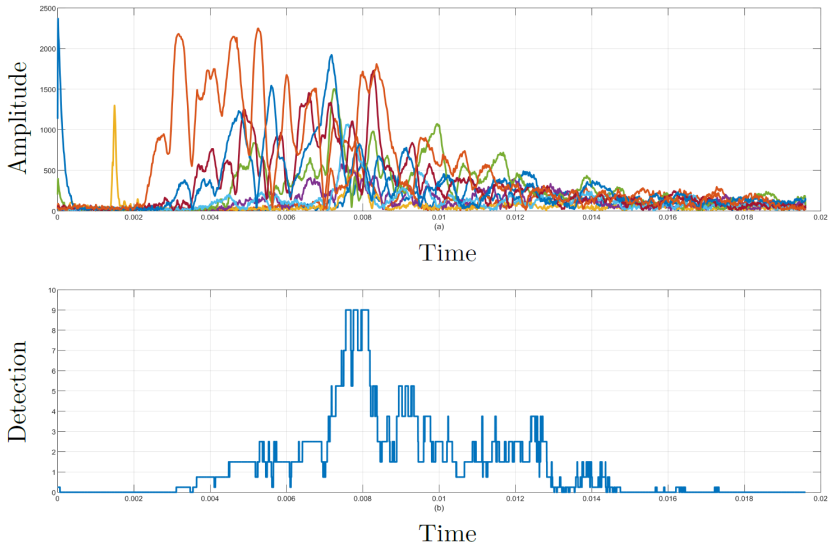

Fig. 5. Above, envelope curves of the acquired signals. Below, cumulative function of the sum of concurrent detections.

many sensors should concur to confirm a detection. On figure 6, with two different colors are presented two possible outcomes with different values of T_b. In conclusion, it is clear from these preliminary analysis how thresholds T_a and T_b

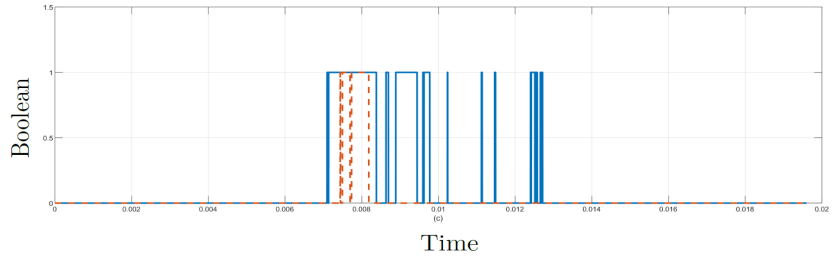

Fig. 6. The solid blue line: detections with concordance of 3 sensors and orange dashed line: concordance of 6.

can greatly affect the overall result of the system, generating false positive alarm or miss detections. So, difficulty in selecting optimized thresholds, the manual interpretation of a considerable amount of acquisitions and boost in performance

exploiting correlations between the different sensors, are the driving motivations for the presented innovative methodologies.

5 Proposed Methodologies

The analysis and processing of the ultrasonic waves has been carried out using two distinct approaches. Initially, only samples value range are took into account, neglecting temporal correlations between subsequent instances. However, the contemporary use of multiple sensors and the derived correlations between different signals, opens the possibility to build and train robust models, able to achieve optimal results. Secondly, a much more sophisticated model has been designed to exploit jointly the time information of the different wave forms and their amplitude values. In the next section both methodology are compared and analyzed in view of the results achieved.

5.1 Independent time correlations analysis

Model-based learning as well as instance-based learning systems have been investigated, showing excellent results on both types of binary classifiers. The pre-processing phase is a standard procedure. Firstly, the acquired dataset has been processed with a simple pipeline, removing non representative samples (initial time steps with saturated signals due to the emission of ultrasound waves) and applying standardization to perform feature scaling. Secondly, the resulting training data, with nine attributes, coming from the designed array of sensors, have been manually annotated and used to train and evaluate the different models. In order to better evaluate the different cases, K-fold cross-validation, with k equal to ten, and randomized search have been used in order to evaluate some initial models by varying some hyperparameters. For the sake of completeness, for the first algorithm, K-nearest neighbors (K-NN) [11], 6 is the number of neighbors used with Euclidean as distance reference metric. For the Random Forest [12], 50 is the number of estimators with 3 as minimum samples split (minimum number of samples required to split an internal node) and maximum depth of 15. The selected function to measure the quality of a split was 'entropy' for the information gain. Finally, for the kernel SVM [13], it has been used a radial basis function kernel 'rbf' with 1.5 as value for the penalty parameter. After this first preliminary phase, the analysis has been focused only on the most promising hyperparamenters, fine tuning it with a grid search strategy. In table 1 the different architectures are shown, with their corresponding results generated by the K-fold cross-validation process [14].

It is possible to see the difference between results achieved by the feed forward fully connected neural network and all other supervised algorithms. That is made possible by the presence of multiple hidden layers, capable of mining deep correlations between the different samples and represent knowledge in a hierarchical structure [15]. After the grid search analysis, the selected configuration accounts five hidden layers with 32, 32, 32, 52 and 62 rectified linear

units (ReLU) respectively, for a total of 7,497 trainable parameters. Moreover, a batch of 128 instances has been used with AMSGrad [16] optimizer, an adaptive learning rate method which modifies the basic ADAM optimization algorithm. The overall algorithm update rule, without the debiasing step, is shown in Eq.1. The first two equations are the *exponential decay* of the gradient and *gradient squared*, respectively. Instead, with the third equation, keeping an higher v_t term results in a much smaller learning rate, η, fixing the exponential moving average and preventing to converge to a sub-optimal point of the cost function.

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \\ \hat{v_t} &= max(\hat{v_{t-1}}, v_t) \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{v_t} + \varepsilon}} m_t \end{aligned} \tag{1}$$

Finally, only Dropout regularization technique has been applied at each layer with a drop rate of 30%. It is also worth pointing out that, due to the online learning nature of neural networks, once a system is trained on the original dataset, it can be further improved with new mini-batches coming from the actual application. That opens the possibility to devise a dynamic setup able to adjust and evolve to the specific application and to new environmental changes.

5.2 Time correlations analysis

In this second part, time correlations, between subsequent data samples, are no more overlooked. On the contrary, that additional information is exploited to achieve an even better model, more prone to generalize over new sample data points. That is achieved using a deep learning model based on long short term memory (LSTM) layers that increasingly extracts longer range temporal patterns. Moreover, in order to avoid random interactions between different time-steps and channels (LSTM hidden units), a many-to-many architecture [17] has been adopted. For this purpose, a TimeDistributedDense layer is applied which executes a Dense function across every output over time, using the same set of weights, preserving the multidimensional nature of the processed data. Figure 7 presents a simplified overview of the devised model. The adoption of this strategy, opens the possibility to exploit the temporal correlations between the different samples without leaving out the interactions among the nine signals.

Dataset pre-processing For this second case study in the pre-processing pipeline few more steps were needed. Indeed, the devised recurrent neural network (RNN), requires data points to be collected in slice of time series. So, before feeding the model, the samples of each sensor have been collected in chunks of data, forming a three-dimensional array, used for the training of the model. In order to maximize the number of possible new instances of the training dataset the following rule has been adopted:

$$X_{n,:,:} = x^{(i-t):(i)} = (x^{(i-t)}, ..., x^{(i)}) \tag{2}$$

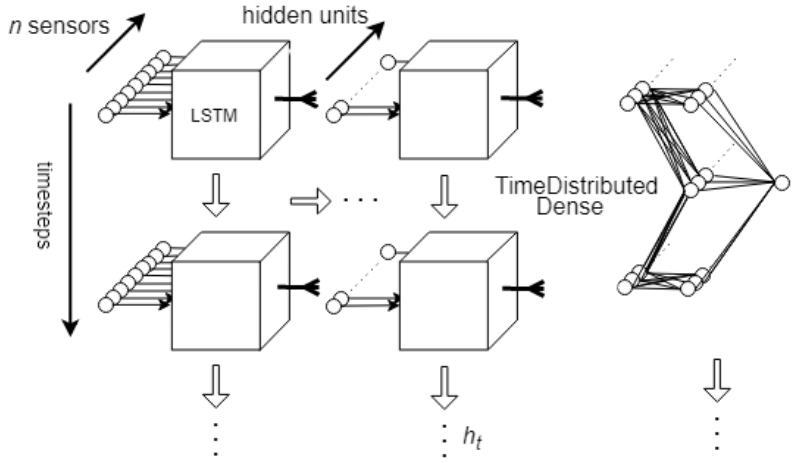

Fig. 7. Overview of the adopted RNN architecture based on LSTM units.X^i samples feed a sequence to sequence model based on LSTM units that gradually extract higher level temporal representations. After that, a TimeDistributedDense layer is applied which executes a Dense function across every output over time using the same set of weights. Finally, a fully connected layer is applied with a sigmoid end point unit to perform the predictions.

where t is the chosen time-step range, $\underline{X} \in \mathbb{R}^{i \times j \times k}$ is the 3rd-order dataset output tensor and $x^{(i)}$ are the feature vectors of the old dataset which has $(x^{(i)})^T$ as rows. Finally, for the ground truth array, the following approach has been adopted:

$$Y^{(i)} = H(\sum_{n=i-t}^{i} y^{(n)} - t/l) \tag{3}$$

where $H(x)$ is the Heaviside step function, $Y \in \mathbb{R}^i$ is the new label array, $y^{(i)}$ is the old one and t the time-step value. The proposed approach introduced a new hyperparameters, l, which is required to be set and can affect the reactivity of the system. Figure 8 presents the architecture of the generated data structure. The left cube of data represents the different instances used as input for the RNN. At the centre, there are the feature extracted by the signals of the sensors and finally, on the right, all individual sample data points, generated by the nine sensors in t time steps. After a randomized search, l has been set to 2 and the time-step parameter to 20 that is, feeding the RNN with twenty subsequent acquisitions with each instance. Finally, non-representative samples were removed and standard deviation applied.

Training and results The final designed model presents three LSTM layers with 32, 15 and 32 hidden units, respectively. Then, a TimeDistributedDense layer with 52 ReLu neurons and finally, a single sigmoid unit to perform the

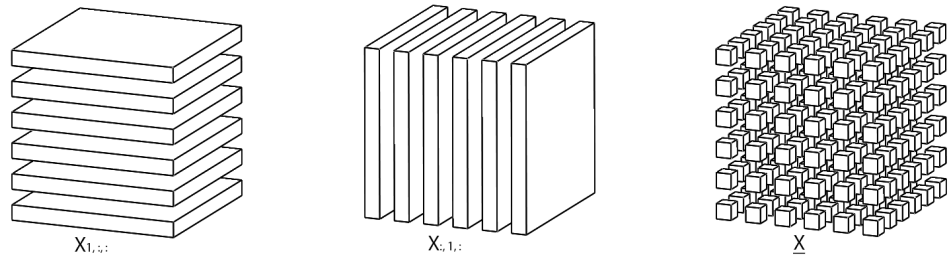

Fig. 8. Overview of the LSTM dataset structure.

predictions. In this second case, particular attention has been given to regularization. As before, Dropout has been used at each layer, but in order to prevent over-fitting also weight decay [18] has been exploited. Again, this is a simple fix to Eq. 1 updating rule, but that has shown, in our case study, for better results than L2 regularization:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} m_t - \eta w_t \theta_t \tag{4}$$

The new term $\eta w_t \theta_t$ added in Eq. 4 subtracts a little portion of the weights at each step, hence the name decay. On the other hand, L2 regularization is a classic method to reduce over-fitting, and consists in adding to the loss function the sum of the squares of all the weights of the model, multiplied by a given hyper-parameter. So, the two strategies are only the same thing for vanilla stochastic gradient descent (SGD), but as soon as we add momentum, or use a more sophisticated optimizer like Adam or AMSGrad, L2 regularization and weight decay become different giving completely results. After, finding the maximum value of learning rate to start with [19], training the RNN with cosine annealing strategy and evaluating the architecture with K-fold cross-validation, the final model has

Supervised Learning Alg.	**Accuracy**	**Precision**	**Recall**	**F1 Score**
K-nearest neighbors (K-NN)	0.9280	0.9979	0.8516	0.9190
Random forest	0.9229	0.9798	0.8004	0.8811
Kernel SVM (rbf)	0.9042	0.9703	0.8004	0.8772
Feed Forward NN	0.9576	0.9796	0.9311	0.9547
Recurrent Neural Network (RNN)	0.9615	0.9677	0.9782	0.9729

Table 1. Results of the independent and dependent time correlation analysis kept separate due to the different trainig methodologies. It is clear, from the last model, how the contemporary exploitation of temporal and instantaneous interactions can greatly boost the final scores. Moreover, it is clear from the F1-score that the last model is much more balanced in performance than others.

been shown to achieve over 97% (table 1) of F_1-score, setting a huge gap between all other tested systems based on a independent time correlations analysis. All training have been carried out on a GPU 2080 Ti with 11 GB of memory exploiting the parallel computing platform CUDA version 10 created by Nvidia. The time spent for the training of the independent time correlation networks was negligible. Instead, the RNN training, due to the complexity of the networks, has required approximately 3 hours.

5.3 Future works

The proposed algorithms could be easily applied in real time applications that would not require the presence of hardware with high computational capabilities. A future development of the proposed methodologies could be an embedded deployment of the overall system, taking advantage of hardware accelerators for the time correlation solution. Indeed, a simple solution could include the usage of hardware like a simple micro-processor for the acquisition of raw data coming from the sensors array, their analysis through the chosen model and generation of warning signals in presence of detected objects. This configuration could solve huge security problems, with a low cost solution, combining two important requirements for present companies.

6 Conclusions

Two novel methodologies have been presented as valuable alternatives to traditional approaches for processing information of ultrasonic proximity sensors. Classical methodologies, in critical situations, struggle to find a correct level of threshold and filtering resulting in miss detections and high false positive rate. Several binary classifiers have been trained in a supervised matter and tested on real case applications where classical techniques struggles to detect the presence of an object. The contemporary usage of multiple sensors, in a cluster disposition, revealed clear correlations between the different signals. These recurrent and marked patterns enables the possibility to build more robust and reliable models able to build their knowledge on these information. Finally, results have shown a significant boost in accuracy exploiting not only the instantaneous correlation of the different signals but also the temporal pattern of the acquired sample. Indeed, the simultaneous observation of multiple samples and their temporal correlation can deeply mine into data, producing more elaborated and articulated predictions.

7 Acknowledgements

This work has been developed with the contribution of the Politecnico di Torino Interdepartmental Centre for Service Robotics PIC4SeR (https://pic4ser.polito.it) and SmartData@Polito (https://smartdata.polito.it) .

References

1. Shrivastava, AK and Verma, A and Singh, SP; Distance Measurement of an Object or Obstacleby Ultrasound Sensors using P89C51RD2, *International Journal of Computer Theory and Engineering*, IACSIT Press, 2010
2. Houghton, Richard and DeLuca, Frank; Ultrasonicensordevice, *Patentinspiration*, 1964
3. Li, Shih-HsiungUltrasound sensor for distance measurement, *Google Patents*,2002
4. Wu, Kaizhi and Chen, Xi and Ding, Mingyue; Deep learning based classification of focal liver lesions with contrast-enhanced ultrasound, *Optik-International Journal for Light and Electron Optics*, Elsevier, 2014
5. Wu, Kaizhi and Chen, Xi and Ding, Mingyue; Deep learning based classification of focal liver lesions with contrast-enhanced ultrasound, *Optik-International Journal for Light and Electron Optics*, Elsevier, 2014
6. Carneiro, Gustavo and Nascimento, Jacinto C and Freitas, António; The segmentation of the left ventricle of the heart from ultrasound data using deep learning architectures and derivative-based search methods, *IEEE Transactions on Image Processing*, 2012
7. Farias, Gonzalo and Fabregas, Ernesto and Peralta, Emmanuel and Vargas, Héctor and Hermosilla, Gabriel and Garcia, Gonzalo and Dormido, Sebastián; A Neural Network Approach for Building An Obstacle Detection Model by Fusion of Proximity Sensors Data, *Multidisciplinary Digital Publishing Institute*, 2018
8. De Simone, Marco and Rivera, Zandra and Guida, Domenico; Obstacle avoidance system for unmanned ground vehicles by using ultrasonic sensors, *Multidisciplinary Digital Publishing Institute*, 2018
9. Lee, Donghoun and Kim, Sunghoon and Tak, Sehyun and Yeo, Hwasoo; Real-Time Feed-Forward Neural Network-Based Forward Collision Warning System Under Cloud Communication Environment, *IEEE*, 2018
10. Parrilla, M., J. J. Anaya, and C. Fritsch; Digital signal processing techniques for high accuracy ultrasonic range measurements, *IEEE Transactions on instrumentation and measurement*, 1991
11. Cover T, Hart T P; Nearest neighbor pattern classification, *IEEE*, 1967
12. Breiman, Leo; Random forests, *Machine learning 45.1*, 2001
13. Cortes, Corinna, and Vladimir Vapnik; Support-vector networks *Machine learning 20.3*, 1995
14. Tharwat, Alaa; Classification assessment methods, *Applied Computing and Informatics*, 2018
15. LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton; Deep learning, *nature 521.7553*, 2015
16. Reddi, Sashank J and Kale, Satyen and Kumar, Sanjiv; On the convergence of adam and beyond, 2018
17. Lipton, Zachary C and Berkowitz, John and Elkan, Charles; A critical review of recurrent neural networks for sequence learning, *arXiv preprint arXiv:1506.00019*, 2015
18. Loshchilov, Ilya and Hutter, Frank; Fixing weight decay regularization in adam, *arXiv preprint arXiv:1711.05101*, 2017
19. Smith, Leslie N; Cyclical learning rates for training neural networks, *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, 2017