Chapter 4

A Robust Environmental Sound Classification

In the previous chapter, an effective technique is presented for environmental sound source separation. In this chapter, a method is proposed for Environmental Sound Classification which recognizes the sound source separated audio streams into one of the predefined categories. Section 4.2 discusses the extraction of MFCC and frequency-domain features (Spectral features). Section 4.3.1, Section 4.3.2 and Section 4.3.3 describe the principle of various pattern classification models namely Backpropagation Neural Network (BPNN), Radial Basis Function Neural Networks (RBFNN) and Neuro Fuzzy (NF) classifier for classifying the different categories of the separated audio respectively. Experimental results are discussed in Section 4.5.

4.1 Introduction

Sound event classification is a growing popularity recently in the field of acoustic signal analysis not only because it bears great interest for application in multimedia search based on sound, but it is also one of the most important key components to analyze environments, e.g., in surveillance, monitoring of people in need of care, or detecting, localizing, tracking and classifying sources of military interest in real time [45], [55]. Obviously, there is also great benefit for humanoid and general robots, such as the one introduced in for kitchen tasks, to better understand their acoustic environment.
Finally, there is hope to better recognize and enhance speech and music, once the sound type of disturbance can be identified [121].

People consider the task of classified environment sounds for the understanding of a scene (or context) surrounding an audio sensor. By auditory scenes, we refer to a location with different acoustic characteristics such as a coffee shop, park or quiet hallway. Consider, for example, applications in robotic navigation and obstacle detection, assistive robots, surveillance, and other mobile device based services. Many of these systems are dominantly vision-based [122], [67]. When being employed to understand unstructured environments, their robustness or utility will be lost if visual information is compromised or totally absent. Audio data could be easily acquired, in spite of challenging external conditions such as poor lighting or visual obstruction, and is relatively cheap to store and compute than visual signals. To enhance the system’s context awareness, we need to incorporate and adequately utilize such audio information [123], [124].

Most application of acoustic event recognition is based on a classification task. That is, given a short clip of audio, a recognition system must determine which acoustic event in its training database is the closest match with this new sound. This typically requires a longer audio clip than for individual acoustic events. The final application is indexing and retrieval, where an acoustic event can be queried by its audio content. Fig. 4.1 shows the block diagram of Environmental Sound Recognition system.

4.2 Acoustic Features for Sound Classification

The first step in building a recognition system for auditory environment is to investigate on techniques for developing an event classification system using audio features. The study is performed by first collecting real world audio from the web which provides a
large amount and variety of sound events from real life and then building a classifier to discriminate different environments, which allows us to explore and investigate on suitable features and the feasibility of designing an automatic environment recognition system using audio information. MFCC and Spectral features are used in this work for sound classification.

The purpose of feature extraction is to extract useful discriminative information from the waveform which will result in a compact set of feature vectors [125], [51]. Many different types of features for classification of auditory events are used for experimentation. Environmental sounds in general are unstructured data comprising of contributions from a variety of sources, and unlike music or speech, no assumptions can be made about predictable repetitions nor harmonic structure in the signal. Due to the inherent diverse nature, there are many features that can be used, or are needed, to describe audio signals. The appropriate choice of these features is crucial in building a robust recognition system. A considerable number of audio features are used in this work from 13 dimensional MFCC and 5 dimensional spectral are extracted to construct an environmental audio classification system.
4.2.1 Mel Frequency Cepstral Coefficients

Mel frequency cepstral coefficients is discussed in Section 3.2.2.

4.2.2 Spectral Skewness

Spectral skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable that in this context is the spectrum of the signal. For a sample of \( N \) values forming a frame, the skewness is:

\[
\text{Skewness} = \frac{m_3}{m_2^{3/2}} = \frac{\frac{1}{N} \sum_{n=0}^{N-1} (x(n) - \bar{x})^3}{\left(\frac{1}{N} \sum_{n=0}^{N-1} (x(n) - \bar{x})^2\right)^{3/2}} \tag{4.1}
\]

In this equation, \( \bar{x} \) represents the mean of the magnitudes, \( m_3 \) is the sample third central moment and \( m_2 \) is the sample variance.

4.2.3 Spectral Decrease

Section 3.2.6 describes the process of extracting Spectral Decrease features from an audio signal and is summarized as follows: Spectral decrease also represents the amount of decrease of the spectral amplitude.

4.2.4 Spectral Slope

Section 3.2.7 describes the process of extracting Spectral Slope features from an audio signal and is summarized as follows: The spectral slope represents the amount of spectral energy decrease as a function of frequency.

4.2.5 Spectral Crest

Spectral crest factor indicate how flat or “peaky” the power spectral density is in a given subband. The Spectral Crest Factor or peak-to-average ratio is a measurement of a waveform, calculated from the peak amplitude of the waveform divided by the
mean value of the waveform.

\[
CrestFactor = \frac{|x_n(i)|_{\text{peak}}}{\sqrt{\frac{\sum_{i=1}^{N} x^2(i)}{N}}}
\]  \hspace{1cm} (4.2)

In this equation, where \( N \) is the frame length, and \( X_n[i] \) represents spectral amplitude of the \( i^{th} \) sample in the \( n^{th} \) frame.

### 4.2.6 Spectral Flatness

Spectral Flatness is a measure of distribution of spectral power in an audio spectrum. The spectral flatness is calculated by dividing the geometric mean of the power spectrum by the arithmetic mean of the power spectrum. The spectral flatness used is measured across the whole band. The formula is:

\[
\text{Flatness} = \sqrt{\frac{\prod_{n=0}^{N-1} x(n)}{\sum_{n=0}^{N-1} x(n)}}
\]  \hspace{1cm} (4.3)

In this equation, \( x(n) \) represents the magnitude of bin number \( n \) of the power spectrum. High spectral flatness indicates that the spectrum has a similar amount of power in all spectral bands and this would sounds similar to white noise. Low spectral flatness, instead indicates that the spectral power is concentrated in a relatively small number of bands and it is typical for tonal sounds.

The Spectral Decrease (SDF), Spectral Slope (SSF) are extracted as described in Sections 3.2.

### 4.3 Modeling the Acoustic Features for Sound Classification

Neural networks which have been widely used in image and signal processing are very effective for solving multiple class classification problems. Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the connections between elements largely determine
the network function. Neural networks have been trained to perform complex functions in various fields, including pattern recognition, identification, classification, speech, vision, and control systems. Neural networks can also be trained to solve problems that are difficult for conventional computers or human beings [126].

The recognition performance of the neural network will highly depend on the structure of the network and training algorithm. It consists of three layers forward structure that has hidden layer between input layer and output layer interconnected by links that contains weights.

4.3.1 Back Propagation Neural Network

The Back propagation neural network (BPNN) is a feed-forward neural network with back propagation learning algorithm is used. Two different components make up the construct of phases: one is the feed forward phase in which the external input information at the input nodes is propagated forward to compute the output information signal at the output unit, and a backward phase in which modification to the connection strengths is made based on the differences between the computed and observed information signals at the outputs units [127], [128]. Fig. 4.2 shows the Back propagation neural network frame.

4.3.1.1 Backpropagation Training Algorithm

1. Select the next training pair from the training set and apply to the network.

2. Calculate the output of the network.

3. Calculate the error between the output of the network and the desired output.

4. Adjust the weights (V,W matrix) in such a way that it minimize the error.

5. Repeat steps 1 to 4 for all the training pairs.
6. Repeat steps 1 to 5 until the network recognizes the training set or for certain number of iterations called epochs.

The activation function used by BPNN training algorithm is sigmoid or squashing or logistic function, and it is defined as

$$\text{OUT} = 1/(1 + e^{-\text{NET}}) \quad (4.4)$$

BPNN training algorithm uses the derivative of activation function and defined as

$$\frac{\partial \text{OUT}}{\partial \text{NET}} = \text{OUT}(1 - \text{OUT}) \quad (4.5)$$

BPNN training algorithm consists of two passes: (i) Forward pass and (ii) Reverse pass. 

(i) Forward pass:
In this pass, output of the network is calculated as,

\[ \text{NET}_{1j} = x_1 w_{11} + x_2 w_{21} + \ldots + x_n w_{n1} \] (4.6)
\[ \text{OUT}_{1j} = 1/(1 + e^{-\text{NET}_{1j}}) \] (4.7)
\[ \text{NET}_{1k} = \text{OUT}_{11} v_{11} + \text{OUT}_{21} v_{21} + \ldots + \text{OUT}_{n1} v_{n1} \] (4.8)
\[ \text{OUT}_{1k} = 1/(1 + e^{-\text{NET}_{1k}}) \] (4.9)

This is repeated for all the neurons.

(ii) Reverse pass:

This pass consist of two parts. They are

(a) Adjusting the weights of the output layer:

To adjust the weights of the output layer generalized delta rule is used.

\[ \delta_{qk} = \text{OUT}_{qk} (1 - \text{OUT}_{qk})(\text{Target} - \text{OUT}_{qk}) \] (4.10)

where \( \delta_{qk} \) is error for neuron \( q \) in the output layer \( k \),
\( \text{OUT}_{qk} \) is output of neuron \( q \) in the output layer \( k \) and
Target is required output.

The new weight of the \( V \) matrix is calculated as

\[ V_{pq}(n+1) = V_{pq}(n) + \eta \delta_{qk} \text{OUT}_{pj} \] (4.11)

where \( V_{pq}(n+1) \) is new weight,
\( V_{pq}(n) \) is old weight and
\( \eta \) is learning or training rate coefficient.

(b) Adjusting the weights of the hidden layer:

\[ \delta_{pj} = \text{OUT}_{pj} (1 - \text{OUT}_{pj}) \sum_{q=1}^{n} \delta_{qk} V_{pq} \] (4.12)
Where \( \delta_{pj} \) is error for neuron \( p \) in the hidden layer \( j \),

\( O U T_{pj} \) is output of neuron \( p \) in the hidden layer \( j \),

\( \delta_{qk} \) is error neuron \( q \) in the output layer \( k \) and

\( V_{pq} \) is weight from neuron \( p \) in the hidden layer to neuron \( q \) in the output layer.

The new weight of the \( W \) matrix is calculated as

\[
W_{mp}(n + 1) = W_{mp}(n) + \eta \delta_{pj} x_m
\]  

(4.13)

Where \( W_{mp}(n+1) \) is new weight,

\( W_{mp}(n) \) is old weight and

\( \eta \) is learning or training rate coefficient.

The integrated signal is transformed to activation via a transfer function such as the sigmoidal function. Sigmoidal function is a continuous activation function, designed to respond relative to the amount of excitation received [129]. It is the most widely used function in various BPNN applications.

### 4.3.2 Radial Basis Function Neural Network

The Radial basis function neural networks (RBFNN) forms a special architecture with several distinctive features. A typical RBF neural network classifier has three layers, namely input, hidden, and output layer. The input layer of the network is made of source nodes that connect the coordinates of the input vector to the nodes in the second layer. The second layer, the only hidden layer in the network, includes processing units called the hidden basis function units which are located on the centers of well chosen clusters. Each hidden layer node adopts a radial activated function, and output nodes implement a weighted sum of hidden unit outputs [130], [53].

The output layer is linear, and it produces the predicted class labels based on the hidden units. The structure of multi-input and multi-output (MIMO) RBF neural
network is represented by Fig. 4.3. The parameters of an RBF type neural network consist of the centers \( m \) and the spread \( \sigma_m \) of the basis functions at the hidden layer nodes and the synaptic weights \( W_{ij} \) of the output layer nodes. The RBF centers are also points in the input space. It would be ideal to have them at each distinct point on the input space, but for any realistic problem, only a few input points from all available points are selected using clustering.

For an input vector \( X_i \), the \( j^{th} \) hidden node produces an activation function \( h_j \) given by

\[
h_j = \exp \left\{ -\frac{||X_i - \mu_j||^2}{2\sigma_j^2} \right\}
\]  

(4.14)

In this equation (7) where \(-||X_i - \mu_j||^2\) is the distance between the point representing the input \( X_i \) and the center of the \( j \)th hidden node as measured by some norm. In this study, the Euclidean norm is used. The output of the network at the \( k^{th} \) output node is given by

\[
y_{ik} = \sum_{j=1}^{L} h_j w_{kj}
\]  

(4.15)
In this equation (8) where $h_j$ is Gaussian function and $w_{kj}$ is the weight between the hidden and output layer.

The performance of the RBF network depends highly on the number and initial locations of the hidden units. Generally, the positions of the hidden units are initialized using unsupervised clustering algorithms such as $k$-means or Expectation Maximization or supervised clustering algorithms such as the ones introduced in [131], [132]. In this study, we initialized the hidden unit centers using the $k$-means clustering. The $\mu_i$ and $\sigma_i$ are calculated by using suitable clustering algorithm. Here the $k$-means clustering algorithm is employed to determine the centers. The algorithm is composed of the following steps:

1. Randomly initialize the samples to $k$ means (clusters) $\mu_1 \ldots \mu_k$.
2. Classify $n$ samples according to nearest $\mu_k$.
3. Re-compute $\mu_k$.
4. Repeat steps 2 and 3 until no change in $\mu_k$.

The number of activation functions in the network and their spread influence the smoothness of the mapping.

4.3.3 Neuro fuzzy classifier

The Neuro-Fuzzy (NF) classification system extracts feature-wise information of input pattern to different classes. Since all features are not equally important in discriminating all classes, the feature-wise belonging is expected to help in the classification process. The block diagram of the NF model is shown in Fig. 4.4. The NF model works in three steps. In the first step, the system takes an input and fuzzifies its feature values using membership functions (MF), and provides the membership of individual features to different classes. A membership matrix thus formed contains number of rows and
columns equal to the number of features and classes, respectively, present in a data set. In the second step, the membership matrix is converted into a vector by cascading all rows or columns. This vector becomes the input to the NN and thus the number of input nodes of the NN is equal to the product of the number of features and classes. The number of output nodes in the NN is the same as the number of classes present in the data set. The last step of the proposed NF classifier is a hard classification by performing a MAX operation to defuzzify the output of the NN.

![Diagram of Neuro-fuzzy classifier process](image)

**Fig. 4.4:** The detailed process of Neuro fuzzy classifier.

### 4.3.3.1 Fuzzification

The input values are extracted features from the multiple event sound. Now Mel Frequency Cepstral Coefficients (MFCC) and spectral features is employed to extract the features. These input features values are fuzzified by means of membership functions that make it easy to decide the membership of each feature to dissimilar classes. The unseen and inter-related information are removed from the features to the classes through the MF, which leads to get more accuracy of the classification phase by means of Neuro-fuzzy classifier. The membership matrix includes with sum of the window
size and three columns, in which the number of rows is the same to the number of features and the number of columns is the same to the number of classes.

The membership matrix \(mf_{a,c}(S_m)\) explains the degree of belonging of different features \((a)\) to different classes \((c)\).

Where, \(S_m\) is \(m^{th}\) feature value of pattern \(S\). \(m - 1, 2, 3, 4, \ldots, m\) here number of features is the sum of the window size. \(c - 1, 2, 3, 4, \ldots, c\) here number of classes. The depiction of pattern is as follows,

\[
S = [\text{sum of the window size}]^T
\]  

(4.16)

The formula employed to calculate the membership values is represented as below,

\[
mf(s) = \begin{cases} 
0, & \text{if } s \leq a; \\
\frac{s-a}{b-a}, & \text{if } a \leq s \leq b; \\
\frac{c-s}{c-b}, & \text{if } b \leq s \leq c; \\
0, & \text{if } s \geq c.
\end{cases}
\]  

(4.17)

Fig. 4.5 illustrates a triangular membership function for a single fuzzy set. Now, we can see that, \(a\) and \(c\) value is zero and it attains progressively to a maximum of value one at the centre point \(b\) between \(a\) and \(c\). Fig. 4.6 illustrates the plot considering all the three membership functions containing overlapping values. Now, the curves for, low, medium and high are revealed for a specific one attribute.

The membership function after the fuzzification process is uttered for a pattern \(S\) as follows,
Fig. 4.5: Triangular membership function.

Fig. 4.6: Triangular membership function with defined parameters and their values.

$$mf(S) = \begin{pmatrix}
m_{f_{1,1}}(x_1) & m_{f_{1,2}}(x_1) & m_{f_{1,3}}(s_1) \\
m_{f_{2,1}}(x_2) & m_{f_{2,2}}(x_2) & m_{f_{2,3}}(s_2) \\
m_{f_{3,1}}(x_3) & m_{f_{3,2}}(x_3) & m_{f_{3,3}}(s_3) \\
\vdots & \vdots & \vdots \\
m_{f_{s,w}}(x_n) & m_{f_{s,w}}(x_n) & m_{f_{s,w}}(s_n)
\end{pmatrix} \tag{4.18}$$

All rows and columns in the membership matrix are tumbled and changed into a vector by this cascading. This produced vector is specified as the input to the Neural Network.
4.3.3.2 Neural Network

In this, Feed Forward Multi-layer Perceptron classifier is employed which has three layers such as input layer, hidden layer and output layer. The entire number of input nodes of the NN is the same to the product of the number of features and classes. In this document the total number of output nodes from the NN is similar as that of the number of classes, and now the output nodes are produced from the NN. The total number of unseen nodes is equal to the square root of the product, of the number of input nodes and output nodes. In Fig. 4.7 the configuration of neural network is shown.

![Neural Network Diagram](image)

**Fig. 4.7:** The detailed block diagram of Neuro fuzzy classifier.

4.3.3.3 Defuzzification

Next the defuzzification process is performed on the output nodes of NN, by executing a MAX (maximum) operation. The output is a single value, from this value; we can competent to categorize the multiple event sound.
4.4 Performance Measures

Performance measurement is the process of collecting, analyzing and/or reporting information regarding the performance of a system or component. Sensitivity and specificity are statistical measures of the performance of a binary classification test, also known in statistics as classification function.

**Accuracy:** The accuracy of a measurement system is a level of measurement that yields true (no systemic errors) and consistent (no random errors) results.

\[
Accuracy = \frac{TruePositives + TrueNegatives}{TotalNo.ofSamples}
\]  

(4.19)

**Precision:** In a classification task, the precision for a class is the number of true positives (i.e. the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class (i.e. the sum of true positives and false positives, which are items incorrectly labeled as belonging to the class).

\[
Precision = \frac{TruePositives}{TruePositives + FalsePositives}
\]  

(4.20)

**Recall:** Recall is defined as the number of true positives divided by the total number of elements that actually belong to the positive class (i.e. the sum of true positives and false negatives, which are items which were not labeled as belonging to the positive class but should have been).

\[
Recall = \frac{TruePositives}{TruePositives + FalseNegatives}
\]  

(4.21)
4.5 Experimental Results

4.5.1 Dataset

The database for the experiments contains 1000 samples which are taken from AU-RORA database. The recordings are categorized into general classes according to common characteristics of the scenes (220 kitchen noise, 180 living room noise, 210 laundry sounds, 230 meeting sounds, 160 office sounds) and events (Pan boiling, steel plate, music player, paper scrap, washing machine, flush, overlapped speech, footsteps, typewriter, dust bin, etc.). The categorization of the scenes was somewhat ambiguous; some of the recordings are associated with more than one higher-level class. The recordings are manually labelled and are separated into 2-second, 3-second and 5-second fragments. Every sound signal was stored with some properties that are also the initial conditions and criteria for the well-functioning of the algorithm. The sample database is split into training sets and test sets. Randomly select 80% sounds of each class for the training set. The remaining 20% sounds form the test set. Thus we have taken different proportion of samples based on class dependency in each category as shown in Table 4.1.

<table>
<thead>
<tr>
<th>Context</th>
<th>Total amount of Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kitchen</td>
<td>22%</td>
</tr>
<tr>
<td>Living Room</td>
<td>18%</td>
</tr>
<tr>
<td>Laundry</td>
<td>21%</td>
</tr>
<tr>
<td>Meeting</td>
<td>23%</td>
</tr>
<tr>
<td>Office</td>
<td>16%</td>
</tr>
</tbody>
</table>

Table 4.1: Acoustic Database descriptor.
4.5.2 Acoustic Feature Extraction

To extract the features from the acoustic signal, the signal must be pre-processed and divided into successive windows or analysis frames. The training data is segmented into fixed-length and overlapping frames (in our experiments we used 20 ms frames with 10 ms overlapping). When neighboring frames are overlapped, the spectral characteristics of audio content can be taken into consideration in the training process. Since a 16 kHz sampling rate is deployed, 20 ms frames consists of 320 values. These 320 values are converted into 13 dimensional MFCC and 5 dimensional Spectral features coefficients (combined) which are for one frame \((320 \times 18)\). So there are 100 such frames for 1 second audio data and coefficients are extracted for the audio data as described in Section 4.2. The combined (MFCC and spectral) feature extraction process is repeated for the audio data of varying durations ranging from 2 Secs, 3 Secs and 5 Secs for all the five categories. Experiments are conducted for MFCC and spectral features and the performance of BPNN, RBFNN and Neuro Fuzzy(NF) classifier is studied.

4.5.3 Evaluation using BPNN, RBFNN and NF classifier

Back propagation neural network classifier to discriminate various events. Classification parameters are calculated using BPNN learning. The training process analyzes audio training data to find an optimal way to classify audio frames into their respective classes. The training data should be sufficient to be statistically significant. The BPNN learning algorithm is applied to produce the classification parameters according to calculated features. The derived classification parameters are used to classify the context of the audio data. The results show in Table 4.2, we observe that the overall classification accuracy is high for 3-second samples when compared to other duration.

To determine the performance of BPNN, we examine the results by varying the
number of hidden layers. Using the same settings as the rest of the experiments, we examined hidden neurons of 5, 10, 15 and 20 and used the same number of neurons for each environment type. The overall recognition rates are plotted in Fig. 4.8. We see that the classification performance peaks around fifteen and the performance slowly degrades as the number of neurons varies. The highest recognition rate for each class across the number of mixtures was obtained with 15 neurons.

![Performance of BPNN for different hidden neurons.](image)

**Fig. 4.8: Performance of BPNN for different hidden neurons.**

For RBFNN training, 13 dimensional MFCC and 5 dimensional Spectral features are extracted from the audio frames for each category. These features are given as input to the RBFNN model. The RBF centers are located using $k$-means algorithm.
The weights are determined using least squares algorithm. The means value of \( k = 3, 4, \) and 5 has been used in our studies for each category. The system gives optimal performance for means of \( k = 4 \). When the value of means \( k \) is increased to 10, the system shows no considerable increase in performance.

Figs. 4.9 show the performance of RBFNN for different means respectively.

![Performance of RBFNN for various means.](image)

\textbf{Fig. 4.9: Performance of RBFNN for various means.}

There performance are measured for various sample durations for RBFNN network structure with mean=4. The results are shown in Table 4.3.

\textbf{Table 4.3: Performance of RBFNN for means \( k = 4 \).}

<table>
<thead>
<tr>
<th>Sample Duration (in seconds)</th>
<th>2 Seconds</th>
<th>3 Seconds</th>
<th>5 Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recognition Accuracy for RBFNN</td>
<td>77.24%</td>
<td>90.65%</td>
<td>89.36%</td>
</tr>
</tbody>
</table>

For Neuro fuzzy classifier, 13 dimensional MFCC and 5 dimensional spectral features are extracted from the audio frames for each category. The input features values
are fuzzified by means of membership functions that make easy the membership of each feature to dissimilar classes. The unseen and inter-related information are removed from the features to the classes through the MF, which leads to get more accuracy of the classification phase by means of Neuro-fuzzy classifier.

To determine the performance of NF classifier, we examine the results by varying the number of hidden layers. Using the same settings as the rest of the experiments, we examined hidden neurons of 5, 10, 15 and 20 and used the same number of neurons for each environment type. Table 4.4 shows that the overall classification accuracy is high for 3-second samples when compared to other duration. The overall recognition rates are plotted in Fig. 4.10. We see that the classification performance peaks around fifteen and the performance slowly degrades as the number of neurons varies. The highest recognition rate for each class across the number of mixtures was obtained with 15 neurons.

Table 4.4: Performance of NF for 15 hidden neurons.

<table>
<thead>
<tr>
<th></th>
<th>2 Seconds</th>
<th>3 Seconds</th>
<th>5 Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recognition Accuracy for NF</td>
<td>75.36%</td>
<td>83.45%</td>
<td>79.20%</td>
</tr>
</tbody>
</table>
Table 4.5: Performance of BPNN, RBFNN and NF Classifier for Environmental sound Classification

<table>
<thead>
<tr>
<th>Classifiers</th>
<th>BPNN</th>
<th>RBFNN</th>
<th>NF Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance (in %)</td>
<td>Acc</td>
<td>Prec</td>
<td>Rec</td>
</tr>
<tr>
<td>MFCC</td>
<td>94.21</td>
<td>93.95</td>
<td>91.50</td>
</tr>
<tr>
<td>Spectral features</td>
<td>83.26</td>
<td>88.23</td>
<td>85.96</td>
</tr>
<tr>
<td>Combined MFCC + Spectral features</td>
<td>98.23</td>
<td>97.56</td>
<td>96.32</td>
</tr>
</tbody>
</table>

Fig. 4.10: Performance of NF for different hidden neurons.

Performance of BPNN, RBFNN and NF Classifier for Environmental sound Classification is given in Table 4.5.

Figs. 4.11 show the snapshot of the Environmental sound classification using BPNN.
4.6 Summary

In this chapter, the performance of Environmental acoustic scene recognition using BPNN, RBFNN and NF classifier has been analyzed. MFCC and Spectral features are extract characterize audio content. For recognition of all the 5-classes, it is shown that the BPNN did achieve significant overall recognition for every independent event. By varying the parameters of the learning rate and hidden neurons in Section 4.5.3, managed to increase the average recognition rate to 91.7%. RBFNN and NF on the other hand is also accurate as it is non-iterative, highly parallel and takes less time duration the training and testing phase. The performance of BPNN shows an accuracy of 91.7% than RBFNN and NF classifier. BPNN performs fairly well on most of them.