Sensor Fusion-Based Activity Recognition for Parkinson Patients

Majid Bahrepour, Nirvana Meratnia, Zahra Taghikhaki and Paul J. M. Havinga
Pervasive Systems Group, University of Twente, Netherlands

1. Introduction

Parkinson disease (PD) is a slow destructive disorder of the central nervous system in which dopamine, i.e., catecholamine neurotransmitter in the central nervous system is lost. PD hurts patients' movement and speech ability. Sometimes, it can also affect patients’ mood, behavior, and thinking ability. Falling down is a common problem in PD patients and on time fall detection is important to assist PD patients and prevent them from being injured. To this end, being able to correctly distinguish various activities, e.g. walking, sitting, standing still, is a must. To monitor activities and moving patterns of PD patients, a wireless body sensor network (BSN) may prove to be useful. By attaching various wireless sensor nodes on the body of PD patients or integrating them into their shoes or cloths, their activities and physiological conditions can be checked regularly and an alarm can be generated in case of emergency or need for additional assistances.

A wireless body sensor network consists of a number of wireless sensor nodes that cooperatively monitor physical (e.g. motion) and physiological (e.g. heart rate) conditions of a person. In addition to sensors, each sensor node is typically equipped with a radio transceiver or other wireless communication devices, a small microcontroller as processing unit, and an energy source in a form of a battery. Sensor nodes may vary in size and type of sensors they are equipped with. Size and cost constraints on sensor nodes cause limitations on their resources in terms of energy, memory, and computational processing. Figure 1 shows an example of a body sensor network.

Previous studies for activity recognition of PD patients mostly use accelerometer and occasionally gyroscope sensors attached to various parts of patients’ body (JJ., HA. et al. 1991; Aminian, Robert et al. 1999; JL., AA. et al. 2001; JL., V. et al. 2004; N., T. et al. 2004; White, Wagenaar et al. 2006; Moorea, MacDougalla et al. 2007; Salarian, Russmann et al. 2007). One of the main criticisms on the previous studies is that they use centralized techniques which not only require expensive equipments to monitor physiological conditions and activities of patients [e.g. Vitaport 3 (White, Wagenaar et al. 2006)] but also introduce delays in the detection process. Also due to having a single point of failure they are more prone to failures and crashes. In contrary, we propose a fusion-based distributed algorithm which can be easily implemented on resource constrained wireless sensor nodes and detect and distinguish activities in (near) real-time. Our approach offers three main advantages: (i) distributed processing and reasoning which decreases the data processing
time and provides fast activity detection and classification, (ii) robustness against sensors failure, and (iii) accurate detection through use of sensor fusion.

![An example of a body sensor network.](image)

Fig. 1. An example of a body sensor network.

Our approach is based on implementing classification techniques on wireless sensor nodes attached to patients’ body, online evaluation of the classification results on individual nodes, and fusing results of various nodes to resolve possible conflicts between sensor nodes and reach a consensus. Previously in (Bahrepour, Meratnia et al. 2009; Bahrepour, Meratnia et al. 2009; Bahrepour, Zhang et al. 2009; Bahrepour, Meratnia et al. 2010; Bahrepour, Meratnia et al. 2010; Bahrepour, van der Zwaag et al. 2010), we have shown capability of machine learning based classification techniques in distributed detection of environmental events such as fire. There is no reason to believe that classification techniques used in other domains are not applicable for medical domain. The challenge here, however, is twofold: (i) investigating capabilities of these classification techniques for a more complex data such as activity data, and (ii) being able to determine the most relevant sensor data among a large number of sensor types for a specific purpose, e.g., fall detection of PD patients. In what follows we address both challenges.

The organization of this paper is as follows. Section 2 provides an overview of the related work. Section 3 introduces the machine learning based classification and sensor fusion techniques that will be used in this study for the purpose of activity recognition. In section 4, our processing models are introduced, which will be followed by our proposed approach explained in Section 5. Section 6 describes the dataset and presents experimental results. Conclusions are drawn in Section 7 and the obtained results are discussed.

2. Related work

Related work on use of wireless body sensor network in medical domain can be generally classified into three groups:
Sensor Fusion-Based Activity Recognition for Parkinson Patients

173

i. those related to hardware platform design [e.g. (Kern, Schiele et al. 2003; Park, Liu et al. 2005; Lorincz, Kuris et al. 2007; Ying, Schlösser et al. 2008; Nabar, Banerjee et al. 2010)];

ii. those related to activity recognition methods and algorithms (Aminian, Robert et al. 1999; Veld, M.H.A. et al. 2005; Moorea, MacDougalla et al. 2007; Osmani, Balasubramaniam et al. 2007; Salarian, Russmann et al. 2007; Osmani, Balasubramaniam et al. 2008; Lee, Kim et al. 2009; Khattak, Vinh et al. 2010);

iii. those related to making the information flow between patients and medical team more efficient [e.g. (Centeno, Giachetti et al. 2003; Martinez-Garcia and Menndez-Olague 2003; Wijewickrama and Takakuwa 2006; Sanchez, Tentori et al. 2008)].

While the first group focuses on design and implementation of small, easy to wear, and cheap sensor node platforms equipped with a set of sensors, the second group focuses on design of computationally light signal processing, feature extraction, classification, and pattern recognition techniques. The focus of the third group is on providing timely information to the doctors in charge, reducing patients waiting time, and improving interaction between doctors and patients, etc.

The main objectives of performing activity recognition in medical domain are:

- Prevention (e.g., (Steele, Belza et al. 2003; Wu, Bui et al. 2008; Le and Pan 2009; Benocci, Tacconi et al. 2010)

- Rehabilitation (e.g., (Jarochowski, Shin et al. 2007; Soini, Nummela et al. 2008; Zhang and Sawchuk 2009)

- Assistance and care giving (e.g., (Hou, Wang et al. 2007; S. Bosch 2009)

To perform activity recognition usually the following steps are taken (Krishnan, Juillard et al. 2009; Avci, Bosch et al. 2010; Horst and Meratnia 2011):

- Sampling, which refers to the process of taking measurements from the body sensors. Success of this process heavily depends on robustness of the sensor node platform, radio communication, as well as availability of accurate timing.

- Preprocessing, which involves refining the sensor data. More specifically this step deals with noise reduction through for example filtering (Yun, Lizarraga et al. 2003) as well as signal transformation through use of for example Fourier transformation (Wu, Pan et al. 2009).

- Segmentation, which is the process of identification of beginning and end of an important feature in the sensor data stream. Segmentation can be performed manually, e.g., (Jafari, Li et al. 2007) or automatically, e.g., (Guenterberg, Bajcsy et al. 2007; Guenterberg, Ostadabbas et al. 2009).

- Feature extraction, classification, and pattern recognition, which refer to the process of giving semantic to the identified feature and identification of repeated patterns and trends. During feature extraction, time-domain, frequency-domain, or time-frequency domain features can be identified. Techniques such as neural network, Bayes network, hidden markov model, support vector machine, and decision trees are often used for the classification purpose.

- Pattern matching, which deals with comparing the identified patterns and features either with those identified on other nodes or with pre-defined templates.

- Feedback, which is the process of taking appropriate action based on output of classification.

For a more elaborate overview of state of the art on use of wireless body sensor networks in medical domain, reader is referred to (Abbate, Avvenuti et al. 2010; Avci, Bosch et al. 2010; Horst and Meratnia 2011).
3. Classification and fusion techniques

For the purpose of activity recognition and classification, we use Feed Forward Neural Network (FFNN), Naïve Bayes (NB), and Decision Tree (DT). For the purpose of sensor fusion, we use reputation-based voting and majority voting. In this section, we first provide a short explanation of the concepts used by these techniques.

3.1 Feed forward neural network (FFNN)
Feed forward neural network (FFNN) is a type of the neural network, in which each layer is fed by its back layer (Mehrotra, Mohan et al. 1996). FFNN consists of one input layer, one or more hidden layers and one output layer. The challenge faced in using FFNN as a classifier is finding correct weights. The process of finding these weights, which is called ‘learning’, can be carried out using algorithms such as gradient descent (GD) algorithm (Wikipedia; ALPAYDIN 2004).

3.2 Naïve bayes classifier (NB)
A Naïve Bayes classifier uses Bayesian statistics and Bayes theorem to find the probability of each instance belonging to a specific class. It is called Naïve because of its emphasis on independency of the assumptions. Naïve Bayes classifier finds the probability of belongingness of each instant to a specific class.

3.3 Decision tree
A decision tree is a learning algorithm that uses tree-like graphs to model and evaluate discrete functions (Russell and Norvig 2003; ALPAYDIN 2004). Construction of a decision tree for classification purpose requires a training phase (ALPAYDIN 2004), which employs a set of data and a learning algorithm to find a minimum depth decision tree. The tree should contain the minimum required nodes (or minimum depth) to reduce time and memory complexities. Therefore, the training algorithm is usually a local search greedy algorithm to find an optimum decision tree (ALPAYDIN 2004). Once the decision tree is created, the tree can be used by evaluating nodes from the root down to the leaves. The final leaf contains a value that shows the result of the classification. After the decision tree is constructed in the learning phase, it can be pruned to save memory. Pruning is only required for large trees to alleviate computational complexity.

3.4 Reputation-based voting
Reputation based voting is a fusion technique that happens after classification. Once each classifying entity makes its individual decision about belongingness of an instance to a class, a consensus needs to be reached among the classifying entities. Reputation-based voting approaches are based on finding reputation of individual classifying entity and choosing the decision made by the classifying entities having the highest reputation. Assuming that classifying entities have correctly classified instances, they should judge how well the other entities have performed the classification. To do the judgment, each entity first sends its classification result, called Detection Value (DV), to all other entities in its neighborhood. The DVs received from the neighbors will be stored in a table called Neighbors Detection Value Table (NDVT). In the next step, each entity should judge about its neighboring entity by considering itself as the reference. The judgment is accomplished by comparing the difference between value of entity itself and value of the other neighboring classifying
entities. If the difference is less than a threshold value $\theta$ (which is chosen based on the context), the “judging” entity gives a positive vote ($V_{\text{new}} = V_{\text{old}} + 1$) to the other entities. Otherwise, the “being judged” entity receives a negative vote ($V_{\text{new}} = V_{\text{old}} - 1$). Finally, NDVT tables are sent to a voter to reach a consensus among different opinions. The challenging part of reputation-based voting is how to assign a global reputation value to each entity in order to choose high reputed entity and its classification result. There are various ways to do so, two of which are explained here:

### 3.4.1 Reputation technique 1

This reputation technique checks local reputation of every individual entity from the other entities’ perspective. The local reputation value is obtained based on average value of $V_i$ (positive or negative votes which were given by the other entities) for each classifying entity. Then, the average local reputation is multiplied by the weight of sensor nodes calculated using Equation 1 to assign global reputation values. The class with the highest reputation weight ($W$) is the result of the voting procedure. Equation 1 shows how the weights are calculated.

$$W_i = R_i \times Acc_i$$

where $W_i$ is the reputation value corresponding to classifying entity $i$, $R_i$ is the local reputation value of classifying entity $i$ from other entities’ perspective, and $Acc_i$ is weight of classifying entity $i$ (Bahrepour, Meratnia et al. 2010).

### 3.4.2 Reputation technique 2

In this reputation technique, two threshold values, i.e., $\theta_1, \theta_2$ are used. Comparing the local reputation value ($R_i$) with $\theta_1$ and $\theta_2$ gives an insight about how well classification is performed. If ($R_i \geq \theta_1$), then classifying entities have made perfect decisions, if ($\theta_1 > R_i \geq \theta_2$) then classifying entities have made OK decisions, and if ($\theta_2 \geq R_i$) then classifying entities have made poor decisions. We assign 0.5 to poor classification performance, 1 to normal classification performance, and 2 to perfect classification performance. Based on these values, this reputation technique uses Equation 2 to assign reputation to each classifying entity.

$$W_i = S_i \times Acc_i$$

where $W_i$ is the reputation value corresponding to classifying entity $i$, $S_i$ is obtained from Eq. 3, and $Acc_i$ is weight of classifying entities (Bahrepour, Meratnia et al. 2010).

$$S_i = \begin{cases} 
2 & \text{if } (R_i \geq \theta_1) \\
1 & \text{if } (\theta_1 > R_i \geq \theta_2) \\
0.5 & \text{if } (\theta_2 \geq R_i)
\end{cases}$$

$\theta_1$ and $\theta_2$ are application dependant.

### 3.5 Majority voting

Majority voting is a simple voting technique, which selects the classification result obtained from majority of sensor nodes.
4. Classification models

In this section, we present our general classification models regardless of which classifier is being used for the classification purpose. For more information about properties of these models and their performance, reader is referred to (Bahrepour, Meratnia et al. 2009; Bahrepour, Meratnia et al. 2009).

4.1 Local classification model

Local model assumes that each sensor node performs classification individually without communicating and cooperating with others. Figure 2 illustrates processing model of local classification, which consists of (i) a number of sensors providing input to the classifier, (ii) the classifier, which is responsible for activity recognition and determining the belonging of each instance to an activity class, and (iii) classification output, which is called activity. One should note that not all sensor nodes need to have the same classifiers.

![Fig. 2. Local classification model](image)

4.2 Fusion-based classification model

The local approach is simple and works fine in situations, in which the sensor nodes are highly accurate and not prone to noises. However, generally speaking sensors, sensor nodes, and communication links are not always reliable and their failure is a common practice. Fusion-based classification model tolerates individual sensor and sensor node failures and involves more than one sensor node in the classification process. By doing so, it ensures that there are always some sensor nodes contributing to the classification process and compensating for the errors. The fusion-based approach uses the basic notions of the local approach and lets individual sensor nodes first classify and detect activities on their own. Then, the classification results are all sent to a fuser/voter node (e.g., a cluster head) to reach a consensus. Figure 3 illustrates processing model of fusion-based classification. Similar to the local model, not all sensor nodes (including the fuser node) need to have the same classifiers.

In this study we use fusion-based classification model for activity recognition and investigate its performance by making use of various classifiers. Choice of classifier not only has direct effect on classification accuracy but also on complexity, as presented in the following sections.
4.3 Computational complexity consideration

4.3.1 Complexity of local model using FFNN and naïve bayes

Complexity of FFNN is a function of \( m \) (number of features or number of nodes in the input layer), \( z \) (number of neurons in the hidden layer), and \( p \) (number of neurons in output layer). Equation 4 presents time complexity of FFNN in the local model (Bahrepour, Meratnia et al. 2009):

\[
O_{\text{FFNN}} = O(m \times z \times p)
\]  

(4)

Complexity of Naïve Bayes is a function of \( m \) (number of features), \( c \) (number of classes), and \( j \) (number of partitions for distribution estimation). Equation 5 presents time complexity of Naïve Bayes classifier (Bahrepour, Meratnia et al. 2009) in the local model:

\[
O_{\text{NaïveBayes}} = O(m \times c \times j)
\]

(5)

Since training phase of FFNN and Naïve Bayes is conducted offline and only once, the computational complexity of training part can be disregarded.

4.3.2 Complexity of fusion-based model using FFNN and naïve bayes

When naïve Bayes or FFNN are used in the fusion-based model, the data-fuser has to wait till all classification results from the sensor nodes are available. In the worst case for \( n \) time duration (where, \( n \) is the number of sensor nodes involved in the classification process):
FFNN or naïve Bayes classifiers on sensor nodes run in parallel. In this case, computational complexities of classifiers running on the fuser and on the sensor nodes are added together. The computational complexities of the fusion-based model using FFNN and Naïve Bayes are presented by Equation 6 and Equation 7, respectively (Bahrepour, Zhang et al. 2009).

\[
O_{\text{FFNN}} = O[(m \times z \times p) + (m \times z \times p)] = O(m \times z \times p) \tag{6}
\]

\[
O_{\text{Naïve Bayes}} = O[(m \times c \times j) + (m \times c \times j)] = O(m \times c \times j) \tag{7}
\]

### 4.3.3 Complexity of local model using Decision Tree

Complexity of decision tree appraisal is a function of the depth of decision tree. Equation 9 presents this time complexity:

\[
O(\text{local approach}) = O(\text{Decision tree appraisal}) \tag{8}
\]

\[
O(\text{Local approach}) = O(m)O(\text{Local approach}) = O(m) \tag{9}
\]

where \(m\) is depth of the decision tree

### 4.3.4 Complexity of fusion-based model using decision tree and reputation theory

Time complexity of the fusion-based model using decision tree as classifier and reputation theory as fuser is a function of three parameters: (i) complexity of making the decision tree on each node, (ii) complexity of performing local classification on each node, and (iii) complexity of performing fusion and reaching consensus between classification results. Equation 12 presents the final complexity.

\[
O(\text{Fusion based decision tree with reputation}) = \max[O(\text{Decision tree appraisal}) + O(\text{process on the node}) + O(\text{reputation voting})] \tag{10}
\]

\[
O(\text{Fusion based decision tree with reputation}) = \max[O(m) + O(n(n-1)) + O((n(n-1)) + n + c)] \tag{11}
\]

\[
O(\text{Fusion based decision tree with reputation}) = O(n^2) \tag{12}
\]

where \(n\) is the number of nodes

### 4.3.5 Complexity of fusion-based model using decision tree and majority voting

Complexity of classification process of fusion-based model using decision trees is \(O(m_1 + m_2 + \cdots + m_n) = O(m)\); where \(n\) is the number of nodes involved in the classification and \(m\) is depth of the decision tree. Since the voting is independent from the classification, its time complexity is added to the classification time as shown in Equation 13. Equation 16 presents the final complexity of fusion-based model using decision trees and majority voting.

\[
OO(\text{Distributed approach using majority voting}) =
O([m]) + O[\text{Majority voting} 1] = \max[[m], \text{Majority voting}] \tag{13}
\]
Sensor Fusion-Based Activity Recognition for Parkinson Patients

\[ O(Distributed\, approach\, using\, majority\, voting) = O(M\, \text{Max finding}) + O(m) \]

\[ O(\text{max finding}) = O(c) \]

\[ O(O(Distributed\, approach\, using\, majority\, voting) = O(c) + O(m)) = O(m) \]

where \( m \) is depth of the decision tree, and \( n \) is number of nodes in the network.

4.4 Computational comparison

The time complexities of local and fusion-based models using different classifiers and fusers are summarized in Table 1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Time complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion based model using FFNN</td>
<td>( O = O(m \times z \times p) )</td>
</tr>
<tr>
<td>Fusion based model using Naïve Bayes</td>
<td>( O = O(m \times c \times j) )</td>
</tr>
<tr>
<td>Fusion based model using Decision Tree and Reputation-based voting</td>
<td>( O(n^2) )</td>
</tr>
<tr>
<td>Fusion based model using Decision Tree with majority voting</td>
<td>( O(m) )</td>
</tr>
</tbody>
</table>

Table 1. Complexity comparison of local and fusion-based classification models with different classifiers

where \( n \) is the number of sensor nodes in the network, \( c \) is number of classes, \( m \) is number of features, and \( j \) is number of partitions for distribution estimation.

5. Data description and empirical results

5.1 Activity dataset and feature reduction process

To investigate applicability of the aforementioned classifiers for the activity recognition task in the medical domain, we used an activity dataset provided by Enschede Hospital (Medisch Spectrum Twente). The dataset consists of data from sensors attached to a 25 year old person, while he had been walking, sitting and standing still. The dataset contains 30 features from 5 tri-axial accelerometers and 5 tri-axial gyroscope sensors located on feet, shank, thigh and trunk. To find the most important features of the dataset, we first developed a feature reduction technique using genetic algorithm and decision tree. Feature reduction is the technique of selecting a subset of relevant features for making robust learning models. By removing redundant features from the dataset, feature selection helps improve the performance of learning models by (i) alleviating the effect of the curse of
dimensionality, (ii) enhancing generalization capability, and (iii) speeding up learning process (Isabelle Guyon 2006). Genetic algorithms (GAs) belong to the larger classes of evolutionary algorithms (EAs) that generate solutions to optimization problems using techniques inspired by natural evolution, e.g., inheritance, mutation, selection, and crossover (Goldberg 1989). GA optimizes the features in order to find those most contributing to the classification process. Using GA for feature reduction, we first selected four most contributing features from the dataset. A decision tree was then created using these four features and its classification accuracy was considered as the fitness value for the four features. After the feature reduction process, our activity dataset contained four features, 8330 instances of data and three classes namely walking, sitting, and standing still. Figure 4 shows data distribution for these three classes and Table 2 presents statistical information of dataset, while the four features of the dataset are: ‘Z’ vector of gyroscope installed on the right foot, ‘Y’ vector of accelerometer installed on trunk, ‘Z’ vector of accelerometer installed on trunk, and ‘X’ vector of accelerometer installed on left foot. As it can be seen from Figure 4, data representing three classes have a high degree of overlap, which makes the dataset very complex for the classification process.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Min</th>
<th>Max</th>
<th>Mean</th>
<th>STD</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9.233</td>
<td>10.775</td>
<td>-0.040</td>
<td>1.4355</td>
<td>right foot gyroscope Z vector</td>
</tr>
<tr>
<td>2</td>
<td>78.7644</td>
<td>50.0057</td>
<td>-3.9921</td>
<td>6.942</td>
<td>trunk accelerometer Y vector</td>
</tr>
<tr>
<td>3</td>
<td>-24.5415</td>
<td>20.3732</td>
<td>-2.8579</td>
<td>2.2486</td>
<td>trunk accelerometer Z vector</td>
</tr>
<tr>
<td>4</td>
<td>29.8074</td>
<td>1.8017</td>
<td>-4.4615</td>
<td>2.753</td>
<td>left foot gyroscope X vector</td>
</tr>
</tbody>
</table>

Table 2. Activity data set information (8360 instances, 3 classes: stand still, walking, sitting)

To simulate the classification process, we use Matlab® to train and test every individual node separately. In the training phase, first the classifiers (local model) running on the sensor nodes are trained. Then, according to the trained classifiers on sensor nodes, the fuser (fusion-based model) is trained. In the test phase, an instance of data is given to all sensor nodes and then their outputs are fused using either another classifier or a voting technique. Training phase is conducted with 2/3 of data and testing is performed with 1/3 of data.

5.2 Empirical results
We perform each activity classification experiment ten times and accuracy ratios are reported in Table 3.

According to Table 3, fusion based models using decision trees and reputation-based voting (technique 1 and 2) and using majority voting provides more accurate activity recognition. Additionally, in this specific dataset (activity dataset) reputation based voting technique 1 and 2 provides the same accuracy ratio.
Fig. 4. Data distribution of three classes per each feature: (a) ‘Z’ vector of gyroscope installed on the right foot, (b) ‘Y’ vector of accelerometer installed on trunk, (c) ‘Z’ vector of accelerometer installed on trunk, (d) ‘X’ vector of accelerometer installed on left foot.
Table 3. Accuracy and standard deviation of activity classification and recognition

<table>
<thead>
<tr>
<th>Model</th>
<th>Classification Accuracy (Mean)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion based model using FFNN</td>
<td>73.6022</td>
<td>1.4858</td>
</tr>
<tr>
<td>Fusion based model using Naïve Bayes</td>
<td>73.4624</td>
<td>1.2771</td>
</tr>
<tr>
<td>Fusion based model using Decision Tree and Reputation-based voting (technique 1)</td>
<td>78.9881</td>
<td>0.7820</td>
</tr>
<tr>
<td>Fusion based model using Decision Tree and Reputation-based voting (technique 2)</td>
<td>78.9881</td>
<td>0.7820</td>
</tr>
<tr>
<td>Fusion based model using Decision Tree with majority voting</td>
<td>78.6712</td>
<td>0.9836</td>
</tr>
</tbody>
</table>

5.3 Parameter study
There are some parameters in our classification process, which can affect classification accuracy. These parameters include (i) type of the classifiers, (ii) internal parameters of the classifiers (e.g. number neurons in hidden layer for FFNN), (iii) number of sensors, and (iv) type of sensors. In (Bahrepour, Meratnia et al. 2010), we have shown that the most effective parameter in fusion-based classification is the sensor types. This means that it is important to identify the most contributing sensors and ensure their presence in the classification process. As mentioned in Subsection 5.1, using a genetic algorithm and decision tree in the feature reduction process, we have identified the most contributing features of the dataset. In this section we study internals parameters of classifiers and their effects on classification accuracy.

5.3.1 Effects of internal parameters of feed forward neural network (FFNN) on classification
Internal parameters for FFNN are number of hidden layers and number of neurons in each hidden layer. The hidden layer in FFNN can grow vertically, horizontally and bi-directional (in both horizontal and vertical). The vertical growth means keeping number of hidden layers as low as possible and increasing the number of neurons in hidden layers. Number of neurons can also be kept as low as possible while number of hidden layers is increased. This process is called horizontal growth. A combination of both vertical and horizontal growth is also possible (bi-directional growth). To see the effects of growth in different directions, we performed several experiences, whose results are reported in Table 4.
Generally speaking, vertical growth of neural network leads to more accurate results comparing to horizontal growth. Moreover, by increasing number of hidden layers and/or neurons in the hidden layers, the accuracy enhances; however, at a certain point increasing hidden layer elements decreases the accuracy. Therefore, number of hidden layers as well
as their neurons in each layer should be increased to the point which generates the highest accuracy. This is obtained experimentally.

![Accuracy and standard deviation of activity classification and recognition.](image)

Table 4. Effect of neural network’s internal parameters on classification accuracy

<table>
<thead>
<tr>
<th>Hidden Layers</th>
<th>Neurons</th>
<th>Accuracy</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal growth</td>
<td>1</td>
<td>45.52</td>
<td>0.92</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>46.33</td>
<td>0.61</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>46.59</td>
<td>1.01</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>64.27</td>
<td>13.07</td>
</tr>
<tr>
<td>Vertical growth</td>
<td>1</td>
<td>55.77</td>
<td>1.37</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>58.53</td>
<td>1.22</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>44.57</td>
<td>18.49</td>
</tr>
<tr>
<td>Bi-directional growth</td>
<td>10</td>
<td>51.44</td>
<td>0.92</td>
</tr>
</tbody>
</table>

5.3.2 Effects of internal parameters of naïve bayes (NB) classifier on classification

The only parameter for Naïve Bayes classifier is number of partitions for making a histogram. Table 5 shows how this parameter affects classification accuracy.
As it can be seen in Table 5, generally speaking increasing number of partitions leads to increase of the classification accuracy. However, at a certain point either the accuracy stays roughly the same or drops.

### 5.3.3 Effects of internal parameters of decision tree (DT) on classification

Decision trees have almost no internal parameter because the whole tree is made during the training phase. However, after the training phase, the obtained tree can be pruned and some less-contributing branches can be removed. We performed a number of experiments, in which we prune the decision tree created for the activity dataset to a degree that tree has only one node. Figure 6 illustrates the effect of this pruning on classification accuracy. As it can be seen, pruning causes degradation of classification accuracy, in general. One also notices that for our activity dataset, pruning first increases the accuracy to some extent and then decreases it. The reason for this behavior is that the tree is over branched (i.e., almost 240 level tree is sophisticated enough to create some ambiguity and inaccuracy in the classification). Reducing some branches helps tree be simpler and perform classification slightly more accurate. However, at a certain point the accuracy drops. A general conclusion is, pruning causes reduction in accuracy ratio because the decision tree has to perform classification with fewer nodes.

![Fig. 6. Pruning and its effect on classification accuracy](www.intechopen.com)
6. Conclusion

Fall detection is a common problem in PD patients. Accurate activity classification and recognition is one of the first steps towards on time fall detection. Body sensor networks are an emerging technology enabling online and distributed monitoring of both physical and physiological conditions of patients. Integrating this monitoring capability with (near) real-time activity recognition and classification can speed up the fall detection process and enable provision of timely feedback and assistance. To this end, having a thorough understanding of capabilities and performance of classification techniques is essential.

In this paper we investigated applicability of three classification techniques, i.e., Naïve Bayes, Feed Forward Neural Network, and Decision Tree for activity recognition in medical domain and showed how fusion-based classification can improve classification accuracy. In addition to considering classification accuracy, we studied effects of internal parameters of the classifiers on the classification performance and compared various techniques in terms of their complexity.

Activity recognition in medical domains requires inexpensive and easy wearing hardware components and computationally-light algorithms. To this end, our fusion-based classification technique offers three main advantages: (i) distributed processing and reasoning which decreases the data processing time and provides fast activity detection, (ii) robustness against sensors failure, and (iii) accurate detection through use of sensor fusion.

We performed a number of experiments using an activity dataset collected from a 25 year old person walking, sitting, and standing still. Performing an offline analysis shows high complexity of the dataset, as data representing three classes have a high degree of overlap. The evaluation results show that, considering this high complexity of the dataset, our fusion-based technique can reach reasonable classification accuracy.

Our future plans include design of temporal learning techniques to increase detection accuracy over time.

7. Acknowledgment

The authors would like to acknowledge Dr. J. P. P. van Vugt from Medisch Spectrum Twente and Sarvi Sharifi from the University of Twente for sharing their activity dataset. This work is supported by the EU Seventh Framework Programme and the SENSEI project.

8. References


www.intechopen.com


www.intechopen.com
Sensor Fusion - Foundation and Applications comprehensively covers the foundation and applications of sensor fusion. This book provides some novel ideas, theories, and solutions related to the research areas in the field of sensor fusion. The book explores some of the latest practices and research works in the area of sensor fusion. The book contains chapters with different methods of sensor fusion for different engineering as well as non-engineering applications. Advanced applications of sensor fusion in the areas of mobile robots, automatic vehicles, airborne threats, agriculture, medical field and intrusion detection are covered in this book. Sufficient evidences and analyses have been provided in the chapter to show the effectiveness of sensor fusion in various applications. This book would serve as an invaluable reference for professionals involved in various applications of sensor fusion.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:
