Power-Aware Action Recognition with Optimal Sensor Selection: An AdaBoost Driven Distributed Template Matching Approach

Pasquale Panuccio
Univ. of Calabria
Rende(CS), Italy
panuccio@si.deis.unical.it

Hassan Ghasemzadeh
Univ. of California, Los Angeles
Los Angeles, CA 90095
hassan@cs.ucla.edu

Giancarlo Fortino
Univ. of Calabria
Rende(CS), Italy
g.fortino@unical.it

Roozbeh Jafari
Univ. of Texas at Dallas
Richardson, TX 75083
rjafari@utdallas.edu

Abstract

In this paper, we present a distributed action recognition framework that minimizes power consumption of the system subject to a lower bound on the classification accuracy. The system utilizes computationally simple template matching blocks that perform classifications on individual sensor nodes. A boosting approach is employed to enhance accuracy by activating only a subset of sensors optimized in terms of power consumption and can achieve a given lower bound accuracy criterion. Our experimental results on real data shows more than 85% power saving while maintaining 80% sensitivity to detected actions.

Categories and Subject Descriptors

C.3 [**Computer Systems Organization**]: Special Purpose and Application-Based Systems—*Real-time and embedded systems*; J.3 [**Computer Applications**]: Life and Medical Science—*Health*; H.1.2 [**Information Systems**]: Models and Principles—*User/Machine Systems Human information processing; Human factors.*

General Terms

Design, Algorithms, Experimentation.

Keywords

Action Recognition, Collaborative Classification, AdaBoost, Power Optimization

1 Introduction

With the growing interest in wireless health technologies and their potential applications, efficient design and development of wearable medical devices is becoming unprecedentedly important to researchers in both academia and industry. The main driving factors include cost, power consumption, and wearability, with power consumption being the center of many research efforts due to its dramatic influence on other design objectives. An important angle of the low-power design is development of efficient signal processing and data reduction algorithms that reduce computation load of the processing units. Designing power-aware signal processing algorithms for action recognition is challenging as special care needs to be taken to maintain acceptable classification accuracy while minimizing the energy consumption.

In this paper, we propose a novel power-aware action recognition framework that is designed to minimize power consumption of the system while guaranteeing a lower bound on the classification accuracy of the system. The low-power signal processing is accomplished by: 1) using lightweight classifiers that operate based on template matching on sampled inertial data; and 2) eliminating sensors that are irrelevant to the action recognition. The high classification performance is maintained using a boosting approach that combines results obtained from distributed sensors and enhances the accuracy by taking into account contribution of individual sensors. Our approach is novel in the sense that it combines two concepts 1) classifier boosting to enhance classification performance 2) sensor selection to minimize power consumption of the system. To the best of our knowledge, this approach has not been investigated previously.

2 Related Work

Reducing amount of active nodes is a common approach for power optimization and wearability enhancement in BSNs. Authors in [1] formulate coverage problem in the context of movement monitoring using inertial on-body sensors to minimize the number of sensor nodes that produces full action coverage set. Authors in [2] propose to optimize the system energy consumption by selecting the required subset of sensors with the help of the meta-classifier sensor fusion. Therefore it is sufficient to turn on sensors only when their values are needed to ensure correctness.

The idea of combining simple classifiers to achieve higher accuracy is discussed in [3] where authors suggest using a

mHealthSys'11, Nov 1, Seattle, Washington, USA.

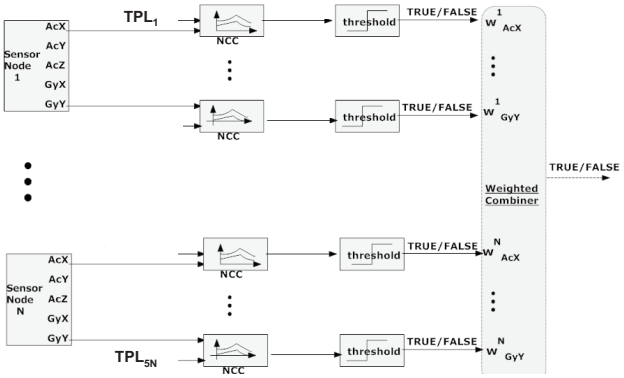

Figure 1. Template matching and boosting approaches for action recognition.

single accelerometer for activity monitoring and combining classifiers using Plurality Voting. In [4] AdaBoost is used to select a small number of features in order to ensure fast classification. The algorithm automatically selects the best features and ranks them based on their classification performance.

3 Collaborative Action Recognition

Our first contribution in this paper is an effective algorithm for detecting human actions. The algorithm takes into consideration collaboration between sensor nodes to make a classification decision. A block diagram of our classification process is illustrated in Figure 1. Each sensor stream is associated with a binary classifier that runs on the sensor node. The classifier performs coarse analysis on the motion signals by comparing the incoming signal with a predefined template associated with the target action. The template matching generates a similarity score between the signal and the target template. The calculated score is compared against a threshold in order to classify the incoming signal into one of the true/false values. A true class indicates that the current action is classified as target while a false decision infers a non-target action.

We assume that, due to the nature of the application, the sensor nodes do not require to know the classification results. Furthermore, we note that the target action is an "a priori" known class label. Thus, every node knows what target class is going to be detected. The goal is to find out whether or not the current action is the target action.

3.1 Template Matching

Our classification approach is based on template matching as illustrated in Figure 1. The goal is to find portions of the signal are most similar to a given template. There are different ways to calculate the similarity of a time series signal with a predefined template. For this study, we use Normalized Cross Correlation (*NCC*). While we have found NCC effective in enabling template matching operations for action recognition, our classification algorithms are completely independent of the choice of the similarity measure.

NCC is robust to different amplitudes of the signal and less sensitive to the noise. In our case the time series are represented by the data sampled from inertial sensors (accelerometers and gyroscopes) whose signal characteristics may vary from one action to another, or from one subject to another. Let $S(x)$ be the time series signal that is associated with the signal segment generated by a given sensor. Further, let $TPL(x)$ be the time series signal corresponding to the target template, the NCC value between these two signals is given by

$$\gamma(TPL,S) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left((S(x_i)-\bar{S})(TPL(x_j)-\bar{TPL})\right)}{\sqrt{\sum_{i=1}^{M}\sum_{j=1}^{N}\left((S(x_i)-\bar{S})^2(TPL(x_j)-\bar{TPL})^2\right)}} \quad (1)$$

where M and N denote lengths of S and TPL respectively and $\bar{S}$ and $\bar{TPL}$ represent their mean values respectively.

3.2 Template Generation

The raw sensor readings are collected during the experiment along with the video recording of the actions. The collected signals are then segmented and labeled manually for each experimental action. Video recording is used to segment the data in a more fine-grained manner. This manual segmentation ensures precise segmentation and labeling of the data, which would prevent injecting automatic segmentation errors to subsequent signal processing and pattern recognition blocks. Our templates are generated by comparing every pair of training instances and choosing the one which is most similar to the others. As such, the NCC measure is used to calculate similarity score between pairs of training instances. Furthermore a simple normalization is used to make the incoming signal and target template uniform in length.

3.3 Weak Classifiers

In-node classification is accomplished using a threshold value for each axis of the sensors. This threshold is calculated during training and is set to a value between an upper bound and a lower bound. The bounds are the average of the cross correlation values by comparing the template with target and non-target classes. Each weak classifier makes a decision on the incoming signal as follows. For each axis of the sensors (e.g. 3 for accelerometer and 2 for gyroscope data used in our experiments), the corresponding signal is used to perform the template matching and compare the incoming signal with the previously determined template. If the result of the comparison is greater than the threshold, the signal is labeled as target action. Otherwise, the signal is classified as a non-target action.

3.4 Weighted Combiner

A decision fusion approach is used for final classification. Our decision fusion algorithm is a classifier combiner which takes into account the amount of contribution each weak classifier can make to the overall accuracy of the system. The algorithm works with a set of training instances that are correctly labeled with the appropriate actions. Each instance consists of a sensor reading (i.e. x_i) and a label (i.e. y_i), and is associated with a weight that is equal for all the instances of the same action. Let L be the learning algorithm that generates the hypothesis $h_t(x)$. The algorithm L is the method of finding the threshold, and $h_t(x)$ is the function of our weak classifier that produces true/false labels. If the outcome of the comparison between instance x and target action

Algorithm 1: Weighted Combiner Model

data: TS training set $(x_1,y_1),(x_2,y_2),\ldots,(x_N,y_N)$,
L learning algorithm generating hypothesis $h_t(x)$ classifier,
N size of training set

initialize: d_n weight of instance n (d is a distribution with $1=\sum_{n=1}^{N} d_n$) $d_n = 1/N$

input : T max number of hypotheses in the ensemble

begin

  for $t = 1,\ldots,|T|$ **do**

    1. Train weak learner and obtain hypothesis h_t
    2. Compute error $\varepsilon_t = \sum_{n=1}^{N} d_n I(y_n \neq h_t(x_n))$
    3. Compute hypothesis weight $\beta_t = \frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$

output: combined hypothesis $f_{Com}(x) = \sum_{t=1}^{T} \beta_t h_t(x)$

template is greater than or equal to the threshold value, a true label is generated. Each weak classifier is associated with a weight, depending on the classification error that is obtained during training. The weak classifiers that achieve a smaller error are assigned larger weights to signify their contribution to the overall system performance.

4 Power-Aware AdaBoost

Classical AdaBoost learns from a set of weak classifiers and boosts classification performance by allocating weights to the individual local classifiers. There are two main drawbacks with this approach: 1) AdaBoost examines all classifiers even if they provide less informative data for classification. Contribution of different sensors to action recognition varies depending on the placement of the sensor, type of inertial information, and actions of interest. 2) AdaBoost does not take into account the power requirements of the individual weak classifiers while learning from weak classifiers. However, power consumption of the classifiers varies depending on the type of sensors, computing complexity, and data communication requirements. For example, gyroscopes generally consume much more power than accelerometers. Therefore, the optimal subset of the classifiers needs to be selected by making appropriate design tradeoff between accuracy and power consumption.

4.1 Problem Formulation

Let S=$\{s_1, s_2, \ldots, s_n\}$ be a set of n sensor nodes used to distinguish between a target action, $\hat{a}$, and a set of m non-target actions, A=$\{a_1, a_2, \ldots, a_m\}$. A sensor node, s_i, is a physical wearable node that has limited processing power and storage and can communicate within certain range, and is composed of l sensors that capture inertial data from human movements. The sensor node used for our experimental verification has l=5 sensors including three axes of accelerometer and two axes of gyroscope readings.

DEFINITION 1 (WEAK CLASSIFIER). *Each sensor node s_i consists of l weak classifiers, $C_i = \{C_{i1}, C_{i2}, \ldots, C_{il}\}$. Each classifier C_{ik} is associated with one of the sensors available on s_i, and is a binary classifier that operates based on template matching and makes a classification decision using the similarity score obtained from NCC. The classifier determines whether or not an incoming signal is classified as the target action.*

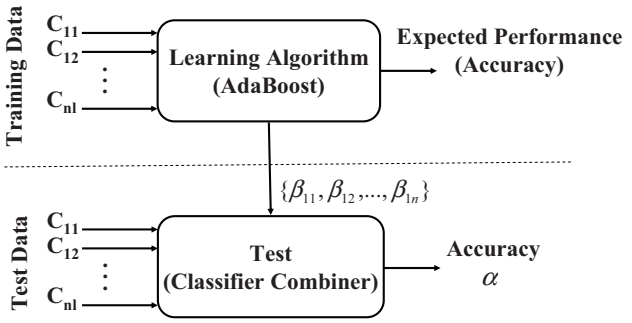

Figure 2. Learning algorithm and classifier combiner during training and test

Given the n nodes and l sensors per node, the system has a total of $T = n \times l$ weak classifiers. Figure 2 shows how learning parameters (e.g. weights $\beta_{11},\ldots,\beta_{nl}$) are generated during training. The learning algorithm can also provide an estimate of the expected accuracy of the entire classification, β.

To calculate power consumption of the set of active classifiers, we consider two sources of power consumption, namely computation and communication costs. We assume that each classifier is associated with a specific sensor (e.g. x-accelerometer, y-gyroscope) and therefore has a fixed computation cost depending on whether or not it is activated.

DEFINITION 2 (COMPUTATION COST). *For each classifier C_{ij}, we define w_{ij} as the computation cost associated with power consumption of the corresponding sensor. This value is a priori known and has a non-zero value for active classifiers while it is zero for non-active ones. Thus, the total computation costs is given by*

$$P_{comp} = \sum_{i=1}^{n}\sum_{j=1}^{l} x_{ij} w_{ij} \tag{2}$$

where x_{ij} is a binary variable that denotes whether or not classifier C_{ij} is active.

DEFINITION 3 (COMMUNICATION COST). *For a set of weak classifiers used for learning, the communication cost is given by*

$$P_{comm} = \sum_{i=1}^{n} f(\sum_{j=1}^{l} x_{ij} b_{ij}) \tag{3}$$

where $f(.)$ denotes the communication cost due to transmission of certain amount of data, and x_{ij} denotes if classifier C_{ij} is activated, and b_{ij} represents the amount of data that is generated by classifier C_{ij} and needs to be transmitted to the basestation.

We note that the communication costs are calculated for each sensor node rather than individual sensors/classifiers. This is mainly due to merging results of all active classifiers at each node prior to transmission to the basestation. The power consumption of the system due to activation of a set of weak classifiers used for learning is then given by

$$Z = P_{comp} + P_{comm} \tag{4}$$

PROBLEM 1. *Given a set of* $T = \bigcup C_i = \{C_{11}, C_{12}, \ldots, C_{nl}\}$ *classifiers, and also data units* b_{ij} *and computation cost* w_{ij} *for each classifier* C_{ij}*, the problem of Minimum Cost Classifier Selection (MCCS) is to find a subset of* C_{ij} *with minimum total cost while a lower bound of* $\alpha \geq F$ *on the overall accuracy is met.*

Therefore, the optimization problem can be written as follows.

$$\text{Minimize} \sum_{i=1}^{n} \sum_{j=1}^{l} x_{ij} w_{ij} + \sum_{i=1}^{n} f(\sum_{j=1}^{l} x_{ij} b_{ij}) \tag{5}$$

Subject to:

$$\alpha \geq F \tag{6}$$

4.2 Problem Complexity

The optimization problem presented in (5) poses several difficulties in arriving at an optimal solution: 1) The communication cost in (3) is a concave function and minimizing a concave function is considered to be hard in general [5]. 2) The constraints of the optimization problem in (6) are nonlinear inequalities, which makes the minimization problem even harder. In order to develop a polynomial-time algorithm for the MCCS problem, we simplify some of the assumptions made for our classification model. In particular, transforming the concave communication cost function into a linear function of the transmitted data can turn the objective function into a convex function. One specific property of our weak classifiers is that they produce very small amount of data per classification. Therefore, local results can be combined to form a small fixed-size packet. This would turn our optimization problem into minimizing the overall computation cost. Thus, the objective function in our optimization problem can be rewritten as follows.

$$\text{Minimize} \sum_{i=1}^{n} \sum_{j=1}^{l} x_{ij} w_{ij} \tag{7}$$

We note that the problem could be transformed to a covering problem if contribution of individual sensors was mutually exclusive. Unfortunately, information inferred from the sensors are highly correlated. That is, the accuracy obtained from combining two sensors is not equal to the summation of the accuracy values from individual sensors.

4.3 Greedy Approach

In this section, we present a greedy heuristic algorithm that takes power consumption and accuracy of the weak classifiers into account and finds a subset of least power consuming classifiers that maintain an overall accuracy of at least equal to a given value. Our greedy algorithm is a backward elimination algorithm that starts with all classifiers being considered for AdaBoost-driven learning. The set of all classifiers can potentially achieve a maximum accuracy that might be much higher than the lower bound accuracy. However, this configuration is highly power consuming as it uses all available sensors for action recognition. The algorithm iterates through different steps until it stops given a condition on overall accuracy of the classification. At each stage of the algorithm one of the classifiers is eliminated. The elimination criterion is validated as a tradeoff between power consumption and accuracy. Specifically, the classifier whose $\frac{w_{ij}}{\beta_{ij}}$ is maximized is the candidate for elimination. We note that w_{ij} and β_{ij} represent the power consumption and significance of the classifier C_{ij}. Yet, at each step, the algorithm checks whether the overall accuracy is still above the minimum desirable value. The algorithm stops if elimination of the next classifier would decrease the overall accuracy to a value below the given desirable threshold, F. The algorithm is linear in the number of weak classifiers. Thus, the algorithm has a complexity of $O(n \times l)$.

No.	Description
1	Stand to Sit
2	Sit to Stand
3	Sit to Lie
4	Lie to Sit
5	Bend and Grasp
6	Kneeling, right leg first
7	Kneeling, left leg first
8	Turn clockwise 90 degrees
9	Turn counterclockwise 90 degrees
10	Move forward (1 step)
11	Move backward (1 step)
12	Jump

(a) Transitional actions

(b) Subject

Figure 3. Experimental setup

5 Experimental Verification

To demonstrate the effectiveness of our classification framework and power-aware sensor selection algorithm, we present in this section our results on both accuracy and power consumption of the system under the developed models.

The data were collected in a single experimental session from three male subjects all in good health conditions. The data included acceleration and angular velocity for the 12 transitional actions listed in Figure 3(a). Each subject was asked to perform each one of the actions 10 times while wearing the four sensor nodes shown in Figure 3(b). The sensor nodes were programmed to sample each sensor at 50 samples per second and transmit to a personal computer using a TDMA approach. The data were then segmented manually with the help of video. The data collected from the experiments were stored on a Laptop computer where we developed our signal processing algorithms and conducted the performance evaluation.

A set of experiments were conducted to identify each action listed in Figure 3(a). For each experiment, the particular action was considered as target action and the rest of the actions formed a non-target class. For each target action, we generate a template for parameter setting and threshold calculation. During execution of the system, an incoming signal is compared against the template.

To calculate the threshold, we compared the template with the 'true' and 'false' instances. By taking an average over all the similarity scores between the template and target action instances, we obtained an upper bound as well as a lower bound on the value of the threshold. For our experiments, we use a threshold that is calculated as given by

$$Thr = \frac{Thr_{upper} + Thr_{lower}}{3} \tag{8}$$

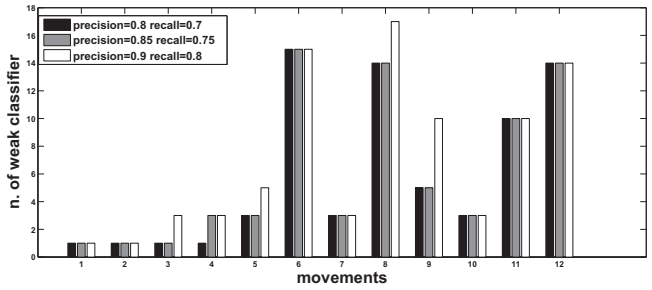

Figure 4. Number of active classifiers reported by the backward elimination greedy algorithm.

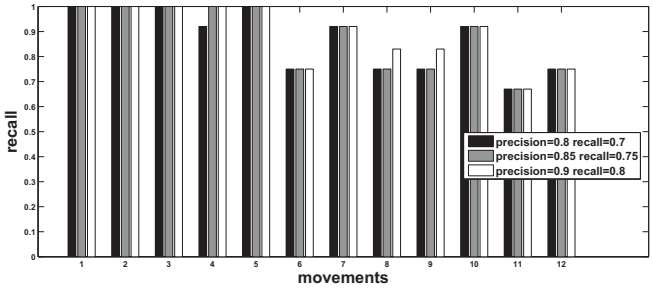

Figure 5. Measured recall due to using only active classifiers. Measured precision was 100**% for all actions except actions** 3**,** 4**, and** 6**.**

The next step in development of our classification framework is to build the *Weighted Combiner Model*. Inputs to this model are the threshold values. It generates a set of weights (β_{ij}) as output, each of which is associated with a weak classifier. Lower error rates for a particular classifier would result in larger weights assigned to that classifier. Classifiers that have larger weights would contribute more to the overall accuracy of the classification. This way, more accurate weak classifiers are considered as more significant contributors. Each axis of the sensors could be more or less significant depending on the location of the sensor node and movement of interest. For example, the sensor readings of the node placed on the ankle will carry more indicative information to detect the 'Moving Forward/Backward', compared to the action 'Sit to Stand'.

5.1 Detecting 'Sit to Stand'

In our first analysis, we trained the system for detecting 'Sit to Stand' as target action. The greedy algorithm was used to find the best subset of the sensors that can detect this action with a precision of $P \geq 90\%$ and a recall of $R \geq 80\%$. The backward elimination algorithm removed all weak classifiers except one sensor axis. The resulting active sensor was the 'Z-axis accelerometer' of the 'right thigh' node. This observation can be interpreted as follows. The Z-axis acceleration is the axis pointing the frontal plane of the subject and has a unique pattern during 'Sit to Stand'. This pattern is not repeated for the rest of actions, and therefore, is a distinguishing pattern which can by itself distinguish between this action and the others. This results in 95% reduction in the number of active sensors (from 20 to 1).

5.2 Per-Action Results

In this section, we present the results for individual actions being considered as target. We analyzed these results

Table 1. List of active sensors (weak classifiers) for detecting each target action.

Action	Desired accuracy threshold values(P=Precision,R=Recall, wk=N. of weak classifiers)					
	wk	**P=0.8 R=0.7**	**wk**	**P=0.85 R=0.75**	**wk**	**P=0.9 R=0.8**
1	1	AcZ-3	1	AcZ-3	1	AcZ-3
2	1	AcZ-3	1	AcZ-3	1	AcZ-3
3	1	AcZ-4	1	AcZ-4	3	AcX-1,AcY-2 AcZ-4
4	1	AcY-4	3	AcY-2,AcZ-3 AcY-4	3	AcY-2,AcZ-3 AcY-4
5	3	AcX-1,AcZ-2 AcZ-3	3	AcX-1,AcZ-2 AcZ-3	5	AcX-1,AcY-1 AcZ-2,AcZ-3 AcY-2
6	15		15		15	
7	3	AcX-1,AcZ-1 AcY-4	3	AcX-1,AcZ-1 AcY-4	3	AcX-1,AcZ-1 AcY-4
8	14		14		17	
9	5	AcX-2,AcY-3 AcX-4,AcY-4 AcZ-3	5	AcX-2,AcY-3 AcX-4,AcY-4 AcZ-3	10	
10	3	AcZ-1,AcZ-2 AcY-3	3	AcZ-1,AcZ-2 AcY-3	3	AcZ-1,AcZ-2 AcY-3
11	10		10		10	
12	14		14		14	

for three scenarios where performance of the classification varies. For each performance level, we found the minimum set of weak classifiers that achieve the given sensitivity (or recall) and precision values. The three performance levels are: "precision=0.8, recall=0.7" (level 1), "precision=0.85, recall=0.75" (level 2), and "precision=0.9, recall=0.8" (level 3).

5.2.1 Classifier Performance Analysis

Figure 4 shows the number of sensors (or weak classifiers) that are required to obtain the desirable performance. The number of active sensors clearly depends on the action being recognized. In most cases, only few sensors are enough to monitor the target action. However, there are cases such as actions 6, 8, 11 and 12 where a larger number of active sensors are needed. The list of the weak classifiers used during the classification task, for different levels of accuracy, are shown in Table 1. For visualization, the list of the active sensors is eliminated from the table for those actions that require a large number of active sensors. This is the case for actions 6, 8, 11 and 12 ('Kneeling', 'Turning', 'Move backward', and 'Jumping'), which require 15, 17, 10 and 14 active sensors for the highest calculated performance respectively. The first column in Table 1 refers to the actions listed in Figure 3(a). The second and third columns show number and name of active sensors detected with the first performance level (precision=0.8, recall=0.7) respectively. Similarly, fourth and fifth columns refer to active sensor desired performance is set to level 2. Finally, the last two columns show the results for performance level 3.

Figure 5 illustrates the results of the classification, in terms of the measured recall. The measured precision was 100% for 9 actions (all except actions 3, 4, and 6) in all the three desired performance levels. When classifying action 3, the measured precision was 85%, 85% and 90% for performance levels 1, 2, and 3 respectively. For action 4, the precision values were 85%, 100%, and 100% for the three performance levels respectively. Finally, when action 6 is considered as target action, the actual precision of the system was 85% in all the three performance levels.

Table 2. Power consumption due to power-aware learning

Mov.	W/ Alg. [mW]	W/o Alg. [mW]	Saving [%]
1	0.88	135.96	99.3
2	0.88	135.96	99.3
3	2.64	135.96	98.0
4	2.64	135.96	98.0
5	4.4	135.96	96.7
6	47.02	135.96	65.4
7	2.64	135.96	98.0
8	88.93	135.96	35.4
9	8.8	135.96	93.5
10	2.64	135.96	98.0
11	8.8	135.96	93.5
12	41.91	135.96	69.2
Avg.	17.68	135.96	87

5.2.2 Power Analysis

Reducing power consumption of the system usually results in a decrease in accuracy. In order to calculate the power consumption of individual sensors, we found the nominal values of power used by the accelerometers (LIS3LV02DQ) and gyroscopes (IDG-300) that are used in building our motion sensor platform [1]. The power consumption for each axis of the accelerometer is 0.88 *mW* while for the gyroscope this value is 15.67 *mW*.

In Table 2, the power consumption of the classification for each target action and for different values of accuracy are reported. As shown, the values of the power consuming remain very low (1-3 *mW*) for almost all the actions. The power needed for the classification increases suddenly when we start to use the weak classifiers associated with the gyroscope sensors. Such power consuming classifiers are activated in order to improve the accuracy of the classification and to achieve the desirable classification performance as requested by the user.

The algorithm described in Section 4 will drastically decrease the power consumption of the classification system. The amount of power savings achieve by using our power-aware action recognition framework is reported in Table 2. For this particular test, the performance level is set to P=90% and R=80%. In the table, power consumption and percentage of power savings are reported for each target action for a system without intelligent sensor selection as well as a system with our sensor activation approach.

6 Discussion and Ongoing Research

Our classification approach can be compared with several previous studies in terms of accuracy, number of classified actions and amount of sensors used in the study. [6] reports 85% accuracy for classifying 4 actions using one sensor node placed in the pocket. It also achieves 86%, 87%, 87%, 89%, and 92% accuracy when the sensor node is placed on necklace, belt, wrist, shirt, and bag respectively. [7] reports 84% accuracy for classifying 4 actions with a sensor node placed on the chest.

Currently, our classification approach is based on template matching. Most previously studied classification approaches rely on feature extraction and either parametric or non-parametric classification algorithms. As part of our future work, we will explore feasibility of the approach where the cost associated with each feature (rather than each sensor) is considered.

As an alternative for sensor selection, one may consider a feature selection approach to find significant features and their associated sensor nodes. However, we note that feature selection algorithms may not take advantage of our classifier boosting methodology.

At the current stage of our research, we use a greedy heuristic algorithm to select active sensors for classification. As part of our ongoing research we derive analytical bounds on the performance of the proposed algorithm.

Currently, we perform static analysis of the power-accuracy trade-offs. Thus, our greedy sensor selection algorithm is running offline. In future, we will investigate a combinatorial algorithm that can run in real-time and selects active sensors dynamically.

In this study, we demonstrated the effectiveness of our classification algorithms on a small experimental population (3 subjects). In future, we will investigate generalizability of our results to larger populations.

7 Conclusion

In this paper, we designed a novel classification approach that incorporate two design criteria, energy consumption and classification accuracy. Our system uses simple template matching blocks to perform coarse classification of human movements on wearable sensor nodes. Only a subset of the sensors is activated for classification purposes where active sensors are determined according to their power consumption as well as their contribution to the overall classification performance of the system. The results obtained by active sensors are further combined through a boosting approach to achieve high classification accuracy.

8 References

[1] H. Ghasemzadeh, E. Guenterberg, and R. Jafari, "Energy-Efficient Information-Driven Coverage for Physical Movement Monitoring in Body Sensor Networks," *IEEE Journal on Selected Areas in Communications*, vol. 27, pp. 58–69, 2009.

[2] P. Zappi, C. Lombriser, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Troster, "Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection," *Lecture Notes in Computer Science*, vol. 4913, p. 17, 2008.

[3] N. Ravi, N. Dandekar, P. Mysore, and M. Littman, "Activity recognition from accelerometer data," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 20, no. 3. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 1541.

[4] J. Lester, T. Choudhury, and G. Borriello, "A practical approach to recognizing physical activities," *Pervasive Computing*, pp. 1–16, 2006.

[5] K. Murty and S. Kabadi, "Some np-complete problems in quadratic and nonlinear programming," *Mathematical Programming*, vol. 39, no. 2, pp. 117–129, 1987.

[6] U. Maurer, A. Smailagic, D. Siewiorek, and M. Deisher, "Activity recognition and monitoring using multiple sensors on different body positions," in *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*, 3-5 2006, pp. 4 pp. –116.

[7] R. Jafari, W. Li, R. Bajcsy, S. Glaser, and S. Sastry, "Physical Activity Monitoring for Assisted Living at Home," in *IFMBE Proceedings*, vol. 13. Springer, 2007, p. 213.