Chapter 5

Early classification on time series by using deep learning approach

In previous chapters, we addressed the problem of early classification on univariate as well as multivariate time series. The proposed models are adaptive to missing values by developing a series of probabilistic classifiers, namely GP. Earlier models also work immensely well for small datasets. However, the limitation of these works is that they need feature transformation to perform classification. To overcome this limitation, we developed a hybrid deep learning classifier that capture hidden temporal information from raw time series data without any feature transformation. Further, the optimal reliability threshold is defined for early decision-making. Specially we considered the problem of early transportation mode detection based sensory time series data.

5.1 Introduction

Nowadays smartphones embedded with a rich set of sensors such as accelerometer, magnetometer, gyroscope, proximity, etc. These sensors act as a portable sensors and provide flourishing support in the development of various useful applications such as physical activity recognition, human fall detection, patients monitoring and so on. In

recent times, the applicability of these sensors also utilized in automation of transportation systems that leads to construct the Intelligent Transportation System (ITS) applications [114]. One of the valuable application of ITS is to know the movements of users in a particular area or region. Thus, transportation modes are the essential elements of ITS in the development of its various applications from the user's context that signify how the users are moving around. By considering transportation modes, the movements are classified as stationary, walking, running and traveling with a bike, bus, train, or car [115]. Transportation Mode Detection (TMD) is considered as a sub-field of activity recognition which aims to automatically recognize the mode of transportation that a person uses [116]. TMD provides valuable support in various ITS applications, such as driving behaviour monitoring [117, 118], human activity monitoring [119], urban transportation planning [120], road environment and traffic prediction [121][122], etc.

Most of the existing methods for TMD heavily depend on handcrafted features extracted from raw sensor data [123, 124]. These methods are shallow in nature and require human expertise. However, recent developments in deep learning (DL) for feature extraction has been adopted in a variety of challenging real-world applications [2, 3, 125]. Thus, it has influenced the researchers of ITS and other fields to use the DL model to overcome the above mentioned issues. A Deep Neural Network (DNN) based model has been proposed to learn transportation modes from three sensors data and it outperformed the feature-based methods [115]. Liang *et al.* [116] proposed a state-of-the-art energy efficient TMD through a deep Convolutional Neural Network (CNN) model using the magnitude of 3-axis accelerometer sensor data. Nawaz *et al.* [126] introduced a convolutional Long Short Term Memory (LSTM) model for identifying the transportation mode from GPS trajectory data, by considering four transportation modes only. Finally, Wang *et al.*[127] presented a combined residual and LSTM Recurrent Neural Network (RNN) based method for TMD using smartphone's light-weight

sensors data.

The above study informs that the TMD problem has been addressed using traditional data-driven approaches based on finite length time series data. However, in time-sensitive applications such as driver activity recognition, it is recommended to take early decisions based on partially observed data to overcome the accident problem. Therefore, In this chapter, we have proposed an early classification approach for transportation modes detection as depicted in Figure 5.1.

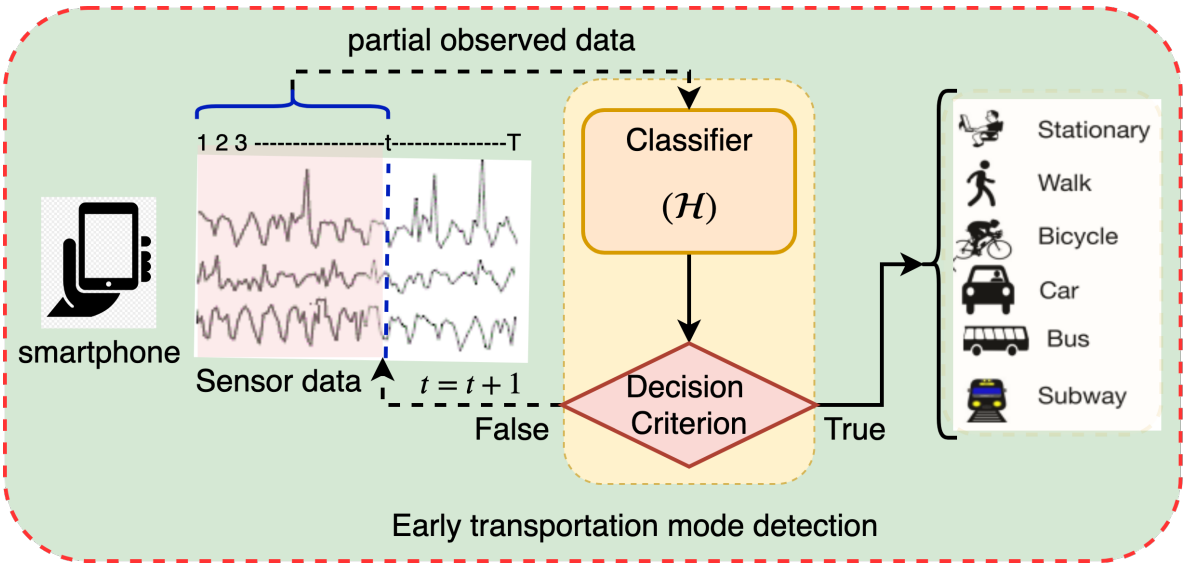


Figure 5.1: Classification model for sensory time series data.

5.2 Motivation and significant contributions

This work addressed the problem of early classification of transportation mode based on time series data. The significant contributions of this work are as follows:

- We propose an Early Transportation Mode Detection (ETMD) model that focuses on classifying the transportation modes as early as possible based on partially observed time series data. The learning of ETMD follows the two-fold process.
- In the first fold, ETMD exploits the capability of DL models such as CNN, RNN, and DNN for developing the hybrid-DL classifier, which mines the hidden tem-

poral information from the time series to classify the transportation modes effectively. This base classifier is trained by considering accuracy only as an objective.

- In the second fold, ETMD utilizes the capability of a base classifier to classify the time series at successive time points and defines the confidence threshold by keeping balance between accuracy and earliness. Moreover, this confidence threshold works as a decision policy for making a reliable prediction based on incomplete time series data.
- Finally, the proposed ETMD model is evaluated on two real-world time series datasets that have been collected using a smartphone for the TMD problem [116] [128]. The performance of ETMD is analyzed by considering accuracy, earliness, and confusion matrices as evaluation measures.

5.3 Preliminaries

Definition 5.1 (Confidence) *It measures the reliability of class prediction to determine whether the partially observed data points in time series, are sufficient for classification or not. At any time step t, the* $\hat{\delta}_t = \mathcal{F}_t^{conf}(\hat{y})$ *represents the confidence of* X *to predict* $\hat{y} \in \mathcal{Y}$*, where* $\hat{y} = \mathcal{H}(X_t)$*. The* X_t *is classified only if* $\hat{\delta}_t \geq \delta$*, where* δ *represents the confidence threshold.*

Definition 5.2 (Accuracy) *It is the performance measure of classifier* $\mathcal{H}$*. It defines the proportion of correctly classified time series to the total number of time series. It is computed as:*

$$\mathcal{H}_{acc} = \frac{\parallel \{X^i\} \, | \mathcal{H}(X^i) = y^i \parallel}{N}, i \in [1..N] \tag{5.1}$$

where N is the number of testing samples.

Definition 5.3 (Earliness) *It is the performance measure of early classifier* $\mathcal{H}$ *which*

defines the average prediction time of time series. It is computed as:

$$\mathcal{H}_{ear} = \frac{1}{N}\sum_{i=1}^{N}\frac{t^{i*}}{T}, i \in [1..N] \tag{5.2}$$

where t^{i*} is the prediction length of X^i, used for classification.

5.4 Model description

This section provides the complete description of the proposed ETMD model which is shown in Figure 5.2. The ETMD comprises of training phase and prediction phase. The training phase deals with the learning of ETMD model and accomplishes it in two-folds. The objective of first fold is to trained the base classifier $\mathcal{H}$ with labelled training set $\mathcal{D}$ that can provide accurate prediction for incoming time series. The second fold objective is to learn the decision criteria as confidence threshold to make early reliable prediction. Thus, second phase takes the trained classifier $\mathcal{H}$ as a pre-trained model and learns the decision criteria for ETMD. The following subsection explain the training and prediction phases in more details.

5.4.1 Training phase

5.4.1.1 Build the base classifier

Time series data generated over the time and as an effect, the temporal relationship among the values exists. Specifically, in sensor data, each value in time series does not hold significant useful temporal information. Therefore, the sequential DL classifiers like RNN are unable to provide acceptable results [116]. Thus, the proposed work exploits the properties of CNN, RNN, and DNN to mine the noteworthy hidden temporal information at different scales from the time series [129], as shown in Figure 5.2(a). Initially, X is presented to a convolution block that consists of $L_c \in \mathbb{N}$ layers. Each L_c layer is composition of 1-D convolution layer and pooling layer. The feature map

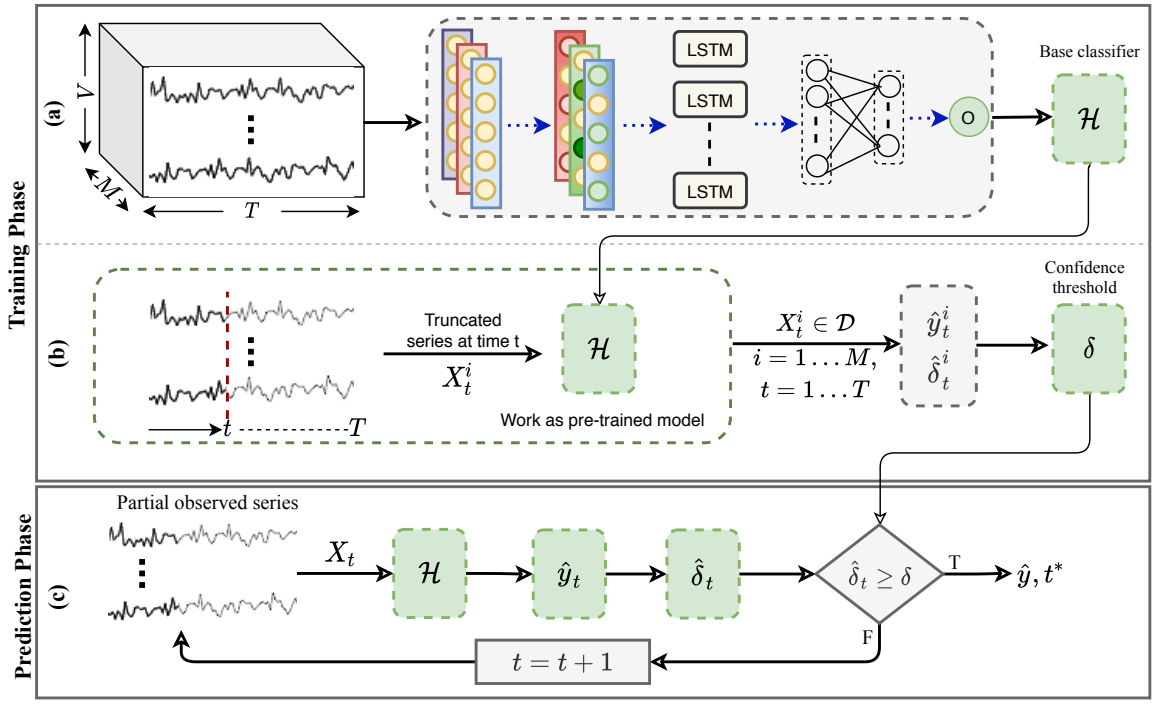


Figure 5.2: The Block diagram of the proposed ETMD model is divided into two phases: training and prediction. Part (a) denotes the training process of base classifier $\mathcal{H}$ and its architecture. Part (b) represents the confidence threshold selection for early decision making. Part (c) represents the prediction process on incoming time series data.

generated after the last operation of the convolution layer is given to $L_r \in \mathbb{N}$, where L_r is a RNN layer. In the proposed architecture of the $\mathcal{H}$, a special kind of recurrent block, i.e., LSTM cell, is specifically chosen due to its capability of learning the long term relationship in the temporal sequence. After the LSTM layer operation, the output is transferred to $L_f \in \mathbb{N}$, where L_f represents the fully connected layer. Finally, the output layer has q neuron to perform classification task where $q = |\mathcal{Y}|$ is the number of classes in the training set.

Convolutional block : The layers present inside this block works as a feature extractor by reducing the temporal variation in data [129]. In particular, sparse weights are the characteristic of CNN and it supports to capture the informative feature by applying convolution filters, which are small in size as compared to the length of an

input sequence. For given input sequence X, the feature maps Z_k^l are obtained with application of convolution operation for l^{th} layer, and it is defined as:

$$Z_k^l = \mathcal{F}\left(\sum_{i \in N_k} Z_i^{l-1} * w_{ik}^l + b_k^l\right) \tag{5.3}$$

where w_{ik}^l and b_k^l denote the weight and bias of k^{th} convolutional filter, respectively and N_k is the input feature maps. The output of each convolutional layer is followed by a pooling layer that reduces the dimension of data by preserving useful information and it is presented as:

$$\widetilde{Z}_k^l = \mathcal{F}\left(\beta_k^l . \psi(Z_i^{l-1}) + b_k^l\right) \tag{5.4}$$

where β_k^l, b_k^l, and $\psi(\cdot)$ denote the weight, bias, and pooling function, respectively.

The convolutional block consists of four sequential combinations of convolutional and pooling layers. Former two layers use 64 feature maps and later two layers use 32 feature maps. The filters of 5×1 size are operated on each convolutional layer with stride 1 except the first layer, which employs a filter of size 15×1. This helps in getting higher-order approximations from the raw input sequence. In each layer, max-pooling strategies are adopted without overlapping. The first three layers use the pooling size of 2 and the fourth layer uses a pooling size of 4 in the proposed architecture.

LSTM block : LSTM is an advanced RNN architecture introduced to mitigate the drawback of vanishing gradient problem of RNN. Thus, it has the capability to capture long term temporal relationships and, hence it is appropriate for modeling sequence data [130]. LSTM cell controls its state with the assistant of three gates (input, forget and output) that helps in preserving long term dependencies. Let us consider at each time step τ, LSTM cell outputs h_τ as the hidden state, c_τ as cell state then computation

of h_τ and c_τ is expressed by following equations:

$$\begin{aligned} i_\tau &= \sigma(\mathcal{W}_i \cdot [h_\tau, z_\tau] + b_i) \\ f_\tau &= \sigma(\mathcal{W}_f \cdot [h_\tau, z_\tau] + b_f) \\ o_\tau &= \sigma(\mathcal{W}_o \cdot [h_\tau, z_\tau] + b_o) \\ c_\tau &= f_\tau \circledast c_{\tau-1} + i_\tau \circledast \tanh(\mathcal{W}_c \cdot [h_{\tau-1}, z_\tau] + b_c) \\ h_\tau &= o_\tau \circledast \tanh(c_\tau) \end{aligned} \tag{5.5}$$

where z_t is the feature map returned by the convolution layer. $(\mathcal{W}_i, b_i), (\mathcal{W}_f, b_f)$ and $(\mathcal{W}_o, b_o)$ represent the weights and biases of input gate, forget gate, and output gate, respectively. In the proposed architecture, one layer of LSTM is designed with 100 nodes, which captures the temporal relationship among the features. These features more likely represent the composite information of small sub-sequences at different scales in X.

Fully Connected block : This block produces the higher-level feature representation that is more appropriate to discriminate the classes. In the proposed architecture, one fully connected layer and one output layer are used of size 100 and $|\mathcal{Y}|$, respectively. Except for the output layer, all layers in the architecture uses *Rectified Linear Unit* (ReLU) as a nonlinear function that provides fast convergence with stochastic gradient descent (SGD)[131]. ReLU in its simple form is defined as $f(x) = max(0, x)$. The output layer used *softmax* function that provides the class label having highest probability.

The proposed model contains over 2 lakh parameters and, therefore, to reduce the problem of over-fitting, two dropout layers with a drop-rate of 50% each are employed. The first dropout layer is placed before the LSTM layer, and the second dropout layer is applied before the output layer. Finally, the model is trained through an SGD optimizer with *cross-entropy* as a loss function. The model's hyper-parameters, such as optimizer,

Algorithm 5.1: Model Training Phase

Input: Training set $\mathcal{D} = \{(X^i, y^i)_{,1 \leq i \leq M}\}$, set of time steps $Ts = \{t_1, t_2, \ldots, t_n\}$, and threshold parameter α.

Output: Base classifier $\mathcal{H}$, Classifiers performance matrices$\{\mathcal{H}_t^P(y|\hat{y})|y, \hat{y} \in \mathcal{Y}, t \in Ts\}$ and Confidence threshold δ.

```
/* Train the base classifier                                          */
```
$\mathcal{H} \leftarrow trainModel(\mathcal{D})$

```
/* Compute the performance of base classifier at different time step t */
```
for *t in Ts* **do**
- $\mathcal{D}_t \leftarrow truncateData(D, t)$
- predictClasses $\leftarrow modelPredict(\mathcal{H}, \mathcal{D}_t)$
- Calculate $\mathcal{H}_t^p(y|\hat{y})$ using Eq. (5.6)

```
/* learn the confidence threshold                                     */
```
for *i=1 to M* **do**
- **for** *t in Ts* **do**
  - $\hat{y}_t^i \leftarrow \mathcal{H}(X_t^i)$
  - Calculate the $\hat{\delta}_t^i \leftarrow \mathcal{F}_{:t}^{conf}(\hat{y}_t^i)$ using Eq. (5.9)
  - Append $\hat{\delta}_t^i$ into Δ
  - Store $\hat{y}_t^i$, and $\hat{\delta}_t^i$

Sort and remove duplicate values of Δ

for *j=2 to len(Δ)* **do**
- $\delta' \leftarrow mid(\Delta[i-1], \Delta[i])$
- **for** *i = 1 to M* **do**
  - **for** *t in Ts* **do**
    - **if** $\hat{\delta}_t^i \geq \delta'$ **then**
      - Store $\hat{y}_{t*}^i$, t^*
      - *break*
- Calculate $\mathcal{F}_{cost}(\delta')$ using Eq. (5.1),(5.2),(5.10)

$\delta \leftarrow \arg\min_{\delta' \in \Delta}\{\mathcal{F}_{cost}(\delta')\}$

activation, learning rate, etc., are selected by initial testing on the validation set.

5.4.1.2 Learn confidence threshold for early classification

The objective of this phase is to learn reliable confidence threshold δ that works as decision criteria for early TMD. For learning δ, we adopted similar approach as defined in [12]. The pseudo-code for training the model is presented in *Algorithm 5.1*.

Let X is classified as $\hat{y}_{t'}$ and $\hat{y}_{t''}$ at two successive time steps t' and t'' with 92% probability each. In this scenario, generally the confidence is estimated based on class probability. However, it is not sufficient for reliable early classification. For example, $\hat{y}_{t'}$ and $\hat{y}_{t''}$ have the same probability. Even the confidence of prediction may not be the same because $\hat{y}_{t''}$ may have higher precision, due to having more number of data points at time step t'' as compared to t'. Thus, for reliable TMD, the confidence at successive time steps is evaluated by fusing classifier's class prediction capability [12].

Let us consider that $\mathcal{H}^P(y|\hat{y})$ denotes the performance of classifier $\mathcal{H}$ that measures the possibility of having actual class label y while predicted class label is $\hat{y}$. It is defined as:

$$\mathcal{H}^P(y|\hat{y}) = \frac{\parallel \{X^i|y^i = y \ \& \ \mathcal{H}(X^i) = \hat{y}\} \parallel}{\parallel X^i|\mathcal{H}(X^i) = \hat{y} \parallel}, i = 1, 2, \ldots, M \tag{5.6}$$

Thus, if a sample X is classified as $\hat{y}$ at time step t then the confidence of prediction $\hat{y}$ is measured as:

$$\mathcal{F}_t^{conf}(\hat{y}) = \mathcal{H}_t^P(\hat{y}|\hat{y}) \tag{5.7}$$

Similarly, if sample X is classified as $\hat{y}_{t_1} = \mathcal{H}_{t_1}(X)$ and $\hat{y}_{t_2} = \mathcal{H}_{t_2}(X)$ at two successive time steps t_1 and t_2, respectively, then composite confidence at time step t_2 is computed by the following expression:

$$\mathcal{F}_{:t_2}^{conf}(\hat{y}_{t_2}) = 1 - (1 - \mathcal{H}^P(\hat{y}_{t_2}|\hat{y}_{t_1}))(1 - \mathcal{H}^P(\hat{y}_{t_2}|\hat{y}_{t_2})) \tag{5.8}$$

Eq. 5.8 considers the confidence of prediction $\hat{y}_{t_2}$ at two different time steps t_1 and t_2. As a result, two cases are possible. (i) When prediction at two time steps are same $\hat{y} = (\hat{y}_{t_1} = \hat{y}_{t_2})$, then composite confidence will be high as compared to individual prediction confidences $\mathcal{F}_{t_1}^{conf}(\hat{y})$ and $\mathcal{F}_{t_2}^{conf}(\hat{y})$, because $\hat{y}$ is predicted two times. (ii) If $\hat{y}_{t_1} \neq \hat{y}_{t_2}$, the classifier $\mathcal{H}$ predicts $\hat{y}_{t_1}$ at time step t_1, even though there is a possibility of prediction $\hat{y}_{t_2}$ at time step t_1, and it is computed as $\mathcal{H}^P(\hat{y}_{t_2}|\hat{y}_{t_1})$. Thus, intuitively, in this case, the composite confidence will be lower as compared to the first case.

Generally, sensor data X is collected over the time, and as a result, the multiple output labels at different time steps are obtained. Thus, the composite confidence of X at time step t_n is computed as:

$$\mathcal{F}^{conf}_{:t_n}(\hat{y}_{t_n}) = 1 - \prod_{k=1}^{n}(1 - \mathcal{H}^{P}_{t_n}(\hat{y}_{t_n}|\hat{y}_{t_k})) \tag{5.9}$$

where $t_n \leq T$ is the n^{th} time point and the confidence score of X at any time point t is bounded as $\mathcal{F}^{conf}_{:t}(\mathcal{H}(X_t)) \in [0, 1]$. Now, for providing the reliable early TMD, the value of δ is essential. One way is to set δ with empirical knowledge of data that needs high expertise because infinite values are possible. Another way, adopted in ETMD is to select an optimum value of α by minimizing the cost of earliness and accuracy such that,

$$\mathcal{F}_{cost}(\delta) = \alpha * (1 - \mathcal{H}_{acc}) + (1 - \alpha) * \mathcal{H}_{ear} \tag{5.10}$$

where $\alpha \in [0, 1]$ is the balancing parameter and thus higher value of α assign higher weight to accuracy and vice-versa. Therefore, the following steps are performed to learn the optimum δ value.

- Compute the performance of $\mathcal{H}$ at each time step t by considering the base classifier as pre-trained model. To achieve this task, at each time point t, X_t is padded with its current mean to make full-length sequence.
- Besides, determine the confidence $\hat{\delta}$ for each $X \in \mathcal{D}$ at each time step t. Further, to learn the optimal δ, all $M * T$ confidence values are sorted and then eliminate the duplicate values. Next, compute the mean of every two nearest pair that works as a candidate for confidence threshold.
- Further, calculate the accuracy and earliness for each candidate δ' using Equs. (5.1) and (5.2), respectively. Finally, select the best threshold candidate δ that minimizes the total cost for the training set.

5.4.2 Prediction phase

The prediction phase of ETMD is discussed in this section and demonstrated in Figure 5.2(c). At time t, the partially observed series X_t is presented to $\mathcal{H}$ for predicting the class label $\hat{y}_t$ and it is defined as $\hat{y}_t = \mathcal{H}(X_t)$. Based on $\hat{y}_t$, the model computes the confidence $\hat{\delta}_t$ by using the Eq.(5.9). The $\hat{\delta}_t$ value is compared with δ at time t, and if $\hat{\delta}_t$ is found to be equal or grater than δ, then the proposed model classifies the series with $\hat{y}_t$ (corresponding transportation mode). If the condition fails, then the model waits for more data points to be added in X and repeats this process until $\hat{\delta}_t$ satisfies the confidence threshold criteria.

It is important to note that δ is a crucial parameter in ETMD model that is assisted by parameter α in achieving the trade-off between accuracy and earliness. Moreover, ETMD accepts raw sensor data directly with minimal preprocessing and provides early decision in realtime.

5.5 Experimental evaluation

This section presents the analysis of the results using both the datasets. Firstly, it is vital to examine the confidence learning step of the training phase. Two parameters are associated with confidence learning, a set of time steps and trade-off parameter α. For experimental analysis, twenty equi-length time steps of time series are considered with step value of 5%, and a set of values of α are considered in the range of 0.1 to 0.9 at the interval of 0.1.

5.5.1 Datasets description

Two publicly available datasets are utilized to examine the effectiveness of the proposed model. The significant difference between both the datasets is the experimental conditions of the data collection process. In the case of "The University of Sussex-Huawei

Locomotion and Transportation (SHL) Dataset" [132, 128], the orientation and location of a smartphone are fixed while TMD dataset [116] is free from this restriction. The brief descriptions of datasets are as follows:

- *Smartphone sensing data for transportation mode detection* (TMD) [116]: The accelerometer sensor of the smartphone is utilized to develop a supervised dataset. A particular android-application has been developed and installed into smartphones for the purpose of collecting the data under seven transportation modes. These modes are walking, bicycling, moving via bus, driving a car, traveling in a train or subway, and being in still position. In the collection process of data, transportation mode activity of each traveler has been recorded and labeled manually. The sampling frequency of the data collection was 50 Hz, and the duration of each transportation mode activity was 2 hours. During the complete process, travelers were not restricted from carrying the smartphones in some specific position or orientation. This unrestricted environment of the data collection process makes the dataset challenging to detect the different transmission modes. However, the data collected from the accelerometer has put into 3-dimensional vectors and each data point represents the x-y-z coordinate system of smartphone.
- *The SHL dataset* [132, 128]: The SHL dataset was recorded in the United Kingdom. Three participants assisted in recoding the dataset. Each participant carries smartphones, which have specially designed android application for data recording. The multimodality is the noteworthy characteristic of SHL dataset. Moreover, the synchronized multimodal data have been recorded for different modality including accelerometer, magnetometer, gyroscope, GPS, and others. Each participant carries four smartphones, and they were placed in four different positions: in hand, at the torso, in the hip pocket, and in a bag. The data collection process was performed for 2812 hours in 703 days, and each day, only 4 hours of data was captured. The eight different transportation modes were considered in the pro-

cess, and they are *Still, Walk, Run, Bike, Car, Bus, Train,* and *Subway.* In our experiment, we have considered accelerometer sensor data with a Bag position.

Data preparation:

It is important to note that most of the TMD application has been designed and developed by utilizing the accelerometer data only, due to its ability to carry significant information about modes [133]. This work also considers accelerometer data to evaluate the ETMD model. In the pre-processing step of the proposed method, the magnitude of 3-axis accelerometer data is employed due to its robust characteristics with respect to smartphone orientation [116]. The magnitude of sensor data $x_t = (x_{t_x}, x_{t_y}, x_{t_z})$ is computed by following expression:

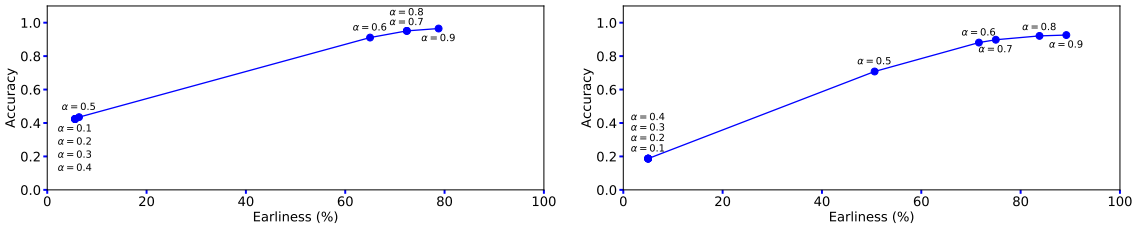
$$|x_t| = \sqrt{\sum_{v=x,y,z} x_{t_v}^2} \tag{5.11}$$

In addition, the dataset is prepared by considering the sliding window of size 512 with 75% overlapping for evaluating the performance of the model [115, 116].

5.5.2 Results analysis

5.5.2.1 Effect of α parameter

The parameter α plays a crucial role in achieving a decent trade-off between accuracy and earliness. Figure 5.3 illustrates the relationship between accuracy and earliness for both the datasets. It is observed that with increasing the α value, classification accuracy increases and simultaneously reduces the earliness effect. Besides, it is noted that the trend of α is somewhat similar for both the datasets. The accuracies are almost constant for α ranging from 0.1 to 0.4 that are 42% and 19% respectively for TMD and SHL datasets. Likewise, the identical stable characteristic is seen in the case of TMD dataset for $\alpha \in \{0.7, 0.8\}$, and the accuracy is around 95%. However, the accuracy of the model gradually improves by changing the α value from 0.4 to 0.9 for both the

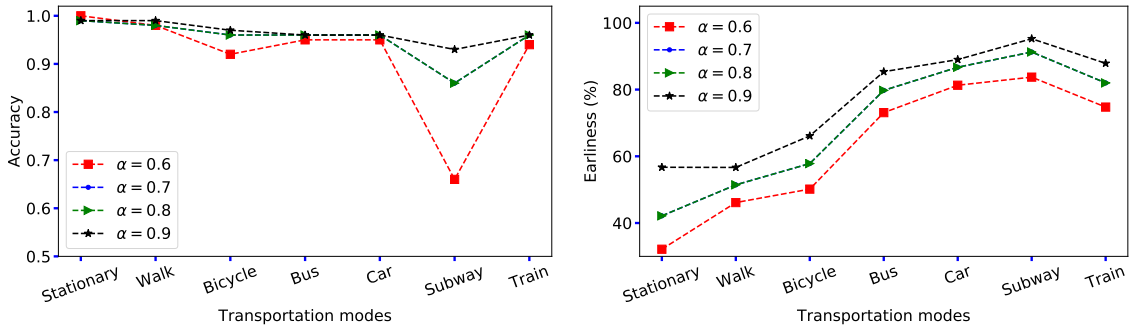


(a) TMD dataset (b) SHL dataset

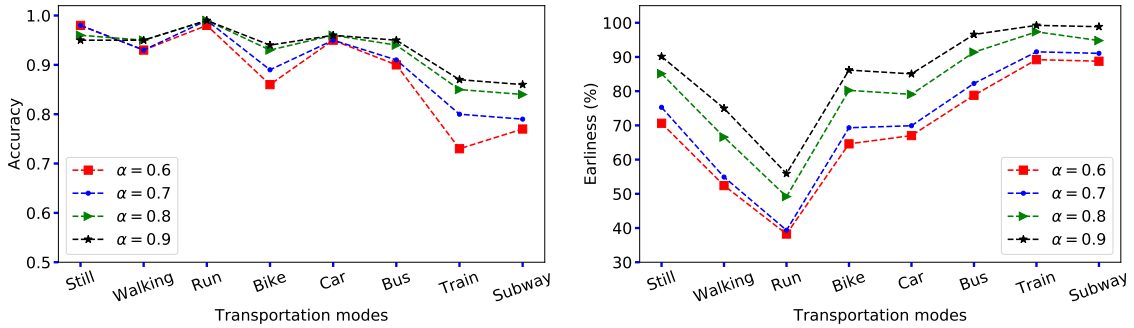
Figure 5.3: Accuracy vs. earliness curve for α.

datasets. Thus, for better achievement of earliness, low value of α is preferred whereas, in need of high accuracy, the value of α should be considered high. It is also observed that, for $\alpha \geq 0.6$, change in the accuracy value is very low as compared to earliness value. For example, the differences in accuracy and earliness values are around 5% and 14% respectively for the TMD dataset, as shown in Figure 5.3(a). Similarly, for the SHL dataset, differences in accuracy and earliness are 5% and 18%, respectively, as shown in Figure 5.3(b).

Further, Figure 5.4 shows the impact of α parameter on different transportation modes. Figure 5.4(b) and 5.4(d) demonstrate that the earliness value increases for all the transportation modes as α changes from 0.6 to 0.9. However, the same trend is not seen for accuracy. As shown in Figure 5.4(a), the accuracy for transportation modes (*Bicycle*, *Subway*) improves by 5% and 27%, respectively, whereas accuracy improves by 1% only for *Walk*, *Bus* and *Car* modes. Similarly, for the SHL dataset, the accuracy improves for transportation mode *Train* from 0.73 to 0.87 whereas 0.98 to 0.99 for *run*, as shown in Figure 5.4(c). It signifies that the impact of α is more on transportation mode *train* and *Subway* as compared to *Still*, *Walking*, or *Run*. Interestingly, it is notable that ETMD requires very fewer data points to classify the transportation modes *Run* and *Walking* as compared to motorized modes *Train* and *Subway*. Thus, based on the above analysis, it is identified that $\alpha = 0.8$ provides a acceptable trade-off between accuracy and earliness. As a result, 0.8 is considered as a default value of α for the rest



(a) TMD dataset (b) TMD dataset



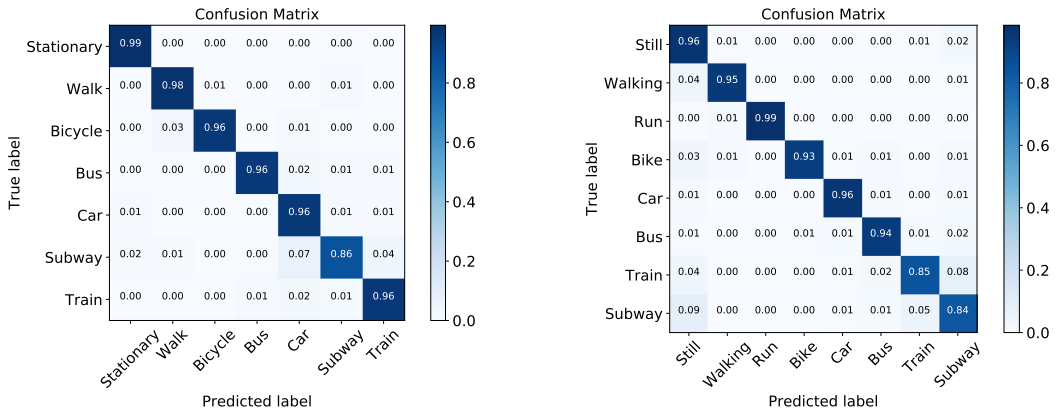
(c) SHL dataset (d) SHL dataset

Figure 5.4: Effect of α on accuracy and earliness over individual transportation mode.

of the analysis. However, in a practical scenario, the value of α will vary based on the need of the ITS applications.

5.5.2.2 Model performance

The number of samples for each transportation mode is an imbalance in both the datasets. The SHL dataset has a high imbalance ratio of minor and major class that is (1:34) while the TMD dataset has a moderate class imbalance ratio (1:3). Thus confusion matrix is a good measure to analyze the performance of the proposed model. The confusion matrix for dataset TMD and SHL are shown in Figure 5.5(a) and in Figure 5.5(b), respectively. The proposed model demonstrated its capability to detect the transportation mode accurately. It is observed that the proposed model is able to classify all transportation modes with decent accuracy ($\geq$ 95%) except *Subway*. It is challenging to classify the motorized transportation modes such as *Bus*, *Car*, and



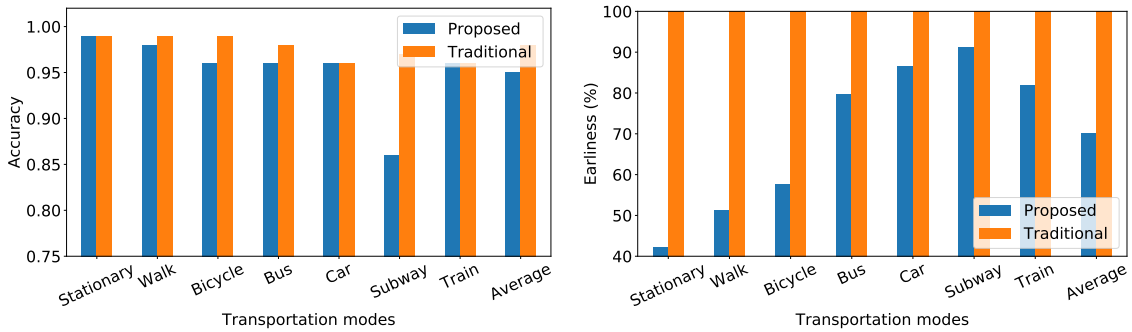
(a) TMD dataset (b) SHL dataset

Figure 5.5: Confusion matrix (a) TMD dataset (b) SHL dataset

Subway accurately [116]. The proposed model demonstrates the acceptable performance for the modes like *Bus* and *Car* for both the datasets. But, it is still a difficult task to achieve the adequate categorization of the *Train* and *Subway* modes. However, the higher value of α assists in attaining the high prediction reliability of the system.

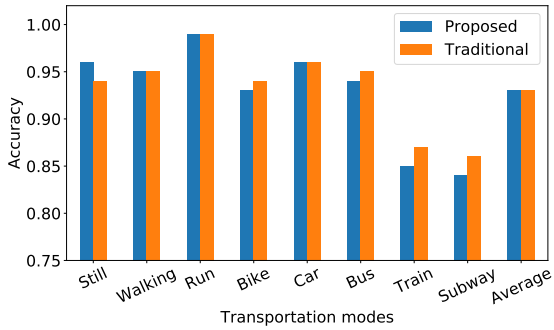
5.5.2.3 Model comparison with the traditional approach

The applicability of the proposed ETMD model is testified by comparing it with traditional approach. The comparative results are displayed in Figure 5.6, and also the average value of performance is calculated for both the datasets to perform the relative analysis of outcomes. The Figure 5.6(a) and 5.6(b) illustrate that ETMD utilizes approximately 50% of data points of full-length activity to predict the transportation modes (*Stationary*, *Walk* and *Bicycle*) with almost similar performance. It is also noticed that the proposed model achieves 70.16% earliness by compromising roughly 3% overall accuracy in comparison to average performance. For the SHL dataset, *running* mode is detected at a very early stage by using about 45% of data points of full-length activity while attaining the same accuracy as the traditional approach. It is also observed that the motorized modes such as *Train* and *Subway* require more data points

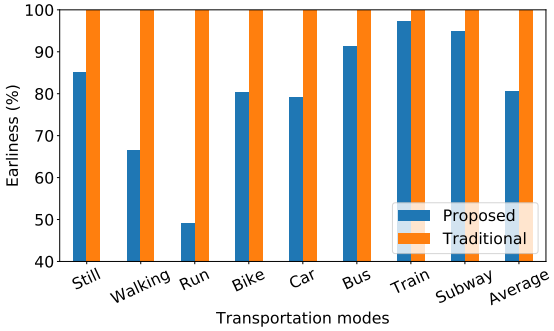


(a) TMD dataset

(b) TMD dataset



(c) SHL dataset



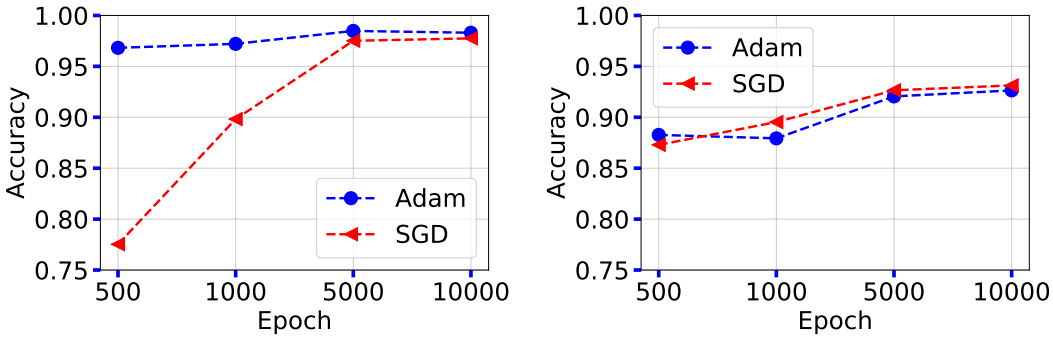
(d) SHL dataset

Figure 5.6: Comparison between proposed ETMD and traditional approach. The last *bar*, named as average, denotes the mean accuracy or earliness over all transportation modes.

as compared to *walking* and *Run* for reliable prediction. It is evident from Figures 5.6 (c)-(d) that the proposed model outperforms the traditional approach and achieves 80% earliness while attaining almost similar accuracy. Based on the above observation, it can be concluded that the proposed model is capable of classifying the transportation modes at an early stage based on partially observed activity information.

5.5.2.4 Effect of optimizer

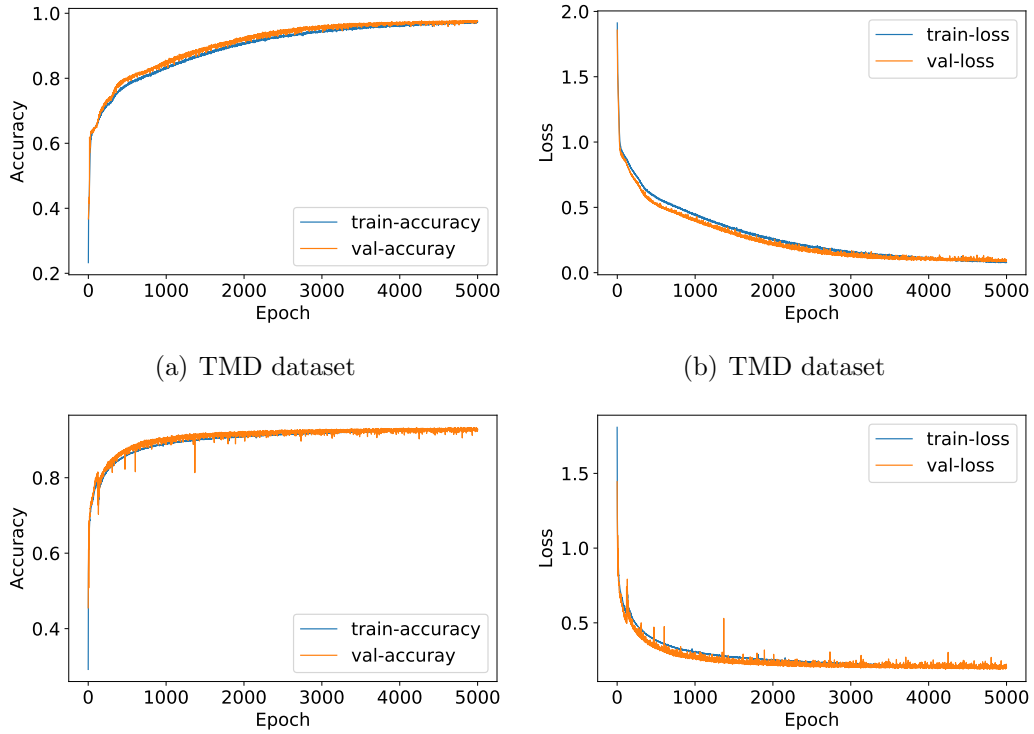
The effect of the two optimizers SGD and Adam are analyzed to learn the transportation modes by the base classifier in the proposed approach. SGD and Adam are used with default parameter settings except for the learning rate $lr = 0.001$. It is observed that Adam optimizer is able to provide good accuracy after few hundred of the epoch, but



(a) TMD dataset (b) SHL dataset

Figure 5.7: Effect of optimizers SGD and Adam

as the learning of the model progresses for the higher number of epochs, SGD provides a more stable solution as compared to Adam as shown in Fig 5.7 (a)-(b). Therefore, SGD optimizer is chosen in ETMD model. It can be observed in Figure 5.8, that the accuracy and loss graph for both the datasets is very smooth and stable.



(a) TMD dataset (b) TMD dataset

(c) SHL dataset (d) SHL dataset

Figure 5.8: Accuracy vs. epoch and loss vs. epoch learning curve of hybrid-DL classifier.

Table 5.1: Comparison of base classifiers in ETMD model

	TMD		SHL	
	Accuracy	Earliness (%)	Accuracy	Earliness (%)
DNN	0.56	09.50	0.41	60.80
LSTM	0.58	05.00	0.62	71.64
CNN	0.92	79.40	0.82	53.20
Proposed	**0.95**	**72.40**	**0.92**	**83.80**

5.5.2.5 Comparison of base classifiers in ETMD model

To verify the effectiveness of the proposed hybrid DL-based ETMD model, the famous DL models such as CNN, LSTM, and DNN are used for the comparison. The architecture of these models is considered similar to the proposed model. Table 5.1 illustrates that the sequence model LSTM is unable to capture the temporal pattern and demonstrates an unsatisfactory performance of 58% and 62% for both the TMD datasets. CNN model achieves better accuracy as well as earliness in comparison to DNN and LSTM. This effect is observed in CNN due to its potential of capturing the local patterns from multiple transportation modes data. However, CNN is unable to remember the temporal relationships among these local patterns and as a result, the proposed hybrid-DL model mitigates the limitations of individual DL models like CNN, LSTM and DNN. Hence, the proposed model outperforms them in terms of both accuracy and earliness. It can also be seen from Figure 5.8, the hybrid-DL model is able to learn the transportation modes accurately for both the datasets, as depicted by accuracy and loss curves in Figure 5.8(a)-5.8(d).

5.6 Summary

The ease of sensors through smartphones has assisted in developing various meaningful applications in multiple domains. Also, it has widespread usability for constructing intelligent transportation applications. It is essential to determine the transportation mode as early as possible to make adequate early decisions without waiting for complete

series. In this Chapter, we have presented an early transportation mode detection model based on smartphone sensor data. The proposed model is able to detect transportation mode based on a partially observed sequence. The time series data carries the useful hidden temporal pattern. As an effect, it is mined by exploiting the capabilities of CNN, RNN and DNN and construct the hybrid DL classifier to achieve satisfactory performance. The earliness is achieved by designing the decision policy on the top of a classifier to predict reliable transportation mode.

The experimental results demonstrated acceptable performance on publiclly available supervised transportation datasets. It has been noted that the hybrid DL model is able to capture the temporal features for better performance as compared to the individual DL model. Moreover, the trade-off between accuracy and earliness depends on the selection of the proper value of α. The proposed method provides a satisfactory result in comparison to existing alternatives.

Chapter 6

Conclusion and Future Directions

This chapter summarizes the important conclusions obtained from the contributions in this thesis. Additionally, it provides promising future directions to explore the problem of the early classification further.

6.1 Conclusion

In this thesis, we studied the problem of early classification of time series data by learning optimal decision criteria. The objective of early classification is to predict the class label of time series as early as possible with acceptable accuracy. The early classification problem is applicable in many domains, where data points are obtained over time. Moreover, it is highly desirable, where either collecting data points are expensive or timely decision is required.

In Chapter 2, we reviewed the existing literature on early classification of time series to find the research gap and limitations of the existing works. Broadly early classification approaches can be categorized into three groups: instance-based, shapelet-based and model-based. Shapelet-based methods are highly interpretable to the user. However, they have some limitations. Firstly, they are highly computationally expensive. Secondly, it is likely very hard to define the shapelet threshold if the time series be-

long to different class groups and do not have distinguishable patterns. On the other hand, model-based approaches are computationally moderate, but they are lacking in interpretability.

The problem of early classification has been identified as the composition of two sub problems. The first one is to design the early classifier that can label the incomplete time series. The second is to define the decision policy that can estimate the right time for making an online decision. Basically, the early classification problem has two conflicting objectives, i.e., accuracy and earliness. Existing approaches consider that the balancing between accuracy and earliness is essential for early classification problems. Even a very few methods have considered trade-off optimization between these two objectives.

In Chapter 3, we addressed the problem of early classification on univariate time series. A series of probabilistic classifiers have been developed to predict the class label for incomplete time series. Then two different strategies have been designed for decision making. The first method has been designed based on two critical aspects safeguard point and confidence threshold. The safeguard point reduces the unnecessary overhead of training the classifiers and ensures the desired accuracy. The confidence threshold ensures reliability in class prediction defined by measuring uncertainty in the predicted output. In the proposed approach, we have analyzed the impact of different probabilistic classifiers such as Naive bays, SVM, and GP. The GP classifier provided a good approximation of class labels as compared to others.

To achieve the trade-off between accuracy and earliness is a key challenge. However, the proposed early decision criterion has not taken it into consideration and is inclined toward accuracy only. Thus, the second method considered an optimization-based approach and designed the early stopping rules that have been learned by optimizing the trade-off between accuracy and earliness. The proposed model demonstrated good balance between accuracy and earliness as compared to the other methods when evaluated

on publicly available synthetic as well as real datasets. Moreover, the applicability of the proposed approach has been validated for early malware detection on the publicly available malware API call sequence dataset and demonstrated decent performance. These two approaches have been validated on UTS problem.

The many real-world applications generate multivariate time-series data that is more challenging compared to univariate time series. Thus we have extended the optimization-based early classification approach for MTS data in Chapter 4. In the proposed method, we have developed a series of probabilistic classifiers for each variable separately to capture the variate-wise information and adopted an ensemble-based classification approach to predict the class label for incomplete time series. Moreover, ESRs have been proposed to perform early decision tasks. In the proposed method, the trade-off between accuracy and earliness has been defined through α parameter. The proposed approach has been analyzed on existing real-world datasets, and it is found that the model is not generalized. In fact, the trade-off between accuracy and earliness depends on the characteristics of application data. However, the proposed model is able to maintain a good balance between earliness and accuracy.

The above methods have two limitations in terms of defining baseline classifier. First, a series of probabilistic classifiers have been developed for labelling the incomplete time series. Moreover, the number of classifiers depends on the number of data points in a complete time series. Second, feature transformation is needed for training the classifiers.

Therefore, in Chapter 5, we have proposed an early classification approach to overcome these issues by developing a deep learning-based early classifier that can capture hidden patterns from raw sensory data directly. The proposed model adapted an imputation-based approach for labelling the incomplete time series, and decision criterion is defined as the reliability threshold. To test the effectiveness of the proposed model, we have considered the problem of early transportation mode detection based

on smartphone sensor data. The proposed model has been evaluated on two real-world transportation data sets and demonstrated excellent performance. Besides, it has been observed that the hybrid DL model is able to capture the temporal information from the raw time series more effectively compared to the individual DL models.

6.2 Future directions

Based on the research work presented in this thesis, the following are promising future directions to explore more.

- The problem of early classification has two sub-problems, (i) designing of the early classifier and (ii) developing of good decision policy. The design of decision policy is a crucial part of an early classification problem. In the future, more complex weighted ESRs can be designed by assigning the higher weight to more informative components in MTS. Furthermore, the proposed model can be optimized for specific applications such as early voice detection, and gait recognition.
- Interpretability is also a desirable parameter in many applications for making an acceptable decision for the user in field, such as health, agriculture, etc. Therefore, to tackle early classification problem, developing interpretable decision rules with trade-off optimization between accuracy and earliness can be a potential future direction.
- This work does not consider multimodal data; therefore, the development of application-specific early classifier by considering multimodal data can be a good research direction. For example, driving behaviour analysis is a potential problem in ITS that can be monitored using multimodal data such as steering wheel angle, acceleration pressure, and the gear shift position.
- Deep learning models have automatic feature extraction capabilities. Therefore, in the future, domain-specific early classification approaches can be developed by adding the capability to handle unseen class labels while making an early decision. In this line, transfer learning and federated learning could be helpful.