International Journal of Automation and Computing 16(6), December 2019, 800-811
DOI: 10.1007/s11633-019-1185-8

A Wide Learning Approach for Interpretable Feature Recommendation for 1-D Sensor Data in IoT Analytics

Snehasis Banerjee[1] Tanushyam Chattopadhyay[1] Utpal Garain[2]

[1] TCS Research & Innovation, Tata Consultancy Services, Ecospace, Kolkata 700160, India

[2] Computer Vision & Pattern Recognition Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata 700108, India

Abstract: This paper presents a state of the art machine learning-based approach for automation of a varied class of Internet of things (IoT) analytics problems targeted on 1-dimensional (1-D) sensor data. As feature recommendation is a major bottleneck for general IoT-based applications, this paper shows how this step can be successfully automated based on a Wide Learning architecture without sacrificing the decision-making accuracy, and thereby reducing the development time and the cost of hiring expensive resources for specific problems. Interpretation of meaningful features is another contribution of this research. Several data sets from different real-world applications are considered to realize the proof-of-concept. Results show that the interpretable feature recommendation techniques are quite effective for the problems at hand in terms of performance and drastic reduction in development time.

Keywords: Feature engineering, sensor data analysis, Internet of things (IoT) analytics, interpretable learning, automation.

1 Introduction

According to the Gartner report[1], there will be about 21 billion connected "Things" by 2020. This unprecedented level of connectivity mandates new ideas and innovations encompassing several domains such as e-governance, health care, transportation, retail and utilities[2]. It also enables the development of cross-domain solutions[3–5].

Development of Internet of things (IoT) applications requires domain knowledge, expertise in sensor signal processing and machine learning, knowledge about platforms and infrastructures, grasp over programming and design. This are traditionally known as the four stake holders of IoT based applications:

1) Domain expert: who understands the problem domain and can make sense of features of a model for causality analysis. An example is the mechanical engineer of a production plant with background domain as machine prognostics.

2) Signal processing (SP) expert: who can suggest suitable signal processing algorithms (such as spectrogram based feature derivation) and their algorithm level tuning parameters (such as spectrum type and window overlap form). This aids in feature listing for the next modeling stage.

3) Machine learning (ML) expert: who can perform data analysis, select apt features from many features, design models for a ML task such as classification or regression.

4) Coder or developer: who can construct a deployable solution (to be used by the end users) by integrating required modules based on inputs shared by other stakeholders.

The typical steps of the work-flow for an IoT based sensor data analytics task as pointed out in a survey[6] is shown below:

1) Domain expert explains the problem′s goal (pertaining to the use case and application area) to the SP and ML resource persons.

2) SP expert provides a list of algorithms that can be used as features (such as data transforms to make the data easy to analyse) for the given problem and data category.

3) ML expert recommends the optimal feature set based on analysis of the available dataset and their knowledge of solutions in similar problems and scenarios.

4) SP expert tunes the parameters of signal processing algorithms (such as window size, decimation for a fast Fourier transform algorithm), and the ML expert tunes the (hyper) parameters to derive a solution model.

5) Recommended feature set is presented to domain expert for validation and verification, to check if extracted features and models are sound and meaningful in the domain.

6) If current features and models are unintelligible (which is often the case), then Steps 2–5 are repeated with a change in strategy by incorporating the domain expert′s feedback.

7) If current model and features are acceptable in

Research Article
Manuscript received May 1, 2018; accepted April 19, 2019; published online July 2, 2019
Recommended by Associate Editor Hong-Nian Yu

terms of performance and soundness, final solution with finalized model is deployed by the developer or coding expert.

Now, the problem of developing such a system is that each of the stake holders speaks their own language and terms. Also, it is not possible for a single person to have this varied knowledge for diversified use cases. Also, due to the boom in IoT and data analysis space, there are more problems to solve than available human resources to solve them. As a consequence, there is a need for automation, so that the time to market and the cost of finding and hiring human resources with niche skill sets can be reduced. This paper addresses the aforementioned problems and presents a machine learning-inspired solution.

In the literature, there is a limited research effort towards automation of IoT analytics[7]. An effort[8] was made to capture the expert knowledge of a sensor signal processing professional: the algorithm which an expert would have suggested is predicted by machine and its corresponding code is obtained via some open source resources. However, the approach was still manual and served as a better way for a developer to stitch together well-known algorithms to form a workflow. An ontology based approach[9] was taken for automation of IoT Analytics by modeling the physical world and the knowledge associated with it. However, scope of automation of feature engineering was not addressed there.

In order to identify the scope of automation, a survey[6] was conducted to investigate the following three points: a) the superset of steps that are followed in IoT analytics (refer to Fig. 1); b) the pain areas of an application developer both in terms of technical and domain knowledge; c) average effort in terms of coding time spent in relevant sub tasks. The first step in a typical IoT data analytics task is data collection from various sensors. This can be done by a data aggregator by means of on-line analysis, or off-line analysis (the approach discussed in this paper). Next comes the pre-processing step that deals with removing noise and outliers, fixing missing value conditions and finally taking the data to a processable format. Subsequently, data transformations are carried out on the instances of a dataset to derive basic level features. Higher level feature extraction for a transform domain and feature selection are next carried out. Finding an appropriate machine learning-based model for the given problem by trying various hypothesis and parameter tuning is carried out next. Higher level inferences are optionally explored by applying semantics on the machine learning models. Finally, visualization helps the stakeholders to analyse results with ease. The survey reveals that the most time consuming step in an intelligent IoT analytics-based application development is the feature selection step where a suitable representation of the input sensor signal is obtained to achieve the target classification task. Moreover, the participants of the survey expressed that feature engineering requires the maximum technical as well as domain-specific knowledge. Hence, in automating IoT analytics[10], we can have a significant gain in solution time if the feature selection step can be automated.

Recent advances in deep learning algorithms[11] namely variations of deep multi-layer perceptron (MLP) and convolutional neural networks (CNN) could have been a choice for the purpose of automatic feature selection[11]. A study[12] was carried out regarding selection of going deep or shallow for a neural network-based machine learning task, mainly targeted for image datasets. But, in the case of IoT analytics, especially in the prognostics domain, there is a requirement of feature interpretability and hence, a method like CNN based representation learning is not a good choice. For images (2-dimensional (2-D) signal), the representation layer (automatically extracted features) of a Deep Network when viewed, seems to make sense in terms of edges and gradients. But, for 1-dimensional (1-D) sensor signals, no apparent understanding could be mapped by visualizing that representation layer. This paper attempts to automate the two major steps, namely feature listing and recommendation of features by retaining interpretable features. A Wide Learning[13] based architecture is followed which consists of two major modules, namely feature generation and feature selection. In the feature generation module, all the possible features which are so far proven good for different signal analysis tasks are generated at different levels. Features generated at each level are passed to the feature selection module to choose a subset of features that meets expected performance. Later, an exhaustive search is done for a limited number of features which are recommended by the feature selection module.

Issues addressed in this work are enlisted below:

1) It takes a huge time to come up with an IoT analytics solution, especially in the feature engineering stage. This costs a significant research and development effort.

2) More IoT analytics problems exist than the number of trained analysts and domain experts to tackle them.

3) Existing automatic methods of feature engineering (like transforms or deep learning) lose feature interpretability, which in the prognostics domain is usually unacceptable.

The main contributions of this work are shown below:

1) A Wide Learning-based approach is taken to automate the feature engineering aspect of data analysis for 1-

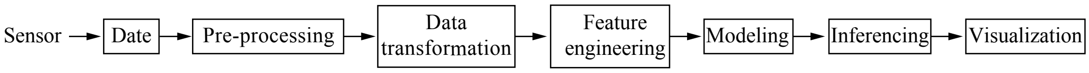

Fig. 1 A typical IoT analytics workflow most suited for one-dimensional sensor data

D data.

2) A comparison of predictive accuracy with the state of the art (SoA) on standard datasets of multiple domains is provided to show the superiority of the proposed approach.

3) Experimental results show that the proposed method can produce significant gain in reducing projected human effort, without sacrificing the solution′s predictive accuracy.

4) A selection of interpretable features and its usefulness in the domain is shown through a real-world problem.

Innovations pertaining to the work are highlighted below:

1) A Wide Learning approach-based feature recommendation methodology for 1-D signals was built by investigating ways to generalize automation of feature engineering that can cater to several use cases and domain.

2) A large number of features found in the literature of a domain were curated and a master list was compiled. This enabled more accurate feature selection based on knowledge gathered from domain and application areas.

3) The design of feature interpretation module has enabled seamless description of features irrespective of specific data windows where their signatures were manifested.

The proposed approach is tested on different data sets in the domain of health care, psychology and machine prognostics. Five different case studies have been reported in this paper as follows: a) classification of machine bearing accelerometer data that is used in manufacturing domain; b) classification of human activity based on accelerometer data for healthy living domain; c) classification of coronary related disease on photoplethysmogram (PPG)[14] dataset for healthcare; d) classification of phonocardiogram (PCG)[15] heart rate data that is used in a healthcare challenge; e) classification of human emotion based on photoplethysmogram readings for psychology domain.

The rest of the paper is organized as follows. Section 2 presents the method of generating and selection of features through a suitable realization of a Wide Learning architecture. Section 3 summarizes the real-world datasets that were used to evaluate the proposed approach. Experimental setup and results are also presented in this section. Section 4 illustrates the usefulness of interpretable features with an example while Section 5 concludes the paper.

2 Method description

The Wide Learning system as shown in Fig. 2 accepts a set of annotated input signal data. The signal data undergoes standard pre-processing steps like outlier removal and noise cleaning. Automation of the pre-processing step is kept out of the current scope of work due it its huge dependence on application and domain demanding years of research. Data is formatted into a standard matrix format with corresponding labels. Next, data is partitioned into Train, Eval (Dev) and Test in multiple folds (usually five folds in splits of 60%, 20% and 20%, respectively). The system is programmed to automatically determine the number of folds depending on the number of data instances available, based on threshold values and splitting logic. The partitioning of data takes place following Train-Eval-Test principles[16] in folds, with each partition retaining data characteristics. This is achieved by clustering the data and assigning equal portions (if possible) of cluster members to Train, Eval, Test as per their percentage of total data splitting in folds. The ideal number of clusters were determined based on Silhouette coefficient[17], a cluster quality metric. Possessing similar

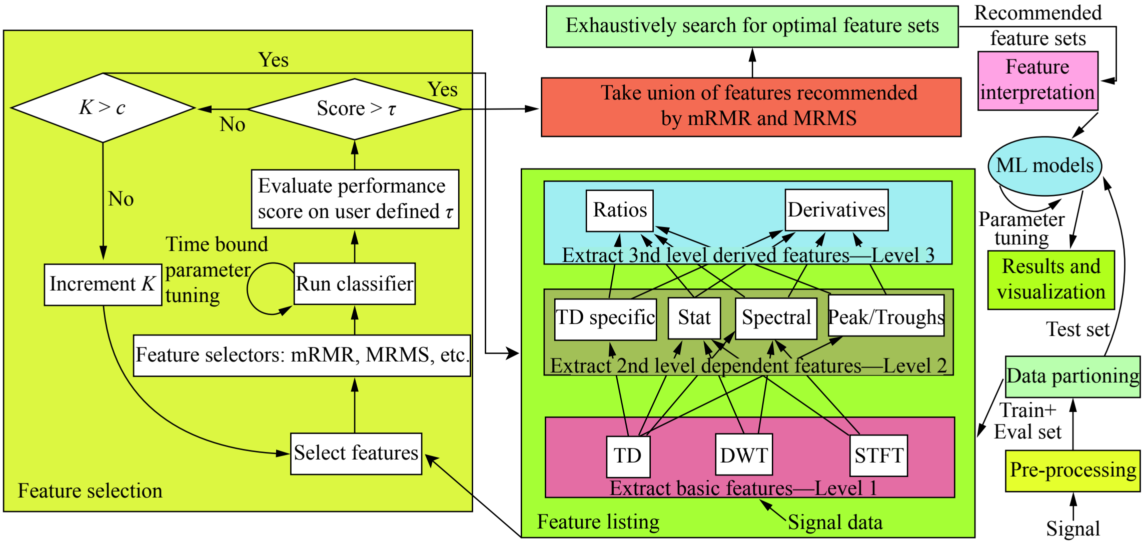
Exhaustively search for optimal feature sets
Recommended feature sets
Feature interpretation
Take union of features recommended by mRMR and MRMS
Yes
$K > c$
No
Score > τ
Yes
No
Evaluate performance score on user defined τ
Time bound parameter tuning
Increment K
Run classifier
Feature selectors: mRMR, MRMS, etc.
Select features
Feature selection
ML models
Parameter tuning
Results and visualization
Test set
Data partioning
Train+ Eval set
Pre-processing
Signal
Ratios
Derivatives
Extract 3nd level derived features—Level 3
TD specific
Stat
Spectral
Peak/Troughs
Extract 2nd level dependent features—Level 2
TD
DWT
STFT
Extract basic features—Level 1
Feature listing
Signal data

Fig. 2 Wide Learning system for 1-D sensor signals

data attributes in Train is important as unknown data signatures in Test will confuse the model, resulting in random predictions. The performance (say accuracy) is reported on the hidden Test set, while the rest is used for feature recommendation. The Train data is passed to extract the features at various levels of feature extraction. The "Eval" set is used for classifier-in loop evaluation (wrapper method of feature selection) on obtained features derived from the Train set. The classifiers used are an ensemble of Random Forest and linear and Gaussian kernels for support vector machine (SVM) with time bounded parameter tuning[18, 19]. The intuition is that even using under-tuned models, good features reveal themselves. The general principles[16] integrated in the system to carry out a machine learning task are 1) if results on Train are not good, then it means a different architecture or algorithm type needs to be tried out; 2) next, if the results obtained on "Eval" set are not good, this usually means more training data is required and more effort should be spent in regularization of the model; 3) next, if results on Test data are not good that usually means that more data representative of Test data needs to form a part of "Eval" data. The aforementioned rules of thumb[20] were automated by assigning threshold values to each stage of the operation. In future, an effort will be made to bring in a mathematical model of the overall process.

Basic features reported in literature of sensor data analytics can be mainly classified in three types: a) time domain features (TD); b) fourier transformation based features (STFT); c) discrete wavelet transformation based features (DWT). So, at Level 1, basic features are extracted and passed on to feature selection module. DWT requires input of a mother wavelet type[1] as one of its parameters, but automated mother wavelet identification is a challenging problem. The appropriate mother wavelet for carrying out the wavelet transform is selected by comparing the input signal with a library of mother wavelets in terms of having maximum energy to entropy ratio. As the purpose of a feature is to distinguish between two or more groups, so an alternative distance based approach, which is less error-prone, is also applied. Here, each mother wavelet′s energy to entropy ratio is ranked and the one that has maximum distance to the set of training classes are added as a feature. In Level 2, spectral, statistical, time domain-based and peak-trough features are extracted. Spectral features used include centroid, crest factor, decrease, flatness, flux, kurtosis, roll-off, skewness, slope and spread which are computed for each window of mean subtracted signal. Statistical features used include mean, variance, standard deviation, root mean square, skewness, kurtosis. Peak-trough features used include average peak amplitude, average trough amplitude, average peak-to-peak distance and average trough-to-trough distance. Level 3 features includes different meaningful ratios and derivatives of the Level 2 features. Feature subsets are selected by iteratively applying a combination of two powerful feature selection techniques in the wrapper approach of feature selection, namely mRMR[21] (minimum redundancy and maximum relevance) and MRMS[22] (maximal relevance maximum significance). They cover different aspects of feature selection. For instance, mRMR is classifier-independent whereas MRMS is effective in reducing real-valued noisy features which are likely to occur in sensor data. Other feature selection techniques[23, 24] were investigated, but this combination has been empirically found the most effective for 1-D signal analysis. We briefly introduce mRMR and MRMS before presenting our overall feature recommendation scheme in details.

mRMR: In order to select effective features, mRMR optimizes an objective function, either mutual information difference (MID) or mutual information quotient (MIQ), by minimizing the redundancy and maximizing the relevance of the features. MID (additive) and MIQ (multiplicative) are defined as follows:

$$MID = \max(V - W)$$

$$MIQ = \max\left(\frac{V}{W}\right)$$

where V minimizes redundancy by computing F-statistics and W maximizes relevance by computing correlation between a pair of features.

MRMS: This technique uses fuzzy-rough set selection criteria to select relevant and non-redundant (significant) features. The objective function is defined as follows:

$$J = J_{rel} + \beta J_{sig}$$

where J_{rel} computes relevance of a recommended feature with respect to a class label and J_{sig} computes the significance of a pair of recommended features by computing their correlation, and β is the weight parameter.

The system by design is open to add more feature selectors as per need, as the top "k" (from the union of features ranked by feature selectors) are taken to next step. The scope "k" is kept large initially to include good as well as moderate features.

Let, x and y be the sets of features recommended by mRMR and MRMS, respectively. Then the final recommended set of features is z, where $z = x \cup y$, where $|z| = k$.

The system finds two feature sets of cardinality "k" for a particular performance metric (such as one of accuracy, sensitivity, specificity): a) Fe1 – that produces the highest performance in any fold of cross-validation. This means a feature having excellent performance in one fold

[1] Various wavelet family listing: http://in.mathworks.com/help/wavelet/gs/introduction-to-the-wavelet-families.html

and bad performance in rest will be included here. b) Fe2 – that is most consistent and performs well across all folds. This is derived by taking the average performance value of a feature across different folds. The above steps of feature selection is done hierarchically – if Layer 1 does not produce expected results set by a user defined preset threshold τ or maximum possible goodness value of a selected metric (say 1.0 on a scale of 0 to 1 for a metric such as accuracy), then Layer 2 (higher level features) is invoked, etc. In this case, "c" is a regularizer for "k" to limit the exhaustive search, and is dependent proportionately on the hardware capabilities of the experimentation system. For the current system configuration described in Section 3.3, "c" was set to 30 to yield results in a few hours for the given datasets. Post feature selection, an exhaustive search is done on the finalized "k" features from "Fe1" and "Fe2" to find the ideal feature combination (best among $2^k - 1$ feature subsets) for the task. It has been shown in the literature that without applying brute-force, apt feature combination cannot be arrived with certainty. This selected feature recommendation set "f" is passed to a machine learning model that can be any standard classifier like artificial neural network (ANN), SVM, random forest. Parameter tuning by a combination of grid search and random search is carried out to derive results on the hidden Test set. The working principle of the method is shown in Fig. 3. There is provision to add manually obtained features "m" in case there is strong domain-based confidence behind them. This addition can be done at two places:

1) At the time of the exhaustive search, so that "f" is derived by ranking $(z \cup m)$ features, subject to $|z \cup m| < c$.

2) At the time of final modeling, so that the model is built on $(f \cup m)$ features. Addition at second case is not recommended as weight should be given to data analysis of earlier step, so that manually added "m" features can compete fairly with automatically recommended "f" features.

3 Experiments

3.1 Dataset

The experiments are performed on five one-dimension-

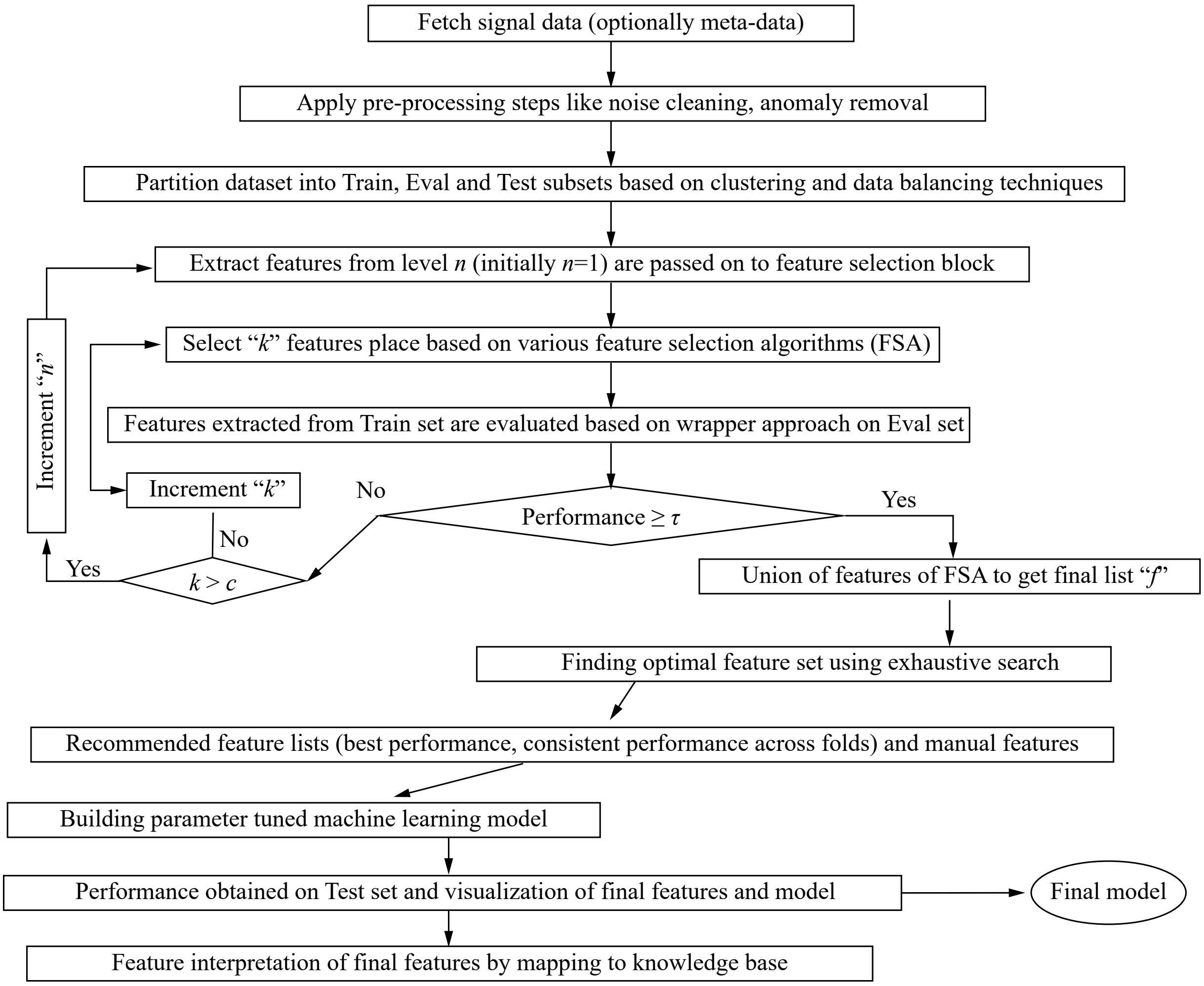
Fetch signal data (optionally meta-data)
Apply pre-processing steps like noise cleaning, anomaly removal
Partition dataset into Train, Eval and Test subsets based on clustering and data balancing techniques
Extract features from level n (initially n=1) are passed on to feature selection block
Select "k" features place based on various feature selection algorithms (FSA)
Features extracted from Train set are evaluated based on wrapper approach on Eval set
Increment "n"
Increment "k"
No
Performance ≥ τ
Yes
No
Yes
k > c
Union of features of FSA to get final list "f"
Finding optimal feature set using exhaustive search
Recommended feature lists (best performance, consistent performance across folds) and manual features
Building parameter tuned machine learning model
Performance obtained on Test set and visualization of final features and model
Final model
Feature interpretation of final features by mapping to knowledge base

Fig. 3 Method of automated feature engineering for 1-D sensor signals

Table 1 Description of data sets used for experiments

Dataset (D)	Total number of instances	Class-0 number of instances	Class-1 number of instances	Number of samples	Sampling rate (Hz)	Time window size$^{@}$(s)
D1.1: NASA all	3932	282	3650	20480	20000	0.5
D1.2: NASA subset	647	282	365	20480	20000	0.5
D2: Mobifall	258	132	126	230	50	1
D3: MIMIC CAD	99	56	43	15000	125	12
D4: PhysioNet	3153	665	2488	10612*	1000	1
D5: Emotion	66	33	33	12000	60	10

* The number of samples per data instance varied in the range of 10612 to 71332, hence, truncated for uniformity

$^{@}$ Time window size is the recommended size as per SoA. It serves as the starting point when trying out various window sizes.

al (1-D) sensor signal datasets among which four are openly available and one is private, the specification being tabulated in Table 1 and described as follows:

1) D1.1 and D1.2: The National Aeronautics and Space Administration (NASA) Bearing[2] data set contains 4 bearing data instances each having 984 records. The first bearing fails after the 700th record among the total 984 recorded readings. The last two readings are not considered due to the presence of missing values. So, we get 282 "bad bearing" (class 0) records as ground truth for a class, while the remaining 700 of the first bearing and 982 values each from the rest 3 bearings that do operate without failure form the "good bearing" class 1. To handle data unbalancing and see its effects, we have used two data-sets: D1.1: that contains the full dataset instances, D1.2: that comprise a randomly selected equally numbered small subset of the "good bearing" instances along with all the "bad bearing" instances. We have restricted ourselves to binary classification tasks to get comparable results. State-of-art (SoA)[25] for classification on this dataset, uses an SVM and Markov Model-based approach.

2) D2: The Mobifall[3] data set is a popular fall detection data-set created by volunteers aged 22–47 years. Although the data-set contains various levels of activities, however the data-set was partitioned into "fall" (class 0) and "not fall" (class 1), in order to restrict the task to binary classification. The state of the art work[8, 26] achieved an accuracy of 0.97619. It was based on an adaptive threshold-based method that used physics-based filtering of sensor data.

3) D3: The multiparameter intelligent monitoring in intensive care (MIMIC)[27] PPG-based coronary artery disease (CAD) classification data set is prepared from the waveform dataset of the MIMIC II[4] dataset, after validation of patient records with ICD-9 codes of disease classification. Finally, CAD (class 0) and non-CAD or healthy (class 1) ground-truth is derived. State of the art (SoA)[28] for this work applied a 0.5–10 Hz filter on the input signal and used several time domain features to build a SVM model.

4) D4: The PhysioNet 2016 challenge[5] dataset comprises of PCG signals with the task of abnormal heart sound classification. The ground truth label (normal or abnormal heart sound) of each record is manually annotated by expert doctors. Raw PCG is further down sampled to 1 kHz from 2 kHz, and the hidden semi-Markov model (HSMM)[29] algorithm is applied. The state of art (SoA) or winner[30] for this challenge used an ensemble of AdaBoost and CNN for classification. There was also a non-automated effort[31] from our team in this challenge, where 5 different approaches were tried out to see the efficacy. Among the alternative methods, using a hierarchical classification approach to handle the noisy signal separately, proved to be the best performing approach.

5) D5: Emotion: This dataset (used to classify the emotion into happy and sad) was generated by recording the fingertip pulse oximeter data of 33 healthy subjects (Female: 13 and Male: 20) with an average age of 27 years. Pulse oximeter was used to detect and record the PPG signal. Standard video stimuli was shown to subjects which itself served as ground-truth. The state of the art (SoA)[32] for this dataset used common time domain and frequency domain heart rate variability (HRV) features and applied SVM based classification to report accuracy.

3.2 Experiments using PCA and SVM

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to derive principal components representative of the features under consideration. This has two outcomes: 1) dimension of feature space can be reduced by selecting the most prominent principal components; 2) derived features is supposed to represent the feature space better. Experiments has been carried out with aforementioned datasets

[2] NASA Bearing set 3 at https://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/publications/#bearing

[3] Mobifall: http://www.bmi.teicrete.gr/index.php/research/mobiact

[4] MIMIC II waveform data at https://physionet.org/mimic2

[5] PhysioNet challenge at https://physionet.org/challenge/2016

and both linear and Gaussian kernels are used for SVM-based classification. The different dimension reduction techniques used are singular value decomposition (SVD), eigen value decomposition (EIG) and alternating least squares (ALS). Various numbers of principal components (in range 5 to 30) were tried to get the best performance, and for the given dataset selecting the top 5–10 components seemed to give the best results across datasets.

3.3 Experiments using deep learning

Experiments have been carried out using Theano[6] on a 8-core Intel 2.66 GHz machine having NVIDIA GTX 1080 GPU with CuDNN[7] library support. MLP, CNN and long short term memory (LSTM) were configured following standard rules of thumb and principles to obtain results on the aforementioned datasets. For hyper-parameter tuning[33] grid search and random search were tried out. Dropout[34] was added at each layer with varying range of probabilities (0.1 to 0.3) to prevent model over-fitting. Variants of stochastic gradient descent (SGD)[35] with varying learning rate (0.1 to 0.001) was used as the optimizer. Some other optimizers such as AdaGrad, Adam, RMSProp were also tried, but SGD performed the best in the current scenario. Categorical cross-entropy as the objective function for calculating loss was applied for model fitting based on metric accuracy. Different epochs (5 to 50) have been tried to see how the weight and bias update rate affects performance. Different activation functions like rectified linear unit (RELU) and its variations, tanh, softmax, sigmoid, etc. has been tried out at different layer levels to get an ideal architecture for classification tasks for the given problems. The dataset was reshaped for feeding into CNN with different overlapping window sizes. The initial CNN layer with an L1 regularizer was followed by a Max-pooling layer with pool length 2 and a Flatten layer to again reshape the feature vector for passage to the next fully-connected NN layers. Simple RNN (recurrent neural network) as well as LSTM (long short term memory RNN) was run with varying overlapping sequence lengths. The range values were influenced by domain expert's knowledge for each dataset extracted from the literature.

3.4 Results and analysis

Table 2 lists the obtained result for a dataset along with the corresponding effort for each of PCA (with SVM as classifier), MLP, CNN, LSTM, state-of-art (SoA) and proposed Wide method. It shows that PCA based methods (where features are projections and not interpretable) are outperformed by the Wide method. Deep learning (DL) approaches were applied on both raw data as well as features recommended by the proposed method. It is seen that DL-based techniques fail when compared to SoA as well as the proposed Wide Learning method, probably because of fewer data instances. The two major problems with a DL-based approach for the given problems were 1) It needs a lot of data for training which is often not available for 1-D sensor signals. Moreover, the data availability is skewed where mostly data of the "good" class is available, with trace amounts of "bad"

Table 2 Comparison in terms of accuracy (PCA, MLP, CNN, LSTM, manual SoA, proposed Wide method)

Dataset (D)		PCA	MLP	CNN	LSTM$^{\$}$	MLP*	CNN*	LSTM*	SoA	Wide
D1.1. NASA prognostics		0.94	0.93	0.93	0.93	0.96	0.97	0.97	0.99	1.0
D1.2. NASA subset		0.52	0.5	0.5	0.5	0.55	0.56	0.56	0.99	1.0
D2. Mobifall		0.51	0.44	0.44	0.44	0.44	0.55	0.44	0.97	0.98
D3. MIMIC II CAD		0.55	0.56	0.56	0.56	0.56	0.56	0.56	0.85	0.878
D4. PhysioNet challenge		0.74	0.79	0.8	0.8	0.81	0.8	0.82	0.89^{W}	0.84
D5. Emotion		0.5	0.5	0.5	0.6	0.7	0.7	0.72	0.9	0.91
Approximate effort in person days to build a solution corresponding to each dataset	D1.1	2	3	5	6	4	6	7	30	1
	D1.2	2	3	5	8	4	6	9	30	1
	D2	1	4	7	9	4.2	7.2	9	90	0.2
	D3	2	4	7	10	4.5	7.5	10.5	60	0.5
	D4	3	6	10	12	8	12	14	120	2
	D5	1	3	4	4	5	6	7	60	0.3
Interpretable features		No	No	No	No	Yes	Yes	Yes	Yes	Yes

$^{\$}$ Output of CNN layers are fed to LSTM;
* Performance measured on features extracted by Wide method; W = score of winner of D4 using an ensemble of ANNs; Score of our team for dataset D4 using manually obtained features was 0.85.

[6] Theano v. 0.8.2, http://deeplearning.net/software/theano

[7] CuDNN library for NN, https://developer.nvidia.com/cudnn

class (say healthy and failing machine parts); 2) There is no way to interpret the features for causal analysis. It was observed that DL techniques classify all the test instances into one class that can be found by calculating the ratio between classes of Table 1 for NASA Bearing dataset D1.1 and D1.2. The performance for the Mobifall (D2) dataset is not at par in the case of DL that can be attributed to the low number of input vectors for training the deep models. For dataset D3, it was observed that the deep-learning approach[36] failed to work on raw data or Wide extracted features, however the proposed method outperformed the state of art with a huge reduction of time to solution. For dataset D4, the deep learning approach worked well owing to the large number of samples. But, automated feature learning was unable to beat the proposed method. For dataset D5, due to a small number of training samples, the manual and proposed method surpassed other methods. Another notable observation is that, in no instance, has classification performance on recommended features trained on a deep learning model fallen in comparison with totally automated feature learning. Hence, the proposed Wide Learning approach was found to be effective for the above varied cases with a huge reduction of development time and at par performance. It is to be noted that the person-day effort (8 working hours) estimate is based on careful monitoring and logging of effort of moderately proficient persons for the task at hand. Each of the approaches were implemented by the same equally skilled persons.

Table 3 Recommended features for “D1.1”, window = 0.5 s

Number	Feature description	
1	STFT	Frequency: 1 851.18 Hz Frequency: 1 853.18 Hz Frequency: 1 153.11 Hz Frequency: 1 837.18 Hz Frequency: 1 845.18 Hz
2	Difference of standard deviation values of windowed DWT coefficients	
3	Standard deviation of STFT coefficients	
4	DWT frequency: (harmonic) 14.49 Hz	

Table 4 Recommended features for “D1.1”, window = 1 s

No.	Feature description	
1	STFT	Frequency: 1 613.58 Hz Frequency: 1 829.59 Hz Frequency: 1 830.59 Hz Frequency: 1 837.59 Hz
2	Kurtosis of DWT coefficients	
3	Standard deviation of DWT coefficients	
4	Standard deviation of STFT coefficients	
5	Zero crossing of DWT coefficients	
6	DWT frequency: (harmonic) 14.37 Hz	

The reasons why the proposed Wide approach has beaten state of the art methods in most cases are as follows:

1) Usage of popular features: The feature database used for feature extraction was carefully curated over months by studying the usual features used by researchers for different problems and datasets targeted for 1-D signals. At stage 1 of the task, all features including irrelevant ones are extracted, however usage of a union of feature selection methods will yield the best features at the final stage for the given task.

2) Discovery of unknown features: The features extracted at stage 1 form a huge list, including features that have never been tried out for a given problem and domain in the literature. This new features and their derivatives gave the edge over the state-of-art features used.

3) Combination of feature ranking methods: Instead of relying on a single feature ranking method, a study was conducted on various datasets to investigate any generic feature ranking method suitable for most 1-D signal analytics. It was found that a combination of mRMR and MRMS methods (used in the proposed approach) yielded the same best features as reported in the state-of-art for a number of given tasks.

4) Exhaustive search on features: It is important to carry out exhaustive searches on features, as two moderate features in combination can beat a single good feature. Hence, instead of selecting just the top good features, a relatively time-consuming search on moderate-to-good features were carried out, so that the best feature combination set can be found. This exhaustive approach has been missed by the state-of-art for the classification tasks on the aforementioned datasets.

4 Feature interpretation

4.1 Feature interpretation module

Tables 3 and 4 show some of the sample feature sets obtained for the classification task in dataset D1.1 (NASA Bearing prognostics). It can be seen that the recommended features differ based on the specified window size. The window size plays a major role which is usually supplied by the domain expert (for dataset D1.1 ideal window size is 1 s as per literature). This listing of features along with ranges of values obtained for the feature type aids the domain experts to map the obtained feature values to the physical world and the problem domain, so that deeper insights can be gained. Also if the same feature is recommended irrespective of step wise variations of window size, then that feature can be considered as a robust good feature for the task. In the example below, standard deviation of STFT coefficients has been found to be a robust feature.

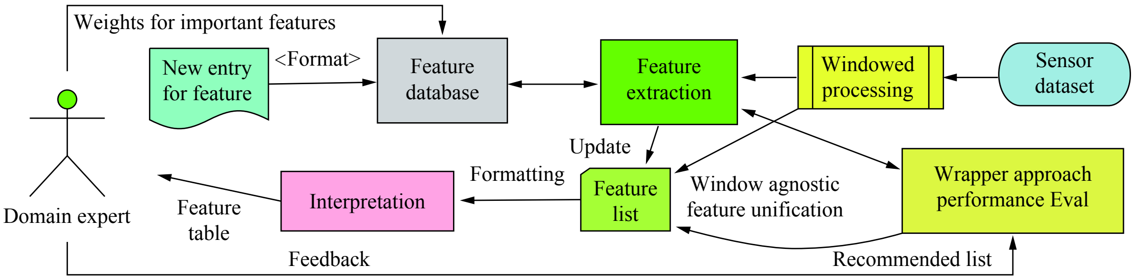

Fig. 4 Feature interpretation module

From the implementation perspective, any feature set recommendation framework would recommend only the corresponding indices of the relevant features. Such a feature identification mechanism is sufficient to trace back the recommended features from the generated feature pool. However, such a practice does not leave any room for further refinement of the recommendation through incorporation of the domain expert′s opinion. Also, when dealing with windowed processing, often the same features of different windows can get reported. So there needs to be means to identify features in different windows and compress them together instead of multiple window-wise reporting. This is true in cases of features that are not dependent on time variation. To address this issue, the proposed framework consists of a feature interpretation module as shown in Fig. 4. This module accepts the recommended feature indices as input and returns any granular information that can be obtained by analyzing its step-by-step genesis process across windows of data processing. While feature values were derived to form an input-derived feature pool, a mapping table is iteratively maintained that stores the details of the steps through which each indexed feature value is being generated. The steps of each indexed value generation would typically include information regarding the domain of transformation, transformation technique, location of the feature value in the transformed vector, and so on. This is in contrast to a hard-coded repository of feature names tagged to unique identifiers, so that new feature-extracting modules can be added and the meta-data update happens at the time of component plug-in. A format for feature extraction algorithm entry in the database is maintained, that includes algorithm description, and value ranges which can aid in interpretation later. Another feature is that domain experts can add weights to those features which seem to have a physical world connection, so that related feature space can be explored with more weightage. As an example, if domain experts tag spectral features as relevant, more algorithm level parameter tuning will be carried out on a variety of spectral features.

4.2 Physical interpretation example

Traditionally, a feature selection method is a manual effort where at step 1, domain expert(s) identifies some features using their domain expertise and experience. At step 2, the domain expert plots them for various class labels to conclude whether the features are relevant or not for a given problem. In line with that, the NASA Bearing dataset (D1.1) is selected here for interpretation analysis. Similar interpretations were also found in other data sets under consideration. The automated feature recommendation method predicted features at 14 Hz (DWT feature) harmonic space of the fundamental frequencies of the bearings rotating elements as reported below. Therefore the recommended features can be mapped to the physical world elements for further introspection and analysis by the in-loop domain expert. The bearing physics[37] as per the literature suggests fundamental frequencies as:

1) Outer race frequency = 236.4 Hz
2) Inner race frequency = 296.9 Hz
3) Rolling element frequency = 279.8 Hz
4) Shaft frequency = 33.33 Hz
5) Bearing cage frequency = 14.7 Hz.

In this case, it can be predicted that bearing fault may arise because of all possible reasons other than the problem in shaft frequency (features do not reveal that frequency as a differentiator), whereas bearing cage frequency seems to be the most causally related to failure. Hence, the reasons of failure can be suggested to the manufacturer by physical mapping of the recommended features and tallying their approximate values for future bearing defect prevention.

5 Conclusions

This paper presents a novel machine learning-based approach for efficient automation of IoT analytics. One of the most time consuming and expertise-hungry methods, namely, feature recommendation, has been automated by using a Wide Learning technique. The outcome of interpretable features is another significant achievement of this research. Six different datasets covering five real-world problems have been used to evaluate the efficiency of the approach. Experimental results show the effectiveness of the proposed feature recommendation method in terms of both performance as well as drastic reduction in time to develop a solution. The current focus of the work was for 1-D signals, future work will explore similar approaches for 2-D (image) and 3-dimensional (3-D) (video) signal processing. Integration with knowledge-bases (OWL based sensor and domain ontologies) and reasoning approaches[38] for improved interpretation is planned in future. In future, human-in-loop system for IoT applications is also planned so that sub steps for a given task in which automation does not seem to perform well, can be improved by involvement of human experts.

References

[1] N. Eddy. Gartner: 21 Billion IoT Devices to Invade by 2020. InformationWeek, 2015.

[2] P. Raj, A. C. Raman. *The Internet of Things: Enabling Technologies, Platforms, and Use Cases*, New York, USA: CRC Press, 2017.

[3] M. C. Zhou, G. Fortino, W. M. Shen, J. Mitsugi, J. Jobin, R. Bhattacharyya. Guest editorial special section on advances and applications of internet of things for smart automated systems. *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1225–1229, 2016. DOI: 10.1109/TASE.2016.2579538.

[4] K. Cao, G. Xu, J. L. Zhou, T. Q. Wei, M. S. Chen, S. Y. Hu. QoS-adaptive approximate real-time computation for mobility-aware IoT lifetime optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, to be published. DOI: 10.1109/TCAD.2018.2873239.

[5] M. Khakifirooz, C. F. Chien, Y. J. Chen. Bayesian inference for mining semiconductor manufacturing big data for yield enhancement and smart production to empower industry 4.0. *Applied Soft Computing*, vol. 68, pp. 990–999, 2018. DOI: 10.1016/j.asoc.2017.11.034.

[6] S. Banerjee, T. Chattopadhyay, A. Pal, U. Garain. Automation of feature engineering for IoT analytics. *ACM SIGBED Review*, vol. 15, no. 2, pp. 24–30, 2018. DOI: 10.1145/3231535.3231538.

[7] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, A. P. Sheth. Machine learning for internet of things data analysis: A survey. *Digital Communications and Networks*, vol. 4, no. 3, pp. 161–175, 2018. DOI: 10.1016/j.dcan.2017.10.002.

[8] D. Jaiswal, P. Datta, S. Dey, H. Paul, T. Chattopadhyay, A. Ghose, A. Singh, A. Pal, A. Mukherjee. Demo: A smart framework for IoT analytic workflow development. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, ACM, Seoul, South Korea, pp. 455–456, 2015. DOI: 10.1145/2809695.2817851.

[9] S. Dey, A. Mukherjee. Towards automation of IoT analytics: An ontology-driven approach. *Application Development and Design: Concepts, Methodologies, Tools, and Applications*, Information Resources Management Association, Ed., Hershey, USA: IGI Global, pp. 947–971, 2018. DOI: 10.4018/978-1-5225-3422-8.ch041.

[10] N. Dey, A. E. Hassanien, C. Bhatt, A. S. Ashour, S. C. Satapathy. *Internet of Things and Big Data Analytics Toward Next-generation Intelligence*, Cham, Germany: Springer, 2018. DOI: 10.1007/978-3-319-60435-0.

[11] E. J. Humphrey, J. P. Bello, Y. LeCun. Moving beyond feature design: Deep architectures and automatic feature learning in music informatics. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, ISMIR, Porto, Portugal, pp. 403–408, 2012.

[12] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, Q. L. Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, vol. 14, no. 5, pp. 503–519, 2017. DOI: 10.1007/s11633-017-1054-2.

[13] H. T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. C. Hong, V. Jain, X. B. Liu, H. Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, ACM, Boston, USA, pp. 7–10, 2016. DOI: 10.1145/2988450.2988454.

[14] M. Elgendi. On the analysis of fingertip photoplethysmogram signals. *Current Cardiology Reviews*, vol. 8, no. 1, pp. 14–25, 2012. DOI: 10.2174/157340312801215782.

[15] A. Ganguly, M. Sharma. Detection of pathological heart murmurs by feature extraction of phonocardiogram signals. *Journal of Applied and Advanced Research*, vol. 2, no. 4, pp. 200–205, 2017. DOI: 10.21839/jaar.2017.v2i4.94.

[16] A. Ng. Machine Learning Yearning. [Online], Available: http://www.mlyearning.org, 2017.

[17] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987. DOI: 10.1016/0377-0427(87)90125-7.

[18] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, vol. 46, no. 1–3, pp. 131–159, 2002. DOI: 10.1023/A:1012450327387.

[19] T. Eitrich, B. Lang. Efficient optimization of support vector machine learning parameters for unbalanced datasets. *Journal of Computational and Applied Mathematics*, vol. 196, no. 2, pp. 425–436, 2006. DOI: 10.1016/j.cam.2005.09.009.

[20] P. Domingos. A few useful things to know about machine learning. *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012. DOI: 10.1145/2347736.2347755.

[21] H. C. Peng, F. H. Long, C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005. DOI: 10.1109/TPAMI.2005.159.

[22] P. Maji, P. Garai. Fuzzy-rough MRMS method for relevant and significant attribute selection. *Advances on Computational Intelligence*, S. Greco, B. Bouchon-Meunier, G. Coletti, M. Fedrizzi, B. Matarazzo, R. R. Yager, Eds., Berlin Heidelberg, Germany: Springer, pp. 310–320, 2012. DOI: 10.1007/978-3-642-31709-5_32.

[23] J. A. Mangai, V. S. Kumar, S. A. alias Balamurugan. A novel feature selection framework for automatic web page classification. *International Journal of Automation and Computing*, vol. 9, no. 4, pp. 442–448, 2012. DOI: 10.1007/s11633-012-0665-x.

[24] D. A. A. G. Singh, S. A. A. Balamurugan, E. J. Leavline. An unsupervised feature selection algorithm with feature ranking for maximizing performance of the classifiers. *International Journal of Automation and Computing*, vol. 12, no. 5, pp. 511–517, 2015. DOI: 10.1007/s11633-014-0859-5.

[25] S. J. Dong, S. R. Yin, B. P. Tang, L. L. Chen, T. H. Luo. Bearing degradation process prediction based on the support vector machine and Markov model. *Shock and Vibration*, vol. 2014, Article number 717465, 2014. DOI: 10.1155/2014/717465.

[26] V. Chandel, A. Sinharay, N. Ahmed, A. Ghose. Exploiting

IMU sensors for IOT enabled health monitoring. In *Proceedings of the 1st Workshop on IoT-enabled Healthcare and Wellness Technologies and Systems*, ACM, Singapore, pp. 21–22, 2016. DOI: 10.1145/2933566.2933569.

[27] M. Saeed, C. Lieu, G. Raber, R. G. Mark. MIMIC II: A massive temporal ICU patient database to support research in intelligent patient monitoring. In *Proceedings of Computers in Cardiology*, IEEE, Memphis, USA, pp. 641–644, 2002. DOI: 10.1109/CIC.2002.1166854.

[28] R. Banerjee, R. Vempada, K. M. Mandana, A. D. Choudhury, A. Pal. Identifying coronary artery disease from photoplethysmogram. In *Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, ACM, Heidelberg, Germany, pp. 1084–1088, 2016. DOI: 10.1145/2968219.2972712.

[29] D. B. Springer, L. Tarassenko, G. D. Clifford. Logistic regression-HSMM-based heart sound segmentation. *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 4, pp. 822–832, 2016. DOI: 10.1109/TBME.2015.2475278.

[30] C. Potes, S. Parvaneh, A. Rahman, B. Conroy. Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds. In *Proceedings of Computing in Cardiology Conference*, IEEE, Vancouver, Canada, pp. 621–624, 2016. DOI: 10.23919/CIC.2016.7868819.

[31] R. Banerjee, S. Biswas, S. Banerjee, A. D. Choudhury, T. Chattopadhyay, A. Pal, P. Deshpande, K. M. Mandana. Time-frequency analysis of phonocardiogram for classifying heart disease. In *Proceedings of Computing in Cardiology Conference*, IEEE, Vancouver, Canada, pp. 573–576, 2016. DOI: 10.23919/CIC.2016.7868807.

[32] R. Rakshit, V. R. Reddy, P. Deshpande. Emotion detection and recognition using HRV features derived from photoplethysmogram signals. In *Proceedings of the 2nd workshop on Emotion Representations and Modelling for Companion Systems*, ACM, Tokyo, Japan, Article number 2, 2016. DOI: 10.1145/3009960.3009962.

[33] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, Cambridge, USA: MIT Press, 2016.

[34] Y. Gal, Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning*, JMLR, New York, USA, pp. 1050–1059, 2016.

[35] I. Loshchilov, F. Hutter. SGDR: Stochastic gradient descent with Warm restarts. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR, Toulon, France, vol. 10, pp. 3, 2016.

[36] Z. J. Yao, J. Bi, Y. X. Chen. Applying deep learning to individual and community health monitoring data: A survey. *International Journal of Automation and Computing*, vol. 15, no. 6, pp. 643–655, 2018. DOI: 10.1007/s11633-018-1136-9.

[37] Y. L. Li, H. R. Li, B. Wang, H. Q. Gu. Rolling element bearing performance degradation assessment using variational mode decomposition and Gath-Geva clustering time series segmentation. *International Journal of Rotating Machinery*, Article number 2598169, 2017. DOI: 10.1155/2017/2598169.

[38] D. Mukherjee, S. Banerjee, P. Misra. Towards efficient stream reasoning. In *Proceedings of OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, Springer, Graz, Austria, pp. 735–738, 2013. DOI: 10.1007/978-3-642-41033-8_97.

Snehasis Banerjee received the M. Eng. degree in software engineering from Jadavpur University, India in 2015. He is a scientist at TCS Research & Innovation. Currently, he is working on signal processing automation for IoT Analytics. He is a secretary, Association for Computing Machinery (ACM) Kolkata Professional Chapter and Treasurer, Computer Society of India (CSI) Kolkata Chapter and recognized as resource person, CSI Region-2 India East. He is India's representative to ISO Software & Systems Sectional Committee, nominated by Computer Society (CS) of India. He is also the lead of CS Pathsala program (joint initiative of TCS and ACM) to bring computing to all schools in India, sponsored by Google for Kolkata region. He has been part of technical program committee and organizing committee of major events like CSI Convention 2012, 2017, ACM Annual Event 2017, IEEE Tensymp 2019, Tata Innovista Regionals 2017, 2018. He has served as jury in many contests such as Digital Impact Square National Hackathon and ACM Kolkata B.Tech/B.E. Project Contest. He was a winner of TCS BDA Stylus Paper Contest 2019 and awarded Best Paper at Tactics Analytics Symposium 2015.

His research interests include modern aspects of the artificial intelligence field including IoT analytics and cognitive computing.

E-mail: snehasis.banerjee@tcs.com (Corresponding author)
ORCID iD: 0000-0001-6497-2085

Tanushyam Chattopadhyay is currently working as a principal scientist at TCS Research & Innovation, Kolkata to deploy and automate analytics on IoT platform. He was awarded with the University Gold medal in master of computer applications (MCA) from Indian Institute of Engineering Science and Technology (IIEST), India in 2002. He started his career as research in Indian Statistical Institute, Kolkata and later on, joined Tata Consultancy Services Limited in 2004. Later he was awarded with the Ph. D. degree from Jadavpur University for the work done at ISI, India in 2012. He had training on speech signal processing from IISC Bangalore, image processing from Indian Institute of Technology (IIT) Kharagpur, and video processing from IIT Delhi. He has 20+ granted patents across the globe. He also authored a book and some book chapters. He has published nearly 60+ papers in peer reviewed journals and conferences. He serves as a member of Board of Governor of several academic institutions. His current research is evolved around developing an analytics solution for TCS built IoT platform namely TCS Connected Universal Platform (TCUP) which involves both research and engineering in different areas of data science.

His research interests include image processing and IoT analytics.

E-mail: t.chattopadhyay@tcs.com

Utpal Garain received the B. Sc. and M. Sc. degrees in computer science and engineering from Jadavpur University, India 1994 and 1997, respectively, and the Ph. D. degree from Indian Statistical Institute, India in 2005. He is a professor in Indian Statistical Institute, India and the coordinator of the Center for Artificial Intelligence and Machine Learning (CAIML). He is one of the associate editors of *International Journal on Docu-*

ment Analysis and Recognition (IJDAR). Previously, he served as the Chair for International Association for Pattern Recognition (IAPR) Technical Committee (TC-6) on Computational Forensics for 2013-2017. He has been serving as program committee member for several international conferences including *International Conference on Pattern Recognition* (ICPR), *International Conference on Document Analysis and Recognition* (ICDAR), *International Conference on International Conference on Frontiers in Handwriting Recognition* (ICFHR), etc. Moreover, he has been regularly reviewing papers for several international journals in the field of natural language processing (NLP), vision and image analysis. For his significant contribution in pattern recognition and its applications for language engineering, he received the Young Engineer Award in 2006 from the Indian National Academy of Engineering (INAE), the prestigious Indo-US Research Fellowship (IUSSTF) in the field of Engineering Sciences in 2011 and JSPS Invitational Fellowship for Research in Osaka University, Japan in 2016.

His present research interest is focused on exploring deep learning methods for language, image, video and IoT analytics.

E-mail: utpal@isical.ac.in