Evaluating the Impact of ANN Architecture for Driver Activity Anticipation in Semi-autonomous Vehicles

Shilpa Gite, Ketan Kotecha

***Abstract*—Artificial neural networks (ANNs) consist of multiple intermediate layers known as hidden layers stacked together through which the input is passed to obtain the desired output. The hidden layers are crucial for feature extraction which in turn impacts the performance of the entire model. However, they do not have a fixed number and the ideal value is traditionally derived iteratively by assessing the performance of the architecture. It is thus desired to analyze and derive the standard amount and deciding criteria for these hidden layers, as well as the impact of altering their configuration on the performance of the system. In this paper, we present our findings for this problem when working on Spatio-temporal data to predict the activity of drivers in semi-autonomous vehicles. The performance of the different architectures is assessed on the standard brain4cars dataset. Exploratory research is conducted to understand the impact of variation in the hidden layers in a deep neural network architecture. A detailed modeling procedure is followed out to present an unbiased analysis of the impact which the architectural changes hold on the performance of the system. The performance is assessed by considering the accuracy and other performance metrics of the system on the testing data. We also evaluate the time required by the system for delivering the inference. Both these factors are seen to be significantly affected by the architecture configuration.**

***Index Terms*—Artificial neural networks, deep learning, LSTMs, RNNs, brain4cars**

I. Introduction

ARTIFICIAL neural networks or ANNs consist of multiple layers of nodes connected together. The layers in the architecture could be categorized as input, output, and hidden layers. There is always only a single input and output layer but the hidden layers could be varying in number. The input neurons or nodes correspond to the number of features present in the training dataset. The output layer corresponds to the number of classes associated with every input. However, when it comes to the hidden layers, the researcher possesses the liberty in deciding the configuration of the hidden layers and the nodes present in each of them [1].

While the input layer accepts the case data in a model-friendly format and the output layer produces an output prediction in a determined form, the hidden layers are responsible for extracting and propagating the feature values from the input layer to the output layer [2]. Hidden layers are private to the neural network and are crucial in facilitating the learning and extracting the features, especially for non-linear data. Currently, the number of hidden layers and nodes is determined through an iterative approach of configuration and testing process by computing the discrepancy between the obtained performance and the desired performance. It is desired that insights are shared upon the best-suited configuration of nodes and hidden layers to get the ideal deep network architecture for a given task [3].

In the past, there have been attempts to review and analyze the impact of altering the configuration of hidden layers and neurons within the ANN architecture for performing a diverse set of tasks [4] [5] [6]. These efforts have stressed making changes in the activation functions, and the number of nodes and layers on a small scale. However, there has been paltry work in this area when working on an image or video-based data of driver activity anticipation. Further, the analysis has been performed up to a restricted small scale of layers and only with one or two architectures [7] [8]. There is thus ample scope for more comprehensive research in this domain.

The performance of an ANN architecture could be assessed in multiple ways. Firstly, the standard performance metrics for the task in hand are a good indicator of the model's learning. These performance metrics evaluate how close the predicted outcome is to the actual outcome or ground truth for the given sample [9]. Overfitting is a scenario where the model performs excellently on the performance metric for training data but fails to get similar levels of results for the testing data [10]. The model architecture should be such that it does not overfit while training. Further, another factor to be considered while assessing the system performance is the inference time required by the architecture. While it does not affect the prediction value, it has a significant impact on the cost incurred for running the system and it also directly impacts the entities which are in the environment of the model. Generally, for a particular architecture, the training time is in direct proportion to the depth of layers and trainable weights present along with other factors. An architecture that gives good results on the given performance metric while also having a lesser inference time would be considered optimal.

In this paper, we present our evaluation of the impact which the architecture configuration of ANNs has on their performance for a given task. Specifically, we work on the task of driver activity anticipation [11] in semi-autonomous vehicles which involves Spatio-temporal data and conducts

Manuscript received January 24, 2021; revised March 17, 2021.

Dr. Shilpa Gite is an Associate Professor at Symbiosis Institute of Technology and Associate Research Faculty at Symbiosis Centre for Applied Artificial Intelligence, Symbiosis International (Deemed University), Pune, Maharashtra, India. e-mail: (shilpa.gite@sitpune.edu.in).Corresponding author

Dr. Ketan Kotecha is the Head of Symbiosis Centre for Applied Artificial Intelligence and Dean Director, Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, Maharashtra, India. e-mail: (head@scaai.siu.edu.in).

the study for multiple ANN architectures. The performance of the different architectures is calculated on the standard brain4cars [12] dataset. We follow an in-depth methodology for analyzing the system and elucidate our findings in the later sections. Specifically, the novelties and contribution of our work are as mentioned below:

1) Unlike previous attempts, an assessment of the impact of variation in hidden layers configuration is done for not just one, but five different ANN architectures.
2) We have carried out the analysis on Spatio-temporal data which requires sharper feature extraction methods while also maintaining lesser training cost.
3) We analyze the variations in the architecture configuration for a significant depth up to 25 hidden layers, unlike previous approaches that have worked on only up to 10-12 hidden layers.
4) Our performance evaluation provides the reader with reasoning not just based on the results obtained on the performance metrics but also based on the architecture complexity and training costs.

The rest of the paper is structured as: We provide information about the work previously done in this domain. Our work methodology and the used architecture configurations are described in the next section while the dataset used is reported in the succeeding section. This is followed by elucidation of the results obtained and their analysis, and the conclusion and future work scope are mentioned in the final section of the paper.

II. Background and Related work

Considerable amount of research is being conducted in analyzing the working and the applications of Artificial Neural Networks. Previously, there have been multiple attempts to assess and formulate the impact of the configuration of these architectures. These attempts have focused on the issues of optimizers [13], parameters [14], and the number of layers and types of features which should be present in the ANN architecture [15] [16].

In one of the earliest works in this area, Li et al. [3] proposed an optimization algorithm and theory to estimate the suitable amount of neurons in a feed-forward neural network. Tamura and Tateishi demonstrated the capability of a four-layer feedforward network as compared to a three-layer network [7]. Heaton [2] proposed a thumb rule to be followed for deciding the optimal configuration of the hidden layer. Xu and Chen proposed the calculation of the number of hidden layer neurons by considering the known target function [8]. Shibata and Ikeda presented a general set of guidelines while taking into account the input features and output classes and also mentioning the need for task-specific modifications [9]. Hunter et al. [17] conducted a comparative analysis before determining the neural network size and architecture and inferred a generalized solution by considering all the neurons in the network. One of the more extensive works was conducted by Sheela and Deepa where they tested over 101 different criteria to come up with their mathematical method [18]. Alvarez and Salzmann proposed the parameters of the neural network to be regularized using a group sparsity regularizer to reduce the size of the network [19]. Hu et al. [20] made use of a network pruning approach for improving the efficiency of CNNs.

Some researchers in the past have also tried to analyze these variations for a specific task. Shafi et al. [4] investigated the effect of changes in ANN architecture for a highly concentrated time-frequency distribution target. Xu et al. [5] observed the learning undergone by various layers in an autoencoder. Tangkraingkij explored the appropriate number of neurons for performing private authentication using delta signals of the brainwave [6]. Quite recently, Zhang and Shen commented on ways of choosing the hidden layers setup when working with time-series-based stock price data and its prediction [21]. A major observation from these previous approaches has been that majority of them have focused on a fixed mathematical function-driven approach and have worked only on structured or sequential data while making use of primitive ANN architectures. There is ample scope to analyze the impact of layer configurations when working on an image or video-based task, especially with contemporary ANN architectures and increased depth in terms of layers stacked together.

III. Work Methodology and Architecture Configurations

Assistive driving helps the driver in various ways like a lane change, parking, and collision detection, etc. by giving him a warning or informatory message. This research problem delves around assistive driving in semi-autonomous vehicles. Detecting the driver's actions well in advance can alert the drivers and also offers ADAS more response time to equip them to avert the possibility of an accident. However, it's a challenging research problem to design computer vision-based methods that can improve the performance not only in terms of action detection accuracy but also the anticipation time. The focus of this research is to improve the anticipation time, which is very critical in a real-time scenario like driving. Reaction time for an average person is 1-2 sec, and ADAS aims to provide more reaction/anticipation time for driver safety. More anticipation time means the system has informed the driver that many seconds before so that possible incidents are avoided. Our research work is inspired by the Brain4cars research group of Cornell University, USA. They have used a fusion of three types of features; namely, the car's inside features, outside features, and the maps. Their sensory fusion architecture using deep learning gives 3.5 sec anticipation time with 86% action accuracy. We further try to improve these parameters in our research by novel preprocessing techniques, which form the baseline for our study.

Timely anticipation of driver intention offers a possible solution for allowing ADAS to prevent potential accidents at an early stage. This made vehicle maneuver prediction an established research topic in the last decades, mostly addressed by classifying handcrafted features with a variety of approaches, such as Support Vector Machines, Relevance Vector Machines, Hidden Markov Models, and Recurrent Neural Networks. Despite recent progress in deep end-to-end learning, the majority of previous work on driver maneuver anticipation has been based on manually designed feature descriptors, often employing eye gaze, head, and body pose or other context features, such as GPS and car speed. Though deep learning techniques extract features automatically, we extract more features manually or using a handcrafted way

in the preprocessing step. Then we apply deep learning classifiers, more specifically RNN+LSTM, to find out the driver's maneuver. Here the reason for using RNN is, it works very well as a sequential classifier where data keeps on coming continuously, and previous data gives input to the next data, which helps in our problem to identify and anticipate action.

In this paper, we have worked on the driver activity anticipation task in semi-autonomous vehicles. Given the recent context of the drive video and related features, we are supposed to predict the future maneuver of the driver from one of the following possibilities: shift to the lane on the left, left turn, shift to the lane on the right, right turn, or drive straight. The task thus involves Spatio-temporal kind of data and to model this data, we try out five different architectures as follows:

1) Feed Forward Neural Network (FFNN):
Feedforward neural network is the traditional kind of ANN architecture where individual layers hold uni-dimensional data and layers are fully connected in the architecture [22]. The video of the driver and the external scenario for the previous five seconds are combined with the structured features for better modeling and training of the architecture. The general equations for a hypothesis in FFNN are given as follows:

$$h(x) = w_1x_1 + w_2x_2 + ... + w_nx_n + b \qquad (1)$$

where $w_1, w_2, .., w_n$ are the weights corresponding to the n input features represented by $x_1, x_2, .., x_n$ and b is the bias included in calculation. Both the w and b terms are trained for the nodes present in the hidden layer of the architecture [23].

2) Fusion-RNN-Exponential Loss (FRE):
In this architecture, we include several Recurrent Neural Networks (RNNs) together through which the sensor stream data is passed [24]. The higher level feature representations from the various RNNs are then fused together to be passed through a fully connected neural network. The RNN layer could be formulated as follows:

$$h_t = Wf(h_{t-1}) + W^{hx}x_{[t]} \qquad (2)$$

where h_t represents the hypothesis for the t^{th} term in the sequence and $f(h_{t-1})$ and $x_{[t]}$ indicate the intermediate representation and the input at the t^{th} state in the sequence respectively [25]. In F-RNN-EL, the model loss grows exponentially with time and could be modeled as follows:

$$Loss = -e^{-(T-t)}log(y_t^k) \qquad (3)$$

The introduction of exponential loss layer along with the fusion of RNNs helps to minimize the loss function of the architecture [24]. Figure 1 indicates the architecture of an individual RNN when being trained on the anticipation task.

3) FRE-PreProc:
The architecture is entirely similar to the previous one except for some changes in the preprocessing techniques. The driver's anticipation time is identified using advanced preprocessing techniques involving the fusion of the context variables for efficient training.

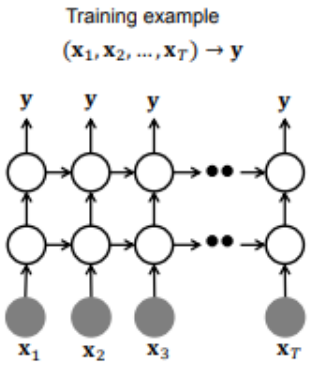

Fig. 1. Training RNN for activity anticipation task

4) FRE-STIP:
As the given data is Spatio-temporal in nature, we also intend to take benefit of this characteristic by including the suitable method in our model architecture. Spatio Temporal Interest Points or STIP refers to such feature spaces that we need to focus upon to improve the model performance. It should be noted that we make use of these STIP techniques in the model while they work only on the internal context of the driver and not the external factor.

5) FRE-DMT:
DMT stands for Driver Movement Tracking and involves a special focus on the internal tracking features i.e. stressing the factors related to the driver. It consists of the preprocessing techniques, STIP methods, and the extraction of the driver's stance and their optic movements [26] [27].

For all these architectures, we analyze the effect of change in the configuration by first incrementing the hidden layers from one to five. We also carry out the same analysis for even deeper architectures by further expanding the number of hidden layers to 25, in the multiples of 5 (i.e. 5, 10, 15,..), and observe the variation in the performance. For a particular intermediate layer, the neurons could be determined as follows:

$$No.\ of\ Neurons = \frac{Training\ Data\ Samples}{f * (Input\ + Output\ neurons)} \qquad (4)$$

where f is a factor used to prevent the overfitting of the model and is a number chosen between 1 and 10. This provides us with the optimum number of neurons to be kept per layer. The models were trained with each of such configuration and architecture types.

For prediction, the trained model provides probabilities for each of the four possible change events. A threshold value is decided and if none of the probabilities across the threshold, we consider the prediction to be the fifth possibility of no activity i.e. 'go straight'. However, if any of the other four predictions are above the threshold, those are retained for the next 5 seconds. In all other cases, the anticipation of driver activity is done every 0.8 seconds based on the context. The

algorithm for inference of activity anticipation is written in Algorithm 1.

The predicted maneuver is then predicted with the ground truth to get the final performance estimate. The input context features of the data are explained in the next section.

Algorithm 1 Inference of activity anticipation

Input: Context features X[$(x_1, x_2, ..., x_T)$,$(z_1, z_2, ..., z_T)$], threshold value p_t
Output: Next maneuver m_{next}

```
Initialize m_0 = driving straight
M : Four possible activity maneuvers
while t = 1 to T do
    Predict probability y_t of each m ϵ M
    m^t_pred = argmax_{mϵM} y_t
    if m^t_pred ≠ driving straight and y_t > p_t then
        m_next = m^t_pred
    end if
end while
return m_next
```

IV. Dataset description

For our work we make use of the publicly available[1] brain4cars dataset. The dataset consists of multiple types of features: inner context and outer context. The inner context features x_i include a video recording of the activity of the driver's head. They are used to follow the driver's face, their head motion, and identify any facial landmarks [12]. The outside features z_i include multiple components of video, GPS as well as speed. The GPS coordinates of the vehicle are used to check its proximity to any road artifact. The minimum, maximum, and average speed of the vehicle for the last 5 seconds are maintained. The outer camera provides video used to keep track of lanes on either edge of the driver's vehicle. Sample images from the dataset including both the inside and outside features are shown in Figure 2.

Fig. 2. Sample images from the dataset

The dataset consists of 2 million video frames from different landscapes covering over 1180 miles of city and freeway driving videos. Over 700 events are annotated for these frames including 131 instances of turning the vehicle, 274 cases of changing the lane, and 295 randomly chosen instances of driving straight. As mentioned earlier, the five possible driver activities are: turn left, change to the left lane, turn right, change to the right lane, and drive straight. 80% of the dataset is used for training while testing of the models is carried out on the remaining data.

The performance of machine learning methods is profoundly dependent on the choice of data representation (or features) on which they are applied. For the data used in this thesis, the sensory input of the system is the multi-modality sequences derived from the following: (i). Face camera: Head pose and eye gaze; (ii) Dashboard camera: Right and left lane change, the relative position of the vehicle (iii) GPS+map: distance and direction from the nearest intersection. We use both handcrafted feature extraction methods and deep learning techniques for extracting useful features from the videos.

A particular configuration of hyperparameters is decided depending on the number of layers present in the architecture. These hyperparameter configuration details are tabulated in Table I.

TABLE I
Hyperparameters used for different layers

Hidden Layers	Layer Delay	Epoch	Performance	Gradient	Batch Size	Initial Learning Rate
5	1:2	7	0.494	0.655	150	0.01
10	1:2	7	0.643	1.08	150	0.01
15	1:2	7	1.58	2.28	150	0.01
20	1:2	7	1.94	3.06	150	0.01
25	1:2	7	1.98	4.01	150	0.01

V. Result and analysis

To evaluate the predictions which are made, we first define the following terminologies:

- True Positive (TP): When the correct driver activity is predicted
- False Activity Prediction (FAP): A driver activity is predicted but the driver keeps driving straight
- False Prediction (FP): When a wrong driver activity is predicted
- Missed Prediction (MP): When the driver performs an activity but we predict them to drive straight

Concerning these terms, we can define accuracy as the fraction of correctly predicted maneuvers from the total maneuvers. Apart from accuracy, we also evaluate our architectures for two other performance metrics: precision and recall.

Precision refers to the ratio of predicted maneuvers that are accurate while recall refers to the ratio of maneuvers that are correctly predicted. The precision and recall values are calculated as:

$$Precision = \frac{TP}{TP + FAP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FP + MP} \tag{6}$$

Apart from the performance-based metrics, the amount of time required for making the prediction is also crucial for

[1] http://www.brain4cars.com

the driver's safety. We term this as anticipation time and calculate it as follows:

$$T = T^A - T^P \tag{7}$$

where T^A is the instance at which the actual activity takes place, and T^P is the instance at which the activity is predicted to take place at T^A. The higher the anticipation time, the better it would be for the system as the driver and vehicle would be more prepared for the activity. Similarly, a higher value of accuracy, precision, and recall is desired.

As mentioned earlier, we first check the impact of layers by gradually increasing them from one to five in each of the architectures. Firstly, we start by calculating the accuracy of the architectures in Table II, followed by precision and recall in Tables III and IV respectively.

It can be seen that there is no difference in the accuracy of the architectures when layers move from 1 to 4. This indicates that the model is stuck at a particular minimum when working with the respective nodes and layers. However, there is a significant increase in the accuracy of the architecture when 5 hidden layers are configured. An increase of up to 9% has been observed in the accuracy when the number of layers has been increased. The highest accuracy is achieved by the FRE-DMT architecture. Next, we compare the variation and impact on the anticipation time and tabulate our results in Table V.

TABLE II
EFFECT OF VARIATION IN HIDDEN LAYERS ON ACCURACY (IN %)

Hidden Layers	**FFNN**	**FRE**	**FRE -PreProc**	**FRE -STIP**	**FRE -DMT**
1	80.2	81.11	83.23	84.67	88.34
2	80.2	81.11	83.23	84.67	88.34
3	80.2	81.11	83.23	84.67	88.34
4	80.2	81.11	83.23	84.67	88.34
5	**85.76**	**87.76**	**92.12**	**87.89**	**96.21**

TABLE III
EFFECT OF VARIATION IN HIDDEN LAYERS ON PRECISION (IN %)

Hidden Layers	FFNN	FRE	FRE -PreProc	FRE -STIP	FRE -DMT
1	75.23	78.19	82.74	86.52	89.45
2	77.59	79.56	88.89	88.56	90.91
3	80.21	80.19	88.96	88.66	91.23
4	81.54	82.66	90.44	88.76	92.15
5	82.75	83.56	91.12	89.44	94.11

TABLE IV
EFFECT OF VARIATION IN HIDDEN LAYERS ON RECALL (IN %)

Hidden Layers	FFNN	FRE	FRE -PreProc	FRE -STIP	FRE -DMT
1	72.35	74.90	81.77	84.15	88.82
2	75.20	75.44	83.44	87.41	91.35
3	82.67	78.49	93.22	87.87	94.73
4	84.39	80.34	91.25	88.39	95.10
5	87.76	88.56	94.12	91.33	97.56

TABLE V
EFFECT OF VARIATION IN HIDDEN LAYERS ON ANTICIPATION TIME (IN SECONDS)

Hidden Layers	**FFNN**	**FRE**	**FRE -PreProc**	**FRE -STIP**	**FRE -DMT**
1	2.87	2.62	2.53	4.23	4.17
2	2.92	2.9	2.87	4.65	4.43
3	3.39	3.32	3.31	4.54	4.41
4	3.68	3.63	3.6	4.71	4.64
5	**3.94**	**3.72**	**3.69**	**4.92**	**4.89**

We observe that for all of the architectures, there is a direct proportionality between the number of layers and the anticipation time. Improvements of upto 45% have been observed from layer 1 to layer 5 in terms of anticipation time in case of FRE-PreProc. The best results have been given by the FRE-STIP architecture with 5 layers which has also seen an improvement of over 16% when going from layer 1 to layer 5.

It could thus be inferred that as the number of hidden layers is improved, the system performs better on the performance metric while making faster predictions as compared to a smaller number of layers. This is true for Spatio-temporal data in our case with an increase of hidden layers from 1 to 5, owing to sufficient training samples and suitable architecture types.

Next, we perform a similar kind of analysis by increasing the hidden layers even further up to 25 in increments of five. The accuracy, precision and recall for each architecture for these varying configurations are listed in Table VI, VII, and VIII respectively.

TABLE VI
EFFECT OF VARYING HIDDEN LAYERS FROM 5 TO 25 ON ACCURACY (IN %)

Hidden Layers	**FFNN**	**FRE**	**FRE -PreProc**	**FRE -STIP**	**FRE -DMT**
5	**85.76**	**87.56**	**92.12**	**87.89**	**96.21**
10	79.12	87.2	91.23	87.45	94.56
15	77.78	85.4	90.56	86.77	93.1
20	73.45	84.67	89.2	86.24	91.45
25	68.49	79.62	84.93	83.44	86.98

TABLE VII
EFFECT OF VARYING HIDDEN LAYERS FROM 5 TO 25 ON PRECISION (IN %)

Hidden Layers	**FFNN**	**FRE**	**FRE -PreProc**	**FRE -STIP**	**FRE -DMT**
5	**82.76**	83.56	**91.12**	**89.44**	**94.11**
10	78.12	**85.2**	90.23	89.1	94.01
15	76.78	84.4	88.56	86.7	92.3
20	73.45	84.67	89.2	85.78	90.1
25	68.49	79.62	84.93	83.56	85.78

TABLE VIII
EFFECT OF VARYING HIDDEN LAYERS FROM 5 TO 25 ON RECALL (IN %)

Hidden Layers	**FFNN**	**FRE**	**FRE -PreProc**	**FRE -STIP**	**FRE -DMT**
5	**87.76**	**88.56**	**94.12**	**91.33**	**97.56**
10	80.12	85.2	92.23	91.13	95.34
15	79.78	87.4	91.56	90.45	93.78
20	75.45	86.67	90.2	90.37	93.56
25	68.49	79.62	84.93	83.4	92.34

It can be seen that increasing the layers from 5 to 25 has to lead to a considerable drop in the accuracy of the systems

as high up to 10% in the case of the best performing FRE-DMT architecture. An inverse relation has been observed in the system performance and hidden layers after increasing them beyond a point. This could be attributed to Overfitting as there is an increase in the number of trainable parameters and sufficient training data is not present. As a result, the model 'overfits' or tends to be too constrained to the nature of the training data and thus underperforms for the testing data. There is no improvement but rather a reduction in the generalization ability of the system.

TABLE IX
EFFECT OF VARYING HIDDEN LAYERS FROM 5 TO 25 ON ANTICIPATION TIME (IN SECS)

Hidden Layers	FFNN	FRE	FRE -PreProc	FRE -STIP	FRE -DMT
5	**3.94**	**3.72**	**3.78**	**4.92**	**4.89**
10	3.11	3.21	3.33	4.76	4.75
15	2.89	2.79	2.82	4.34	4.32
20	1.93	1.67	1.78	3.76	3.75
25	1.74	1.62	1.65	2.7	2.69

As a part of ablation study, we also confirm this impact by evaluating the anticipation time of the architectures in Table IX. A significant drop up to 66% in the case of FRE-PreProc can be observed. With such heavy configurations, the model is getting roughly 2.5-3 seconds to anticipate the activity thus not being reliable enough for the driver. It can be seen that increasing the layers from 5 to 25 has neither been beneficial for performance metrics nor for anticipation time for all types of configurations. For better interpretability of the results, we also visualize the variations in the accuracy, precision, recall and anticipation time for both layers 1-5 and layers 5-25. All these visualizations are shown in the subsequent images.

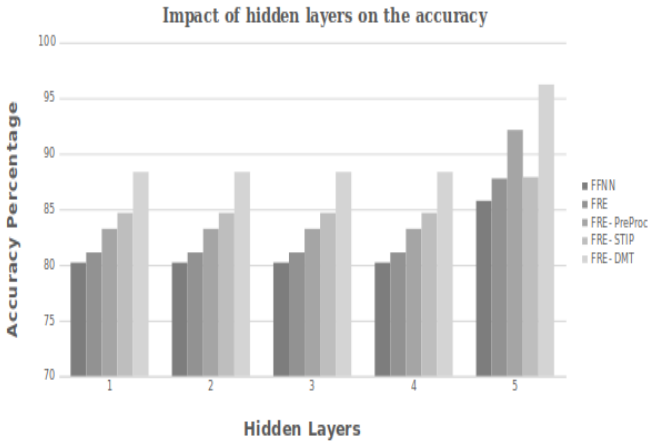

Fig. 3. Impact of hidden layers on accuracy

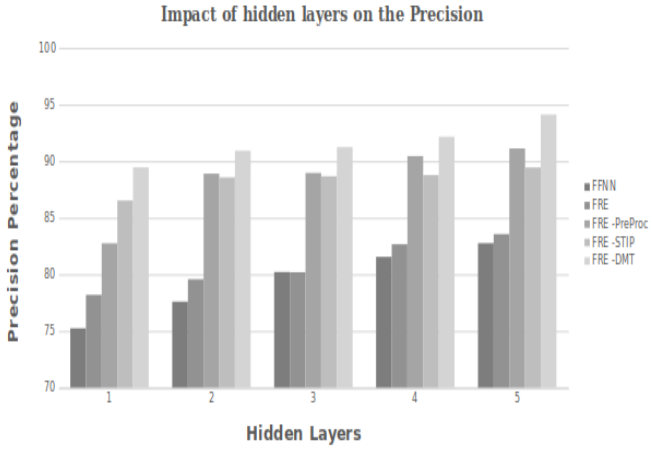

Fig. 4. Impact of hidden layers on precision

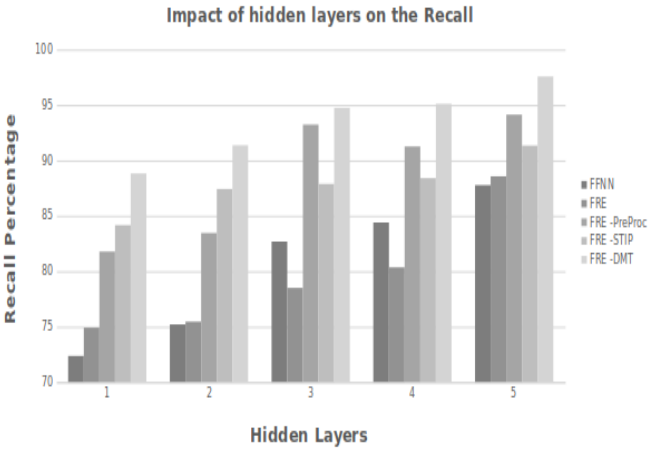

Fig. 5. Impact of hidden layers on recall

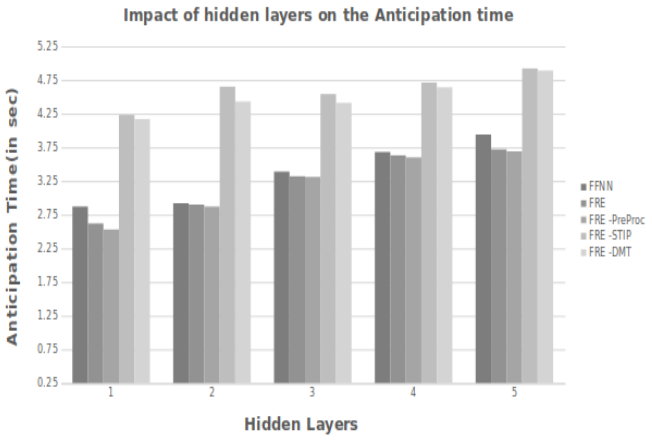

Fig. 6. Impact of hidden layers on anticipation time

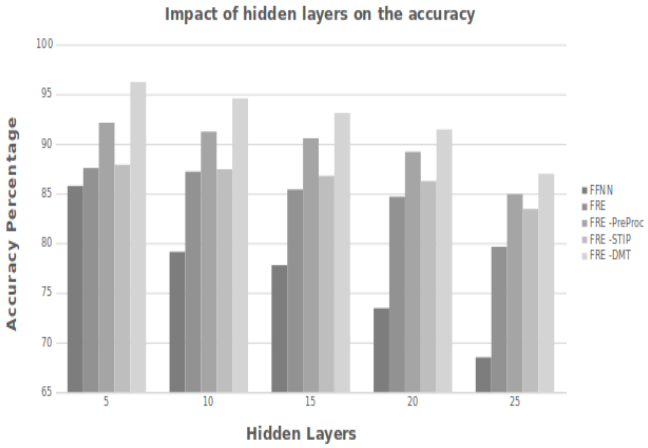

Fig. 7. Impact of hidden layers on accuracy (layers 5 to 25)

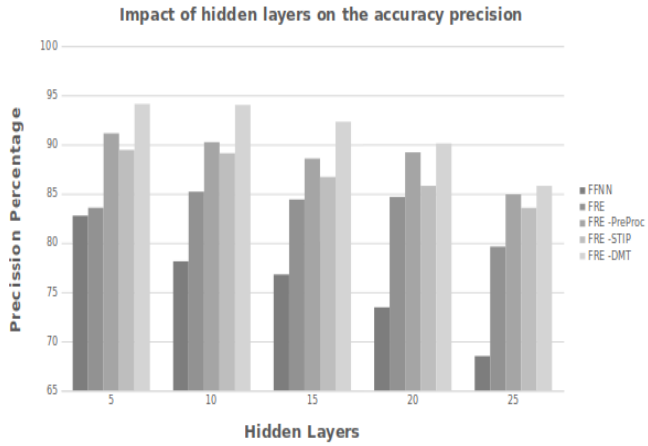

Fig. 8. Impact of hidden layers on precision (layers 5 to 25)

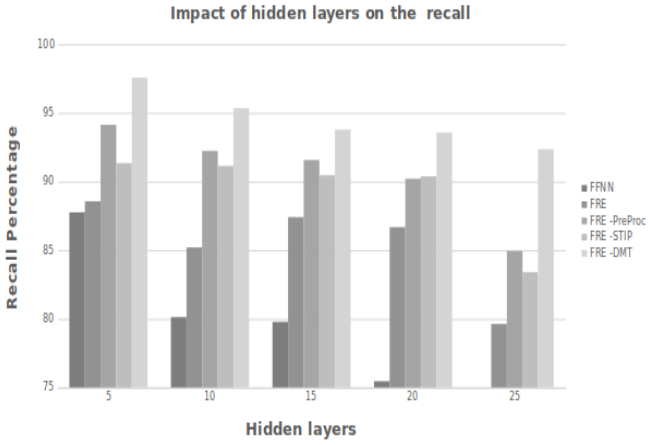

Fig. 9. Impact of hidden layers on recall (layers 5 to 25)

VI. Conclusion

We have performed a thorough investigation on the effect of varying the configuration of the hidden layers on the performance of ANNs, specifically on Spatio-temporal data for driver activity anticipation. We observe that initially, the number of hidden layers is directly proportional to the performance of the system. However later as on stacking more such

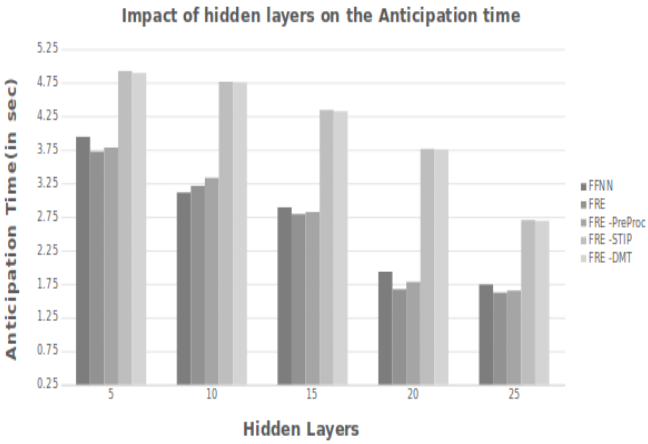

Fig. 10. Impact of hidden layers on anticipation time (layers 5 to 25)

layers together, the training saturates and there is a decline in the performance. These inferences have been confirmed by deducing the performance of the system based on both task-based performance metrics as well as the inference time of the system. ANN architectures with inadequate hidden layers struggle in deriving the right mapping of the input features to the output labels. On the other hand, after a particular level, the model tends to memorize the training data and not improve on the mapping function thus leading to a reduction in the performance. It is thus crucial to have only the necessary and not the excessive stacking of these intermediate layers in the architecture. Further improvements in the work include automated learning of activation functions and improvement in task performance. Natural algorithms like PSO could also be used to decide the ideal ANN configuration. Deep learning methods tend to outperform traditional methods, but it is necessary to come up with architectures that are memory and time-efficient yet effective at the same time.

References

[1] H. Huang, J. Cao, and Y. Qu, "Global robust stability of delayed neural networks with a class of general activation functions," *Journal of Computer and System Sciences*, vol. 69, no. 4, pp. 688–700, 2004.

[2] J. Heaton, *Introduction to neural networks with Java*. Heaton Research, Inc., 2008.

[3] J.-Y. Li, T. W. Chow, and Y.-L. Yu, "The estimation theory and optimization algorithm for the number of hidden units in the higher-order feedforward neural network," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 3. IEEE, 1995, pp. 1229–1233.

[4] I. Shafi, J. Ahmad, S. I. Shah, and F. M. Kashif, "Impact of varying neurons and hidden layers in neural network architecture for a time frequency application," in *2006 IEEE International Multitopic Conference*. IEEE, 2006, pp. 188–193.

[5] Q. Xu, C. Zhang, L. Zhang, and Y. Song, "The learning effect of different hidden layers stacked autoencoder," in *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 2. IEEE, 2016, pp. 148–151.

[6] P. Tangkraingkij, A. Montaphan, and I. Nakavisute, "An appropriate number of neurons in a hidden layer for personal authentication using delta brainwave signals," in *2017 2nd International Conference on Control and Robotics Engineering (ICCRE)*. IEEE, 2017, pp. 232–236.

[7] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layers versus three," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 251–255, 1997.

[8] S. Xu and L. Chen, "A novel approach for determining the optimal number of hidden layer neurons for fnn's and its application in data mining," 2008.

[9] K. Shibata and Y. Ikeda, "Effect of number of hidden neurons on learning in large-scale layered neural networks," in *2009 ICCAS-SICE*. IEEE, 2009, pp. 5008–5013.

[10] D. M. Hawkins, "The problem of overfitting," *Journal of chemical information and computer sciences*, vol. 44, no. 1, pp. 1–12, 2004.
[11] V. Sharma, H.-C. Chen, and R. Kumar, "Driver behaviour detection and vehicle rating using multi-uav coordinated vehicular networks," *Journal of Computer and System Sciences*, vol. 86, pp. 3–32, 2017.
[12] A. Jain, H. S. Koppula, S. Soh, B. Raghavan, A. Singh, and A. Saxena, "Brain4cars: Car that knows before you do via sensory-fusion deep learning architecture," *arXiv preprint arXiv:1601.00740*, 2016.
[13] Ö. F. Ertuğrul, "A novel type of activation function in artificial neural networks: Trained activation function," *Neural Networks*, vol. 99, pp. 148–157, 2018.
[14] J. Tanevski, L. Todorovski, and S. Džeroski, "Combinatorial search for selecting the structure of models of dynamical systems with equation discovery," *Engineering Applications of Artificial Intelligence*, vol. 89, p. 103423, 2020.
[15] M. A. J. Idrissi, H. Ramchoun, Y. Ghanou, and M. Ettaouil, "Genetic algorithm for neural network architecture optimization," in *2016 3rd International Conference on Logistics Operations Management (GOL)*. IEEE, 2016, pp. 1–4.
[16] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1230–1242, 2013.
[17] D. Hunter, H. Yu, M. S. Pukish III, J. Kolbusz, and B. M. Wilamowski, "Selection of proper neural network sizes and architectures—a comparative study," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 2, pp. 228–240, 2012.
[18] K. G. Sheela and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks," *Mathematical Problems in Engineering*, vol. 2013, 2013.
[19] J. M. Alvarez and M. Salzmann, "Learning the number of neurons in deep networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 2270–2278.
[20] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," *arXiv preprint arXiv:1607.03250*, 2016.
[21] P. Zhang and C. Shen, "Choice of the number of hidden layers for back propagation neural network driven by stock price data and application to price prediction," in *Journal of Physics: Conference Series*, vol. 1302, no. 2. IOP Publishing, 2019, p. 022017.
[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[23] J. Han, C. Moraga, and S. Sinne, "Optimization of feedforward neural networks," *Engineering Applications of Artificial Intelligence*, vol. 9, no. 2, pp. 109–119, 1996.
[24] A. Jain, A. Singh, H. S. Koppula, S. Soh, and A. Saxena, "Recurrent neural networks for driver activity anticipation via sensory-fusion architecture," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 3118–3125.
[25] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.
[26] S. Gite, H. Agrawal, and K. Kotecha, "Early anticipation of driver's maneuver in semiautonomous vehicles using deep learning," *Progress in Artificial Intelligence*, vol. 8, no. 3, pp. 293–305, 2019.
[27] S. Gite and H. Agrawal, "Early prediction of driver's action using deep neural networks," *International Journal of Information Retrieval Research (IJIRR)*, vol. 9, no. 2, pp. 11–27, 2019.

Shilpa Gite Dr. Shilpa Gite is a passionate educationalist and researcher at Symbiosis Centre for Applied AI (SCAAI) under Symbiosis International (Deemed University). She has more than 15 years of teaching experience and guided many UG, PG, and Ph.D. students. Her research areas include deep learning, computer vision,Multi-sensor data fusion, Assistive driving. She is currently working in the domains of machine learning, medical imaging, explainable AI, GANs. She has published impactful manuscripts in reputed international conferences and Scopus/ web of science indexed journals and books. In addition to academics and research, she is also a reviewer for reputed journals such as IEEE Transactions on Industrial Electronics, Neurocomputing, PeerJ Computer Science, and many other reputed journals.

Ketan Kotecha Dr. Ketan Kotecha accomplished his Doctorate degree from Indian Institute of Technology (IIT) Bombay in the year 2003. Artificial Intelligence, Machine Learning, and Deep Learning are the core area of research interest along with Computer Algorithms amp; Machine Learning, and Higher Order Thinking Skills, Critical Thinking and Ethics and Value. He had worked as a Principal of GH Patel college of engineering and Technology; then served as Director of Nirma University. He was also a Vice-Chancellor of Parul University. He is presently working as the Dean, Faculty of Engineering amp; Director of Symbiosis Institute of Technology, Symbiosis International (Deemed University). He is the Head of Symbiosis Centre for Applied Artificial Intelligence (SCAAI) amp; CEO of Symbiosis Centre for Entrepreneurship and Innovation. He is a member of the National Advisory Council for Confederation of Indian Industry's (CII) Engineering and Management Curriculum Restructuring Task Force. He is a Technical advisor of a team for "BRTS" implementation at Ahmedabad. He is also appointed as an Independent Director at Gujarat Informatics by Govt of Gujarat. A total of around 45 publications in Scopus indexed journals are to his credit. He has guided 13 Ph.D. scholars and 6 students are taking his guidance for Ph.D.