CS6735 Research Project Footprint Recognition based on Time-series classification

Saeed Kazemi[1], Maryam Sepasi[2]

University of New Brunswick

Abstract

Given present-day security concerns, many buildings have implemented robust authentication techniques. Aside from authentication to enter a building, applications such as border and airport security also administer identification. Therefore, many cities and companies provide technologies like CCTV or fingerprinting for authentication and verification. But each system has its own drawbacks. For example, due to the Covid-19 pandemic, most people wear a mask and avoid touching unnecessary surfaces. Thus, gait recognition could be a solution.

In this paper, we present a review of the time series approaches for classification tasks including conventional machine learning algorithms and Deep Neural Networks. Also, in Appendix B, more approaches have been reviewed for time series classification. These approaches were implemented in verification mode. Experimental results show that the SVM classifier on the contribution of all handcrafted features had the best performance with 91.3%.

Keywords: Footprint recognition, Time-series classification, pressure sensor

1. Introduction

Contemporary security identification has led to a plethora of biometric-based authentication systems. From palm readers at testing centers to facial recognition on smartphones, many systems are now used to regularly verify identities. These inherence-based systems are attractive in comparison to knowledge or possession-based methods of authentication because biometrics are unique, unforgettable, and far more difficult to steal than a password or swipe card.

Although biometric identification appears sophisticated when compared to something like physical keys, it does not come without its own share of caveats. For example, owing to the ongoing Covid19 pandemic, many people wear masks when outside of their house, challenging most facial recognition systems. Additionally, biometrics that rely on touch, such as fingerprinting, raise safety concerns, as the scanner may become a vector for virus transmission. Despite these setbacks, given their merits and widespread deployment, biometric identification systems are unlikely to disappear.

[1]Saeed.Kazemi@unb.ca.
[2]maryam.sepasi@unb.ca.

One behavioral biometric that has gained recent success and is worth further consideration given current constraints is gait recognition. Usage of gait recognition has grown in the security industry in recent decades due to advances in deep learning. Singh et al. [1] categorized gait recognition into two main categories, vision-based and sensor-based. In vision-based approaches, cameras capture data of a person walking for the purpose of gait recognition. Sensor-based gait recognition is performed using either wearable sensors which produce kinematic data, or floor sensors which produce kinetic data [2].

This paper focused on analysing kinetic data. For this purpose, the Stepscan dataset which is a private dataset was used [3]. This dataset was obtained from high-resolution floor tiles that have recently been introduced by Stepscan Technologies Inc. Furthermore, this dataset consists of a spatial-temporal tensor, X, with dimensions $S \times T \times H \times W$ where S represents the number of samples. T is the number of temporal observations or video frames; H and W are the dimension of the image in pixels. Figure 1 indicates three frames from one of samples in the dataset.

In general, there are two modes for footprint recognition or generally in the biometric system: verification or identification mode [4]. In verification mode, the biometric system is used for accessing buildings or data. In other words, the system compares the claimed person with its dataset to determine whether or not the claim is valid. These systems not only consume less processing power and time but also have better performance regarding identification systems [4]. Moreover, verification systems could be implemented on a small scale. This project aims to find some features from the datasets to construct a classifier for verification purposes.

The rest of this paper is organized as follows: Section 2 provides the relevant works and researches. Then, the classification method based on machine learning and deep learning is presented in Section 3. Also a brief survey on time series classification and Deep learning is provided in Appendix B. The results and discussion are described in Sections 4 and 5.

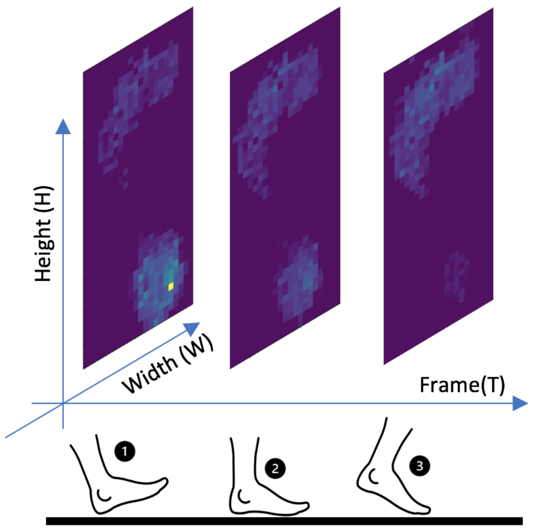


Figure 1: Different frames of footprint video in Stepscan dataset.

Furthermore, this research will be implemented in Python, and the source codes are available on the GitHub repository [5].

2. Literature Review

DARPA, the Defense Advanced Research Projects Agency of the USA, started to research gait recognition by vision data in the early 2000s [2]. Besides vision data [6], some studies have instead used accelerometry from smartphones [7], audio [8], and underfoot pressures data [9].

Addlesee in [10] used a new sensor (Active floor) for the first investigations into footprint recognition. This sensor was a square carpet tile maintained at the corners by some load cells and supplied the Ground Reaction Forces (GRFs). Orr and Abowd [11] extracted ten temporal features from the GRFs curve.

Moustakidis et al. [12] extracted temporal features from the wavelet decomposition of GRFs and then applied a kernel-based support vector machine. These studies have limitations in terms of the small sample sizes used for classification (e.g., 15 [11], 10 [12], and 15 [13]), and moderate classification rates ($CR < 90\%$).

Pataky in [14] achieved a 99.6% classification rate in a 104-participant dataset. This result was based on spatial alignment and automated dimensionality reduction. He used a template image that was made in [15]. Pataky named this template the Munster-104 template.

In 2015, Cantoral-Ceballos [16] introduced an intelligent carpet system. This carpet system (iMAGiMAT) worked based on the deformation of 116 distributed plastic optical fibers (POFs). So that applying pressure to this system would change the intensity of the transmitted light. Thus the nature of the output of this sensor is time-series data.

Costilla-Reyes et al. [17] extracted five features directly from raw data of the iMAGiMAT sensor. These features were spatial Average (SA), standard deviation (SD), adjacent mean (AM), cumulative sum (CS), and cumulative product (CP). They implemented 14 various machine learning methods for classification. The best result belonged to the Random Forest model with a validation score of $90.84 \pm 2.46\%$.

Costilla-Reyes et al. in [18] used an end-to-end convolutional neural network to extract Spatio-temporal features automatically. This technique increased F-score performance to about 97.88% $\pm$ 1.7%. They reconstructed pressure images from the time-series data to feed to deep networks. Some papers used other techniques like Continuous Wavelet Transform (CWT) [19], Recurrence Plots (RP) [20], and short-time Fourier transform (STFT) [21] to produce a 2D representation of time-series. Therefore, time series classification can change to a texture image recognition task.

3. Methodology

3.1. Image to Time-Series Encoding

The nature of data in the Stepscan dataset is a video-based dataset. There are several methods for converting a tensor (like video) to 2D time-series data.

Chen et al. in [6] used contour width for defining a one-dimensional signal. They utilized some morphological operations on the background-subtracted silhouette image to extract the outer contour. Afterwards, according to the contour width of each image row, a one-dimensional signal was generated.

Another method could be that the pixel values in each frame are plotted over frame number. By this means, the $H * W$ time-series will be produced for each sample.

In the final method, some spatial features are extracted from each frame (e.g. centroid and maximum pressure in each frame). Afterwards, we track these values over time (next frames). As a result, 3D videos with size $T \times H \times W$ are converted to the four 2D time-series data. Costilla-Reyes et al. utilized this method to combine the output of 160 distributed POFs [18].

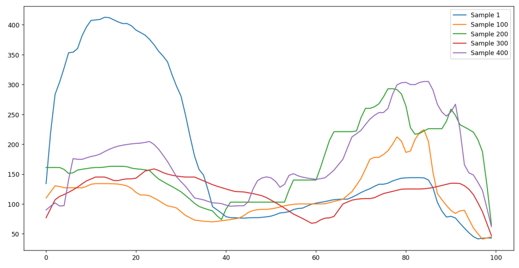
In this research, the last mentioned method was applied to produce time-series data. Figure 2 depicts the time series extracted from the Stepscan dataset. The values of maximum pressure (figure 2a), the center of pressure (COP) (figures 2d and 2c), and the average pressure (figure 2b) have been tracked over each frame to produce these time series.

3.2. Conventional Classifiers

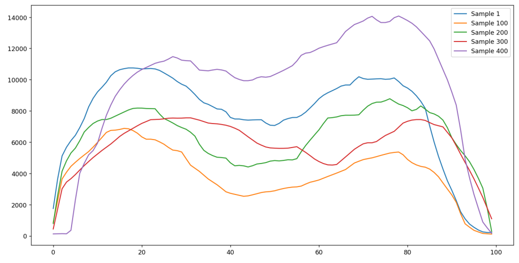
This method consists of feature extraction, feature selection, and classification, as shown in Figure 3. In the following, each step would be described.

3.2.1. Features Extraction and Selection

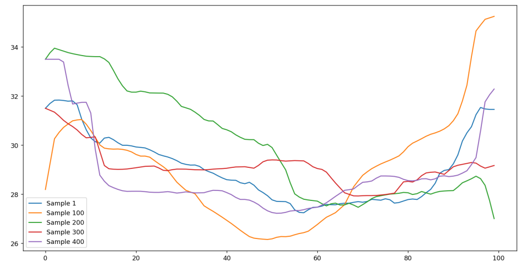
As mentioned before, our goal is to develop a classification model. For this classification task, we use about 32 features in four categories. These feature sets are explained briefly here, and more details about them can be found in Appendix A.



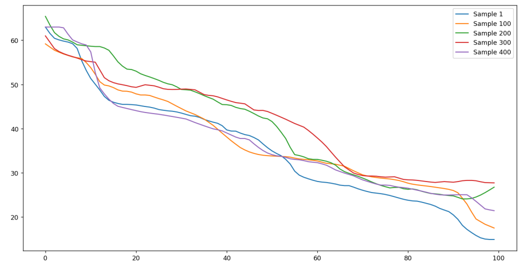
(a) The maximum pressure in each frame



(b) The average pressure in each frame



(c) The x position in the center of pressure (COP) in each frame



(d) The y position in the center of pressure (COP) in each frame

Figure 2: The time series extracted from the Stepscan dataset based on four spatial features. The horizontal axis indicates the frame number.

After extracting features from each time series, some low variance and high-correlated features were eliminated. Feature selection causes the complexity of the model to reduce. Ten percent of data were set aside for testing our classifier, and others were divided into 10-fold cross-validation for evaluation and training.

To have an equally balanced class population in the test set and cross-validation set, the Stratified and StratifiedKFold methods were used. As a result, each set includes approximately the same percentage of samples of each class as the complete set.

Temporal Features; The first set of features extracted was temporal features. In this set, we focused on features that related to the time axis. Features like Entropy, Absolute energy, Centroid, Area under the curve fall into this group. The number of features extracted was 10.

Statistical Features; Statistical information was another feature set that was extracted from the dataset. Min, Max, variance, and standard deviation were some of the statistical features. The total number of features in this group is about 9 for each time series.

Spectral Features; In the third category, both FFT and wavelet transform were used to extract spectral information from the dataset. Not only the time complexity but also the number of features were more than two other feature sets. Max power spectrum, Maximum frequency, Spectral centroid, Wavelet energy, and FFT mean coefficient were spectral features extracted from the dataset.

Autoregressive model (AR) coefficients; The final set of features in this research was the coefficients of AR. In this set, the first-order differencing used to make our data stationary. Then based on significant lag on Partial AutoCorrelation function (PACF), the order of the model was selected. This approach extracted two features for each time-series signal.

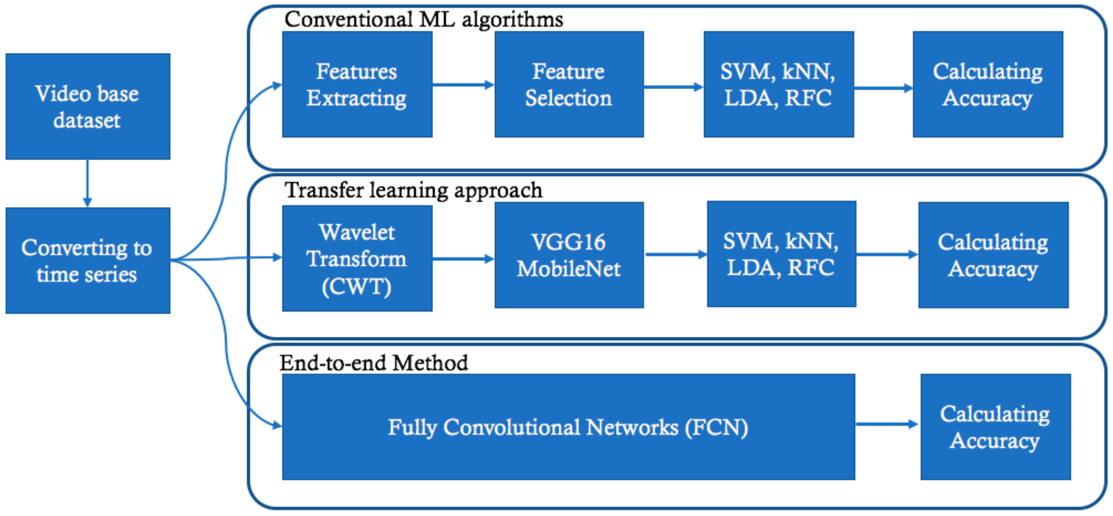


Figure 3: Flowchart of implemented method.

3.2.2. Machine learning algorithms

These time-series features were fed to four different types of machine learning models with tuned hyperparameters. The hyperparameters tuning was implemented by grid search and performance evaluation with 10-StratifiedKFold cross-validation. Table 1 shows these models along with their best-tuned hyper-parameters.

3.3. Deep Learning Approaches

As Figure 3 shows, two approaches of Deep Neural Network were implemented. In this subsection, these approaches would be reviewed.

3.3.1. Transfer Learning Approach

For this project, two famous CNN architectures were implemented, including VGG16 [22] and MobileNet [23]. These architectures ensure complete feature extraction by automatically generating features from the raw data. Due to the small size of the dataset, it is impossible to train networks on our dataset. As a result, pre-trained networks were downloaded from http://image-net.org/. Then, the top layer of these networks was replaced with four machine learning algorithms such as Random Forests classifier, LDA, SVM, and KNN. A major limitation of the pre-trained networks is that the data are required to be of the type of image, while our data are time series. As a result, the 2D-scalogram of each time-series signal was calculated and fed to CNNs.

3.3.2. Fully Convolutional Networks

Another model which was implemented in this project is a Fully Convolutional Network (FCN) [24]. This model is an end-to-end convolutional network with three blocks. Each block has a convolutional layer followed by batch normalization and ReLU activation layer. The output of the third block is averaged over the time dimension, which corresponds to the global average pooling (GAP) layer. Finally, a softmax layer is fully connected to the GAP layer's output to produce the final label.

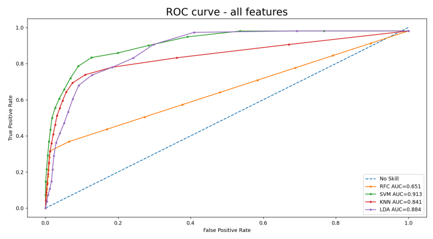
4. Experimental Results

The biometrics industry has utilized two performance measurements, False Rejection Rate (FRR) and False Acceptance Rate (FAR), for accuracy. These measures also are called False Negative Rate (FNR) and False Positive Rate (FPR) in biometric literature. According to these values, a graphical ROC curve was produced by plotting FPR on the x-axis against 1–FNR on the y-axis. Figure 4 shows the ROC curves for different models and features.

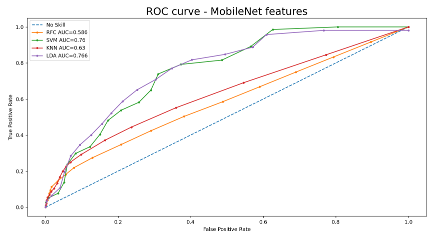
The area under the ROC curve (AUC) shows the performance of verification systems. An AUC of 0.5 represents a system with no discriminating ability (The blue line in figure 4), whereas an AUC of 1.0 represents a test with perfect discrimination. Moreover, many researchers often use equal error rate (EER) as an additional metric to describe the performance of biometric systems. EER refers to the point that FRR equals FAR. The smaller value of EER shows the better performance. Results of four different machine learning algorithms in several types of features are shown in table 1.

For the training scenario, the first 50 users were used for developing models while other users set aside as imposter data. Then for each subject, a binary classifier was trained based on 80 percent data (about 16 in-class samples and $16 \times 49 \approx 780$ samples from impostor class). For testing, a set of four genuine samples and $4 \times 49 \approx 190$ from impostor samples were used. Due to the fact that we developed a model for each subject, the average of FPR and FNR was calculated for each method.

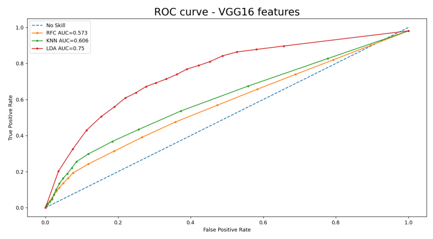
The SVM classifier's best AUC was 91.3%, as shown in table 1. This result was achieved by classifying the combination of all features and the best AUC among all approaches. Other algorithms had similar performance on all features. All kinds of features had the worst results on Random Forests Classifiers. In terms of the EER metric, the SVM classifier achieved the smallest value (about 0.147) on all handcrafted features.



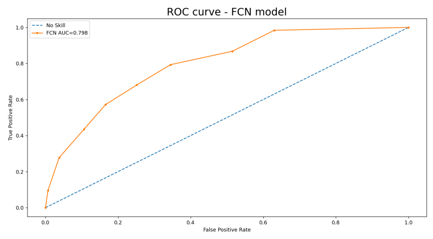
(a) The ROC curve of four Machine learning algorithms on handcrafted features



(b) The ROC curve of four Machine learning algorithms on MobileNet features



(c) The ROC curve of four Machine learning algorithms on VGG16 features



(d) The ROC curve of Fully Convolutional Network (FCN).

Figure 4: The ROC curve of different approaches

Table 1: The EER and AUC metric for Machine learning models implemented in this paper.

	Frame Work	AUC	EER
0	LDA_ALL	0.884	0.202
1	LDA_SPEC	0.870	0.193
2	LDA_STAT	0.877	0.206
3	LDA_TEMP	0.881	0.215
4	LDA_VGG	0.750	0.306
5	LDA_MOBNET	0.766	0.298
6	KNN_ALL	0.841	0.202
7	KNN_SPEC	0.811	0.228
8	KNN_STAT	0.822	0.206
9	KNN_TEMP	0.819	0.222
10	KNN_VGG	0.606	0.418
11	KNN_MOBNET	0.630	0.404
12	SVM_ALL	0.913	0.147
13	SVM_SPEC	0.606	0.412
14	SVM_STAT	0.741	0.306
15	SVM_TEMP	0.883	0.203
16	SVM_MOBNET	0.760	0.286
17	RFC_ALL	0.651	0.402
18	RFC_SPEC	0.598	0.441
19	RFC_STAT	0.565	0.465
20	RFC_TEMP	0.608	0.438
21	RFC_VGG	0.573	0.453
22	RFC_MOBNET	0.586	0.452
23	FCN	0.798	0.285

5. Discussion Progress

The results indicate that spectral features are more powerful features for all machine learning algorithms. The reason behind this might be the number of features and their qualities in this set. Based on table A.2, there are about 300 features for this set, much more than for the other two.

Another feature that has a significant effect on the accuracy is inter-stride distance. This feature has been added only to the all feature set, and it is one of the powerful features. Furthermore, other inter-stride features like the angle used to align the footprint with the template and stride duration should be considered in the feature set to improve the discrimination power of classifiers.

Unbalance dataset also could have a negative effect on the results. For example in the Stepscan dataset, there is about 800 negative class whereas 16 samples associated with the positive class. Consequently, the models were biased towards the negative class.

The poor results of the transfer learning framework might be because only machine learning algorithms were trained based on the Stepscan dataset. It could have a better performance if we train the last two-layer of CNNs.

References

[1] J. P. Singh, S. Jain, S. Arora, U. P. Singh, A Survey of Behavioral Biometric Gait Recognition: Current Success and Future Perspectives, Archives of Computational Methods in Engineering (2019). doi:10.1007/s11831-019-09375-3.

[2] P. Connor, A. Ross, Biometric recognition by gait: A survey of modalities and features, Computer Vision and Image Understanding 167 (2018) 1–27. doi:10.1016/j.cviu.2018.01.007.

[3] P. C. Connor, Comparing and combining underfoot pressure features for shod and unshod gait biometrics, in: 2015 IEEE International Symposium on Technologies for Homeland Security, HST 2015, Institute of Electrical and Electronics Engineers Inc., 2015. doi:10.1109/THS.2015.7225338.

[4] A. K. Jain, A. Ross, S. Prabhakar, An Introduction to Biometric Recognition, IEEE Transactions on Circuits and Systems for Video Technology 14 (1) (2004) 4–20. doi:10.1109/TCSVT.2003.818349.

[5] SKazemii/EE6563.
URL https://github.com/SKazemii/EE6563

[6] C. Chen, J. Liang, H. Zhao, H. Hu, Gait recognition using Hidden Markov model,

Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 4221 LNCS (60402038) (2006) 399–407. doi:10.1007/11881070{_}56.

[7] J. Mäntyjärvi, M. Lindholm, E. Vildjiounaite, S. M. Mäkelä, H. Ailisto, Identifying users of portable devices from gait pattern with accelerometers, in: ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, Vol. II, 2005. doi:10.1109/ICASSP.2005.1415569.

[8] J. T. Geiger, M. Hofmann, B. Schuller, G. Rigoll, Gait-based person identification by spectral, cepstral and energy-related audio features, in: ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2013, pp. 458–462. doi:10.1109/ICASSP.2013.6637689.

[9] K. Nakajima, Y. Mizukami, K. Tanaka, T. Tamura, Footprint-Based Personal Recognition, Tech. Rep. 11 (2000).

[10] M. D. Addlesee, A. Jones, F. Livesey, F. Samaria, The ORL active floor, IEEE Personal Communications 4 (5) (1997) 35–41. doi:10.1109/98.626980.

[11] R. J. Orr, G. D. Abowd, The Smart Floor: A Mechanism for Natural User Identification and Tracking, Tech. rep. (2000).

[12] S. P. Moustakidis, J. B. Theocharis, G. Giakas, Subject recognition based on ground reaction force measurements of gait signals, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 38 (6) (2008) 1476–1485. doi:10.1109/TSMCB.2008.927722.

[13] L. Middleton, A. A. Buss, A. Bazin, M. S. Nixon, A floor sensor system for gait recognition, Tech. rep.
URL www.tekscan.com

[14] T. C. Pataky, T. Mu, K. Bosch, D. Rosenbaum, J. Y. Goulermas, Gait recognition: Highly unique dynamic plantar pressure patterns among 104 individuals, Journal of the Royal Society Interface 9 (69) (2012) 790–800. doi:10.1098/rsif.2011.0430.

[15] T. C. Pataky, K. Bosch, T. Mu, N. L. Keijsers, V. Segers, D. Rosenbaum, J. Y. Goulermas, An anatomically unbiased foot template for inter-subject plantar pressure evaluation, Gait and Posture 33 (3) (2011) 418–422. doi:10.1016/j.gaitpost.2010.12.015.

[16] J. A. Cantoral-Ceballos, N. Nurgiyatna, P. Wright, J. Vaughan, C. Brown-Wilson, P. J. Scully, K. B. Ozanyan, J. A. Cantoral-Ceballos, N. Nurgiyatna, P. Wright, K. B. Ozanyan, J. Vaughan, P. J. Scully, Intelligent Carpet System, Based on Photonic Guided-Path Tomography, for Gait and Balance Monitoring in Home Environments, IEEE SENSORS JOURNAL 15 (1) (2015) 279. doi:10.1109/JSEN.2014.2341455.
URL http://ieeexplore.ieee.org.

[17] O. Costilla-Reyes, P. Scully, K. B. Ozanyan, Temporal Pattern Recognition in Gait Activities Recorded with a Footprint Imaging Sensor System, IEEE Sensors Journal 16 (24) (2016) 8815–8822. doi:10.1109/JSEN.2016.2583260.

[18] O. Costilla-Reyes, P. Scully, K. B. Ozanyan, Deep Neural Networks for Learning Spatio-Temporal Features From Tomography Sensors, IEEE Transactions on Industrial Electronics 65 (1) (2018) 645–653. doi:10.1109/TIE.2017.2716907.

[19] T. Wang, C. Lu, Y. Sun, M. Yang, C. Liu, C. Ou, Automatic ECG classification using continuous wavelet transform and convolutional neural network, Entropy 23 (1) (2021) 1–13. doi:10.3390/e23010119.

[20] N. Hatami, Y. Gavet, J. Debayle, Classification of Time-Series Images Using Deep

Convolutional Neural Networks, Tech. rep. (2017).

[21] J. Huang, B. Chen, B. Yao, W. He, ECG Arrhythmia Classification Using STFT-Based Spectrogram and Convolutional Neural Network, IEEE Access 7 (2019) 92871–92880. doi:10.1109/ACCESS.2019.2928017.

[22] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, International Conference on Learning Representations, ICLR, 2015.
URL http://www.robots.ox.ac.uk/

[23] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv (4 2017).
URL http://arxiv.org/abs/1704.04861

[24] Z. Wang, W. Yan, T. Oates, Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline, Tech. rep.
URL https://github.com/cauchyturing/

[25] P. Esling, C. Agon, Time-series data mining, ACM Computing Surveys 45 (1) (2012). doi:10.1145/2379776.2379788.

[26] J. Gamboa, Deep Learning for Time-Series Analysis, arXiv (2017).

[27] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, Communications of the ACM 60 (6) (2017) 84–90. doi:10.1145/3065386.

[28] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P. A. Muller, F. Petitjean, InceptionTime: Finding AlexNet for time series classification, Data Mining and Knowledge Discovery 34 (6) (2020) 1936–1962. doi:10.1007/s10618-020-00710-y.
URL https://link.springer.com/article/10.1007/s10618-020-00710-y

[29] F. M. Bianchi, S. Scardapane, S. Løkse, R. Jenssen, Reservoir computing approaches for representation and classification of multivariate time series, arXiv (2018) 1–11doi:10.1109/tnnls.2020.3001377.

Appendix A. List of features

The list of features from several categories used in this project (Table A.2):

Table A.2: list of features

#	Features Name	Categories Name	# features	description
1	Histogram	Statistical	10	10-bin Histogram
2	Max	Statistical	1	max(s)
3	Mean	Statistical	1	mean(s)
4	Mean absolute deviation	Statistical	1	$\Sigma_{i=1}^{N}\|S_i^2 - mean(s)\|/N$
5	Median	Statistical	1	median(s)
6	Median abs deviation	Statistical	1	$median(\|s - median(s)\|)$
7	Min	Statistical	1	min(s)
8	Root mean square	Statistical	1	$\sqrt{\Sigma_{i=1}^{N}S_i^2/N}$
9	Standard deviation	Statistical	1	$\sqrt{var}$
10	Variance	Statistical	1	$mean(\|s - mean(s)\|^2)$
11	AR coefficients	AR	8	-
12	Absolute energy	Temporal	1	$\Sigma_{i=0}^{N}S_i^2$
13	Area under the curve	Temporal	1	$\Sigma_{i=0}^{N}(t_i - t_{i-1}) \times (s_i + s_{i-1})/2$
14	Centroid	Temporal	1	$\Sigma_{i=0}^{N}(t_i \times s_i^2)/\Sigma_{i=0}^{N}s_i^2$
15	Entropy	Temporal	1	$-\Sigma P(x)log_2 P(x)$
16	Mean absolute diff	Temporal	1	$mean(\|diff(s)\|)$
17	Mean diff	Temporal	1	$mean(diff(s))$
18	Median absolute diff	Temporal	1	$median(\|diff(s)\|)$
19	Median diff	Temporal	1	$median(diff(s))$
20	Slope	Temporal	1	fitting a line and returning the slope
21	FFT mean coefficient	Spectral	256	mean(fft(s))
22	Max power spectrum	Spectral	1	-
23	Maximum frequency	Spectral	1	-
24	Median frequency	Spectral	1	-
25	Spectral centroid	Spectral	1	-
26	Spectral entropy	Spectral	1	-
27	Wavelet abs mean	Spectral	10	$\|mean(wavelet(s))\|$
28	Wavelet energy	Spectral	10	-
29	Wavelet stand deviation	Spectral	10	$\|std(wavelet(s))\|$
30	Wavelet entropy	Spectral	1	-
31	Wavelet variance	Spectral	10	$\|var(wavelet(s))\|$
32	Inter stride	Temporal	1	
	The total number of features		340	

Appendix B. Deep learning approaches for time series classification

As already mentioned Stepscan dataset, with the nature of video-based, is the dataset that we used in this research. It should be noted that this type of dataset is basically suitable for this project and there is no need to convert the image into a time-series signal; however since our goal in this research is developing machine learning models from the time-series dataset, so we turned the video dataset into a time-series one in order to implement a variety of classic machine learning algorithms and also deep learning methods, named LDA, KNN, SVM and Random forest which are categorized in machine learning algorithms and CNN, inception time and Echo State as deep learning methods. Therefore, in this appendix, we have tried to give a brief explanation of time series classification and main deep learning methods.

Appendix B.1. Introduction

For the last two decades, time series classification has been regarded as one of the most challenging problems in data mining [25]. Any classification problem which uses data taking into account some indication of order can be accounted as a time series classification (TSC) case [26]. The field of computer has been seen as a big bang, once a Deep Convolutional Neural network has been proposed [27]. These deep learning algorithms could handle feature engineering automatically and internally. It caused many applications in many different domains to utilize these networks.

Appendix B.2. Deep learning

Figure B.5 shows a general Deep Learning framework for time series classification. It is a composition of several layers that implement non-linear functions. The input could be either a multivariate time series or univariate data. Every layer takes the output of the previous layer as input, then applies its non-linear transformation to compute its own output. In this research, five different Deep Learning architectures for time series classifications are presented.

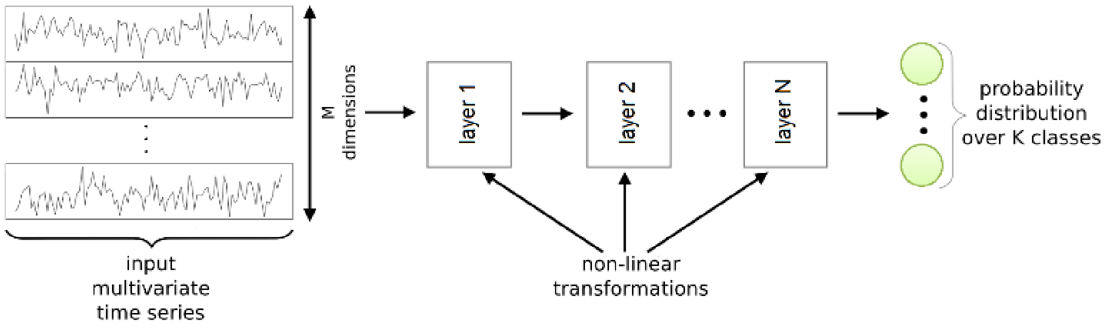

M dimensions
layer 1
layer 2
layer N
probability distribution over K classes
input multivariate time series
non-linear transformations

Figure B.5: The general Deep Learning framework for time series classification.

Appendix B.3. Convolutional Neural Networks

Since AlexNet [27] won the imageNet competition in 2012, deep CNNs have seen a lot of successful applications in many different domains such as reaching human-level performance in image recognition problems as well as different natural language processing tasks. Motivated by the success of these CNN architectures in these various domains, researchers have started adopting them for time series analysis [26].

The challenge of using this architecture is changing time series data to image data. Some papers used several techniques like Continuous Wavelet Transform (CWT) [19], Recurrence Plots (RP) [20], and short-time Fourier transform (STFT) [21] to produce a 2D representation of time-series. Therefore, time series classification can change to a texture image recognition task.

Weng et al. in [24] used a 1D convolutional operation to adapt this architecture to time series data. They tested three deep neural network architectures, MLP, FCN and Residual Network, to provide a comprehensive baseline model in time series.

Appendix B.4. Inception Time

Recently a deep Convolutional Neural Network called Inception Time was introduced by Fawaz et al. in [28]. This kind of network shows high accuracy and scalability. As shown in figure B.6, the Inception Network consists of a series of Inception Modules followed by a Global Average Pooling Layer and a Fully Connected Layer (usually a Multi-Layer Perceptron). Moreover, a residual connection is added at every third inception module. Each residual block's input is transferred via a shortcut linear connection to be added to the next block's input, thus mitigating the vanishing gradient problem by allowing a direct flow of the gradient.

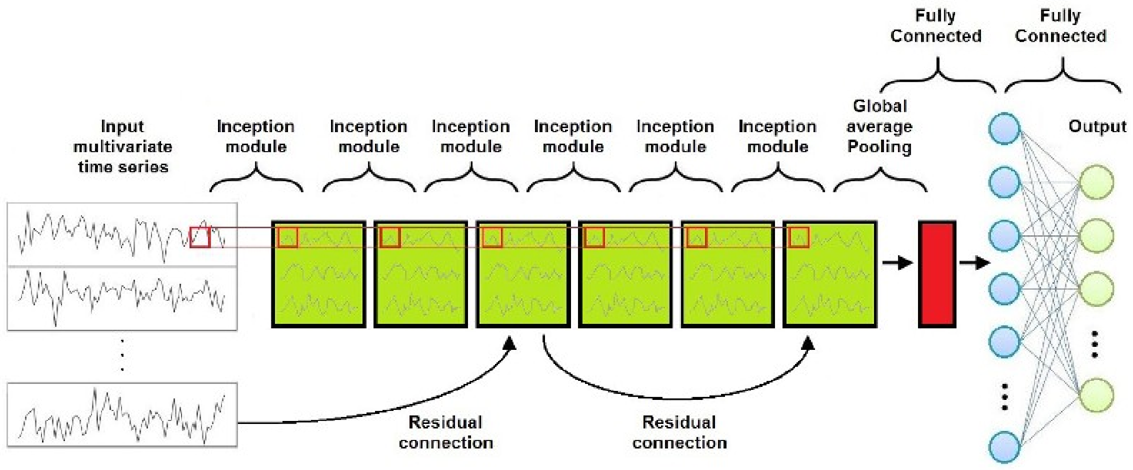


Figure B.6: Inception Network Architecture [28]

Appendix B.5. Echo State Networks

Echo State Networks [29] are a type of Recurrent Neural Network. Hence it can be useful to have a small introduction about them. Recurrent Neural Networks are networks of neuron-like nodes organized into successive layers, with an architecture similar to one of the standard Neural Networks. In fact, like in standard Neural Networks, neurons are divided into the input layer, hidden layers and output layers. Each connection between neurons has a corresponding trainable weight.

As shown in figure B.7 , the architecture of an Echo State Network consists of an Input Layer, a hidden Layer called Reservoir, a Dimension Reduction Layer, a Fully Connected Layer called Readout, and an Output Layer.

In general, the main difficulty in using CNNs is that they are very dependent on the size and quality of the training data. To solve this problem, many new algorithms were recently elaborated, and among these Inception Time and Echo State Networks perform better than the others.

Inception Time, figure B.6 is derived from Convolution Neural Networks and speeds up the training process using an efficient dimension reduction in the most important building block, the Inception Module. Echo State Networks are really helpful to handle chaotic time series. Hence, high accuracy and high scalability make these new architectures the perfect candidate for product development.

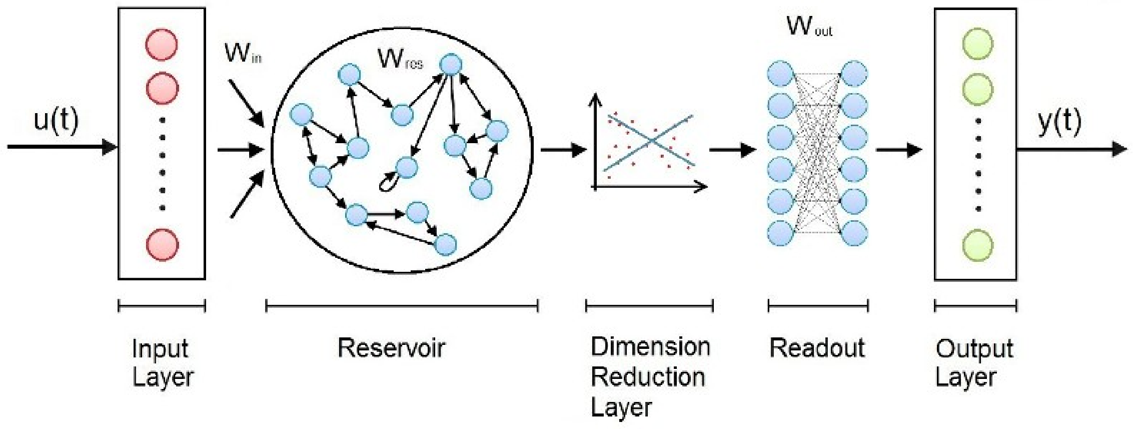


Figure B.7: Echo State Networks Architecture [29]