# Dynamic Models for Entity Trajectory Prediction Using Deep Learning

Dhanya Raghu[1*], Apoorva K H[2], Anjana Anil Kumar[3], S Natarajan[4]

[1] PES Institute of Technology; Vijayanagar, Bangalore, Karnataka, India.
[2] PES Institute of Technology; Rajarajeshwarinagar, Bangalore, Karnataka, India.
[3] PES Institute of Technology; Rajajinagar, Bangalore, Karnataka, India.
[4] PES University; Banashankari, Bangalore, Karnataka, India.

* Corresponding author. Tel.: +91-9731054855; email: dhanyaraghu52@gmail.com
Manuscript submitted May 15, 2018; accepted July 8, 2018.
doi: 10.17706/jcp.13.9.1127-1135

**Abstract:** Human motion tracking and forecasting has posed multiple challenges due to its high dimensional interactions with the physical world. The uncertain nature of human robot interaction environment stresses on the need to develop prognostic methods for determining responses to ambivalent scenarios in the environment. However, recent growth in depth camera technologies has enhanced performance of such sequence prediction tasks. A growing requirement for co-robotic applications that place robots as partners with humans play a role in cooperative and tightly interactive tasks. Smart machines need to enhance social intelligence and the capacity to make stable, safe and consistent decisions to effectively operate in unconstrained crowded scenes. Autonomous robots should be modelled to circumvent obstacles and obey common sense rules by understanding the personal space of neighbors. Intelligent learning techniques are being exploited to build socially smart, safe and efficient systems to predict human motion in various situations. We propose three deep learning approaches namely- pooling technique, hierarchical RNN and attention mechanism to learn general human social movements, collision avoidance and predict their future paths. The results obtained from the different approaches are compared and analyzed.

**Key words:** Attention network, autonomous agents, collision avoidance, hierarchical RNN, pooling, recurrent neural network, trajectory prediction.

## 1. Introduction

Pedestrian navigation behaviour modelling is seen as a requirement in multiple fields such as behavioural science, computer robot games, blind people navigation, and navigation of autonomous agents in malls for movie ticket distribution or for garbage collection. Reliable models for safe and efficient navigation are needed to understand the intent and behavior of neighbours. In navigating, right-of-way and private space are considered as important features in determining future movements [1] in constrained situations. Machine learning techniques are a progressing research topic to develop most safe and effective models for predicting human motion. The latter uses fully observable environments with features that allow for predictions. Collision avoidance and social etiquette play a key role in crowded spaces.

Learning nonlinear dynamical models for high-dimensional data is a key aspect [2]. We investigate deep learning techniques to forecast trajectories of entities belonging to various classes (pedestrians, bicyclists, skateboarders, carts). Recurrent neural networks along with pooled trajectories from neighboring entities are used. Multiple instance learning approach is followed in correspondence to a single RNN instance for each entity.

The chosen models are based on recurrent neural network features considered due to relevance of their architecture in solving the problem at hand. The RNNs inherent ability to maintain temporal context [3] appropriately fits our need of processing several frames sequentially. We propose a novel hierarchical RNN model that considers hierarchy in the weight given to neighbours. A novel attention memory network is modelled to give importance to the most relevant and highly affecting components of the input, which would be the speed of neighbours that affect the entity at hand.

## 2. Related Work

Many deep learning models like RNNs and CNNs have been applied to such sequence-to-sequence modelling problems. [4] addresses the problem of forecasting pedestrians' destinations. They propose to learn the various social affinities which bind people in a crowded scene through a feature called as Social Affinity Map (SAM). All our models include many other output classes other than humans such as bikers, cars, carts, skaters etc., thus making it a regression model with multi class occupants. [5] presents an algorithm for trajectory prediction that incorporates inter-class interactions into trajectory prediction models as opposed to just intra-class interactions. Our predictive models are able to predict non-linear behaviours to cope with their limitation. [6] aims at predicting the motion dynamics in crowded scenes. They propose a model called 'Social' LSTM which can jointly predict the paths of all the people in a scene by taking into account the common-sense rules and social conventions that humans typically utilize as they navigate in shared environments. Our attention mechanism approach is different from this model as it takes the speed of the neighbouring entities into consideration, thus handling collision avoidance and social force requirements. [7] deals with an approach to learn the navigation behaviour of collaborating agents from illustrations. The decision process that ultimately paves way to the observed continuous trajectories of the agents usually consists of discrete decisions, which divides the composite trajectories' space into homotopic classes. Our approaches use non-linear deep learning models with a linear cost function which facilitates faster computation of loss. [8] focuses on anticipating future appearance, given the current frame of a video. They show that CNNs can learn an intrinsic representation of typical appearance changes over time and successfully generate realistic predictions in one step at a deliberate time difference in the near future. Our approaches use RNNs in contrast to their CNN methodology.

## 3. Methodology

In all our models, each occupant trajectory is considered as a sequence. Each entity is modelled with its own neural network instance; however, entities of the same class are parameterized by the same set of weight model parameters collectively. Weight parameters form learned navigation rules, thus, rules are shared by members of a class but are different from class to class. In addition, we define two baseline models for comparison.

### 3.1. Baseline

This approach deals with predicting the future paths of entities only by considering the past trajectory of that entity. Amongst the deep learning methodologies, RNNs have been very successful for sequence prediction and classification tasks. We observed positions from T0 to Tobserved and predict the next time step, Tobserved+1.

The Vanilla RNN baseline model is unrolled for 25-time steps to learn the traversing behaviour of that entity, as depicted in Fig. 1(a). The future path is then predicted using the learned context captured by the hidden states of the RNN. The Naive GRU baseline model uses a similar approach of considering the previous context of entities for future prediction tasks.

## 3.2. Multitarget Trajectory Prediction

Behaviour of people in crowded scenes [9] is dependent on the motion of other people in their vicinity. In this approach, we consider the positions of neighbouring entities of the selected entity along with its past trajectory for learning of the future path.

### 3.2.1. Pooling RNN model

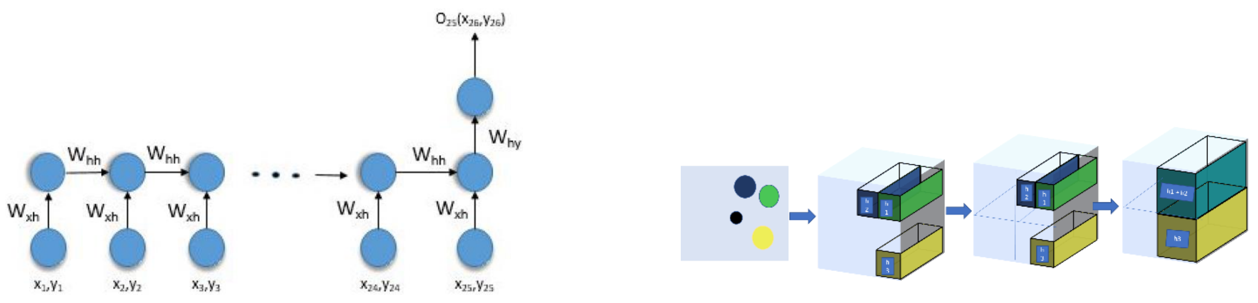
In this approach, each entity is considered to be under the influence of all neighbouring entities within a predefined vicinity. Pre-training of the models' weights is done to capture the traversal rules of various output classes and they are used by the entities in calculating the context at the hidden layer. Further, pooling [6] is done to add the hidden states of the neighbouring entities which represent their navigational behaviour to the current entity in consideration to inculcate the concept of social sensitivity. Pooling [10] is done based on the proximity of the neighbouring entities to the entity under consideration. If an entity is within 100X100 pixels square centered around the entity, it is considered to be a neighbour. The neighbourhood is divided into 5x5 pooling windows, giving 400 in total. The hidden states of all the neighbouring entities within the influential area are added to the entity's hidden state, as depicted in Fig. 1(b). This is performed to handle the interdependencies that occur due to the paths of these neighbours, collision avoidance and also to follow social etiquettes. This method is implemented using RNN.

$$H_t^e = H_t(i^m, j^m) + h^m_t \tag{1}$$

where, for each neighbour '$m$' at time '$t$', we locate the appropriate grid coordinates $i, j$ in the pooled matrix ($H_t(i^m, j^m)$) and summate it with the hidden state of the neighbour '$m$' at time '$t$' ($h^m_t$) around the selected entity's location on the p x q grid.

$$h^e_{t,\ final} = W^s . H_t^e \tag{2}$$

where, $W^s$ is a high density tensor which is learnt as an additional parameter during the back-propagation of weights, which is used to calculate the effective hidden state of the entity '$e$' ($h^e_{t, final}$) and thus aids in fine tuning the navigational skills of our trained model.



(a): Internal structure of RNN (b): Pooling of hidden states in Multi-target Approach

Fig. 1. Baseline and pooling models.

### 3.2.2. Hierarchical RNN model

While we regard the influence of the neighbouring entities, we must take into account the prominence given to each neighbour. Intuitively, neighbours which are nearer to the considered entity have more influence on the entity as compared to a far off neighbour. To solve this, we used a hierarchical approach. The hierarchical approach uses a weighted method to summate the hidden states. The closer entities have more weights, thus, having more hand at hidden state variation. A virtual hierarchy of weights is given emerging from the entity under consideration. Hierarchy is applied on the pooled tensor to obtain the final hidden state, as seen in Fig. 2. The 100x100 pooled tensor has 10 levels of hierarchy based on proximity. The weights provided in this approach for the neighbours are constant for a specific radius around the selected entity. These weights constantly reduce as the neighbours are further away from our entity.

$$H^{e} = H(i^{m}, j^{m}) + w^{m}.h^{m} \quad (3)$$

where, for each neighbour '*m*' at time '*t*', we locate the appropriate grid coordinates *i, j* in the pooled matrix ($H(i^{m}, j^{m})$) and summate it with the product of the constant weight assigned($w^{m}$) and the hidden state of the neighbour '*m*' at *i, j* time '*t*' ($h^{m}_{t}$) around the selected entity's location on the p x q grid.

### 3.2.3. Attention mechanism model

In this approach, the speed with which the neighbouring entities approach the selected entity is considered to handle collision avoidance and to mimic the social sense of humans better. The weightage provided by the hierarchical approach ranks the initial importance of all the neighbours. In attention mechanism, the neighbours are rated in increasing order of their speeds and provide correspondingly increasing weights to prioritise them accordingly, to model the chance of colliding with the selected entity as shown in Fig. 3. This approach also takes the social sense and etiquettes into consideration. We induce this ranked feature into our RNN, which further assists in improving prediction of the trajectories.

$$H_{t}^{e} = \Delta d / \Delta t * h^{m}_{t} \quad (4)$$

where m is each neighbour within the boundaries of the restricted space radius around the selected entity's location on the p x q grid. The total rate of change of distance ($\Delta d/\Delta t$) for each neighbouring occupant coupled with hidden state ($h^{m}_{t}$) plays a key role in aggregation of the pooled hidden state of the entity ($H_{t}^{e}$).

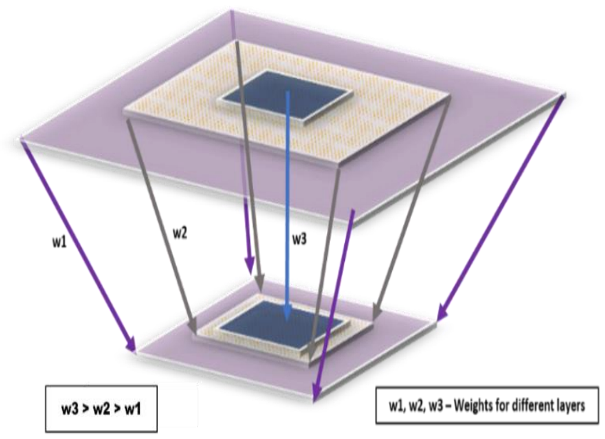


Fig. 2. Hierarchical model.

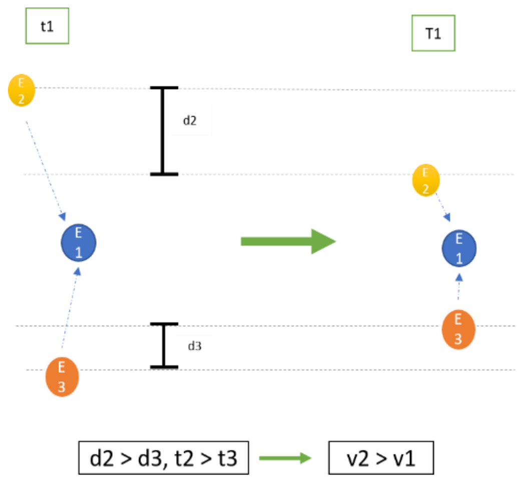


Fig. 3. Attention model.

### 3.2.4. Algorithm

```
Calculating hidden state for entity, e:
    N = neighbours of e
    for each n in N :
    d = Euclidian distance between n and e
        If d < ( 100 x 100 pooling window ) :
            assign weight, w to n based on its speed towards e
            $H_t^e = H_t(i^n, j^n) + h^n_t$
    Reduce dimension of $H_t^e$ to 20x20 from 100x100
    Final pooled hidden state of e, $h^e_{t,final} = W^s . H_t^e$
```

## 4. Experiments and Results

### 4.1. Dataset

The Stanford Drone Dataset is used for our research, which is openly provided by Stanford's Computer Vision and Graphics Lab. This new large-scale dataset contains videos of various types of targets (not just pedestrians, but also bikes, skateboarders, cars, buses, golf carts) that navigate in a real world outdoor environment such as a university campus. It comprises of more than 100 different top-view scenes for a total of 20,000 targets engaged in various types of interactions. The dataset comprises more than 19K targets consisting of 6.4K bicyclists, 1.3k cars, 0.2K golf carts, 11.2K pedestrians and 0.3K skateboarders.

### 4.2. Experimental Setup

The RNN models has a hidden state dimension 32 and 64, input dimension and output dimension of 2. The activation function used at each level is tanh. The entity neighbourhood was set to 100 X 100 and 5 X 5 pooling windows. Parameters were updated using Stochastic Gradient Descent as our back-prop algorithm. The initial learning rate was set to 0.3 and annealed upon increased loss and the model trained for 30 epochs. To train the model, we allowed the RNN to observe 25 frames of an occupant's trajectory and predict the next video frame (time step) as shown in Fig. 4.

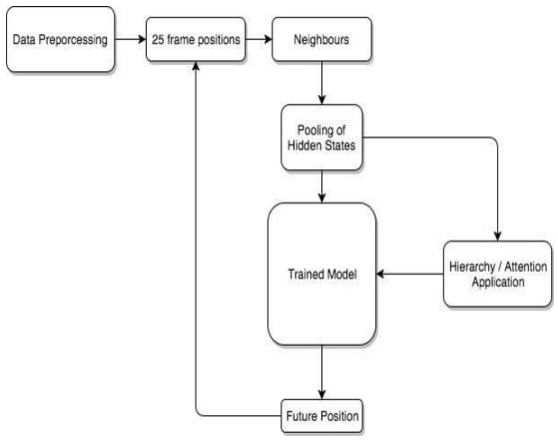


Fig. 4. System design.

The models consider the data with an entity wise and class wise perspective. The forecasting of the path of any given entity is done with respect to each frame. The hidden state of an entity at any particular time gives us the temporal state of the entity. The temporal state represents the characteristic of the entity at present time, for example, it could be current direction, speed, etc. The weight matrices considered for every model are considered class wise. The weight matrices are believed to be holding and constantly learning the characteristics of a class as a whole, for example, collective navigation rules of entities of a class. The accuracy is calculated in terms of path displacement from the original position. Predicted position is considered acceptable if the euclidian displacement falls within the threshold and strengthens the accuracy.

## 5. Results and Analysis

The models are evaluated based on accuracies which are calculated using the path displacement metric. For calculating accuracy of the models, a path displacement is accepted if it falls within threshold of 36 pixels. We also evaluate each of our models based on the error rate metric. The error rate is effectively the path displacement between the predicted trajectory and the actual target trajectory of the entity.

### 5.1. Baseline Approach Using RNN and GRU:

Our single track RNN model achieved an accuracy of 85% with learning rate as 0.3, 40 epochs and 64 as the hidden state dimension. This model does not handle collision avoidance, thus not incorporating social sensitivity skills. The GRU mechanism obtained the best accuracy of 67%. Since the input to the neural network is not too long, the gates of the GRU are plausibly unnecessary for our problem statement The GRU model was thus discontinued for further prediction of paths due to its low accuracy and incompatibility.

### 5.2. Multitarget Tracking Approach

The results are in terms of error rate and accuracies of each of the three models with respect to different perspectives. The Fig. 5(a) shows the error rate across time, for all the three models after model training. As seen in the figure, the losses for all the models decrease consistently over time as the model learns, conforming to the theoretical expectations. The error rate is equivalent to the loss of the model at each time step. Attention method has considerably higher loss when compared to the other two models because of the dynamically changing preference given to neighbouring occupants. The fluctuation in the values of the loss increases from pooling to hierarchical to attention method because of the seemingly unpredictable velocities of the surrounding entities.

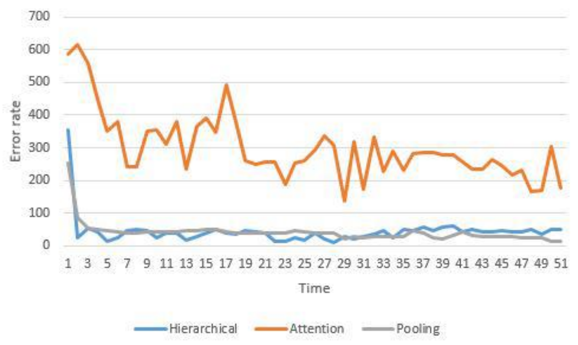
As seen in Fig. 5(b), the accuracy of all the three models increases across time, as the models gradually learning the features of the trajectories of various entities in each frame. According to the figure, the best accuracy is obtained for attention mechanism, followed by hierarchical approach and lastly, the pooling method. While the pooling method takes into account the influence of neighbours, the hierarchical also considers the weight of influence of each neighbor and attention mechanism considers the neighbouring entities' velocities and direction of approach for measuring influence of neighbours.

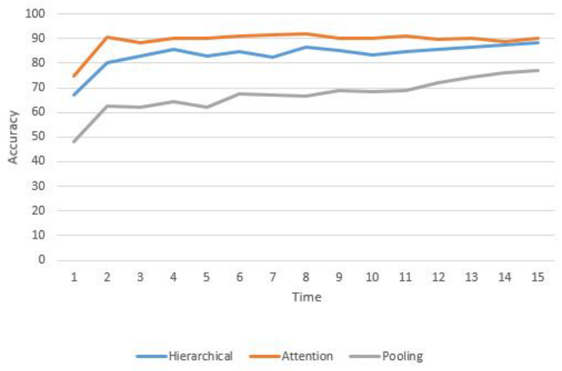
Table 1 and Fig. 5(c) describe the output of class wise accuracies for each approach mentioned in the paper. The skater and cart class gave a good accuracy considering their motion to be linear most of the time. The pedestrian and biker class resulted in slightly lesser accuracy due to high occurrences and added nonlinearity. The Car class resulted in lesser accuracies due to its highly non-linear behaviour as compared to other classes.

Table 2 and Table 3 represent accuracies and error rates, respectively, based on various scenes in the dataset. Each scene has its own characteristic road paths with curves and straight roads. Accuracies for the 'Coupa' and 'Gates' scenes are high because, the entities here mostly have linear paths. Our models can successfully predict non-linear paths of moving entities, as seen with the accuracies of the 'Death circle' scene which has a circular roundabout in the center. Pre- training of weights, however improved the accuracies by a small factor.

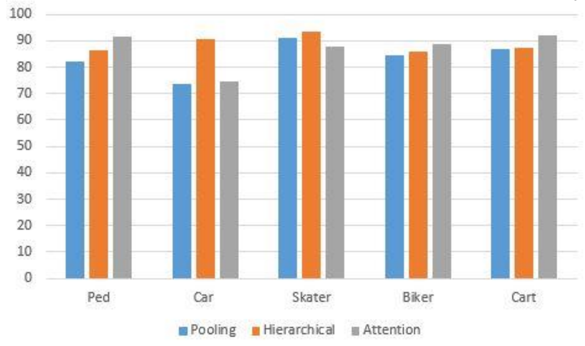
Table 1. Class-Wise Comparison of Accuracies for All Three Approaches

| | Pedestrian | Car | Skater | Biker | Cart | Overall |
|---|---|---|---|---|---|---|
| Pooling | 82.09 | 73.62 | 90.9 | 84.56 | 86.77 | 86.71 |
| Hierarchical | 86.36 | 90.5 | 93.6 | 85.82 | 87.36 | 87.5 |
| Attention | 91.4 | 74.55 | 87.6 | 88.52 | 92 | 91.75 |





(a) Error rate (b) Accuracy



(c) Class-wise accuracy

Fig. 5. The error rate and accuracy v/s time is plotted for the three models.

Table 2. Class-Wise Comparison of Accuracies for the Different Sub-Datasets in the Stanford Dataset

| | Book-store | Death circle | Coupa | Gates |
|---|---|---|---|---|
| Pooling | 82.52 | 83.6 | 88 | 82.5 |
| Hierarchical | 86.7 | 89.16 | 90.6 | 87.78 |
| Attention | 91.3 | 90.37 | 89.2 | 88.92 |

Table 3. Class-Wise Comparison of Error-Rates for the Different Sub-Datasets in the Stanford Dataset

| | Book-store | Death circle | Coupa | Gates |
|---|---|---|---|---|
| Pooling | 785.69 | 2069.31 | 241.68 | 492.34 |
| Hierarchical | 775 | 2056.5 | 227.49 | 514.5 |
| Attention | 9797.78 | 2393.5 | 245.53 | 522.25 |

## 6. Conclusion

We have presented RNN-based modes that can reason across multiple entities to predict multi-class trajectories in various scenes. On comparison of all our proposed methods, attention mechanism with RNN approach showed higher performance on the Stanford publicly available dataset. In addition, we effectively show that our approaches successfully predict various non-linear behaviours arising from social interactions. However, due to many social factors, our current methodologies of pooling cannot comprehensively predict perfect trajectories. Future work will be to enhance our model to incorporate more approaches that share the same space as our current models and implement reinforcement learning for further qualitative comprehension of its results. It will also include a more extensive dataset to increase model training and obtain better results.

## 7. References

[1] Gianluca A., Michel B., & Mats W. (2006). Discrete choice models of pedestrian walking behavior. *Transportation Research Part B: Methodological, 40(8),* 667-687.

[2] Wang, J. M., David, J. F., & Aaron H. (2008). Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(2),* 283–298.

[3] Wang, Y., Wang, S., Tang, J., O'Hare, N., Chang, Y., & Li, B. (2016). *Hierarchical Attention Network for Action Recognition in Videos.*

[4] Alahi, A., Ramanathan, V., & Fei-Fei, L. (2014). Socially-aware large-scale crowd forecasting. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2203-2210).

[5] Robicquet, A., Alahi, A., Sadeghian, A., Anenberg, B., Doherty, J., Wu, E., & Savarese, S. (2016). *Forecasting Social Navigation in Crowded Complex Scenes.*

[6] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., & Savarese, S. (2016). Social lstm: Human trajectory prediction in crowded spaces. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 961-971).

[7] Monfort, M., Liu, A., & Ziebart, B. D. (2015). Intent prediction and trajectory forecasting via predictive inverse linear-quadratic regulation. *AAAI*, 3672-3678.

[8] Zhen Z., Bing S., Gang W., Xiao L., Xingxing W., Bing W., & Yushi C. (2016). Learning contextual dependence with convolutional hierarchical recurrent neural networks. *IEEE Transactions on Image Processing, 25(7)*, 2983-2996.

[9] Yi, S., Li, H., & Wang, X. (2016). Pedestrian behavior understanding and prediction with deep neural networks. *Proceedings of European Conference on Computer Vision* (pp. 263-279).

[10] Raju, K., & Chang, M. (2016). *Predicting Human Trajectories in Multi-Class Settings*.

**Dhanya Raghu was** born on 29th December 1995 in Bangalore, India. She received the B.E, in computer science from PES Institute of Technology, Bangalore, 2017.

**Apoorva K H** was born on 19th June 1995 from Bangalore, India. She received the B.E in computer science from PES Institute of Technology, Bangalore, 2017.

**Anjana Anil Kumar** was born on 24th May 1995 from Bangalore, India. She received the B.E in computer science from PES Institute of Technology, Bangalore, 2017.

**S Natarajan** received the doctor of philosophy (PhD), JNTU. He is a professor at PES Institute of Technology, Bangalore.