Energy Efficient On-Sensor Processing for Online Activity Recognition

Florian Grützmacher[1], Albert Hein[1], Benjamin Beichler[1], Polichronis Lepidis[2], Rainer Dorsch[2], Thomas Kirste[1] and Christian Haubelt[1]
[1]*University of Rostock, 18051 Rostock, Germany*
[2]*Bosch Sensortec GmbH, Gerhard-Kindler-Straße 9, 72770 Reutlingen, Germany*

Keywords: Energy Efficiency, Wearable Sensors, Activity Recognition, Energy Trade-Off, On-Sensor Computations.

Abstract: In sensor-based online activity recognition, the communication of sensor samples at high data rates has a great impact on the energy consumptions of wearables. In our work we investigate the idea of calculating data reducing stages of activity recognition systems on wireless sensor nodes in order to reduce the amount of transmitted data and thus the overall energy consumption. In our experiments, this approach could reduce the energy consumption of a wireless sensor node by up to 27%. Since the benefit of this approach highly depends on design parameters of the activity recognition, we introduce an energy trade-off model for wireless sensor nodes to estimate energy-savings of application specific configurations at design time. By calibrating this model for our wireless sensor node, we could achieve an accuracy of more than 99% in our experiments.

1 INTRODUCTION

Online Activity Recognition with wearables has gained increasingly importance, especially in the domains *healthcare*, *e-health*, and *ambient assisted living* (AAL). Usually, activity recognition is done in multiple consecutive stages, which are known as the Activity Recognition Chain (ARC) like introduced by Bulling et al. in (Bulling et al., 2014). This chain is composed of the stages data acquisition, preprocessing, segmentation, feature extraction, and finally, classification. For a more detailed description we refer to (Bulling et al., 2014).

Common setups involve wearable sensors from which raw data is sent via wireless channels to perform *online* activity recognition. When using wearable sensors, its energy consumption plays a crucial role, as it directly influences the long term usability of the system. Thus, it is of great importance to keep the energy consumption as small as possible, to prolong the lifetime of the batteries. A considerable amount of energy is used for the wireless transmission of raw sensor data from the wearable sensors (Grützmacher et al., 2017; Rault et al., 2017). In order to reduce the overall energy consumption, the amount of wirelessly transmitted sensor data should be kept as low as possible. In order to achieve *application specific data reduction* the feature extraction stage of most ARCs could be utilized. By extracting the features on the wireless sensor node, the amount of communicated data can be reduced.

While some researchers proposed calculating features on wireless sensor nodes for energy efficiency reasons (Van Laerhoven and Aronsen, 2007; Laerhoven et al., 2006; Lorincz et al., 2009), the actual energy consumption highly depends on the design parameters of the activity recognition and does not allow for generalized statements. There exists a trade-off between the added computational effort on the sensor side and the reduced wireless transmissions. This trade-off needs to be estimated at design time in order to substantiate early design decisions. As a solution, this paper introduces an energy trade-off model considering the important design parameters of the activity recognition system. Our contributions are:

- examining important design parameters of feature extraction influencing the energy consumption when performed on wireless sensor nodes,
- an energy trade-off model for application specific feature extraction configurations,
- an evaluation of our model by four feature set implementations on a wireless sensor node, and
- a delay analysis of our implementations.

The remainder of the paper is structured as follows: In Section 2 the related work is discussed, which is followed by a concept description in Section 3. In Section 4 our experimental setup is de-

Grützmacher, F., Hein, A., Beichler, B., Lepidis, P., Dorsch, R., Kirste, T. and Haubelt, C.
Energy Efficient On-Sensor Processing for Online Activity Recognition.
DOI: 10.5220/0006860100850092
In *Proceedings of the 8th International Joint Conference on Pervasive and Embedded Computing and Communication Systems (PECCS 2018)*, pages 85-92
ISBN: 978-989-758-322-3

scribed, followed by the discussion of our results in Section 5. In Section 6 conclusions are drawn.

2 RELATED WORK

As summarized in (Rault et al., 2017), the energy-efficient design of wearable sensor devices has been subject of research in the wireless sensor networks, activity recognition, healthcare, and AAL domains for many years. One general approach to reduce power consumption is the reduction of communication overhead using efficient protocols (Ye et al., 2002) or sensor network topologies (Younis and Fahmy, 2004).

Several software architectures for activity recognition systems have been studied to improve power consumption: disabling hardware sensors using prediction of activities has been proposed in (Gordon et al., 2012) and (Wang et al., 2009). Saving computational effort at a conceptual level can be done using feature selection (Yan et al., 2012), classifier selection (Liang et al., 2014), or fixed-point arithmetic (Anguita et al., 2013). These approaches are orthogonal to the approach investigated in the paper at hand and can be combined for additional energy savings.

The principal of communication reduction by performing on-board computations of wireless sensor nodes has been introduced by several researchers. Their works either focus on offline scenarios (Van Laerhoven and Aronsen, 2007; Laerhoven et al., 2006), sensor selection strategies (Lorincz et al., 2009; Van Laerhoven and Aronsen, 2007; Laerhoven et al., 2006; Zappi et al., 2008), or signal compression and abstraction approaches (Berlin and Van Laerhoven, 2012; Berlin and Van Laerhoven, 2010; Mamaghanian et al., 2011; Marcelloni and Vecchio, 2008).

However, except (Mamaghanian et al., 2011) which focuses on sensor data compression, none of the aforementioned works provide a quantified reduction of energy consumption of their approaches as no explicit comparison to reference scenarios transmitting raw data is done.

Rault et al. already indicated that a trade-off between communication reduction and added computational load has to be considered when performing sensor-side computation (Rault et al., 2017). While they propose to perform ”light” computations on the sensor to keep the energy consumption for the additional computations as low as possible, we can show that even calculating Fast Fourier Transform (FFT) based features on the wireless nodes sensor subsystems can lead to an overall energy reduction.

In (Grützmacher et al., 2017) we have shown for one configuration, i.e. a window-based zero-crossing rate on accelerometer data, that calculating it on the wireless sensor node can reduce the energy consumption by a considerable amount. Besides the evaluation of more sophisticated feature sets, the paper at hand studies the important design parameters of different configurations influencing the energy consumption of wireless sensor nodes.

3 CONCEPT

In online activity recognition with wireless sensor nodes, the prevalent setup involves sensors sampled at high sampling rates, e.g. 100Hz, which are transmitted wirelessly to a host system for further processing. As modern sensors already perform signal correction as a preprocessing step, the wireless communication usually takes place between the preprocessing and segmentation stages of an ARC (see Figure 1). Especially in settings which require energy efficient designs like smartphones, wireless sensor nodes or wearables, highly frequent wireless communication is not desired, when targeting a low energy consumption.

Examining the ARC, data reducing stages can be identified. The feature extraction stage for example is mainly implemented as a sliding window based feature extraction in most of the literature (Huynh and Schiele, 2005; Krüger et al., 2014; Atallah et al., 2011; Capela et al., 2015). It often drastically reduces the data rate, as a relatively small number of features is calculated from a high number of samples of a window. Thus, the calculation of features, which is included in an ARC anyway, could be shifted to the wireless sensor node. Figure 1 shows this principal. While this approach can reduce a considerable amount of energy, it highly depends on the design parameters of the activity recognition (e.g. a sliding window with 50% overlap halfs the data rate, compared to a sliding window with 75% overlap). On the other hand, the added computational effort has to be taken into account as well, as it increases the energy consumption of the processing units. The trade-off between reduced data rate and added workload has to be considered in order to estimate the resulting energy savings.

3.1 Estimation of Energy Savings

As different features are useful to detect different kinds of activities, the number and composition of features varies amongst different applications.

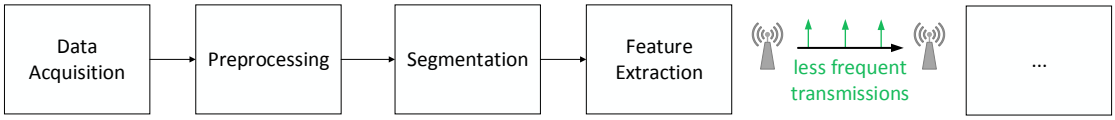

Figure 1: Activity recognition chain with on-sensor calculations up to and including the feature extraction stage.

Two aspects are decisive for the resulting energy savings: the workload of the microcontroller and the rate with which data is communicated over the wireless interface. We introduce an estimation method, which is based on a combination of two energy models of the target architecture: a model which captures the energy consumption of the communication, and a model which captures the additional computational load on the sensor device. As we are interested in the change of energy consumption when varying the data rate or the computational effort, both models are not required to capture the absolute energy consumption, but rather the relative average energy consumption to both parameters. Thus, the resulting model is referred to as *energy trade-off model*. It can be used to calculate the difference in energy consumption of two configurations.

As this model depends on two parameters, it becomes a surface in a three dimensional space. The structure of the surface depends on the relation between the energy consumption and the two model parameters. Note, that the model parameters need to be distinguished from the design parameters of the feature extraction. The model parameter capturing the amount of communicated data can be calculated from the design parameters of the feature extraction.

Since data is acquired from sensors, the amount of data can be naturally captured by the output frequency, of either raw samples or feature values. Therefore, it is intuitive to describe the first model parameter in terms of a data transmission frequency f_d. Thus, the energy dependency on the amount of communicated data is described by function $e_d(f_d)$. Note, that the size of the data types transmitted with f_d must equal, when comparing two configurations.

The computational effort on the other hand can be described in terms of processing load l_p in %, defined by the processing time t_p in a fixed time interval T:

$$l_p = \frac{t_p}{T} \tag{1}$$

Since the calculation of feature extraction usually is done in a sample-based or sliding-window based fashion, which both is triggered with the sampling frequency, it is obvious to use the sampling period as the time interval T. This simplifies the processing load to the average processing time per sample t_{sp}. The resulting energy dependency on the processing load is described as function $e_p(t_{sp})$. Note, that the sampling frequency describes at which rate sensor samples are acquired. This needs to be distinguished from the data transmission frequency, which captures the rate at which either raw samples or feature values are communicated from the wireless sensor node. As we will see in Section 4, both model parameters allow a fast calibration, if $e_d(f_d)$ and $e_p(t_{sp})$ are unknown for a particular sensor node.

The resulting equation for calculating the difference of the energy consumption Δe is:

$$\Delta e = e_d(f_{d_R}) - e_d(f_{d_T}) + e_p(t_{sp_R}) - e_p(t_{sp_T}), \tag{2}$$

with f_{d_R} and f_{d_T} being the output frequencies and t_{sp_R} and t_{sp_T} being the average computation times per sample of the *reference* and the *test* configuration respectively. Note, that this model assumes that computational load and the amount of communicated data influence the energy consumption in a additive way, and both components do not influence each other

As shown in Section 4, the energy consumption of our wireless sensor node on both model parameters shows a linear behavior.

In that case, the resulting model becomes a plane in a 3D space with the slopes m_d for $e_d(f_d)$ and m_p for $e_p(t_{sp})$. Thus, the equation for Δe resolves to:

$$\Delta e = (f_{d_R} - f_{d_T}) * m_d + (t_{sp_R} - t_{sp_T}) * m_p. \tag{3}$$

In the following we will explain, how both model parameters can be acquired from the design parameters of a certain set of features.

The output data rate f_d of the feature values mainly depends on the number of features N_F , and the sliding window rate f_{SW}. The latter is calculated from the sensor sampling frequency f_s in Hz and the sliding window parameters window size S_W in *Number of Samples* and window overlap O_W as a fraction on the interval (0,1), by:

$$f_{SW} = \frac{f_s}{(1 - O_W) * S_W}. \tag{4}$$

The parameter f_d can be calculated by:

$$f_d = f_{SW} * N_F, \tag{5}$$

for a configuration calculating the feature extraction on the wireless sensor node or directly by using $f_d = f_s$ in a configuration of transmitting raw sensor samples. As we can see from Equations (3), (4), and (5),

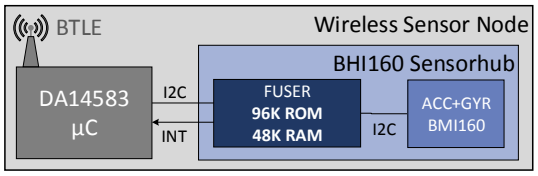

Figure 2: Architecture of our custom wireless sensor node.

the resulting absolute data rate reduction depends on the sampling frequency. Thus, the impact of energy savings by calculating feature extraction on the wireless sensor node increases with the sampling rate of the sensor.

For determining the workload dependency $e_p()$ of the target configuration, different approaches are possible. A simple but fast way is the use of energy models which depend on the execution time of the feature extraction code. Such models can be acquired by experiments as we have done for our sensor node like shown in Section 4.1. The workload can be calculated by Equation (1), which only requires an estimation of the average processing time per sample t_{sp} of the feature extraction code.

The processing time t_{sp} can be acquired by doing worst-case execution time analysis on the assembler code compiled for the target architecture. Feature extractions are usually performed using data flow oriented algorithms with few branches. Furthermore, different execution paths usually have neglectable differences in their path lengths. Thus, the instruction count of the assembler code together with the microcontroller frequency and the cycles per instruction acquired from their data sheets can be a fast way of estimating the execution time. This should be possible for most wireless sensor nodes, as mainly microcontroller architectures are chosen, because of their low power consumption. Typically microcontroller architectures neither have deep pipelines nor operating systems performing dynamic scheduling. Thus, the execution time of the assembler code is not subject to essential variations.

However, there also exist more fine grained approaches, which directly model the energy consumption of assembler instructions for a piece of source code (Ruberg et al., 2015; Bazzaz et al., 2013).

4 EXPERIMENTS

In our experiments we used a custom wireless sensor node which was designed for low power applications. It is equipped with an ultra low power BHI160 sensor hub from Bosch Sensortec (Bosch Sensortec, 2017), a Dialog Semiconductor DA14583 microcontroller with Bluetooth Low Energy (BTLE) interface, acting as a host controller which is connected with the BHI160 via I^2C and a CR1225 coin cell battery as power supply. For an overview see Fig. 2. In our setup, the sensor node either sends raw samples or calculated features to a Samsung Galaxy S5 smartphone, acting as a data aggregating device. All on-sensor feature extractions are performed on the BHI160's integrated *FuserCore* in our experiments.

When evaluating the energy consumption of the system we followed a similar approach introduced by (Russell and Jacome, 1998) since it meets the necessary requirements for comparing average energy consumptions and it is easy to deploy. For current measurement we are using a $10\,\Omega$ shunt resistor R_S in series with the power source. In our experiments we substituted the coin cell by a constant voltage source providing $3.0\,V$ supply voltage. The voltage drop U_S over R_S is captured and averaged by a DSO-X 3034A oscilloscope from Agilent Technologies over a time of at least $100\,s$ with a resolution of $10\,MSamples/s$.

Due to its proportional nature, the average energy consumption can be deduced from the measured average voltage drop with sufficient accuracy. The temperature dependency was considered to be neglectable in our setup, which could be substantiated with our results for the model accuracy in Section 5.1. When comparing the average voltage drop over time of different scenarios of the sensor, the systematic error of the measuring apparatus and the tolerance of the shunt resistor is canceled out and thus, propagation of uncertainty can be neglected for our evaluations.

4.1 Model Calibration

In order to show how much energy could be saved by reducing data communication, we measured the energy consumption of our wireless sensor node providing accelerometer and gyroscope data at different output data rates transmitted over BTLE. While varying the output data rates, all other factors like sensor sampling rate, work load and energy mode are kept constant for all measurements. The output data rate is reduced by just skipping the corresponding number of samples to be propagated to the sensor output. Fig. 3 shows the overall average energy consumption of the sensor node at different output data rates as well as the energy consumption without BTLE transmissions, to show the fraction of energy consumed by wireless transmissions.

It can be seen that the amount of data send via BTLE can have a great impact on the nodes energy consumption, especially at data rates like $100\,Hz$ or more, which are broadly used in the literature con-

cerned with activity recognition.

In order to show the energy increase of the added workload when calculating additional algorithms on a sensor subsystem, we implemented synthetic processing load by performing Multiply and Accumulate (MAC) operations in a loop and measured their energy consumption as well as the average computation time. We used MAC operations, since our feature implementation including online versions of statistical feature calculations as well as an FFT implementation, mainly consist of MAC operations. The processing load is defined as the fraction of time the processing unit is actively performing the additional calculations, as described by Equation (1). The experiments were done in the operating point at transmitting accelerometer and gyroscope values with $2.5\,Hz$ as shown in Fig. 3. The resulting dependency can be seen in Fig. 4. The trade-off between reduced data rate and additional workload gets obvious, as reductions in energy consumption caused by decreasing the output data rate from $60\,Hz$ to $2.5\,Hz$ can already be neutralized when the additional workload for doing that is more than approximately 57%. This shows, that the trade-off between reduced communication and additional processing load needs to be explicitly evaluated for different configurations.

To calibrate our model we used the experimental results of the aforementioned measurings. As the energy dependency of both model parameters is linear, the model is described by Equation (3). Both slopes, m_d and m_p can be determined by calculating a regression of measuring values acquired for the overall energy consumption. These lines are colored green in Figures 3 and 4. The slopes of the regression lines ($m_d = 0.0654172\,mV/Hz$ and $m_p = 0.067873\,mV/average\ workload\ \%$) are used as the calibrated model parameters.

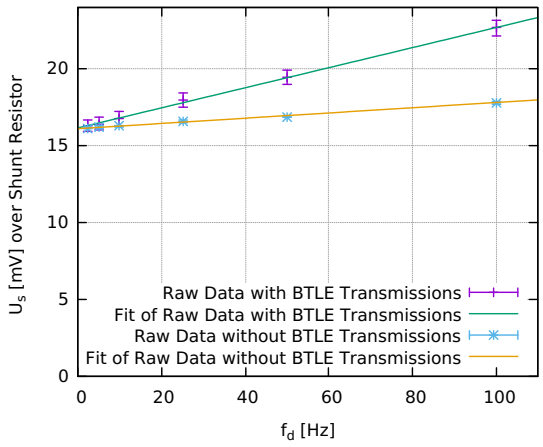

Figure 3: The sensor nodes average energy consumption at different output data rates of both accelerometer and gyroscope with and without BTLE communication.

4.2 Feature Sets

We implemented a feature extraction configuration on the BHI160 sensor subsystem which showed good results for activity and intention recognition for meal preparation and related activities in a kitchen scenario (Krüger et al., 2014). This feature set consists of *mean*, *variance*, *skewness*, *kurtosis*, *peak* (dominant frequency), and *energy* (magnitude of the dominant frequency) on accelerometer and gyroscope data. Both signals are sampled at $100\,Hz$ and segmented by a sliding window of 128 samples with an overlap of 75%. This leads to a 12-dimensional feature vector with a sliding window rate of $3.125\,Hz$ and is referred to as *MVSKPE*. This setup includes frequency domain features by calculating an 128-point FFT. Due to its high computational effort, we also implemented a more lightweight feature set, by substituting peak and energy by a zero-crossing rate (ZCR), since it can also give an idea about with which average frequency an sensor is shaken. This setup is referred to as *MVSKZ*. In some literature, only statistical time-domain features are used with mean and variance at least (see (Damaševičius et al., 2016)). Therefore, we also tested a feature set of mean, variance, skewness and kurtosis referred to as *MVSK*, and a set of mean and variance only, referred to as *MV*. All feature sets have been implemented on the BHI160's FuserCore on our wireless node and its BTLE connection interval was chosen as the maximal possible interval which could still provide the necessary corresponding data rate. In Fig. 5, our setup is shown.

In order to keep the computational effort for calculating features on the sensor as low as possible, we implemented the online version of mean, variance, skewness, and kurtosis from (John D. Cook, 2018), which is based on the one pass algorithms

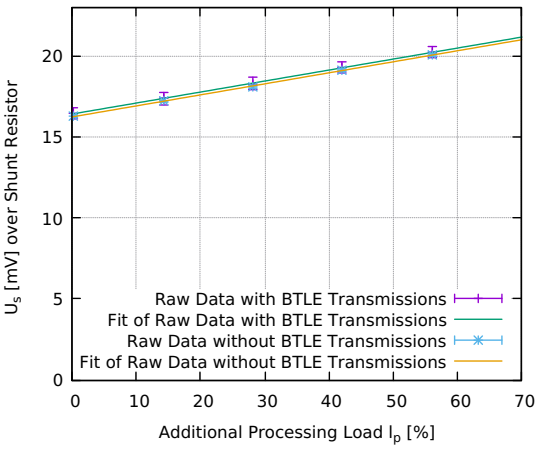

Figure 4: Sensor nodes average energy consumption at 2.5 Hz output data rate and different additional workloads with and without BTLE communication.

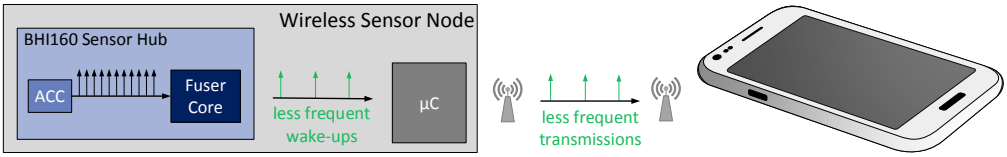

Figure 5: Shifting the feature extraction to the sensor hub reduces the wake-up frequency of the host and wireless transmissions.

for arbitrary-order statistical moments from (Pebay, 2008). This enables those features to be updated for each new sample with a computational complexity O(1). The implemented ZCR is a comparison of a new sample with its preceding sample to detect a zero crossing accumulated over the time of a sliding window. The implemented FFT is a 128-Point Radix-2 implementation with 7 butterfly stages. All algorithms use single precision floating point arithmetics.

5 RESULTS

The energy consumptions of MV, MVSK, MVSKZ, and MVSKPE as well as the reference scenario of sending raw accelerometer and gyroscope data at 100 Hz to a smartphone were measured. The measured improvement in energy consumption of each feature set compared to the reference scenario is shown in column "Meas." in Table 1. It can be seen that by implementing our chosen feature sets on the sensor node, the energy consumption is reduced by 20.53% to 26.8% compared to our reference scenario, depending on the feature set.

Note, that the relative energy savings depend on the overall energy consumption, including the energy consumption independent of the sensors workload and the BTLE transmission. Reducing this part, e.g. by utilizing sleep modes of the host controller, has a potential for considerable further reductions of the overall energy consumption and thus a higher impact of the studied approach.

From the model parameters of the implemented feature sets in Table 2, it can bee seen that the data reduction of our implementations range from 81% to 93%, depending on the chosen feature configuration. This results can cope with state of the art data compression algorithms for wireless sensor nodes, without neither introducing additional processing effort, nor loosing information. Note, that by shifting the feature extraction to the sensor subsystem of our node, less computations on the smartphone have to be performed as well, which should reduce its energy consumption as well. However, this is not further evaluated in this work.

Table 1: Predicted and Measured Average Energy Savings.

Features	Pred. [%]	Meas. [%]	Pred. Error [%]
MVSKPE	20.60	20.53	0.5%
MVSKZ	23.68	23.70	0.1%
MVSK	24.63	24.61	0.1%
MV	27.02	26.87	0.6%

5.1 Model Accuracy

In order to evaluate the accuracy of our energy trade-off model, we acquired the average computation time of the reference scenario and the feature sets calculated on the BHI160's FuserCore. We measured the average computation time instead of using one of the prediction methods from Section 3.1, in order to only evaluate the model accuracy without additional prediction uncertainties, as they may vary among different methods. The results are shown in row l_p of Table 2. It can be seen that by calculating the ZCR instead of the FFT features, the average workload can be reduced from 10% to 4%.

With the differences of the average workload l_p (Equation (1)) and the output frequency f_d (Equation (5)) between the feature set to be tested and the reference scenario, the savings in energy consumption were estimated by our model (Equation (3)). The model parameters used for the evaluation are shown in Table 2. We compared the estimated savings with the measured results for the selected feature sets and calculated the error, which can be seen in Table 1. The prediction error of our model is less than 1%, which is small enough to make feasible estimations of energy savings when considering feature calculations on the sensor node at design time. Note, that the uncertainty of the methods for predicting the additional average workload from an existing implementation (Section 3.1) will further affect the overall prediction error correspondingly.

Table 2: Model parameters for the selected features sets.

Features	avg. workl. l_p [%]	output rate f_d [Hz]
Raw	0.3	100
MVSKPE	10.4	18.75
MVSKZ	3.3	15.625
MVSK	3.2	12.5
MV	1.3	6.25

Table 3: Actual and achieved processing delays.

Features	proc. delay [ms]	delay in our setup [ms]
MVSKPE	22.8	120
MVSKZ	0.34	110
MVSK	0.35	100
MV	0.14	80

5.2 Delay Analysis

As already stated in (Rault et al., 2017), the sensor-side computation of features can influence the latency of the activity recognition system. Therefore, we analyzed the processing delay for the features calculated on the sensor. With timestamp functions we measured the additional processing time of the feature extraction implementations.

We calculated the average processing time of at least 200 measurements per feature set. These results can be seen in Table 3 as "proc. delays". With calculating the feature extraction on the sensor itself, multiple values (feature vector entries) per sliding window are propagated to the sensor output in one batch, in a very short time. As in our setup a microcontroller is acquiring the sensor data for transmitting via BTLE, which has a limited input data rate due to it's firmware implementation, we had to delay the output of features on the sensor subsystem artificially. When the features are calculated for a sliding window, the output of each feature vector entry is delayed until the next sensor value is sampled, which is 10 *ms* in our case. Since the feature vector for scenario MVSKPE has 12 entries which is the greatest feature vector among our feature sets, the maximum delay in our experiments is 120 *ms*. This and the delay for the other feature sets can be seen in Table 3 as "delay in our setup". Note, that the artificial delay in our setup was necessary because of our firmware implementation on the sensor node. With a BHI160 evaluation board, we verified that this is not a general issue, but caused by our particular setup. Only the feature entries for the peak frequency and its magnitude have to be delayed by approx. 30 *ms*, since the FFT calculation for all 6 axis takes longer than the 10 *ms* sampling period and has to be distributed among 3 sampling periods. However, even 120 *ms* is a fairly good result for most activity recognition tasks. Furthermore, the calculation of features on multiple wireless sensor nodes is inherently done in parallel, which might not be the case when calculating the feature extraction for all sensors on a data aggregating device, depending on its parallel processing capabilities.

6 CONCLUSION

Our work investigates the idea of calculating the feature extraction stage of activity recognition algorithms on wireless sensor nodes. We have shown the outcomes of this approach by porting four broadly used feature sets to a BHI160 sensor subsystem residing on a wireless sensor node and compared the energy savings to reference scenarios transmitting raw sensor samples instead. In our experiments, this approach reduced the energy consumption of the wireless sensor node by up to 27%.

To show the applicability of on-sensor feature extractions, we evaluated the processing delay of our implementations, which was 120 *ms* at most.

More importantly, in our work we examined the decisive design parameters for the resulting energy trade-off when calculating data reducing stages like feature extraction on wireless sensor nodes. From these design parameters an energy trade-off model is built which allows to estimate the energy savings of sensor side feature calculations at design time. This improves the development process of activity recognition systems as design decisions can be substantiated early in the design process. By calculating the energy savings from the model fitted to our wireless sensor node and comparing it to four implemented feature sets in our experiments, we achieved a model accuracy of more than 99%.

ACKNOWLEDGMENTS

This work is partially supported by the German Federal Ministry of Education and Research (BMBF), grant number 03ZZ0519D.

REFERENCES

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. (2013). Energy efficient smartphone-based activity recognition using fixed-point arithmetic. *J. UCS*, 19(9):1295–1314.

Atallah, L., Lo, B., King, R., and Yang, G.-Z. (2011). Sensor positioning for activity recognition using wearable accelerometers. *IEEE transactions on biomedical circuits and systems*, 5(4):320–329.

Bazzaz, M., Salehi, M., and Ejlali, A. (2013). An accurate instruction-level energy estimation model and tool for embedded systems. *IEEE Transactions on Instrumentation and Measurement*, 62(7):1927–1934.

Berlin, E. and Van Laerhoven, K. (2010). An on-line piecewise linear approximation technique for wireless sensor networks. In *Proc. of LCN'10*, pages 905–912. IEEE.

Berlin, E. and Van Laerhoven, K. (2012). Detecting leisure activities with dense motif discovery. In *Proc. of UbiComp'12*, pages 250–259. ACM.

Bosch Sensortec (2017). BHI160 / BHI160B - Ultra low-power sensor hub incl. integrated imu. BST-BHI160(B)-DS000-01.

Bulling, A., Blanke, U., and Schiele, B. (2014). A tutorial on human activity recognition using body-worn inertial sensors. *ACM Comput. Surv.*, 46(3):33:1–33:33.

Capela, N. A., Lemaire, E. D., and Baddour, N. (2015). Feature selection for wearable smartphone-based human activity recognition with able bodied, elderly, and stroke patients. *PloS one*, 10(4):e0124414.

Damaševičius, R., Vasiljevas, M., Šalkevičius, J., and Woźniak, M. (2016). Human activity recognition in aal environments using random projections. *Computational and mathematical methods in medicine*, 2016.

Gordon, D., Czerny, J., Miyaki, T., and Beigl, M. (2012). Energy-efficient activity recognition using prediction. In *Proc. of ISWC '12*, pages 29–36.

Grützmacher, F., Wolff, J.-P., Hein, A., Lepidis, P., Dorsch, R., Kirste, T., and Haubelt, C. (2017). Towards energy efficient sensor nodes for online activity recognition. In *Proc. of IECON'17*, pages 8291–8296. IEEE.

Huynh, T. and Schiele, B. (2005). Analyzing features for activity recognition. In *Proc. of sOc-EUSAI '05*, pages 159–163, New York, NY, USA. ACM.

John D. Cook (visited 03/2018). Computing skewness and kurtosis in one pass. https://www.johndcook.com/blog/skewness_kurtosis/.

Krüger, F., Nyolt, M., Yordanova, K., Hein, A., and Kirste, T. (2014). Computational state space models for activity and intention recognition. a feasibility study. *PloS one*, 9(11):e109381.

Laerhoven, K. V., Gellersen, H.-W., and Malliaris, Y. G. (2006). Long term activity monitoring with a wearable sensor node. In *Proc. of BSN'06.*, pages 4–pp. IEEE.

Liang, Y., Zhou, X., Yu, Z., and Guo, B. (2014). Energy-efficient motion related activity recognition on mobile devices for pervasive healthcare. *Mobile Networks and Applications*, 19(3):303–317.

Lorincz, K., Chen, B.-r., Challen, G. W., Chowdhury, A. R., Patel, S., Bonato, P., Welsh, M., et al. (2009). Mercury: a wearable sensor network platform for high-fidelity motion analysis. In *SenSys*, volume 9, pages 183–196.

Mamaghanian, H., Khaled, N., Atienza, D., and Vandergheynst, P. (2011). Compressed sensing for real-time energy-efficient ecg compression on wireless body sensor nodes. *IEEE Transactions on Biomedical Engineering*, 58(9):2456–2466.

Marcelloni, F. and Vecchio, M. (2008). A simple algorithm for data compression in wireless sensor networks. *IEEE communications letters*, 12(6).

Pebay, P. P. (2008). Formulas for robust, one-pass parallel computation of covariances and arbitrary-order statistical moments. Technical report, Sandia National Laboratories.

Rault, T., Bouabdallah, A., Challal, Y., and Marin, F. (2017). A survey of energy-efficient context recognition systems using wearable sensors for healthcare applications. *Pervasive and Mobile Computing*, 37:23–44.

Ruberg, P., Lass, K., and Ellervee, P. (2015). Microcontroller energy consumption estimation based on software analysis for embedded systems. In *Proc. of NORCAS'15*, pages 1–4. IEEE.

Russell, J. T. and Jacome, M. F. (1998). Software power estimation and optimization for high performance, 32-bit embedded processors. In *Proc. of ICCD'98*, pages 328–333. IEEE.

Van Laerhoven, K. and Aronsen, A. K. (2007). Memorizing what you did last week: Towards detailed actigraphy with a wearable sensor. In *Proc. of ICDCSW'07.*, pages 47–47. IEEE.

Wang, Y., Lin, J., Annavaram, M., Jacobson, Q. A., Hong, J., Krishnamachari, B., and Sadeh, N. (2009). A framework of energy efficient mobile sensing for automatic user state recognition. In *Proc. of Mobisys '09*, pages 179–192. ACM.

Yan, Z., Subbaraju, V., Chakraborty, D., Misra, A., and Aberer, K. (2012). Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *Proc. of ISWC '12*, pages 17–24. IEEE.

Ye, W., Heidemann, J., and Estrin, D. (2002). An energy-efficient mac protocol for wireless sensor networks. In *Proc. of INFOCOM '02*, volume 3, pages 1567–1576. IEEE.

Younis, O. and Fahmy, S. (2004). Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on mobile computing*, 3(4):366–379.

Zappi, P., Lombriser, C., Stiefmeier, T., Farella, E., Roggen, D., Benini, L., and Tröster, G. (2008). Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection. In *Wireless sensor networks*, pages 17–33. Springer.