Chapter 4

A sensor-based river water pollution assessment system with noisy data using deep neural network

The previous chapter presented river water data collection process, pre-processing of the data to make it suitable for detecting the river water pollution. Next, we estimated the WQI and obtained the labeled lab dataset. Here, in this chapter, we propose an automatic annotation of massive sensory data instances using labeled lab data. Automatic annotation inserts noisy labels in the data that adversely affect the performance of the model. Therefore, this chapter considers the problem of noisy labels in the dataset and propose the approach to handle the noisy labels. Finally, we also propose a sensor-based assessment of the river water pollution using deep neural network.

4.1 Introduction and major contributions

In water pollution assessment, the sensors measure various water parameters and the generated data can be used to analyze the pollution level in the water bodies (*e.g.,* river, pond, lake, *etc.*). The pollution level in the river water can be identified if and only if

the sensory data is correctly annotated with the class labels (*e.g.*, excellent, good, worst, *etc.*) for training a classifier. Water pollution level assessment is an important part of research because we can live without food for many days but we can not live without water more than three days. The water pollution level can be identified by using different learning techniques such as machine learning and deep learning. The automatic feature extraction capability of deep learning techniques enhances their use in various domains [66]. The researchers are also emphasized on deep learning technique like Convolution Neural Network (CNN) and Long Short Term Memory (LSTM) for identifying different patterns in the sensory data [67]. One of the limitations that comes up with the deep learning technique is the requirement of a large number of truely annotated instances for training. However, annotating a large amount of sensory data collected from the river is impractical [68,69]. Deep learning techniques are competent enough to extract the features automatically from raw data with remarkable performance [22, 70]. However, these techniques require considerable correctly labeled data, but gathering such annotated data is costly and tedious. The data analysts, therefore, handles different labeling techniques such as labeling through crowdsourcing, web-based queries, and so on. In general, non-expert volunteers annotate the data. Therefore, there is a high chance of inserting wrong labels in the dataset. If noisy labels are present in the dataset, the classifier learns a wrong mapping, which diminishes performance [70]. A deep learning-based classifier is built by learning a mapping between raw sensory data and their annotated labels with some wrong labels of water pollution, such as "good" instead of "bad" and "bad" instead of "very good" *etc.*. The traditional techniques for river water classification do not provide the mechanism to assign labels to the dataset automatically [28, 29, 32]. In other words, deep learning technique can not be applied to the river data classification unless the automatic annotation scheme is incorporated. In this work, we are using automatic annotation technique; therefore, the deep learning model can be applied to identify pollution level of river water. The automatic annota-

tion sometimes leads to insertion of noisy labels in the data that must be handled before training. This work uses noise handling mechanism to mitigate the damage caused by the noisy labels.

Later, the authors in [34] employed deep learning technique for retrieval of cyanobacteria pigment in river water. Finally, James rising [35] proposed an integrated assessment model that highlights the importance of river water.

4.1.1 Motivation of this work

Previous studies have following major limitations which motivated this work:

- In existing work [28,29,32,33] on river water datasets, machine learning techniques (*e.g.,* support vector machine, kNN, linear regression *etc.*) are employed which requires statistical features. Such dependency on the statistical features can be eluded by adopting deep learning techniques (such as long short term memory, convolution neural network, *etc.*) that incorporate automatic feature extraction capability.
- The prior studies [29, 32, 34–37] on river water do not consider any automated annotation mechanism to label the data instances. As in the absence of an automated mechanism, it is infeasible to annotate a large dataset that can improve the performance of the built classifier in learning technique. Thus, there is a need for an automated mechanism for labeling a large unlabeled dataset.
- The previous work [28, 29, 34, 37] on river dataset use various water parameters such as pH, electrical conductivity, dissolved oxygen, turbidity, *etc.*, for assigning labels to the data instances. As it is impractical to simultaneously obtain sensory values from all the sensors for a data instance. Therefore, the annotation mechanism must use a few parameters for labeling the data instances.
- The existing work [28,29,32–37] on river water do not adopt any mechanism that is capable of handling noisy labels in training instances. Hence, a noise handling

technique must be incorporated during training that improves the performance of the build classifier. Therefore, in the real-world scenario for recognizing river water pollution level, where the chance of noisy labels are high, we need a mechanism that can improve the performance of the built classifier.

In this chapter, we address the problem: *How to annotate the massive sensory data automatically and how to identify pollution level in the river water using a deep neural network on a large noisy annotated sensory dataset?* To solve this problem, this work proposes a deep learning based approach that uses the concept of automatic annotation of unlabeled sensory dataset. The proposed approch uses Automatic Label Transfer (ALT) algorithm to annotate the sensory data automatically. Further, the proposed approach builts a classifier using the automatically annotated sensory dataset by learning the mapping between sensory data instances and their corresponding noisy labels. Next, this work proposes a noise handling loss function to enhance the performance of the built classifier. Finally, the built classifier can be used to predict the labels (pollution levels) against the noisy sensory data instances.

4.1.2 Major Contributions

To the best of our knowledge, this is the first work to address the problem of identifying pollution level in the river water using a deep learning technique on the annotated noisy sensory dataset. The major contributions are:

- This work proposes an algorithm for automatic transfer of labels from labeled lab dataset to the unlabeled sensory dataset using only GPS coordinate of sensory data instances. The obtained labeled sensory dataset is used to build a classifier by learning a mapping between sensory data instances and their corresponding labels.
- Next, we propose a noise handling loss function that enhances the capability of the built classifier to work with noisy labels. This work conducts various experiments

to evaluate the performance of the proposed approach on the river dataset [1].

- Further, we propose a deep learning based approach that uses LSTM model to identify the pollution level in the river water using labeled large sensory dataset. The approach uses only single layer of LSTM unit with 25 cells for automatically extracting the features from the dataset.

The rest of chapter is organized as follows. Next section describes the preliminaries. Section 4.3 proposes deep learning approach for identifying pollution level of the river water. Next, Section 4.3.1 proposes automatic annoatation algorithm. Further, in Section 4.3.2, we discuss the mechanism for handling wrong labels to enhance the performance of the built classifier. Later, in Section 4.3.4, we discuss the prediction of class labels for new sensory test data instances. Next, Section 4.4 presents the experimental analysis of proposed approach. Further, Section 4.5 presents the comparison of proposed approach with the existing approaches. Finally, the chapter is concluded in Section 4.6.

4.2 Preliminary

Water is the greatest part of human being for sustainability of life. Contamination of water due to industrial, agricultural or anthropogenic activities is termed as water pollution (WP). The easy and convenient availability of low-power sensors facilitate smart sensing for water pollution. Over the last few years, water pollution monitoring methods have allowed researchers to recognize and mitigate water pollution by providing them with sophisticated methodologies that can measure prime water parameters such as, potential of Hydrogen (pH), Dissolved Oxygen (DO), turbidity, Nitrates (NO2), Biochemical Oxygen Demand (BOD), Fecal Coliforms (FC) etc. automatically and transfer that data online to the Cloud for further analysis. DL techniques can handle big online, continuous, real data and can do the real-time analysis of data to detect the water pollution level and alert authorities regarding danger such as flood or adverse con-

ditions for aquatic life. The analysis of parameters incorporates two methods, namely, lab based and sensor based. We studied lab based method in the previous chapter. The sensor based method gives a real-time assessment of river water but suffers from the problem of scarcity of truely annotated labels. The dataset with labels is used for building a classifier that can predict pollution level against a sensory data instance of river water.

4.2.1 Problem statement and overview of solution

Pollution level identification in the river water helps in protecting the river water for sustaining the survival of human beings on the Earth, as discussed in the introduction. The sensory data instances collected from the river water are substantialally large and it is impractical to annotate manually. Therefore, automatic annotation is beneficial. But, the automatic annotation adds noisy labels in the training dataset. This work, therefore, addresses the problem of autoamtic annotation of huge noisy sensory data. This work In this chapter, we propose a deep learning based classifier to classify the new sensory data instances to identify the river water pollution level. **Overview of the solution:** This work proposes a deep learning based approach that uses annotated large and noisy sensory dataset. The sensory dataset is used to build a classifier $\mathbf{\Pi}$ that predicts a class label (pollution level) for a given sensory data instance.

4.3 Sensor-based river water pollution assessment system using deep neural network

In this section, we propose a deep leaning based approach for assessment of river water pollution level using automated annotation. The objective of approach is to recognize a water pollution level even in the presence of wrong labels.The approach comprise major four phases as follows, Phase 1: Estimation of WQI, Phase 2: Automatic annotation of sensory data, Phase 3: Construction of deep learning classifier, and Phase 4: Prediction

of class labels. We studied Phase 1 in the previous chapter. In Phase 2, we propose automatic annotation mechanism for massive sensory data. In Phase 3, we construct a deep neural network model using LSTM unit consisting of 25 cells. In Phase 4, a specified loss function is also proposed that can handle the noisy labels in the river dataset for predicting the class labels of new sensory data instances.

4.3.1 Automatic annotation of massive sensory data

This work initially constructs a mathematical model to estimate WQI using weighted arithmetic method to assign labels to the lab dataset covered in the previous chapter. Here, in this section, we discuss the phase 2: automatic annotation of significant sensory data using limited labeled lab data. Fig. 4.1 illustrates the phase 2 involved in the sensor-based river water pollution assessment system. The label assigned to the lab

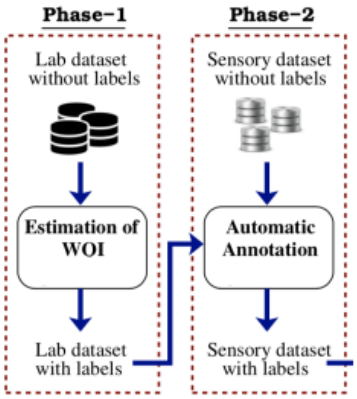


Figure 4.1: Block diagram for automatic annotation of sensory data.

dataset $\mathbf{D}^{L}_{R*p'*T}$ in the previous chapter can be used for assigning the labels to the sensory data $\mathbf{D}^{S}_{R*q*T}$, where q is the set of water parameters in sensory data. In this work, the automatic label assignment of sensory data using lab dataset is termed as Automatic Label Transfer (ALT) mechanism. The mechanism is advantageous by the fact that it only uses GPS coordinate of the sensory data for assigning labels. The primary motivation for using GPS coordinates is to accurately tag the location where

data is collected, as water behavior varies due to factors like industrial waste and city drains entering the river at specific points. Identifying the location and its contributing factors is essential during data collection, necessitating the use of GPS. Additionally, collecting GPS coordinates separately from the data would require specialized tools, increasing the overall estimation cost. Further, the time-stamped and GPS-tagged data is then preprocessed by using different filtering and segmentation techniques and then transferred to the cloud, where it can be permanently stored for analysis.

Let N_1 represents the number of instances in the sensory dataset. d_i and t_i denote date and time stamp of collecting sensory data for i^{th} instance. GPS coordinate (latitude, longitude) for i^{th} instance of sensory dataset is denoted as (g_x^i, g_y^i) and that of lab dataset is (h_x^i, h_y^i). The GPS coordinate $(g_x^i, g_y^i), \forall i \in N_1$ is input to the ALT mechnism. The data instance z_1 from lab dataset is fetch against same date d_i and a time stamp in range of $t_i \pm 60$ minutes. Later, data instance z_2 is obtained by selecting those instances in z_1 that are within 100 meters radius of (g_x^i, g_y^i). The distance between (g_x^i, g_y^i) and (h_x^i, h_y^i) is estimated using the haversine distance [71]. The steps involved in ALT mechanism are discussed in Procedure 4.2.

4.3.2 Noise handling mechanism

Here, we present a noise managing mechanism to handle wrong labels in training data for enhancing the performance of classifier. Constructing a precise and well-annotated dataset poses significant challenges. When starting with an unlabeled dataset, employing various labeling strategies to generate a labeled dataset introduces noise for several reasons. One contributing factor is the constrained time available to human annotators, which can lead to errors. In a different context, complex tasks that require specialized knowledge, such as assessing water pollution levels through mobile-captured or satellite images in water bodies located beyond physical reach, pose significant challenges. The experts facing such tasks may encounter uncertainty, leading to potential mistakes, par-

Procedure 4.2: Automatic Label Transfer mechanism

Input: $\mathbf{D}^{L}_{R*p'*T}$ with labels and $\mathbf{D}^{S}_{R*q*T}$ without labels;
Output: $\mathbf{D}^{S}_{R*q*T}$ with labels;
/*Selecting GPS coordinate of an instance from $\mathbf{D}^{S}_{R*q*T}$ */
for $i \leftarrow 1$ *to* N_1 **do**
  /*d_i and t_i are the date and timestamp of instance i*/
  Select GPS coordinate (g^i_x, g^i_y) with d^s_i and t^s_i. /*Selecting instances from $\mathbf{D}^{L}_{R*p'*T}$*/
  Select z instances from $\mathbf{D}^{L}_{R*p'*T}$, where date = d^s_i.
  for $j \leftarrow 1$ *to* z **do**
    if $|t^s_i - t^l_j| \leq 60$ **then**
      Select z_1 instances, GPS coordinate (h^j_x, h^j_y).
      dist$\leftarrow$*haversine*$(g^i_x, g^i_y, h^j_x, h^j_y)$.
      if *dist* ≤ 100 **then**
        Select z_2 instances with their WQI values.
  /*Label assignment to i^{th} instance of $\mathbf{D}^{S}_{R*q*T}$ */
  $WQI_i \leftarrow$ average value of WQIs for z_2 instances.
  Assign label to instance i of $\mathbf{D}^{S}_{R*q*T}$ as per WQI_i.

Function *haversine*$(g^i_x, g^i_y, h^j_x, h^j_y)$
begin
/*Estimating distance between two GPS coordinates*/

$$\text{dist} = 2r \sin^{-1} \sqrt{\sin^2 \frac{h^j_x - g^i_x}{2} \cos(g^i_x) \cos(h^j_x) \sin^2 \frac{h^j_y - g^i_y}{2}}.$$

return dist.
end

ticularly when water sample images are captured under suboptimal lighting conditions. The time constraints imposed on experts further intensify the pressure to make swift decisions, making the occurrence of noisy labels a natural consequence in these situations. Additionally, assigning labels becomes exceptionally difficult when dealing with very low-quality data. Therefore, noisy labels may emerge organically during the involvement of human annotators. Even the automatic annotation process, facilitated by models, can introduce noisy labels into the dataset. The presence of noise in a data set can increase the model complexity and time of learning which degrades the performance of learning algorithms. This work automatically assigns labels to the sensory dataset, which is used for training the classifier. This automated annotation sometimes leads to the assertion of noisy labels in the dataset [72]. To cope with the noisy labels, we proposed a loss function that can handle noisy labels in the training dataset. The loss function estimates how well a particular algorithm models the provided data. The loss function incorportaes only 2 arguments, which are true label probability and predicted label probability.

In this work, we estimate the predicted probability during the training of the model and employ the predicted probability vector defined as follows: Let $x_{ij} \in X$, $w_{ij} \in W_T$, and $b_j \in \mathbf{b}$ represent an element of the feature matrix, weight matrix, and bias vector for the j^{th} class of the i^{th} training instance, respectively, for $1 \leq i \leq N$ and $1 \leq j \leq k$. Using these values, we estimate an element z_{ij} as follows:

$$z_{ij} = w_{ij}x_{ij} + b_j \tag{4.0}$$

Later, the vector $z_i = \{z_{ij} \mid 1 \leq j \leq k\}$ is passed to a softmax function to compute the predicted class label probability ρ_{ij} as

$$\rho_{ij} = \frac{e^{z_{ij}}}{\sum_{j=1}^{k} e^{z_{ij}}} \tag{4.1}$$

The predicted probability vector for the i^{th} instance is a set of probabilities $\{\rho_{i1}, \rho_{i2}, \ldots, \rho_{ik}\}$ for each class label k. The class label with the highest probability value is the predicted class label. Equation 4.0 estimates the output element z_{ij} for the i^{th} instance and j^{th} class using the feature, weight, and bias. Equation 4.1 applies the softmax function to compute the probability ρ_{ij} for each class label j, ensuring the probabilities sum to 1. The final predicted class label corresponds to the highest probability. Since the predicted probability is simultaneously estimated with the correct label probability; thus, during training it is automatically calculated prior to the estimation of loss.

Let $\mathbf{Y}$ be a prediction matrix of size $N \times k$, where N is the number of training instances, and k is the number of classes in the dataset. The noise handling loss function consists of the two terms:

1. Probability of true label:

   ω_i denotes the probability of the true label for i^{th} instance.

$$\omega_i = \sum_{i=1}^{N} \sum_{j=1}^{k} \mho \log \omega_i \tag{4.2}$$

2. Probability of predicted label:

   ρ_{ij} denotes the predicted probability of $j^{th} \in k$ class in $i^{th} \in N$ instance.

$$\rho_{ij} = (1 - \mho) \log \rho_{ij} \tag{4.3}$$

By aggregating Eq. 4.2 and Eq. 4.3, we define the noise handling loss function $\mathcal{L}_{noise}(\cdot)$ mathematically as follows

$$\mathcal{L}_{noise}(\omega_i, \rho_{ij}) = \sum_{i=1}^{N} \sum_{j=1}^{k} \mho \log \omega_i + (1 - \mho) \log \rho_{ij}, \tag{4.4}$$

where, $\mho$ is a variable whose value is dynamically learned in the training process. The

initial value of $\mho$ is 1, which iteratively adjust to an optimal value that improves the performance of the built classifier. The variable $\mho$ captures the randomness in the dataset and with its dynamic modification that helps in mitigating the noise effect. The loss function $\mathcal{L}_{noise}(\cdot)$ is a combination of two terms separated with summation sign as given in Eq. 4.4. The loss function captures the discrepancy between true and predicted labels. The first term assigns weight to the true labels whereas, the second term assigns weight to the predicted labels. This combination of weight assignment captures the noisy labels by assigning weights to the false prediction alongside with true prediction. The aggregated weights of the noisy labels are less in comparison with that of true labels which are discarded by the LSTM model during training.

4.3.3 Construction of deep learning classifier

Here, in this section, we discuss the phase 3: construction of deep learning classifier. Fig. 4.2 illustrates the phase 3 involved in the proposed approach. Phase 3 incorporates

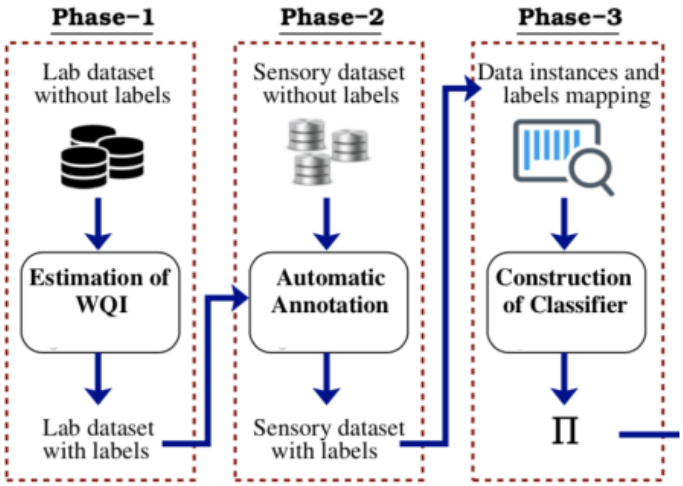


Figure 4.2: Block diagram for construction of deep learning classifier **Π**.

a deep learning based model that extract the features from raw sensory data to learn a mapping between the extracted features and given labels. The mapping constructs a classifier denoted as **Π**, that can predict labels against the testing instances. In

this work, we are using the LSTM model for extracting the features from raw sensory data. We have time-series river water data, making LSTM networks an ideal choice for classification. LSTMs are designed to handle sequential data, such as time series, by capturing temporal dependencies over long periods. They employ memory cells, along with input, output, and forget gates, to regulate the flow of information. This structure allows LSTMs to selectively retain or discard information, ensuring that relevant patterns from earlier in the sequence are remembered while irrelevant ones are forgotten. Importantly, this gating mechanism also helps LSTMs avoid the vanishing gradient problem, a common issue in traditional recurrent neural networks (RNNs), where the network struggles to learn long-term dependencies due to the gradual shrinking of gradients during backpropagation. By overcoming this limitation, LSTMs are particularly effective for tasks like river water data classification, where capturing trends over time is crucial for accurate analysis.

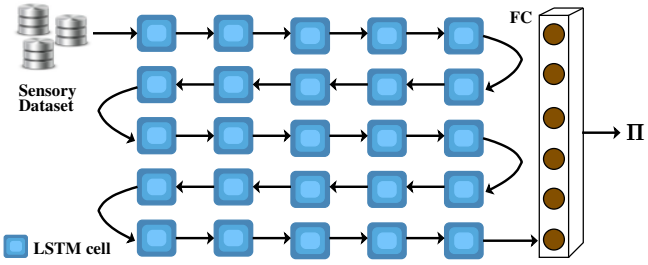


Figure 4.3: Illustration of LSTM model to extract features from labeled sensory dataset to build classifier **Π**.

Fig. 4.3 illustrates the LSTM model used in this paper, where the raw sensory data (D^{S}_{R*q*T}) with their labels is supplied as an input to a sequential combination of LSTM cells. The sequential combination comprises of 25 LSTM cells, the output of 25^{th} LSTM cell is supplied as an input to the Fully Connected (FC) layer. The number of neurons in the fully connected layer is 6, as there are six classes in the sensory dataset (*i.e.,* excellent, very good, good, medium, bad, and very bad.). The output of the FC is a classifier **Π** that is used for predicting the labels against testing instances. The

description of the LSTM cell and FC layer are as follows.

4.3.3.1 LSTM unit

The LSTM model learns the long term dependencies in the sensory input sequence and extracts the temporal features. The core mechanism of the LSTM model is the gated operations performed on a single LSTM cell. LSTM unit comprises of four main components: a cell and three gates (input, forget, and output). The gates determine the information about the content of the memory cell and the cell state provides a way to memorize the relevant features. The main objective of the LSTM unit is to learn weights matrices for all the components. At any time t, the LSTM unit takes three inputs: 1) current input vector, denoted by x_t, 2) output (or hidden) state at time $t-1$, denoted by h_{t-1}, and 3)previous cell state at $t-1$, denoted by c_{t-1}. The single cell operation of LSTM incorporate two activation functions, *i.e.,* $\sigma(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{x}}}$ and $Tanh(\mathbf{x}) = \frac{e^{2\mathbf{x}-1}}{e^{2\mathbf{x}+1}}$ for adding non-linearity in the gated operations. Let $\mathcal{U}_j$ and $\mathcal{W}_j$ denote a weight matrix for a component j corresponding to input vector x_t and previous output state h_{t-1}, respectively, where $j = \{c, i, f, o\}$ and c for cell, i for input gate, f for forget gate, and o for output gate are mathematically represented as:

- **Input gate:** In the input gate, the current input x_t and hidden state h_{t-1} are passed through a sigmoid and a Tanh function.

$$i_t = \sigma(\mathcal{U}_i x_t + \mathcal{W}_i h_{t-1}) \otimes tanh(\mathcal{U}_c x_t + \mathcal{W}_c h_{t-1}), \tag{4.5}$$

  where, symbol $\otimes$ denotes an element-wise multiplication operator.

- **Forget gate** It decides what information of x_t and h_{t-1} should be kept or forgot using sigmoid function ($\sigma(\cdot)$).

$$f_t = \sigma(\mathcal{U}_f x_t + \mathcal{W}_f h_{t-1}). \tag{4.6}$$

- **Cell** It updates the cell state by using previous cell state c_{t-1}, f_t, and u_t as follows:

$$c_t = f_t \otimes c_{t-1} + i_t. \tag{4.7}$$

- **Output gate** It operates on input vector x_t, hidden state h_{t-1}, and updated cell state c_t, to decide the next hidden state.

$$h_t = \sigma(\mathcal{U}_o x_t + \mathcal{W}_o h_{t-1}) \otimes tanh(c_t). \tag{4.8}$$

Fig. 4.4 illustrates the LSTM cell with different gates and cell state used in this paper, where the raw sensory data (D^S_{R*q*T}) with their labels is supplied as an input to a sequential combination of LSTM cells.

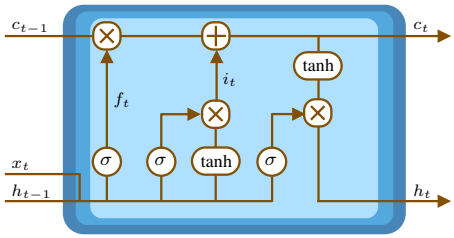


Figure 4.4: Illustration of a LSTM cell with different gates and cell state.

4.3.3.2 Fully connected layer

In a deep neural network, the fully connected layer adds non-linearity in the input data. The input may be from any deep learning model, including convolution neural network or long short-term memory. In the LSTM based model used in this work, the FC layer consists of 6 neurons where each neuron uses a softmax activation function to introduce the non-linearity in the features and generates prediction probabilities against the labels.

4.3.4 Prediction of class labels

This section discusses the Phase 4 of sensor-based river water pollution assessment system in detail. A prediction reflects an individual's anticipation of future events, extending beyond mere weather forecasts. Prediction signifies before, making a prediction of the anticipated labels for new test data. In contrast to typical classification tasks with mutually exclusive class labels, multi-label classification necessitates specialized deep learning algorithms capable of predicting multiple, non-exclusively related classes or labels. Deep learning neural network models inherently support the complexities of multi-label classification problems. This predictive modeling task involves foreseeing zero or more mutually non-exclusive class labels. The configuration of deep neural network models caters to the demands of multi-label classification tasks, with evaluations conducted to make predictions for new data. Labels are the known values for old sensory data instances in the dataset. Prediction is where the deep neural network model tries to predict the correct label for a given new input sensory test data instance, where you do not have a label. The model is fully trained using the training data, and then it is evaluated on test data before being used to perform prediction on new unseen data. Phase 4 incorporates the prediction of the class label. y^{test} for a testing instance $\mathbf{x}^{test}$ of sensory data using the built classifier $\mathbf{\Pi}$. Fig. 4.5 illustrates all the phases involved in the proposed approach. The proposed LSTM based model first extracts features from the given testing instance, using LSTM sequential cells. Later, the classifier $\mathbf{\Pi}$ uses these features to obtain a prediction vector. Finally, the maximum probable class label in the prediction vector is said to be the label of the testing instance. Fig. 4.6 illustrates the process of prediction of a class label for a testing instance $\mathbf{x}^{test}$, using the built classifier $\mathbf{\Pi}$.

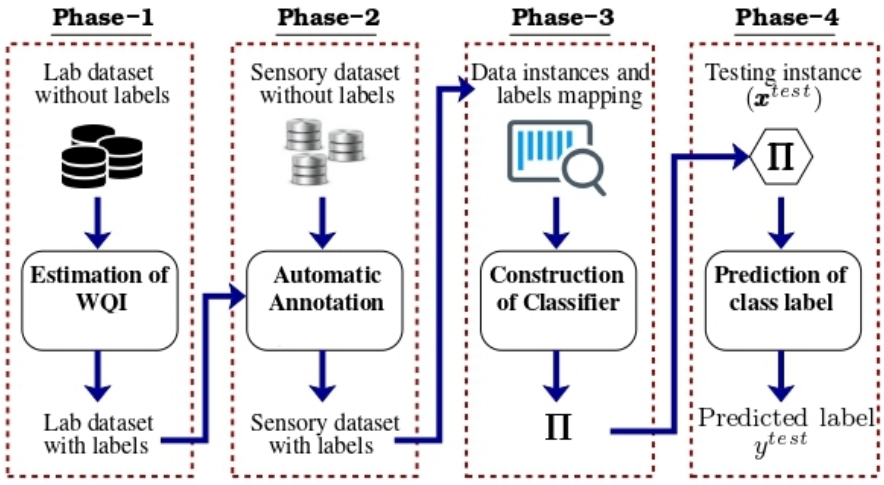


Figure 4.5: Illustration of all phases of sensor-based river water pollution assessment system.

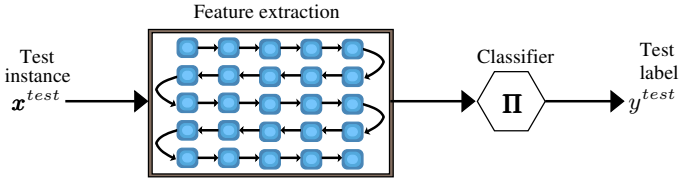


Figure 4.6: Feature extraction of test instance followed by label prediction.

4.4 Experimental evaluation of approach

In this section, we first discuss the implementation details of the proposed approach. Finally, this section presents a comparative analysis of the proposed approach with the existing work.

4.4.1 Implementation details

This section discusses the implementation of the proposed approach. We implemented the proposed approach in Python programming language using PyTorch library. The cell count of LSTM is varied from $10-50$ cells, and we achieve maximal performance at 25 cell in a single layer of LSTM. The number of LSTM layers with a specified number of the neurons is rigorously tested against a range of layers, *i.e.,* $1-10$. The highest accuracy is achieved at 8 LSTM layers by a nominal difference from single-layer LSTM ($\approx 1\%$). Thus, we have taken the single layer of LSTM for our evaluation which preserve

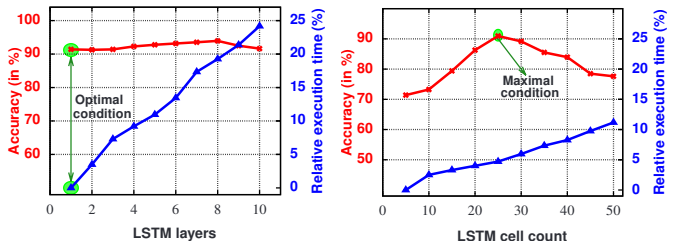
the training time by about 63%. Similar, with the increase in the count of FC layers training and computation times of the system, increases with a little improvement in the accuracy. It enforces to take only FC layer for experiments. The optimizer in the experiment is Adam, and we are using a cross-entropy loss function for capturing the discrepancy between actual and predicted labels. The implementation of two phases are as follows.

1. Algorithm 4.2 is implemented using PyTorch library that maps lab dataset labels with the sensory dataset using GPS coordinates, time, and date as input.
2. We have also implemented the loss function discussed in Section 4.3.2 on PyTorch library in python. The value of the dynamic variable $\mho$ is initialized to 1 and maximum allowable iterations for its estimation is set to 500.
3. Next, the python script with the standard scalar library with 25 LSTM cells and 6 neurons in FC is executed to obtain the accuracy of the proposed approach with test train split of 80 : 20 for 100000 data instances.
4. Finally, the pretrained model in Phase 3 is used for predicting the labels against the testing instances.

4.4.2 Experimental results

In this section, several experiments are carried out to evaluate the performance of the proposed approach and provide answers to the following questions:

- What is the best combination of LSTM layers and cell count for the proposed approach? (Section 4.4.3)
- What is the precision and recall during training and testing on river dataset? (Section 4.4.4)
- What is the effect of using the noise handling loss function? (Section 4.4.5)



(a) LSTM layers (b) LSTM cells.

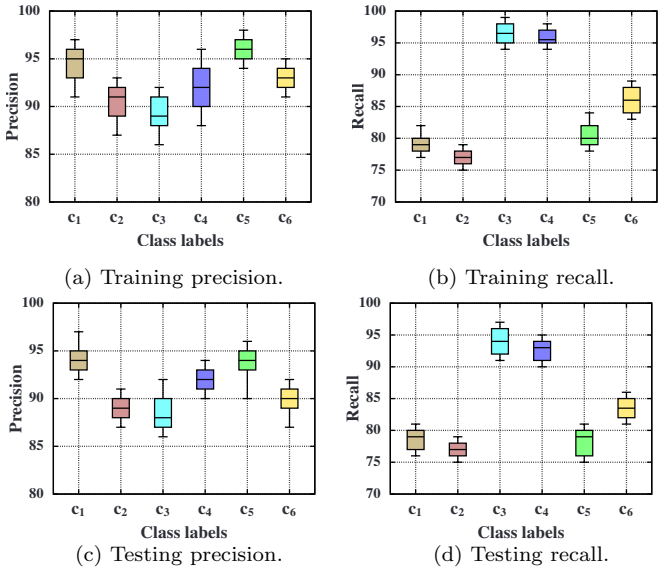
Figure 4.7: Performance measure for different combinations of LSTM layers and LSTM cells.

4.4.3 Number of LSTM layers and cell count

Next, the proposed approach is evaluated by keeping LSTM cell count as constant and LSTM layers as a variable (range 1 to 10). The accuracy of the proposed approach is nearly the same throughout the layer variations and shows a marginal improvement (≈ 1) when layer count is 8. The ratio of execution time (in %) between training on one layer and multiple layers of LSTM, increases with the increase in LSTM layers. Therefore, the optimal performance is achieved at a single LSTM layer with minimum execution time and accuracy of around 90%, as shown in part (a) of Fig. 4.7. Similarly, part (b) of Fig. 4.7 illustrates the accuracy of the proposed approach is maximum when the number of cells in an LSTM layer is 25.

4.4.4 Precision and recall

This work also evaluates the precision and recall of the proposed approach using river dataset and the obtained results at 160 epochs are illustrated in Fig. 4.8. The six classes in the river dataset are denoted by $\mathbf{c_1}$, $\mathbf{c_2}$, $\mathbf{c_3}$, $\mathbf{c_4}$, $\mathbf{c_5}$, and $\mathbf{c_6}$. Part(a) of Fig. 4.8 shows that the precision of $\mathbf{c_5}$ class is highest , whereas the recall of class $\mathbf{c_3}$ reaches to peak, as illustrated in part (b) of Fig. 4.8. The diversification in the value of the precision and recall indicates that the dataset contains imbalance data instances against different classes in the dataset. The imbalance instances are the result of identical behaviour of the river water for a fixed area under observation. Similar, results are obtained for



(a) Training precision. (b) Training recall.

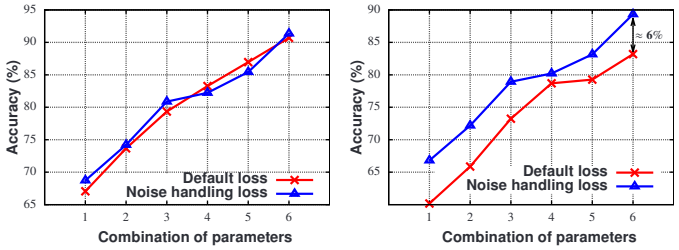
(c) Testing precision. (d) Testing recall.

Figure 4.8: Value of precision and recall on training and testing sensory dataset of river water.

testing data as illustrated in parts (c) and (d) of Fig. 4.8. The precision of class $\mathbf{c_1}$ and $\mathbf{c_5}$ is nearly equal for testing data and is highest among all. The recall for class c_3 in testing shown similar trends to that of training. These results show that the proposed approach captures the class imbalance problem in the dataset and achieves exceptional performance around 90% even in presence of 20% noisy labels. Precision and Recall score are useful performance measures of prediction when the classes are very imbalanced. They offer more insight into the model's skill. High precision and high recall implies an ideal classifier model that has classified ground-truth correctly. Ground-truth is the reality in the real world which you want to model. F1 score which considers simultaneous effect of precision and recall is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy is one of the simplest performance measure and works best when the samples are distributed equally in all classes.

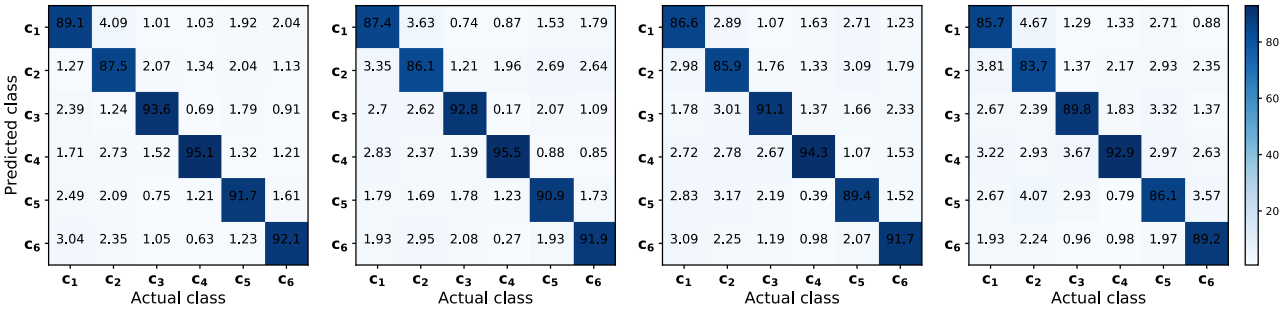
4.4.5 Effect of using noise handling loss

This section discusses the effect of noise handling loss function on the accuracy followed by estimation of precision, recall, and F1-score on different noise concentration.



(a) 0% noise concentration (b) 20% noise concentration.

Figure 4.9: Illustration of noise handling capability of $\mathcal{L}(\cdot)$ with combination of variable parameters.



(a) Godavari (b) Ganges (c) Yamuna (d) Hindon

Figure 4.10: Class-wise accuracy of proposed approach on the data collected from different rivers (*i.e.,* Ganges, Godavari, Yamuna, and Hindon).

4.4.5.1 Parameters versus noise handling loss

The proposed approach performance is also verified in a noisy environment where labels are incorrect. We have inserted random noise in the labels and evaluated the performance of the proposed loss function on river dataset with noisy labels. Fig. 4.9 illustrates the performance of the proposed approach by incorporating the use of both types of loss functions *i.e.,* default (cross-entropy) and noise handling loss. In this work, we proposed a noise managing loss function that reduces the discrepancy between true

labels and predicted labels using dynamic variables. Along with different noise concentrations, we increase the parameters combination (from 1 to 6) for estimating the training labels. Part(a) of Fig. 4.9 demonstrates the performance of default loss function is nearly equal when the dataset is noise-free and increases with the increase in the parameters count. In the presence of 20% noisy labels, the accuracy achieved by default loss function drops to a greater extent. However, the noise handling loss function preserves the accuracy and manage around 91% even in the presence of 20% noisy label in the dataset for 6 parameters combination. The noise handling loss function achieves a performance gain of nearly 6% to that of default loss function in presence of 20% noisy labels. Thus, the noise handling loss function enhances the recognition capability of the LSTM model, without having any prior information about the concentration of noisy labels.

The observed performance gain of 6% despite the presence of 20% noisy labels can be intuitively explained by considering the effectiveness of our noise handling loss function. This function is designed to minimize the impact of noisy labels during training by focusing more on the consistent patterns in the data rather than being swayed by the incorrect labels. By effectively down-weighting the influence of the noisy instances, the model can still learn from the underlying structure of the data, allowing it to generalize better to clean instances. Additionally, the robustness of the deep neural network architecture we employed enables it to discern meaningful features despite the noise, which contributes to maintaining high accuracy. Essentially, the combination of our noise handling approach and the model's inherent capacity to learn from less-than-perfect data led to the performance gain we observed. This highlights the importance of robust methodologies in real-world applications where data quality can be variable.

4.4.5.2 Varying noise concentration

Table 4.1 illustrates the precision, recall, F1-score of the proposed approach for the following condition:

- **Condition 1:** Using default loss function when the noise concentration in the label is 0% *i.e.,* dataset is noise-free and 20%.
- **Condition 2:** Using noise handling loss function when the noise concentration is 0% and 20%.

For condition 1 and 2, at 0% noise concentration the value of precision, recall, and F1-score are more or less equal. At 20% noise concentration, the value of precision, recall, and F1-score for noise handling loss function outperforms to that of the default loss function. This result identifies that the loss function can suppress the noisy labels in the dataset and achieves a higher value for all accuracy measures.

Table 4.1: Precision (**P**), Recall (**R**), and F1-score (**F1**) using default and noise handling functions with different noise concentration.

Noisy labels	Class	Default loss			Noise handling Loss		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
0%	c_1	93.71	79.23	85.86	92.70	80.19	86.23
	c_2	89.23	76.79	82.54	90.19	77.23	83.21
	c_3	86.71	94.23	90.31	87.90	93.18	90.46
	c_4	92.07	93.36	92.71	92.30	92.97	92.63
	c_5	93.90	79.90	86.34	93.18	80.45	86.35
	c_6	90.17	84.10	87.03	91.74	85.13	88.31
20%	c_1	85.73	70.19	77.18	**89.37**	79.73	84.28
	c_2	84.17	68.23	75.37	88.48	75.37	81.40
	c_3	80.93	80.07	80.50	87.61	86.17	86.89
	c_4	83.17	79.23	81.15	85.90	84.37	85.13
	c_5	84.25	78.67	81.36	88.35	**87.07**	**87.71**
	c_6	80.70	75.45	77.98	85.27	84.93	85.10

4.4.6 Class-wise accuracy for different rivers

Finally, this work presents a comparative analysis of class-wise accuracy for the dataset collected from different rivers, including Ganges, Godavari, Yamuna, and Hindon. Fig. 4.10 illustrates that the accuracy of river Godavari for all classes is higher among all rivers whereas, the class-wise accuracy of the Hindon river is lowest. The higher accuracy of the Godavari river is the matter of the fact that dataset of the Godavari

has highest instances for training for all 6 classes considered in this work. The dataset for the river Ganges have fewer instances to that of Godavari. So, its class-wise accuracy slightly lags with the Godavari as illustrated in parts (a) and (b) of the Fig. 4.10. The Hindon river suffers from high pollution throughout the entire catchment area for sensor data collection. Therefore, the dataset has fewer instances of excellent and very good classes and upon equalizing the class instances for training, the dataset shrink, which leads to accuracy compromise. Part (d) of Fig. 4.10 illustrates the decrement in the accuracy of the Hindon river. Similarily, part (c) of Fig. 4.10 shows the class-wise accuracy of Yamuna river.

4.5 Comparison with existing approaches

This section compares the proposed approach with the existing approaches including Early Classification Approach (ECA) [29], Mobile Acoustic Wave (MAW) [37], and Modeling pollutant distribution (MPD) [33]. For fair comparison, we set following parameters:

- In ECA, the desired level of accuracy (α) is set to 1.
- MAW uses a specified acoustic sensor whose procurement is infeasible, so we use sensory values from Hanna multiparameter HI-9829 and processed them on RaspberryPi available at the boat itself.
- In MPD, we reduces the sensor count from 25 to 10 and increase the iteration to 100.

Table 4.2 illustrates the accuracy and F1-score of existing and proposed approaches on the datasets of different rivers. It also illustrates that the proposed approach (default loss or noise handling loss) outperforms over all the existing approaches. The accuracy and F1-score are highest for Godavari river due to colossal data instances, whereas for Hindon river it is lowest. The use of deep learning model along with noise handling technique strengthens the model mapping capability, which reflects in performance

improvement as shown in Table 4.2.

Table 4.2: Performance comparison of the proposed approach with the existing work.

	Approach	Godavari	Ganges	Yamuna	Hindon
Accuracy (%)	MAD	82.31	80.39	79.47	75.29
	MAW	84.19	82.15	80.19	78.31
	ECA	87.13	86.19	83.17	80.13
	Proposed (Default loss)	**91.07**	**90.67**	**88.76**	**85.73**
	Proposed (Noise handling loss)	**92.45**	**91.09**	**89.71**	**87.02**
F1-score (%)	MAD	80.17	79.25	77.95	73.67
	MAW	82.97	81.19	78.37	77.13
	ECA	85.76	84.93	81.90	78.25
	Proposed (Default loss)	**90.10**	**89.49**	**86.76**	**84.29**
	Proposed (Noise handling loss)	**91.57**	**90.67**	**88.15**	**85.73**

Fig. 4.11 illustrates the effect of noise concentration on the accuracy of the different approaches for river dataset. The proposed approach with noise handling loss achieves the best performance on the dataset with noise concentration 20%. The maximum performance degradation is observed in the existing approach when noise concentration is 20% ($\approx$ 20% for MAD approach and 14% for ECA).

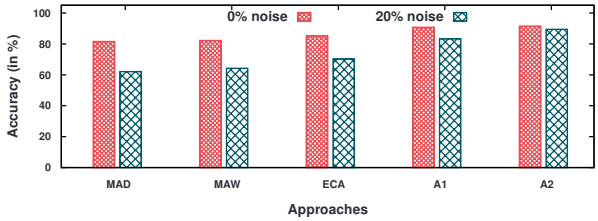


Figure 4.11: Accuracy of existing work on 0% and 20% noise concentration. (**A1**: default loss and **A2**: noise handling loss).

4.6 Conclusion

Here, we proposed automatic annotation of sensory data. Further, we proposed a deep learning based approach that uses LSTM model to identify pollution level in the river

water. This work carried several experiments to evaluate performance of the proposed approach. The experiments are performed on default loss function (cross-entropy loss) and proposed noise handling loss function.

The evaluation results have shown that the proposed approach has achieved remarkable performance on river water dataset. The proposed approach is also capable of achieving the accuracy of around 90% even with high noise concentration in the labels (*i.e.,* 20%). We also carried out several experiments to validate the effectiveness of the proposed approach using three existing approaches.