Dynamic Sensor-Frontend Tuning for Resource Efficient Embedded Classification

Laura Galindez*, Komail Badami*, *Student Member, IEEE*, Jonas Vlasselaer*, Wannes Meert† , Marian Verhelst*, *Senior Member, IEEE*
*Department of Electrical Engineering, ESAT-MICAS, KU Leuven, Leuven, Belgium
†Department of Computer Science, CS-DTAI, KU Leuven, Leuven, Belgium

***Abstract*—Emerging portable applications, including Human Activity Recognition and Robot Navigation, require always-on sensing technologies to continuously monitor the environment. Continuous sensing, however, strongly dominates the hardware devices' power consumption and consequently hampers the systems' always-on functionality. In this paper we propose a circuit-aware Machine Learning scheme that exploits the devices' ability to dynamically tune the quality of its sensors to trade-off system level accuracy versus total system-level power consumption. To this end, we 1) analytically derive the power-quality trade-off space in sensory front-ends; 2) use these equations to make the probabilistic relations between sensory features and their degraded versions explicit in a Bayesian Network classifier; and 3) propose a methodology building upon this model to control the required sensory information quality at run-time. We show how this enables to tune the circuit's power consumption versus inference accuracy trade-off space with fine granularity, and achieve significant power savings at almost no accuracy loss. In addition, our methodology is able to cope with sensor failure and with a dynamically changing environment by means of an efficient on-line tuning strategy. This dynamic Power-vs-Quality scalability is empirically shown on various Machine Learning benchmarking datasets.**

***Index Terms*—Hardware-aware Machine Learning, Bayesian Network Classifiers, Quality Scalable Systems, Embedded Sensing, Embedded Classification**

I. Introduction

SENSORY technologies are essential in many portable devices (e.g. smart watches and phones, domestic gadgets, robots, etc.) as they provide an interface between the user and the environment. Many real-world applications and implementations, however, encounter a fundamental conflict between the desire to uninterruptedly gather, process and fuse a large amount of high quality sensory information and the battery life of the corresponding device.

Consider, for example, a smart-phone based Human Activity Recognition (HAR) application used for daily activity monitoring of elderly people [1]. To be reliable, this application should be able to constantly collect and process high quality sensory information from e.g. accelerometers, gyroscopes and GPS, but its always-on functionality will be impeded by the high energy demands that these sensory tasks entail [2]. State-of-the-art implementations often address limitations on computational bandwidth by running the most complex sensor fusion and inference tasks on the cloud [3], [4]. Yet, the dominant power consumption of sensor interfaces remain the main bottleneck towards the realization of always-on embedded sensing applications. In addition, heavy resource usage has been shown to be a reason leading to poor app reviews in online app stores [5]. In this paper we consider the range of embedded sensing applications whose energy consumption is dominated by the sensing and feature extraction blocks. This is most apparent in applications where classification is done locally for latency and/or privacy reasons, and hence communication overhead is low. Examples are wearable health monitoring [6], home monitoring [7] or mobile activity recognition [1], [8].

To enable their always-on functionality, this class of applications necessitate a new paradigm whereby 1) the circuit level of abstraction is scalable in favor of power consumption savings and 2) the algorithmic level of abstraction is aware of the circuit's properties and can control said scalability towards meeting the user's performance demands. The key insight used in this paper is that it is not always required to operate sensing circuits at their highest possible quality. For example, considering the use case above, it is not necessary to perform high-accuracy high-power GPS sensing when the data collected by the accelerometer indicates that the person is immobile for some period of time.

The most common approach within the Machine Learning community is to first convert sensory information into features which then are used as input for a Machine Learning algorithm. As an example, one can compute the mean and energy of the measurements collected by an accelerometer and gyroscope in order to detect whether a person is walking or sitting. In this paper, however, we extend upon this general scheme and propose the Power-vs-Quality scalable sensing system depicted in Figure 1. The idea is that the system exploits the ability of state-of-the-art circuits to dynamically tune the quality of its sensory signals, and thus adapts the features dynamically, in return for power savings. By being aware of the relationship between a sensor signal's quality and its power consumption, and by tracking each sensory stream's quality and power consumption at run-time, this scheme is able to minimize the system's power consumption while meeting the Machine Learning task's requirements. The design and algorithmic paradigms discussed herewith give rise to Power-vs-Quality tunable inference systems that are capable of operating in various trade-off points between inference accuracy and power consumption, while maintaining the system's robustness to dynamically changing environments.

The main contribution of this paper is threefold, each of them in correspondence to one of the building blocks of the

system as depicted in Figure 1. Concretely, 1) we introduce a tunable feature extracting front-end and derive the power-noise performance scalability equations that describe the circuit components of the system; 2) utilize these equations to propose a circuit-aware Machine Learning scheme, in the form of a Bayesian Network Classifier, that can exploit said scalability; and 3) propose a run-time strategy that tunes the power-noise performance and closes the loop between the scalable circuit and the circuit-aware Machine Learning system.

The remainder of this paper is organized as follows. In Section II, we discuss the the state-of-the-art sensing circuits and circuit-aware Machine Learning techniques. Section III introduces the system architecture of the proposed Power-vs-Quality scalable inference system and briefly outlines the interaction of the different building blocks necessary for such a scalable-system. Next, Sections IV, V, and VI detail the individual building blocks of our system. Finally, we experimentally simulate and evaluate the proposed scalable system in Section VII and conclude in Section VIII.

II. Related Work and Background

We now discuss state-of-the-art approaches towards enabling always-on inference in embedded sensing applications, both from a sensing-circuits and Machine Learning point of view.

A. Circuit Design

The recent surge of applications based on smart low-power sensing systems has sparked significant research into highly power-efficient processors and adaptive sensor front-ends. Many state-of-the-art implementations focus on the optimization and design of hardware accelerators and processors dedicated to run specific inference algorithms [9]. Another body of work, focuses on the sensor interfaces motivated by the often dominant power consumption of the sensor circuits which have to remain always-on and cannot benefit from duty cycling, as the processing unit can. Solutions for signals that can be sparsely represented often exploit compressive sensing [10]–[12] based approaches to reduce the system power consumption. For systems that do not require full precision, but need always-on sensing, adaptive sensing based approaches are targeted [13], [14]. Such approaches enable a scalable resolution performance depending on the complexity of the current information extraction task.

While highly efficient, these sensing circuits are not yet co-optimized with the inference tasks or algorithms, which can enable a further factor in efficiency of such sensing systems. The tight integration of power-noise scalable sensing systems with circuit-aware Machine Learning algorithms opens up opportunities towards co-optimization as they are capable of pre-processing the incoming sensory signal at the scalable sensor front-end thus facilitating a direct interface between the circuit's analog domain properties and the algorithmic implementations.

B. Machine Learning techniques

Several bodies of work in the Machine Learning community focus on enabling efficient embedded classification of sensory data. One of such approaches consists on sequentially deciding what set of observations provide the most information [15] under scarce resources [16] and whether more observations are required to meet the tasks' requirements [17]. However, these techniques are often not suitable for multi-sensor time series where sensory readouts have to be sampled synchronously at real-time. Another stream of research focuses on cost-aware feature sub-set selection where one trades off the value of information in the observations with the impact these observations have on the overall system's resource management, whether computational or circuit oriented [1], [18], [19]. A third line of research exploits application specific temporal characteristics, such as context changes, to optimally switch to low-power sensing modalities that keep inference performance losses to a minimum [20]. Finally, the authors of [21] have devised a run-time power-vs-quality tuning strategy for sensor networks that relies on a cascaded detection system to filter out irrelevant data instances and thus reduces data transmission costs while also increasing the outcoming data quality. However, the two aforementioned techniques exploit application specific properties, context changes in the first case and data transmission in the second. In addition, all the techniques mentioned thus far only enable a few points in the power versus accuracy trade-off space, as they can only decide to observe a feature or not. This fails to exploit all the power saving opportunities of the hardware platform, which, can in fact, be tuned to operate at different levels of quality. Specifically, state-of-the art feature selection enables a trade-off space with 2^N operating points, whereas the proposal in this paper makes available C^N operating points, where N refers to the number of features in the classifier and C to the number of possible levels of quality per feature. As such, our methodology allows for a more subtle inclusion of lower and higher quality (and consequently, power consumption) feature versions, as opposed to the on or off approach of state-of-the-art. In addition, we introduce a heuristic based strategy that comes with very limited overhead, allowing it to function efficiently in dynamic run-time scenarios.

Recent work focuses on exploiting the run-time scalability of quantization noise [22], but fails to harness the more power effective saving opportunities of on thermal-noise based degradation. In addition, it does not provide a run-time strategy for Power-vs-Quality performance tunability that can deal with dynamically changing environmental conditions and sensor failure. In the remainder of this paper, we will propose a methodology that addresses the aforementioned shortcomings to realize an efficient run-time tunable, circuit-aware embedded sensing paradigm.

III. Power-vs-Quality Scalable Sensing System

The noise scalable system has the purpose of enabling sensory based classification in power constrained embedded devices. The focus of this paper, as highlighted in Figure 1 is the tight integration of three crucial building blocks:

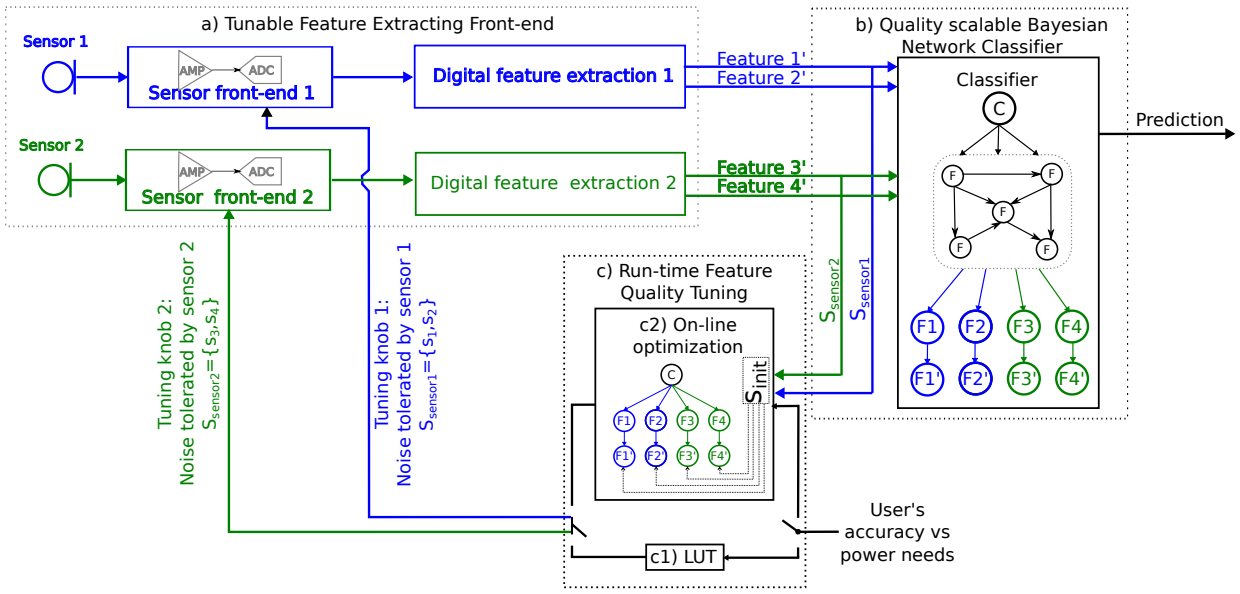

Fig. 1. Power-vs-Quality Scalable Sensing System.

a) Power-vs-Noise tunable mixed-signal feature-extracting front-end: Within this block, the incoming sensory signal is first processed by a set of amplifiers and filters whose settings can be tuned to scale the level of noise tolerance in exchange for power savings. This will be detailed in Section IV, where we also derive the analytical equations highlighting its scalability aspects. The processed signal is then discretized by the Analog-to-Digital-Converter (ADC) and fed to a digital unit where the features — measurable properties of the incoming signal used by the Machine Learning algorithm — are extracted. Each sensory signal is interfaced to its own processing chain and can therefore be individually tuned. As such, the quality of the features generated from a particular sensor are jointly controlled. For example, in Figure 1, the quality of Feature 1 and 2 is regulated by the amount of noise tolerated by Sensor front-end 1 while the quality of Features 3 and 4 is controlled by Sensor front-end 2.

b) Quality scalable Bayesian Network classifier: The end goal of the Power-vs-Quality Scalable Sensing System is to classify the incoming sensory signals, e.g. to identify a specific activity within a Human Activity Recognition application. This task is achieved by a Bayesian Network classifier which was selected over other Machine Learning schemes because: 1) Bayesian Networks facilitate the sensor fusion required by the applications of interest because their probabilistic nature allows to easily model the dependencies (and conditional independencies) among variables of different magnitudes and natures. They also enable the seamless integration of expert knowledge. 2) Bayesian Networks allow to model the probabilistic and statistical properties of the incoming data, including the noise it is subjected to. 3) They allow to encode the properties of the extracted sensory features at varying levels of quality (or circuit noise tolerance) thus enabling the exploitation of the scalable circuit mentioned in the previous paragraph. 4) As a generative model, Bayesian Networks can deal with missing data, e.g. due to failing sensors; and, in contrast to other computationally expensive generative models such as Neural Networks, they can do this in a concise and efficient manner. A Bayesian Network can also be sampled to re-create the statistical properties of the training data thus eliminating the need to have access to ground truths or a data corpus. As will be discussed in Section V, the model thus enables circuit-awareness, since it contains information about the circuit properties and it exploits it from the algorithmic level of abstraction. Note also that the selected scheme supports any Bayesian Network model structure such as naive Bayes or Tree Augmented Naive Bayes (hence the connected features in black in block b) of Figure 1).

c) Run-time Power-vs-Performance tuning algorithm: The circuit-awareness and the noise scalability provided by the previous two blocks is exploited by a run-time control block that monitors the user's classification accuracy and power consumption needs and selects the optimal sensory chain-wise noise tolerance. The proposed strategy relies also on the Quality scalable Bayesian Network classifier to draw the Power-vs-Accuracy search space within which the optimal noise configurations can be selected. In Section VI we provide the details of the tuning strategy and we describe its run-time implementation under static run-time conditions — during which the optimal sensor-wise noise configurations are fetched from an off-line generated Look-Up-Table (LUT) as shown by block c1 in Figure 1 — and under dynamic run-time conditions — during which we select the optimal set of sensor-wise noise configurations according to the current state of the world ($\mathbf{s}_{init}$) as shown by by block c2 in Figure 1. This block closes the loop between the scalable circuit and the circuit-aware algorithmic level of abstraction and it results in a set of Pareto optimal Power-vs-Accuracy operating points that aim to minimize the system's power consumption whilst complying to the classification performance requirements set by the user and the application.

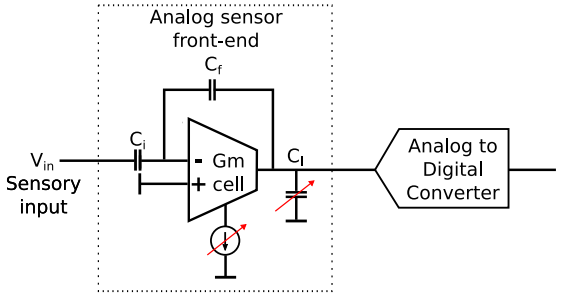

Fig. 2. Power-vs-noise performance scalable front-end.

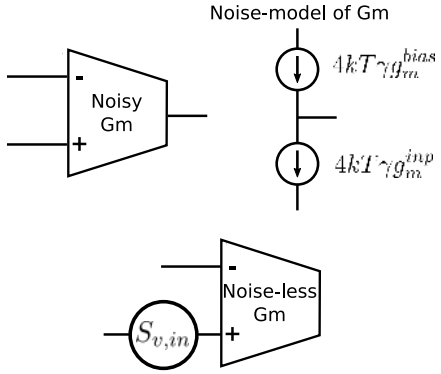

Fig. 3. Noise model for the Gm cell used in Fig. 2

IV. Power-vs-Noise Tunable Feature Extracting Front-end

The Power-vs-Noise performance scalable feature extracting front-end typically consists of the following building blocks (see Figure 1, block a): multiple sensors, a low-noise analog front-end amplifier for each sensor, an ADC to digitize the amplified signal, and a digital feature extraction block where features required by the application are extracted for each sensor. To minimize the power-consumption of the sensing system, we introduce a Bayesian Network based algorithm, that controls the noise performance of the digitized sensory data to settle for the minimum quality necessary for a particular application, and hence minimize the power consumption of the complete system.

For the range of applications under consideration, we assume the use of a 45 nm CMOS technology for the design of both analog and digital blocks. In Sections IV-A and IV-B, we analyze the power consumption contributions of the analog sensing front-end and of the digital feature extraction block.

A. Analog Sensing Front-end

Commonly used analog-mixed signal sensing front-ends use a low-noise amplifier to translate the physical sensor signal to electrical domain and then digitize the signal using an ADC as shown in Figure 2. Such a signal processing front-end exists for each of the sensors used in the system. The low-noise amplifier is modeled as an inverting amplifier with a capacitive feedback network as shown in Figure 2 (dc-biasing not shown), while the ADC is modeled as nb bit quantizer operating at a sampling frequency of $fsamp$.

We now derive the Energy-vs-Noise performance scalability equations for the sensing front-end shown in Figure 2. The total noise performance of the amplifier and ADC is derived first followed by the energy-consumption. The noise performance for the Gm cell used in the amplifier is modeled as shown in the Figure 3. The noise current spectral-density at the output of the Gm cell (see Figure 3) can be expressed as

$$S^I_{gm,op} = 8kT\gamma(gm_{inp} + gm_{bias}) \tag{1}$$

where γ is a process dependent constant typically between 2.3 and 2.5, k the Boltzmann-constant and T the operating temperature. The design variables gm_{inp} and gm_{bias} are the transconductances of the input transistors and the load transistors respectively. Referring the output current noise spectral density to the input, we get:

$$S^V_{gm,inp} = \frac{8kT\alpha\gamma}{gm_{inp}} \tag{2}$$

where $\alpha = 1 + gm_{bias}/gm_{inp}$ and is typically between $1-2$.

The transfer-function for the noise spectral density from non-inverting terminal to output of Gm cell is modeled as (see Figure 3)

$$H(s) = \Big(1 + \frac{C_i}{C_f}\Big)\Big(\frac{1}{1 + s/\omega_p}\Big) \tag{3}$$

where $1 + C_i/C_f$ is the dc closed-loop gain while $\frac{1}{1+s/\omega_p}$ represents the variation of the closed loop gain with the frequency and ω_p being the -3dB cutoff frequency.

From Eq. 2 and 3, the mean-square value of voltage noise at the output of the Gm cell, $V_{n,gm,out}$ can be hence expressed as

$$V^2_{n,gm,out} = \int_0^\infty \mid H(s) \mid^2 S^V_{gm,inp}\frac{df}{(\omega C_L)^2} \tag{4}$$

where C_L is the equivalent load capacitance at the output of the Gm cell. Upon integration the above equation can be simplified to

$$V^2_{n,gm,out} = \Big(1 + \frac{C_i}{C_f}\Big)\frac{2\alpha\gamma kT}{C_L} \tag{5}$$

If an nb-bit ADC completes the sensor front-end, then the total noise contribution of the sensing front-end, $V_{n,rms,tot}$ can be expressed as

$$V^2_{n,rms,tot} = \Big(1 + \frac{C_i}{C_f}\Big)\frac{\alpha\gamma kT}{C_L} + \frac{1}{12}\cdot\frac{V^2_{ref}}{2^{2nb}} \tag{6}$$

where $(1/12)(V^2_{ref}/2^{2nb})$ is the mean-square value of the quantization noise of an nb bit ADC. The quality of the sensory signal generated is based on SNR of the front-end and it can be expressed as

$$SNR = -20\log(2\sqrt{2}V_{n,rms,tot}), \tag{7}$$

$$nb = \frac{SNR - 1.76}{6.02} \tag{8}$$

for a normalized signal swing of $1V_{p-p}$. This SNR based metric of signal quality expressed as number of bits nb is used in Sections V and VI to optimize the Energy-vs-Quality trade-off.

We now derive the energy consumption of the sensing front-end based on the approximation that for an nb bit ADC to yield meaningful output it is necessary that the noise of the amplifier is smaller than the quantization noise of the ADC

$$V^2_{n,gm,out} < V^2_{n,quant,adc} \tag{9}$$

The above equation sets a minimum limit on the load-capacitor C_L and can be expressed as

$$\Big(1 + \frac{C_i}{C_f}\Big)\frac{2\alpha\gamma kT}{C_L} < \frac{1}{12}\frac{V^2_{ref}}{2^{2nb}} \tag{10}$$

where V_{ref} is the ADC reference voltage. Thus

$$C_{L,min} = \Big(1 + \frac{C_i}{C_f}\Big)\frac{24\alpha\gamma kT}{V^2_{ref}}2^{2nb} \tag{11}$$

Further, since there is a feedback factor of $C_f/(C_f + C_i)$ for signals with a bandwidth of f_{sig}, the minimum transconductance necessary for the Gm-cell can be expressed as

$$gm_{inp,min} = \Big(1 + \frac{C_i}{C_f}\Big)2\pi f_{sig} C_{L,min} \tag{12}$$

Combining Eq. (11) and (12)

$$gm_{inp,min} = 48\pi\alpha\gamma kT\Big(1 + \frac{C_i}{C_f}\Big)^2\frac{f_{sig}2^{2nb}}{V^2_{ref}} \tag{13}$$

Using gm/I_D-based design approach, for a simple OTA, the minimum I_D can hence be expressed as

$$I_{D,min} = 2\frac{gm_{inp,min}}{\eta} \tag{14}$$

The parameter η depends on the biasing of the input transistors and is typically < 20 in the saturation and in-between 20 - 25 for the sub-threshold operation, while the factor 2 accounts for the differential nature of the amplifier.

Very often, for designs in scaled CMOS technologies, a single stage open-loop amplification is not sufficient and hence typically 2-3 stage open loop amplification is used. To ensure amplifier stability in a 3-stage amplifier design, the transconductance in 2nd and 3rd stage may need to be as high as $3gm_{inp,min}$ and $5gm_{inp,min}$ respectively thus increasing the bias current in 2nd and 3rd stage proportionately higher. Thus, depending on the amplifier architecture, the total amplifier bias current I_D can be estimated to be range bound as :

$$I_{D,min} < I_D < 9I_{D,min} \tag{15}$$

Based on the above analysis the energy consumption (per digital sample generated) of the sensor interface circuitry highlighted in Figure 2 can be hence expressed as

$$E_{tot} = \frac{V_{DD}I_D + 2f_{sig}2^{nb}FOM_{adc}}{2f_{sig}} \tag{16}$$

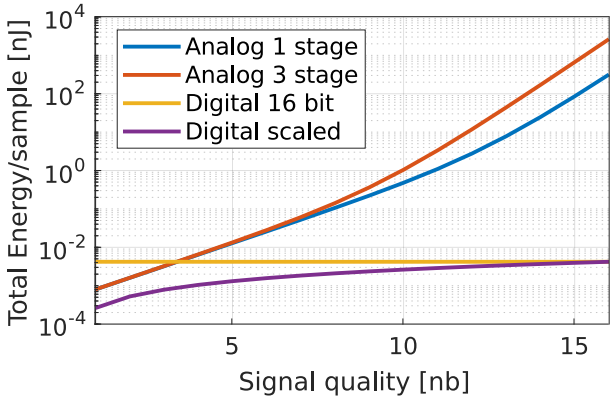

Fig. 4. Energy-vs-Noise performance scaling for the analog sensing front-end.

where FOM_{adc} for STOA ADCs is typically around 10 - 50 fJ / conv-step [23].

Thus from Eq. (13), (14), (15) and (16) we can write

$$E^{1stage}_{tot} = \overbrace{48\pi\frac{\alpha\gamma}{\eta}kT\Big(1 + \frac{C_i}{C_f}\Big)^2\frac{V_{DD}2^{2nb}}{V^2_{ref}}}^{Dominant} + 2^{nb}FOM_{adc} \tag{17}$$

$$E^{3stage}_{tot} = \overbrace{9 \cdot 48\pi\frac{\alpha\gamma}{\eta}kT\Big(1 + \frac{C_i}{C_f}\Big)^2\frac{V_{DD}2^{2nb}}{V^2_{ref}}}^{Dominant} + 2^{nb}FOM_{adc}, \tag{18}$$

where E^{1stage}_{tot} is the energy consumed when using a single stage open-loop amplification and E^{3stage}_{tot} when using three. This range of energy-consumption for the analog front-end is depicted in Fig. 4 as a function of signal quality expressed in i.e. nb.

There is a clear trade-off between the processed signal quality and the required energy consumption. Note that a power vs signal quality relationship can be also derived simply by gaining knowledge of the operational frequency of the sensor-front end, which is application specific. This power vs quality relationship will be exploited by a Bayesian Network based classifier in Sections V and VI to dynamically select the features with just enough quality to maintain the required level of accuracy for the application.

B. Digital Feature Extraction

As highlighted in Figure 1, the digital feature extraction block computes the application specific features from the digitized sensory data. These features are then used by a Bayesian Network Classifier to predict the most likely class to which the incoming sensor streams belong. The energy consumption of this digital feature extraction block depends largely on the type of operation necessary to compute them. In this paper we consider the range of embedded resource constrained sensing applications, where classification has to happen locally for

privacy or latency reasons. Examples of the most common features for this application range are mean, standard deviation, magnitude, maximum, fft coefficients, etc, as discussed by works on Human Activity Recognition [1], [8], health monitoring [6] and home monitoring [7]. All of these features can be computed with a set of multiplications and additions. Therefore, in this work we estimate the energy consumption of the feature extraction block as a function of the number of multiply-accumulate (MAC) operations required per input sample. The level of complexity of the aforementioned features varies. In the calculations of this paper, we will assume that on average, all the arithmetic operations involved in the extraction of a feature, including multiplications, additions and comparisons, can be averaged as 3 MAC operations per sample (e.g. mean, variance and min/max estimation all require one MAC per sample, a 64 point fft would need 6, etc). As explained in Section IV, we assume a 45 nm CMOS technology for our energy calculations. For this particular technology and application range, the estimated energy consumption of a 16 bit MAC operation is 1.5 pJ [24]. Assuming the use of a 16-bit data path and floating point computations, the expected energy consumption per input sample for the feature extraction block is $E_{Fea.Extr.} = \#operations \times Energy/operation_{16b} = 3MAC \times 1.5pJ = 4.5pJ$ as shown by the yellow curve of in Figure 4. However, for many application specific designs, digital blocks can be designed to perform arithmetic operations with variable precision. Considering that energy consumption in the digital domain scales linearly with respect to bit length, the authors in [25] provide an alternative feature extraction energy estimation cost for custom hardware: $E_{Fea.Extr.Custom} = \#operations \times Energy/operation_{1b} \times \#bits$ as shown by the purple curve.

C. Analog vs Digital power consumption

Throughout this section we have shown that, for the hardware properties assumed in this paper, the energy consumption of the analog sensing front end is mostly dominant with respect to the digital feature extraction block. Therefore, we will illustrate how the analog sensing block's Power-vs-Noise scalability, derived in subsection IV-A, can be exploited by our proposed Machine Learning scheme. Note that the generality of our Machine Learning strategy is not lost with these assumptions, as the Energy-vs-Noise relationships derived in Section IV-A can very well be done for any digital block in applications where its power consumption dominates. An example of such cases can be found in advanced health monitoring systems, which do more complex denoising, artifact mitigation, or principal component analysis, resulting in digital subsystems that are less power efficient than the sensors themselves (accelerometers, ECG sensors, etc) and their analog signal conditioning [26]. Just like in the analog case, it is possible to scale the noise in the feature extraction block (by e.g. reducing bit length or sample number in computation or storage) in favor of power consumption savings. The proposed Machine Learning scheme can also exploit this trade-off to determine the optimal digital feature extraction noise tolerance. In this scenario, the control block output in Figure 1 would be fed to the digital feature extraction block instead of to the analog sensing block. In addition, any combination of the energy-vs-quality models represented by the four curves of Figure 4 could be used by our run-time tuning strategy, thanks to its heuristic based optimization methodology. For the experimental section we continue analyzing the impact of only the analog power scaling but will further prove that the overhead digital power consumption does not defeat the benefits of our strategy.

V. Quality scalable Bayesian Network Classifier

In the previous section, we showed how the Power-vs-Quality relation can be derived for each sensory stream in the system, and we concluded that it can be exploited towards minimizing the power required for classification. This calls for a circuit-aware algorithmic strategy that not only takes into account the circuit's properties but also exploits them while meeting the users' performance requirements.

We present a Machine Learning run-time strategy that relies on a Bayesian Network to control the noise degradation of the sensor front-end to meet a desired classification performance while minimizing the power consumption. Bayesian Networks (BN) are directed acyclic graphs that compactly encode a joint probability distribution over a set of random variables [27]. Unlike other deep architectures, which are typically discriminative, Bayesian Networks are generative models and inference can be performed regardless of what nodes are observed or hidden, i.e. without the need to retrain or restructure the model. As such, at any given time, we can decide to stop observing any set of features or start observing sets of previously hidden features. This capability not only allows to completely de-activate sensor streams when desired (due to e.g. very low power consumption requirements) but, as we will discuss in Section VI-D, is essential to account for dynamic run-time conditions, such as sensor failure.

For the noise tuning strategy, we propose to extend a Bayesian Network classifier with nodes representing various (circuit-)noise prone versions of each feature such that each of them can be observed at a specific quality. It is important to realize that the data used to train the model is, in most cases, already noisy and the extent to which this noise affects the classifier's inference performance should determine the tolerated circuit noise in the information gathering stage. In Section VI we introduce a strategy that exploits this circuit noise to trade-off power consumption for inference accuracy.

A. Model

In the rest of this paper, we use the standard notation for variables and their instantiations. Specifically, variables are denoted by upper case letters (F) and their instantiations by lower case letters (f). Sets of variables are denoted by bold upper case letters ($\mathbf{F}$) and their instantiations by bold lower case letters ($\mathbf{f}$).

Figure 5(a) shows the proposed model structure for an example Bayesian Network. The n-feature Tree Augmented Bayesian Network classifier has been extended with n additional nodes that encode the probabilistic relations between

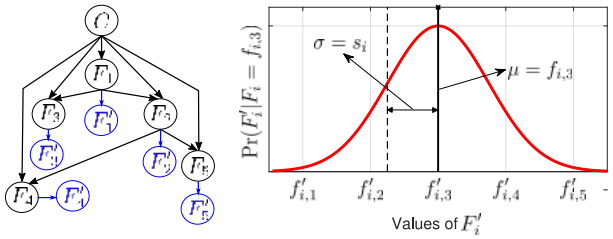

Fig. 5. (a) Example of a Quality scalable Bayesian Network classifier created by extending a Tree Augmented Naive Bayes classifier with noisy feature versions. (b) An example of the probabilistic relation between feature F_i and it's noisy version F_i'.

the features and their low quality versions, denoted by F_i and F_i', respectively.

At run-time, the system observes the features F_i' and uses this information to marginalize their non-observed, high quality counterparts F_i. The proposed structure hence encodes the following joint probability distribution over the original and low quality feature sets and the class variable C:

$$\Pr(C, F_1, .., F_n, ..., F_1', .., F_n') = \prod_{i=1}^{n} \Pr(F_i'|F_i) \cdot \Pr(F_1, \ldots, F_n, C) \ . \quad (19)$$

The distribution describing $\Pr(F_1, \ldots, F_n, C)$ is learned from the training data and the class labels C; while the probabilistic relation $Pr(F_i'|F_i)$ is modeled as a Gaussian distribution with mean $\mu = F_i$ and standard deviation $s_i = V_{n,rms,tot}$ as introduced by Eq. 6. The model thus allows to tune the amount of noise tolerated for the extraction of feature F_i, which results in a proportional power consumption scaling (recall from Eq. 17 that E_{tot} and therefore P_{tot} is proportional to the number of nb bits SNR).

Under this scheme, each feature can suffer from h user defined levels of increasing quality degradation, each characterized by a standard deviation: $\mathbf{s_{i,model}} = \{s_{i,1}, ..., s_{i,h}\}$ and corresponding to a SNR range (in bits) of $\mathbf{N_{i,model}} = \{nb_{i,1}, ..., nb_{i,h}\}$.

As explained in Section IV, each sensory signal is interfaced to a Feature-Extracting Front-end. Thus, the set of features generated from each such sensory stream will suffer from the same degradation at any given time. Consider, for instance, the system shown by Figure 1 which consists of two sensors and four extracted features. Features 1 and 2 suffer simultaneously from the noise of sensor front-end number 1 and features 3 and 4 from the noise of front-end 2. To ease the subsequent explanations we refer to the sets of features of each sensory stream as $\mathbf{F_{sensor,j}}$. Thus, for the previous example $\{F_1, F_2\} \in \mathbf{F_{sensor,1}}$ and $\{F_3, F_4\} \in \mathbf{F_{sensor,2}}$. The selected feature-wise noise configuration would be described by $\mathbf{s}_{select} = \{s_1, s_2, s_3, s_4\}$ with $s_1 = s_2$ and $s_3 = s_4$. The model as such allows to, at any given time t, assess the impact that varying the amount of noise s_i on each of its observed features i might have on classification performance.

B. Inference

At run-time, only the nodes corresponding to the low quality feature versions (F_i' in Figure 5) are observed while the nodes corresponding to the high quality feature versions (F_i) are latent, i.e. they are never observed.

The model is parameterized in terms of the standard deviation s_i of the noise impacting $Pr(F_i'|F_i)$ such that, for various degrees of degradation, the same node F_i' can be (re)used. Note that observing a feature with an infinite standard deviation is equivalent to completely pruning this feature (not observing this sensor, and shutting down the sensory chain). Given an observation $\{f_1', f_2', ..., f_n'\}$, classification is performed by selecting the class that maximizes the posterior probability:

$$\Pr(C|f_1', f_2', \ldots, f_n') \sim \sum_{F_1} \cdots \sum_{F_n} \prod_{i=1}^{n} \Pr(f_i'|F_i) \cdot Pr(F_1, \ldots, F_n, C) \ . \quad (20)$$

The Quality scalable Bayesian Network Classifier thus allows to evaluate classification accuracy under different levels of circuit induced feature quality degradation. This also means that a relationship between classification accuracy and the Feature-Extracting Front-end's power consumption can be established, which, as detailed in the following section, will allow to control the amount of degradation needed in each sensory chain to meet the user's performance needs.

VI. Run-time Feature Quality Tuning

The model introduced in the previous section allows to exploit the Power-vs-Noise scalability of the Feature-Extracting front-end from the algorithmic level of abstraction. In this section, we present the run-time strategy that controls such a tunability and describe its functionality modes under different scenarios.

A. Selection Strategy

The run-time strategy we propose consists of selecting the local optimal set of sensor-wise noise configurations in the classification accuracy versus power trade-off space. The methodology traverses the quality scalable model to perform this optimization procedure and it ultimately pursues two goals: (a) for a given power consumption, find the sensor-wise noise tolerance that maximizes the classification accuracy or (b) for a target accuracy, find the sensor-wise noise tolerance that minimizes power consumption. The available set of such configurations is encoded in the quality scalable model, which allows to choose among h different quality settings for each of the j sensors (see Section V-A) and therefore each of their corresponding feature sets $\mathbf{F_{sensor,j}}$. As such, for each feature F_i within that set, we select the standard deviation s_i that defines the conditional probability distribution $\Pr(F_i'|F_i)$. As described in Section V-A, the selected standard deviation s_i represents the total rms noise tolerated by the sensor front-end as described by Eq. 6.

The number of sensors (and therefore Feature-Extracting Front-ends) of the system and the number of available settings

for each of them define the number of degrees of freedom with which the search space can be traversed. Specifically, the Power-vs-Accuracy search space consists of a total of h^k operating points, where h is the number of possible noise settings defined in $\mathbf{s}_{i,model}$ and k represents the total number of "tuning knobs" or individual sensor streams in the system (for the example in Figure 1, $k = 2$). Since the size of this space increases exponentially with the number of sensors on the system, exploring it exhaustively can become infeasible. Therefore, as common for selection problems of this kind [15], [28], our methodology relies on a greedy search heuristic to explore the search space.

B. Search Strategy

Algorithm 1 provides the details of the search strategy, which is initialized to the current feature-wise noise setting ($\mathbf{s}_{select} = \mathbf{s}_{init}$) defined by the user or by the current run-time conditions. This noise setting also allows the estimation of the initial state within the power-vs-accuracy trade-off space (p and a) which enables the algorithm to search through it until either of the accuracy or power stop criteria (p_{stop}, a_{stop}) have been met.

At every iteration, the algorithm targets to reduce the quality of each sensor-wise feature set $\mathbf{F_{sensor,j}}$, thus producing as many quality reduction candidates as there are "tuning knobs" and selecting the best one by means of a greedy neighborhood search: in each candidate j, the noise tolerated by feature i (included within the set of features $\mathbf{F_{sensor,j}}$) is increased one level with respect to the current setting, hence going from s_{j,v_j} to s_{j,v_j+1}, where v_j refers to the current value of s_j. Each candidate's sensor-wise noise tolerance, described by $\mathbf{s}_{cand,j} = \{s_{1,v_1}, \ldots, s_{j,v_j+1}, \ldots, s_{n,v_n}\}$, thus induces a possible accuracy and power consumption trade-off operating point ($a_{cand,j}$,$p_{cand,j}$). Recall from Eq. 17 that the power consumption of each sensor stream in the front-end (P_i in Algorithm 1) is dominantly proportional to 2^{2nb_i}, where nb_i is the SNR in number of bits, as defined by Eq. 7. The algorithm then selects the candidate that minimizes a predefined cost function $CF = \frac{\Delta a}{\Delta p}$, where the term Δ refers to the predicted state difference between step t and step $t+1$. The accuracy term a is estimated by counting the number of instances in a validation set $\mathbf{F}_{val}$ that were correctly predicted according to their ground truths $\mathbf{C}_{val}$. Note that completely pruning sensors is possible by assigning an infinite value to $\mathbf{s}_i$.

C. Static run-time conditions

Performing this optimization procedure for a range of target accuracy and power consumption settings results in a Pareto-optimal trade-off front. For a selected subset of accuracy or power targets the front-end noise configurations ($\mathcal{S}$) can be stored in a Look-Up-Table (LUT) in the embedded device. Assuming static run-time conditions, feature-noise tuning can then be executed by fetching the configurations that fulfill run-time accuracy or power needs.

Input : BN, a_{stop}, p_{stop}, $\mathbf{F}_{val}$,$\mathbf{C}_{val}$,$\mathbf{s}_{init}$
Output: $\mathcal{S}, \mathcal{A}, \mathcal{P}$
$\mathcal{S} := \emptyset$, $\mathcal{A} := \emptyset$, $\mathcal{P} := \emptyset$
$\mathbf{s}_{select} = \mathbf{s}_{init}$
$a =$ Accuracy(BN,$\mathbf{s}_{select}$,$\mathbf{F}_{val}$,$\mathbf{C}_{val}$)
$p = \sum_{i=1}^{n} P_i(s_i), s_i \in \mathbf{s}_{select}$
$\langle \mathcal{S}, \mathcal{A}, \mathcal{P} \rangle = \langle \mathcal{S} \cup \mathbf{s}_{select}, \mathcal{A} \cup a, \mathcal{P} \cup p \rangle$
while $p \neq p_{stop}$ *AND* $a \neq a_{stop}$ **do**
  for $j = 1$ *to* n **do**
    $\mathbf{s}_{cand,j} = \{s_{1,v_1}, \ldots, s_{j,v_j+1}, \ldots, s_{n,v_n}\}$
    $\Delta a_{cand,j} = a-$Accuracy(BN,$\mathbf{s}_{cand,j}$,$\mathbf{F}_{val}$,$\mathbf{C}_{val}$);
    $\Delta p_{cand,j} = p - \sum_{i=1}^{n} P_i(S_i)$, $s_i \in \mathbf{s}_{cand,j}$
  end
  $\mathbf{s}_{select} = \underset{\mathbf{N} \in \mathbf{N}}{\text{argmin}}$CF($\Delta a_{cand}$,$\Delta p_{cand}$)
  update $a =$ Accuracy(BN, $\mathbf{s}_{select}$,$\mathbf{F}_{val}$,$\mathbf{C}_{val}$);
  update $p = \sum_{i=1}^{n} P_i(s_i), s_i \in \mathbf{s}_{select}$
  $\langle \mathcal{S} \cup \mathbf{s}_{select}, \mathcal{A} \cup a, \mathcal{P} \cup p \rangle$
end
Return: $\mathcal{S}, \mathcal{A}, \mathcal{P}$

Algorithm 1: Feature noise tuning

D. Dynamic run-time conditions

The run-time implementation strategy above is, however, not guaranteed to be robust under certain run-time scenarios as demonstrated in the experimental section. Dynamically changing circuit conditions that are not included in the model, e.g. sensor malfunction, cannot be accounted for during this off-line derivation of a single static optimal set ($\mathcal{S}$). Storing a different LUT for each of those possible conditions is infeasible in an embedded device due to its limited memory capacity. For instance, a 17 feature application with 7 available noise conditions would result in $17^7 = 410$ million settings to store. Moreover, in most cases, these external conditions cannot be predicted well on beforehand or are impossible to monitor.

For such a dynamic setting, we propose to iterate through Algorithm 1 at run-time, therefore selecting the optimal operating points not only in accordance to the user's requirements but also to the current state of the world. To achieve this, the current feature-noise conditions are provided as an input to Algorithm 1 ($\mathbf{s}_{init}$) as shown in Figure 1, thus allowing the search strategy to cater to dynamically changing feature-noise conditions. However, the data driven accuracy estimation — that consists on counting the number of correctly predicted instances from a validation set $\mathbf{F}_{val}$ to which the noise described by s_{cand} has been added — requires a significant amount of memory to store such a validation set. Therefore, we replace the data-driven accuracy estimation by an on-line accuracy estimation approach that utilizes information solely from the Bayesian Network as shown in Figure 6. For that, we rely on a simple forward sampling process executed on-line on the device and we will show that the algorithm can perform adequately, even for small sampling sizes (which are favored due to the embedded device's constraints). At run-time, we perform forward sampling on the Bayesian Network

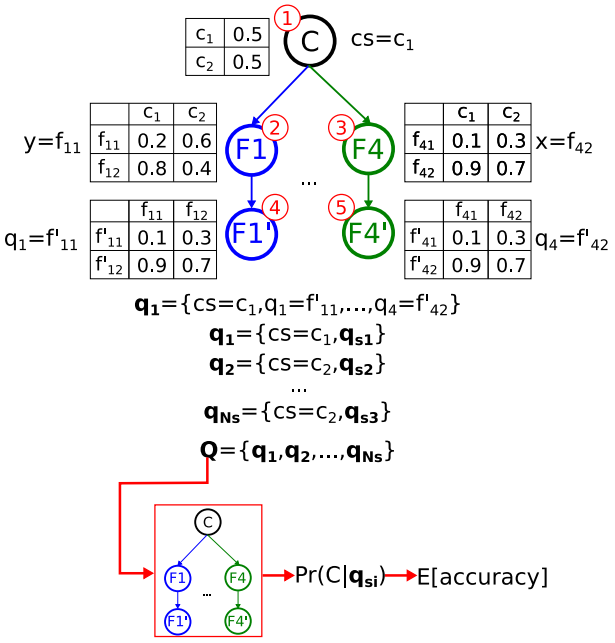

Fig. 6. Process for run-time estimation of expected accuracy: the sample set $\mathbf{Q}$, which comprises N_s data vectors, is sampled from the off-line trained model BN. Each data vector $\mathbf{q_i}$ contains sampled feature values $\mathbf{q}_s$ and the class it was sampled from c_s. The model BN is then used to estimate the class posterior $Pr(C|\mathbf{q}_s)$ and is compared to the sampled c_s to estimate the expected accuracy in equation 21.

parameterized by the current choice of tolerated noise $\mathbf{s}_{select}$. By drawing a total number of N_S instances, we effectively produce the sample set described by $\mathbf{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_{Ns}\}$ with $\mathbf{q} = \{cs, q_1, \ldots, q_n\}$, where cs is sampled from the class node and the instances q are sampled from the noisy feature nodes. Recall from Section V-A that the probabilistic relation $Pr(F'|F)$, as defined by Eq. 19, is parametrized by a Gaussian distribution with standard deviation proportional to the current amount of noise tolerated by each feature. Thus, this sampling process allows to effectively draw a variety of virtual datasets that are capable of emulating the effect of the current noise tolerance setting on the sensory features. Using the probabilistic relations provided by the Bayesian Network, we compute each instance's class posterior probability according to Eq. (20). The expected classification accuracy is then estimated with the following equation:

$$\mathrm{E}[accuracy] = \frac{1}{N_S} \sum_{s=1}^{N_S} \mathbb{1}(cmax_s = cs_s) \ , \qquad (21)$$

where $cmax_s = \arg\max_C Pr(C|\mathbf{q}_s)$ and the choice of N_S is application dependent. Algorithm 1 uses the sampled set $\mathbf{Q}$ and the expected accuracy estimation in Eq. 21 to perform the on-line selection of feature noise settings that satisfy the desired power-accuracy needs (p_{stop}, a_{stop}). Specifically, the set of instances sampled from the noisy feature nodes F' provide Algorithm 1 with the input validation set $\mathbf{F}_{val}$, while the instances sampled from the class node C provide the ground truths $\mathbf{C}_{val}$. The term Accuracy(BN, $\mathbf{s}_{select}$,$\mathbf{F}_{val}$,$\mathbf{C}_{val}$) can then be estimated by $\mathrm{E}[accuracy]$, as defined in Eq. 21.

TABLE I
EXPERIMENTAL DATA SETS.

Dataset	Instances	Features	Classes	Selected F.
Pioneer [1]	6129	27	35	17
HAR	10299	561	6	37
Sonar	208	60	2	8
Glass	214	9	6	6
WDBC	569	30	2	11
Pima	768	8	2	5
Vehicle	846	18	4	8
Banknote	1372	5	2	5
Leaf	340	16	30	10
Ecoli	327	8	5	5
Australian	690	14	2	7

VII. EXPERIMENTAL EVALUATION

We analyze the performance of the proposed noise tuning strategy under static and dynamic run-time conditions on two sensor-based applications of interest: Human Activity Recognition [8] and Mobile Robotics [29]. Furthermore, we implement the methodology in 9 additional Machine Learning benchmarking datasets [30] to demonstrate its general applicability.

A. Experimental setup

All the benchmarks included in this paper underwent a pre-processing step that consisted on removing the nominal features and performing feature selection with Weka's wrapper subset evaluator with a Bayes Net classifier, a best-first search and the default parameters [31]. Table I outlines the details of the chosen benchmarks, including the number of pre-selected features (column denoted Selected F.) as used in our experiments.

For all the experiments, we defined $h = 7$ possible signal quality settings (see Section V-A) equal to $\mathbf{s_{model}} = \{0.0002, 0.0004, 0.005, 0.02, 0.05, 0.09, \infty\}$ — with ∞ corresponding to pruning the feature — normalized according to each feature's dynamic scale and corresponding to a SNR range of $\mathbf{N_{model}} = \{11, 10, 5, 4, 3, 2, 0\}$ bits. The power estimates were then obtained with the Equations detailed in Section IV-A . To ease the performance analysis and due to missing information on the properties of the circuit with which each data set was generated, we normalized those estimated in all experiments from 0 to 1. As such, a normalized power consumption equal to 1 represents the setting where all feature streams are extracted at the highest possible quality (corresponding to $s_i = 0.0002$ or a SNR of 11 bits).

All the experimental results we present throughout this Section were obtained by conducting a 5-trial, 5-fold validation. The reported accuracy percentages reflect the performance of the Quality Scalable Bayesian Network Classifier with a naive Bayes structure (block b of Figure 1) when the tested instances are subjected to the noise tolerance setting selected by the Feature Quality Tuning strategy (Section VI) in either the Static or Dynamic run-time conditions. That is, Gaussian noise parameterized by the current choice of s_i was added to each

[1]From activities that include only linear movement (without turning and gripper activity).

feature and then discretized, thus simulating the impact that the noisy analog component of the Sensor Front-end might have on the quality of the features fed to the classifier. For this, we assume that the relationship between the analog circuit noise from the amplifiers and filters of the Sensor Front-end and the noise we inject in the features is linear. This is because all the operations we do in the digital domain — including feature extraction— consist of additions and multiplications. An addition of noisy data will result in a noisy result with a linear noise dependence: $a + noise + b + noise = a + b + 2 \cdot noise$. The same goes for a multiplication: $(a+noise)\cdot(b+noise) = a \cdot b + (a + b) \cdot noise + noise^2$, where the last term can be neglected since noise is smaller than the signal.

B. Performance of the Feature Quality tuning strategy

In this section we analyze the trade-offs induced by implementing the Feature Quality Tuning strategy under different scenarios and conditions.

We perform most of the analyses on the Human Activity Recognition (HAR) [8] and the Mobile Robotics Pioneer benchmarks [29], as they correspond to the sensor based applications of interest listed in this paper and they have been used by the Machine Learning community for classification tasks. The HAR benchmark includes features generated from six sensor streams (a tri-axial accelerometer and a tri-axial gyroscope) and can be used to identify the activities of 'Walking','Walking upstairs','Walking downstairs','Sitting','Standing' and 'Laying'. The features used for these experiments include properties of time and frequency based physical quantities among which are mean, standard deviation, maximum/minimum magnitude, kurtosis, entropy, skewness and energy of body acceleration and jerk, and gravity [8]. The Pioneer benchmark was collected by a mobile robot with three different types of sensors (sonars, wheel odometers and vision sensors) and it has been used to identify specific navigation occurrences described by the direction of the movement (forwards or backwards), whether the navigation path was obstructed or unobstructed, the activity's speed, and the visibility of objects in the environment [29].

As mentioned in Section V-A the noise scalable Bayesian Network model makes available h^k operating points in the Power-vs-Accuracy trade-off space, where h refers to the available number of noise settings per feature set and k refers to the number of "tuning knobs", or noise-tunable sensors with their own feature generating sensory stream (see Section V-A).

With 6 tuning knobs for the HAR application (a tri-axial accelerometer and tri-axial gyroscope), there are thus 7^6 such operating points available. For the experiments in this paper, we used 17 of the Pioneer features, each corresponding to a particular sensory stream and therefore a particular tuning knob, resulting in a total number of 7^{17} possible operating points. Instead of searching exhaustively across these large trade-off spaces we implement the proposed heuristic, which induces the Pareto-optimal trade-off curves shown by Figure 7. Each of these curves consists of $h \times k$ Pareto-optimal operating points, that is 42 for HAR and 119 for Pioneer.

For the HAR application, the scheme provides power consumption savings of at least an order of magnitude whilst

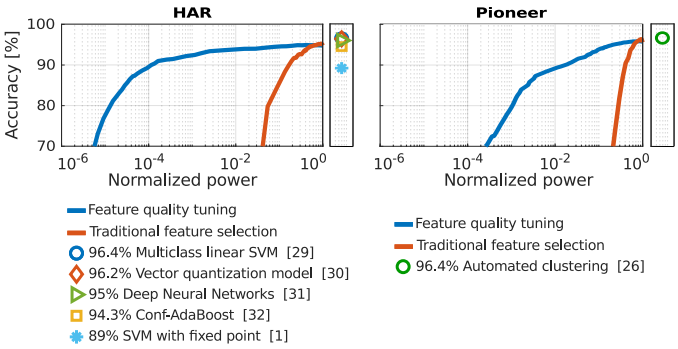

Fig. 7. Power-vs-Accuracy trade-off achieved by the noise tuning strategy on the HAR and on the Pioneer benchmarks compared to other state of the art implementations.

preventing accuracy degradation. Additional power savings of, for example, 4 orders of magnitude can be traded-off for accuracy losses by selecting an operating point that performs at 90% instead of 95% accuracy. The Pareto-front for the Pioneer benchmark avoids accuracy degradation with power savings of an order of magnitude but degrades at a faster rate than the one for the HAR benchmark. This difference in performance, among other factors, relates to the number of instances available in the datasets and the number of classes to predict (see Table I) . The HAR dataset has almost twice as many instances as the Pioneer does, which allows to train a more precise model. In addition, the Pioneer dataset has 35 classes whereas the HAR only has 6, resulting in more overlap of the conditional distributions $Pr(C|F)$, especially when the features are subjected to noise. Such effects can also be observed on the benchmark datasets that will be discussed in Section VII-E.

Figure 7 also shows the results attained by our proposal in comparison to state of the art implementations. For both datasets, we compare with state-of-the-art feature selection strategies, such as the one used by [18], where it is only possible to decide whether a feature is observed or pruned. In addition we include the classification accuracy attained by several other Machine Learning methodologies on the same dataset: we contrast the HAR application with 5 other benchmarks [1], [32]–[35] and the Pioneer application with [29].

Figure 8 illustrates, for the HAR benchmark, four particular trade-off configurations achieved with the presented optimization methodology. Point 1 shows that a power saving of one order of magnitude without accuracy degradation can be achieved, when the features generated by the gyroscope in x have a noise setting of $s = 0.0002$ while the rest use a resolution of $s = 0.0004$. An additional power reduction of three orders of magnitude (point 2) with a degradation of 5% accuracy is achieved when the features from the accelerometer in y tolerate tolerates $s = 0.005$ and the rest tolerate $s = 0.02$. Point 3 and 4 are achieved by pruning the y-gyroscope and the x-gyroscope sensor streams, while allowing more noise for the rest of their features as shown in the Figure.

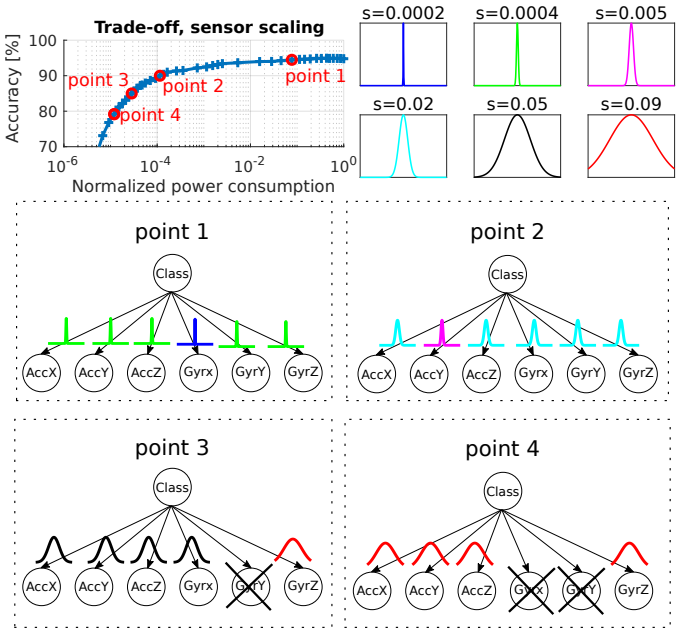

Fig. 8. Examples of four operating points on the Power-vs-Accuracy trade-off for the HAR benchmark.

C. Feature noise tolerance run-time implementation

The trade-offs discussed in the previous section assume static conditions and, under such assumption, can be implemented at run-time by fetching the selected operating points from a LUT. As we have discussed in Section VI, dynamically changing run-time conditions can reduce the robustness of the LUT strategy.

As a motivating example, we now illustrate the LUT strategy's lack of robustness in the case of sensor malfunction or sensor de-activation for the HAR and the Pioneer datasets in Figure 9. The blue curve shows the results from the LUT methodology with entirely functional sensors and serves as a reference. The red and yellow curves show the performance during failure of a particular sensor (listed at the top of the sub-plots) of the on-line strategy and the LUT strategy, respectively. Whereas the on-line methodology is able to, at run-time, pick a feature combination that does not require the usage of the failed or deactivated sensor, the LUT methodology is bound to the settings selected off-line, which are not optimized to account for this situation. Consider, for instance, the failure of the HAR application's z-accelerometer after the first iteration. The on-line methodology (red) is capable of selecting new sensory noise and pruning combinations that achieve almost the same performance as the reference with fully functional sensors (blue). In contrast, the LUT methodology (yellow) experiences an accuracy drop of more than 2% after a few iterations. A similar effect can be observed when the odometers of the Pioneer application lose functionality. Certain applications, however, rely heavily on features from particular sensors, which, for the current experiments, is also a consequence of the feature selection procedure the datasets underwent as a pre-processing step.

Consider, for example, the failure of the y-axis accelerometer in the HAR application. Both in the on-line and off-line case accuracy fall considerably after the first iteration because some of the features extracted from that sensor provide the most valuable observations. A similar effect is observed in the case of failure of all seven sonars of the Pioneer robot as two of them bear the most informative observation.

This is observed as well as of the vision sensors in the same application. Overall, note that regardless of the application and the failed sensor, the LUT methodology's accuracy drops at a faster rate than the on-line optimization methodology. It should be noted that due to its probabilistic nature, Bayesian Network classifiers are capable of successfully inferring classes regardless of what feature subsets are observed or hidden. Other Machine Learning schemes, such as Neural Networks, would not be able to deal with this without having to retrain the model.

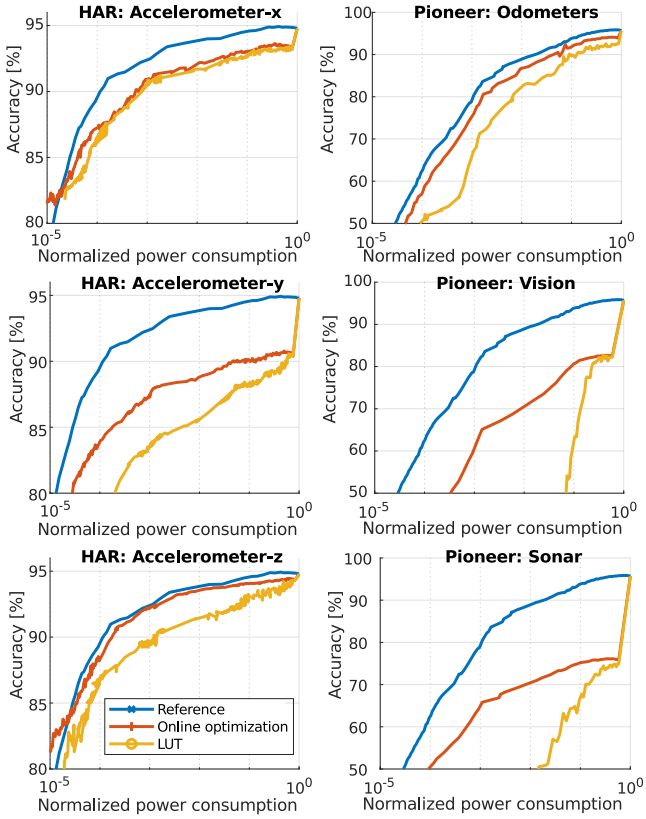

Fig. 9. Performance comparison of the LUT and on-line optimization strategies during sensor stream malfunction.

Regardless of whether the application is faced with a dynamic run-time scenario (as the one just discussed), it is important to assess the performance of the on-line optimization strategy against the LUT methodology, as the on-line strategy suffers from imperfect accuracy estimation due to its limited sampling set. Figure 10 shows a comparison of the trade-off achieved by the off-line LUT methodology (as seen in the last Subsection) and the on-line optimization when using three sampling set sizes N_S for the estimation of expected accuracy in Eq. 21. Specifically, for the blue curve we used a sample size N_S equal to IN, where IN is the total number of instances available in the dataset (see Table I), for the red we used $N_S = IN/5$ and for the yellow $N_S = IN/10$. Overall,

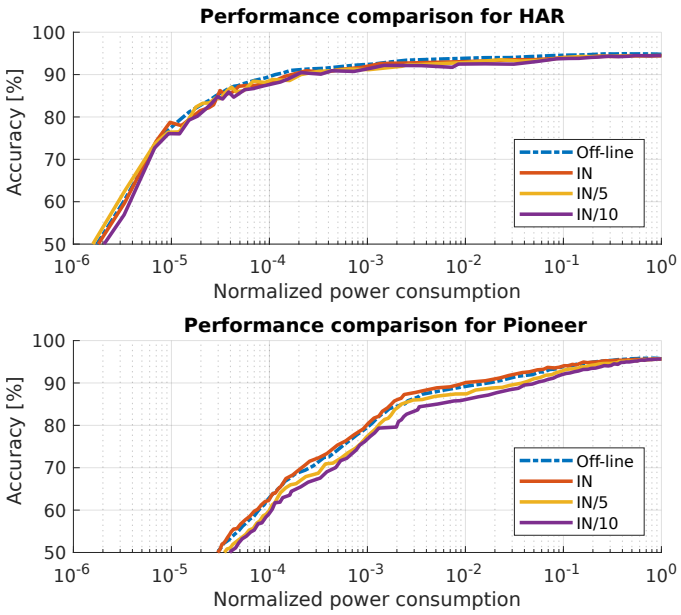

Fig. 10. Performance comparison of the LUT case and the on-line case with different sample sizes

the performance loss from the on-line strategy is very limited, or less than 2%. Note that the smaller the sample size, the larger the standard deviation across the different trials will be.

Overall, we have seen throughout this Section that the on-line optimization methodology is capable of dealing with unforeseen dynamic run-time situations while retaining most of its Power-vs-Accuracy performance, even for small sample sizes ($IN/10$).

D. Digital overhead power consumption

This section will assess the computational overhead due to the run-time tuning strategy, and put it in perspective relative to the achievable system power savings. Recall from Section IV-C, that the analog sensor front-end power consumption is dominant with respect to the digital feature extraction, so we leave the latter out of this analysis. We will illustrate this overhead with the HAR benchmark, described in Table I. Similar to Section IV-B, the energy consumption estimation is based on the number of required elementary operations. Note that, because the probabilistic relations $Pr(F'|F)$ can be parametrized and therefore $Pr(F'|C)$ can be computed in advance, the marginalization is not needed at run-time. Considering the necessary memory operations, as well as all the actions that have to be performed to sample the Bayesian Network and estimate the expected accuracy, the elementary operation count per iteration of Algorithm 1 is: $(d+1)\times 2N_s$ memory fetches; $(d+1)\times 2N_s$ register operations; $(2d+2)\times 2N_s$ multiplications; and $N_s\times(3+c+d\times(1+card(d)))$ additions (where d denotes the number of features and $card(d)$ the number of values each of them can take on, N_s the number of samples and c the number of possible classes).

In Table II we put this overhead power consumption of the tuning algorithm in perspective with the lowest analog sensing power consumption, which will take place when the signal is

TABLE II
OVERHEAD OF RUNNING ALGORITHM 1 EVERY MINUTE WITH RESPECT TO LOWEST ANALOG POWER CONSUMPTION.

Scaling scenario	Relative power consumption
1) Worst:P_{max} to $P_{max}/5$, $N_S = IN$	0.4%
2) Average:$P_{max}/5$ to $P_{max}/10$, $N_S = IN/5$	0.02%
3) Best:$P_{max}/10$ to $P_{max}/20$, $N_S = IN/10$	0.006%

extracted with only 2 bits SNR (see Figure 4). For this analysis we assume that the analog sensor-front end has a frequency of 10 kHz and that the control block is running Algorithm 1 every minute. Let's analyze three different scenarios: 1) The system is currently using all the features at its highest quality and, over the course of the following minute, we need to scale the analog power consumption by 5 fold with minimum accuracy loss (thus $N_S = NI$ for expected accuracy estimation). We can see from the first row of the table that the tuning algorithm would require 0.4% of the lowest analog power consumption. 2) If we would look for a further power consumption scaling of $P_{max}/10$ while relaxing the accuracy specifications (using now $N_S = NI/5$), the relative power consumption decreases all the way to 0.02%. 3) Finally, going from a scaling of $P_{max}/10$ to $P_{max}/20$ with the lowest possible number of samples for the expected accuracy estimation reduces the consumption to as low as 0.006%. Because the overhead of the on-line tuning procedure is negligible in comparison to the overall power savings for several sample sizes, the designer is allowed to freely chose among them in accordance to their optimization targets, i.e. reduce the power as much as possible with a small sample size or maximize the performance as much as possible while still attaining power savings.

Note that, similar to the control block, the classification task itself (block b) of Figure 1) is required at a much lower frequency than that of the sensing and feature extraction blocks. This as the inference runs on features which are extracted over a larger time window (such as min/max/mean/variance/fft/). Moreover, each iteration of the algorithm run by the control block includes several inference steps. Therefore, we do not include any further analysis on the classification blocks power consumption.

E. Methodology's performance for a wide range of benchmarking datasets

For the purpose of generality, we also demonstrate the applicability of our methodology on 9 benchmark data sets from the UCI Machine Learning repository [36]. Most of these datasets are not from sensor-based applications, however, we assumed that they provide numerical strings similar to those collected by sensor interfaces. This also shows the broad applicability of our methodology: it successfully achieved the trade-off of interest regardless of the application and data type it encounters.

Figure 11 shows the trade-off attained by the selection strategy at different Power scaling levels for all the datasets. The curves labeled "Offline" present the results from estimating the accuracy with a validation data set (which would be

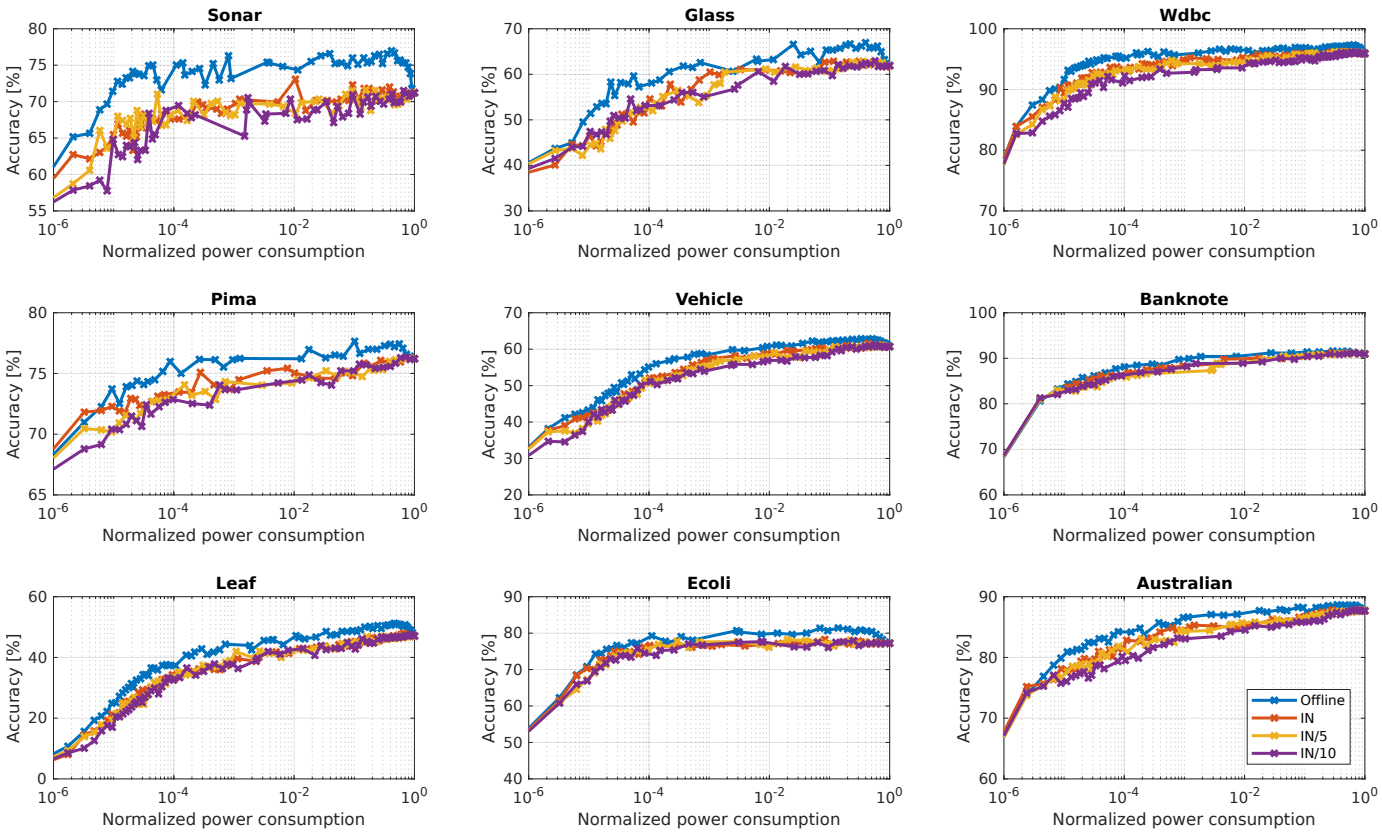

Fig. 11. Performance of the Feature Quality Tuning algorithm for different benchmarks.

implemented at run-time by fetching this values from a LUT). The rest of the curves refer to the on-line optimization strategy as discussed in the last section, for three sizes of samples used for the estimation of the expected accuracy. Accuracy degradation rates depend on the data sets' properties: number of sensor streams or 'tuning knobs', number of possible classes to predict, and number of instances available for training. More tuning knobs result in a finer granularity of the Power-vs-Accuracy trade-off space and thus the possibility to better comply to the particular operational needs requested by the user. However, a larger dimensionality in Bayesian classification problems results in increased occurrences of *decision boundaries* [37] —regions where the curves of $\Pr(F_i|C)$ intersect— and therefore in higher susceptibility to classification errors. In other words, a noisy feature value f_i' can be assigned to a discretization interval different to the one its noiseless version f_i would be, resulting in the prediction of different classes in both cases. This effect can also be heightened by a wider shape of the distribution of $\Pr(F_i|C)$ due to the class overlap this entails. Conversely, data sets with larger training sets result in more robust classifiers. For example, note how the off-line trade-off compares to the on-line trade-offs for the Banknote vs for the Sonar benchmarks — with 1372 and 200 dataset instances, respectively. The performance of all 4 curves in the first case do not differ for more than 3% while in the latter case there is serious accuracy loss regardless of the sample size. In general, we can conclude that Power-vs-Quality Scalable Sensing Systems are expected to be more valuable for multi-sensory systems not only because the number of possible configurations in the Power-vs-Accuracy trade-off space is proportional to the number of available 'tuning knobs' as mentioned before; but because it is more likely that the model does not largely rely on only one sensor, thus keeping classification accuracy from degrading for a significant amount of power consumption savings.

Under the same experimental conditions we can see that, overall, all the data sets benefit from the different noise tuning strategies: in all our experiments, they lose less than 5% accuracy while achieving power savings of at least four orders of magnitude. This also proves that the methodology can be effectively implemented with a wide variety of applications as it allows to select the optimal settings (e.g. amount of noise injected, desired initial and final accuracy/power) according to the data set and the classification task.

VIII. Conclusion

In this paper we presented a Power-vs-Quality scalable sensing system which allows to efficiently tune noise in sensors through a Machine Learning controlled feedback loop. We proposed a Bayesian Network classifier structure that encodes the probabilistic relations between sensory features and their noisy counterparts with multiple levels of circuit dependent degradation. This enables a run-time implementation of dynamic Power-vs-Quality tunable inference for always-on sensing. We demonstrated that this model allows to exploit

the resource saving opportunities of feature tunable sensing systems by dynamically controlling the amount of tolerated noise across sensory features. We discussed the performance of the proposed methodologies through the analysis of the potential power consumption versus inference accuracy trade-offs on several sensor based applications and we demonstrated the general applicability on 11 standard Machine Learning datasets. The methodologies are capable of dynamically finding optimal sensor front-end noise settings, that can provide significant power savings, while maintaining or only slightly degrading accuracy. Additionally, we demonstrated that our on-line methodology is robust against sensor failure while incurring in negligible power overhead, proving thus its implementation feasibility on embedded sensory devices.

IX. Acknowledgements

This work has been supported by the EU ERC project ReSENSE under grant agreement ERC-2016-STG-715037.

References

[1] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine," in *International workshop on ambient assisted living*. Springer, 2012, pp. 216–223.

[2] M. Verhelst and A. Bahai, "Where analog meets digital: Analog to information conversion and beyond," *IEEE Solid-State Circuits Magazine*, vol. 7, no. 3, pp. 67–80, 2015.

[3] R. Arumugam, V. R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, A. S. Kumar, K. D. Meng, and G. W. Kit, "Davinci: A cloud computing framework for service robots," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 3084–3089.

[4] F. H. Bijarbooneh, W. Du, E. C. H. Ngai, X. Fu, and J. Liu, "Cloud-assisted data fusion and sensor selection for internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 257–268, 2016.

[5] G. Pinto and F. Castor, "Energy efficiency: A new concern for application software developers," *Commun. ACM*, vol. 60, no. 12, pp. 68–75, Nov. 2017. [Online]. Available: http://doi.acm.org/10.1145/3154384

[6] S. L. Lau, I. König, K. David, B. Parandian, C. Carius-Düssel, and M. Schultz, "Supporting patient monitoring using activity recognition with a smartphone," in *Wireless communication systems (ISWCS), 2010 7th international symposium on*. IEEE, 2010, pp. 810–814.

[7] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "Chime-home: A dataset for sound source recognition in a domestic environment," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2015 IEEE Workshop on*. IEEE, 2015, pp. 1–5.

[8] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones." in *ESANN*, 2013.

[9] B. Moons and M. Verhelst, "Energy-efficiency and accuracy of stochastic computing circuits in emerging technologies," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 4, no. 4, pp. 475–486, 2014.

[10] F. Chen, A. P. Chandrakasan, and V. M. Stojanovic, "Design and analysis of a hardware-efficient compressed sensing architecture for data compression in wireless sensors," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 3, pp. 744–756, 2012.

[11] Y. Oike and A. E. Gamal, "A 256 x 256 cmos image sensor with delta-sigma-based single-shot compressed sensing," in *2012 IEEE International Solid-State Circuits Conference*, Feb 2012, pp. 386–388.

[12] P. V. Rajesh, J. M. Valero-Sarmiento, L. Yan, A. Bozkurt, C. V. Hoof, N. V. Helleputte, R. F. Yazicioglu, and M. Verhelst, "22.4 a 172??w compressive sampling photoplethysmographic readout with embedded direct heart-rate and variability extraction from compressively sampled data," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 386–387.

[13] K. M. Badami, S. Lauwereins, W. Meert, and M. Verhelst, "A 90 nm CMOS, power-proportional acoustic sensing frontend for voice activity detection," *Solid-State Circuits, IEEE Journal of*, vol. 51, no. 1, pp. 291–302, 2016.

[14] M. Yip and A. P. Chandrakasan, "A resolution-reconfigurable 5-to-10-bit 0.4-to-1 v power scalable sar adc for sensor applications," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 6, pp. 1453–1464, June 2013.

[15] A. Krause and C. E. Guestrin, "Near-optimal nonmyopic value of information in graphical models," *arXiv preprint arXiv:1207.1394*, 2012.

[16] Y. Wang, I. I. Hussein, D. R. Brown, and R. S. Erwin, "Cost-aware sequential bayesian tasking and decision-making for search and classification," in *Proceedings of the 2010 American Control Conference*, June 2010, pp. 6423–6428.

[17] A. Choi, Y. Xue, and A. Darwiche, "Same-decision probability: A confidence measure for threshold-based decisions," *International Journal of Approximate Reasoning*, vol. 53, no. 9, pp. 1415–1428, 2012.

[18] H. Ghasemzadeh, N. Amini, R. Saeedi, and M. Sarrafzadeh, "Power-aware computing in wearable sensor networks: An optimal feature selection," *IEEE Transactions on Mobile Computing*, vol. 14, no. 4, pp. 800–812, 2015.

[19] S. Lauwereins, K. Badami, W. Meert, and M. Verhelst, "Context-and cost-aware feature selection in ultra-low-power sensor interfaces," in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2014, pp. 93–98.

[20] D. Jun, L. Le, and D. L. Jones, "Cheap noisy sensors can improve activity monitoring under stringent energy constraints," in *2013 IEEE Global Conference on Signal and Information Processing*, Dec 2013, pp. 683–686.

[21] L. N. Le and D. L. Jones, "Guided-processing outperforms duty-cycling for energy-efficient systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2414–2426, Sept 2017.

[22] L. Galindez, W. Meert, H. Bruyninckx, and M. Verhelst, "Extending naive bayes with precision-tunable feature variables for resource-efficient sensor fusion," in *2nd Workshop on Artificial Intelligence and Internet of Things, ECAI 2016*, 2016, pp. 23–30.

[23] B. Murmann, "Adc performance survey 1997-2017," *http://web.stanford.edu/ murmann/adcsurvey.html.*

[24] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 10–14.

[25] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Dvafs: Trading computational accuracy for energy through dynamic-voltage-accuracy-frequency-scaling," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, March 2017, pp. 488–493.

[26] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, Jan 2010.

[27] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[28] A. Krause and C. Guestrin, "Optimal nonmyopic value of information in graphical models: efficient algorithms and theoretical limits," 2005.

[29] T. Oates, M. Schmill, and P. R. Cohen, "Identifying qualitatively different experiences: Experiments with a mobile robot," 2000.

[30] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[31] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[32] B. Romera-Paredes, M. S. Aung, and N. Bianchi-Berthouze, "A one-vs-one classifier ensemble with majority voting for activity recognition," in *ESANN 2013 proceedings, 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2013, pp. 443–448.

[33] M. Kästner, M. Strickert, T. Villmann, and S.-G. Mittweida, "A sparse kernelized matrix learning vector quantization model for human activity recognition." in *ESANN*, 2013.

[34] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert Systems with Applications*, vol. 59, pp. 235–244, 2016.

[35] A. Reiss, G. Hendeby, and D. Stricker, "A competitive approach for human activity recognition on smartphones," in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2013), 24-26 April, Bruges, Belgium*. ESANN, 2013, pp. 455–460.

[36] C. L. Blake and C. J. Merz, "Uci repository of machine learning databases [http://www. ics. uci. edu/˜ mlearn/mlrepository. html]. irvine,

ca: University of california," *Department of Information and Computer Science*, vol. 55, 1998.

[37] Y. Yang and G. I. Webb, "Discretization for naive-bayes learning: managing discretization bias and variance," *Machine learning*, vol. 74, no. 1, pp. 39–74, 2009.

Laura Galindez was born in Mexico City in 1989. She received the B.S. degree in Mecatronics Engineering from the Monterrey Institute of Technology and Higher Education, Mexico in 2012; and the M.S. degree in Systems and Control from Eindhoven University of Technology, Eindhoven, The Netherlands in 2015. Immediately after, she joined the MICAS Research Group of KU Leuven, Leuven, Belgium, where she is currently pursuing her Ph.D. degree with a focus on hardware-aware machine learning for resource constrained embedded devices.

Komail Badami is currently a final year PhD researcher at MICAS research laboratories at KU Leuven, Belgium. His research interests include low-power analog/mixed-signal designs for resource-constrained sensor-interfaces, adaptive and context-aware circuits and systems. Before starting his PhD, he was an Analog Design Engineer with Intel Bangalore from Aug. 2011 to Dec. 2012, where he worked on compensation schemes for high speed communication for parallel memory I/O. He has also held internship position with IBM Semiconductor Research and Development, Bangalore during his MS by Research at Indian Institute of Technology, Madras, India. He serves as a reviewer for IEEE Transactions on Circuits and Systems: Express Briefs. He is the recipient of EXPERTS III PhD grant and SSCS Pre-Doctoral award.

Jonas Vlasselaer was born in Leuven, Belgium in 1986. In 2009, he obtained his Bachelor of Industrial Sciences degree and in 2010 his Master of Industrial Sciences degree (specialized in Electronic Engineering) from the Leuven Engineering College (GroupT). He graduated in 2011 as a Master of Science in Artificial Intelligence, major subject Engineering and Computer Science, at the KU Leuven. In 2011, he started as a doctoral student at the DTAI lab (Declarative Languages and Artificial Intelligence) at the KU Leuven under the supervision of Professor Luc De Raedt. He received a personal doctoral scholarship in 2012 from IWT (agency for Innovation by Science and Technology) and successfully defended his dissertation, entitled Probabilistic Inference for Dynamic and Relational Models, in December 2016. From July 2017 till April 2018, he was a postdoctoral researcher at the MICAS research devision of the KU Leuven working on the ERC project RE-SENSE. He is currently working as a Software Developer in PORPHYRIO NV in Leuven, Belgium.

Wannes Meert received his degrees of Master of Electrotechnical Engineering, Micro-electronics (2005), Master of Artificial Intelligence (2006) and Ph.D. in Computer Science (2011) from KU Leuven. He is currently research manager in the DTAI research group at KU Leuven. His work is focused on applying machine learning, artificial intelligence and anomaly detection technology to industrial application domains. For these contributions he received a number of prizes (e.g., EU AAL Smart Ageing Prize, Patient Room of the Future award).

Marian Verhelst is an associate professor at the MICAS laboratories (MICro-electronics And Sensors) of the Electrical Engineering Department of KU Leuven. Her research focuses on embedded machine learning, hardware accelerators, self-adaptive circuits and systems, and low-power sensing and processing for the internet-of- things. Before that, she received a PhD from KU Leuven cum ultima laude in 2008. She was a visiting scholar at the Berkeley Wireless Research Center (BWRC) of UC Berkeley in the summer of 2005. From 2008 till 2011, she worked as a research scientist in the Radio Integration Research Lab of Intel Labs, Hillsboro OR.

Marian has published over 100 papers in conferences and journals. She is a member of the DATE conference executive committee, and was a member of the ESSCIRC and ISSCC TPCs and of the ISSCC executive committee. Formerly, Marian was an SSCS Distinguished Lecturer, a member of the Young Academy of Belgium, an associate editor for TCAS-II and JSSC and a member of the STEM advisory commitee to the Flemish Government. Marian currently holds a prestigious ERC Starting Grant from the European Union.