ABSTRACT Past research on pedestrian trajectory forecasting mainly focused on deterministic predictions which provide only point estimates of future states. These future estimates can help an autonomous vehicle plan its trajectory and avoid collision. However, under dynamic traffic scenarios, planning based on deterministic predictions is not trustworthy. Rather, estimating the uncertainty associated with the predicted states with certain level of confidence can lead to robust path planning. Hence, the authors propose to quantify this uncertainty during forecasting using stochastic approximation which deterministic approaches fail to capture. The current method is simple and applies Bayesian approximation during inference to standard neural network architectures for estimating uncertainty. The authors compared the predictions between the probabilistic neural network (NN) models with the standard deterministic models. The results indicate that the mean predicted path of probabilistic models was closer to the ground truth when compared with the deterministic prediction. Further, the effect of stochastic dropout of weights and long-term prediction on future state uncertainty has been studied. It was found that the probabilistic models produced better performance metrics like average displacement error (ADE) and final displacement error (FDE). Finally, the study has been extended to multiple datasets providing a comprehensive comparison for each model.

INDEX TERMS Uncertainty quantification, Bayesian neural network, Monte Carlo dropout, long short term memory, convolutional neural network.

I. INTRODUCTION

A. MOTIVATION

For a self-driving car, awareness of the surrounding environment is crucial for correct and safe maneuvering [1], [2]. Especially, complex maneuvers require a trustworthy estimate of future states of vulnerable road users (VRUs)
like pedestrians and bicyclists [3]. Continuous progress has been made towards predicting the motion of the vulnerable users with a certain degree of effectiveness [4], [5]. However, most prediction models are deterministic and provide only point estimates of the future states [6]. Such assumptions may be helpful for specific scenarios but in a dynamic environment with multiple interactions, deterministic predictions can be inaccurate. Since, humans tend to change directions swiftly, deterministic predictions may fail to capture this randomness in pedestrian trajectory and thus ignoring the associated uncertainty with the motion. A more robust approach will be to provide a probability distribution based on the likelihood of pedestrian’s location for each predicted state rather than a single point estimate. The uncertainty associated with predicted states can enable autonomous vehicles achieve uncertainty-aware motion planning that will be more robust and trustworthy compared to planning based on deterministic prediction.

For instance, the deterministic prediction outputs point estimates of future states (Figure 1a). However, during long-term forecasts or multiple actor interactions, the deterministic predictions deviate from the ground truth. In such a scenario, the planning algorithm may be uncertain about the future states and the risk of collision will be significantly high. Therefore, a robust and trust-worthy planning algorithm requires improvement in confidence of the predicted states. This can be achieved through probabilistic prediction of states with associated uncertainty (Figure 1b). The risk-aware region around the future state captures all the probable locations the pedestrian can be present at a future time with certain confidence. It enables planning algorithms to either completely avoid the region or plan intelligently to execute uncertainty-aware motion planning that is more robust and less prone to collision [7]. Hence, the current-study proposes a risk-aware motion prediction model for pedestrians which can be further combined with intelligent planning algorithms. The current model can perform long-term motion prediction with high positional accuracy. The uncertainty in motion is estimated in the form of a distribution of trajectories with mean and variance such that the ground truth mostly lies within 95% of the confidence interval of the predictions.

B. RELATED WORK

In recent years, Deep Neural networks (DNNs) have been used extensively for pedestrian trajectory forecasting. Most of the networks are based on recurrent neural network (RNN) architecture which captures the temporal dynamics of sequential data [8]. Although, RNNs should retain complete information of a temporal sequence, practically they fail to propagate long-term dependencies. Hence, Long short-term memory, LSTMs [9], [10] have been used for sequential prediction due to its improved capability in back propagating long-term error. Seminal works like scene-LSTM [11] and social-LSTM [12] have used LSTM network architecture to incorporate either scene information or social pooling between multiple pedestrians for enhanced trajectory forecasting. However, most of the prior work on LSTM focused on improving prediction accuracy and did not stress on quantifying uncertainty. Recently, Convolutional Neural networks have been used for trajectory prediction. Especially, a fast convolutional neural network (CNN) based model compared to 1D convolutional model with LSTM and showed improvement in temporal representation of trajectory [13]. Further, Zamboni et al. [14] elaborated upon the previous work by introducing novel preprocessing and data augmentation techniques to outperform other complex models. However, both the models output deterministic estimates of future state and do not quantify uncertainty. The deterministic predictions may not be trustworthy in complex traffic scenarios. Hence, probabilistic inference of predictions can be useful in safety-critical tasks like collision avoidance [15] and uncertainty-aware motion planning.

Traditionally, Kalman filter has been used for uncertainty estimation [16] but it fails to capture non-linearities during long-term forecasts. Deep learning methods like Gaussian processes [17] and Gaussian Mixture Model [18] have been used for probabilistic forecasting too. For instance, a GP model with unscented Kalman Filter (UKF) was used for long term prediction of obstacle for collision avoidance [19]. However, Gaussian process kernels are infinite-dimensional and require extensive parameter-tuning for accurate prediction. Further, methods like stochastic reachability analysis has been performed by assigning probabilities to future states in the reachable set [20]. However, reachability analysis [21] often estimates all possible future states resulting in a large infeasible set. Recently, non-linear architectures like neural networks are used for probabilistic trajectory forecasting. A CNN based architecture was used to capture the uncertainty but for short-term forecasts only [22]. Further, an LSTM architecture could predict long term probabilistic estimates of future states using an occupancy grid [23]. The problem was formulated as a multi class classification problem using softmax to quantify probabilistic distribution of future states over the occupancy grid. However, softmax often leads to over confident prediction.
and the model can be uncertain with high softmax output [24]. More sophisticated models like the Bayesian Neural networks (BNN) [25] have been used recently to capture the uncertainty in time series.

Typically, a BNN uses a distribution over prior to formulate a posterior distribution which is used to quantify uncertainty during prediction. It can capture three types of uncertainties, model uncertainty termed as epistemic uncertainty, inherent noise in data also known as aleatoric uncertainty, and model misspecification that occurs when testing data is different from the training dataset. In this paper, our main focus is to predict epistemic uncertainty associated with future trajectories. Although, Bayesian network accurately captures uncertainty, the inference becomes challenging due to a large number of model parameters. This often requires the incorporation of variational methods for Bayesian inference that can reduce computational costs. Recently, Monte Carlo (MC) dropout has been used as a variational method for uncertainty estimation in time series forecasting [26] without any significant change to the network architecture.

In the current work, we have elaborated upon this Bayesian approximation of using MC dropout for pedestrian uncertainty estimation. Our work primarily focuses on comparing the prediction performance of deterministic and probabilistic models. We have quantified and compared the uncertainty during trajectory forecasting using three popular neural network architectures for time-series forecasting namely LSTM, 1D CNN and CNN-LSTM. Our novelty lies in showing the importance of probabilistic forecasting of future states over deterministic predictions and also providing a detailed performance comparison between each probabilistic model with its deterministic model. To our knowledge, our work is also novel in the sense that we also show the effect of long-term forecasts as well as stochastic dropout of weights on uncertainty of future trajectory. Moreover, this work also provides a comprehensive performance comparison of both probabilistic and deterministic models on popular pedestrian datasets.

The remainder of the paper is organized as follows. In Section II, we introduce the methods for probabilistic forecasting with Bayesian Neural network followed by describing each neural network architecture; the encoder-decoder model, the convolutional model and the CNN-LSTM model along with Monte Carlo dropout. In Section III, we describe the data preprocessing, implementation details and performance metrics. In Section IV, we discuss the results of uncertainty quantification and provide a comprehensive study on the effect of future prediction horizon and stochastic dropout on performance metrics. Section V concludes our study.

II. METHODS

Deep learning architectures have been used extensively for prediction tasks. However, most networks are deterministic generating point estimates without any confidence interval (Figure 2a). Conversely, a Bayesian Neural Network (BNN) produces uncertainty-aware predictions based on a stochastic network. Although, stochastic networks might provide better point estimates than other standard neural networks, the main aim of stochastic networks is to provide trustworthy uncertainty estimates for predictions. Typically, a stochastic neural network introduces stochastic weights or activation functions (Figure 2b) into the model and is trained using Bayesian approach as:

\[ P(H|D) = \frac{P(D|H)P(H)}{P(D)} \]  

(1)

\( P(H|D) \) is the posterior which implies the hypothesis, \( H \) is statistically updated using inference based on data, \( D \). It is the main aspect of a stochastic network as it captures the model uncertainty known as epistemic uncertainty. The probability \( P(D|H) \) is called likelihood which captures the inherent noise in the data also known as aleatoric uncertainty. Meanwhile, \( P(H) \) is the prior while \( P(D) \) is the evidence. The prior samples from a stochastic distribution of weight with probability, \( P(H) \) unlike standard NNs which have deterministic weights (Figure 2). As a result, the posterior distribution also becomes stochastic and the metrics can be estimated by computing the mean of predicted distribution along with its associated variance. Particularly, stochastic networks such as Bayesian neural networks can be used for time-series forecasting with applications expanding to risk-aware prediction of vulnerable road users. A BNN has several advantages over traditional neural networks such as estimating uncertainty, classifying the uncertainty into epistemic and aleatoric uncertainty as well as integrating the knowledge of prior into the model. For instance, provided a set of training input states, \( X = x_1, x_2, \ldots, x_T \) and outputs, \( Y = y_1, y_2, \ldots, y_T \) for a pedestrian, the distribution of its future predicted states, \( y^* \) can be computed by marginalizing the posterior, \( p(\theta|X, Y) \) over some new input data point, \( x^* \):

\[ p(y^*|x^*, X, Y) = \int \theta p(y^*|\theta^*, \theta') p(\theta'|X, Y) d\theta' \]  

(2)

\( \theta' \) represents weights and \( p(\theta') \) refers to the probability of sampling from prior weight distribution. During Bayesian prediction, computing the posterior, \( p(\theta'|X, Y) \) can be quite challenging and many sampling algorithms like Monte Carlo Markov Chain (MCMC) and variational inference [27] have been used to approximate the posterior. But, most of the sampling methods are either computationally expensive or introduce a lot of parameters into the model which stems
into a complex problem. Further, a BNN incurs additional computation costs due to its non-linearity. Hence, we use the Monte Carlo (MC) dropout method [24] which has been used to accurately approximate a Bayesian neural network without significantly changing the model. The MC dropout method assumes that weights are stochastically dropped during inference. This process is then repeated for N forward passes to generate a random distribution of predicted values. Therefore, we have applied Monte Carlo dropout to different neural network architectures for uncertainty quantification during predictions.

### A. ENCODER DECODER MODEL

We developed an encoder-decoder based simple LSTM architecture to predict future trajectory of pedestrians up to multiple time horizons. An encoder creates an embedding of essential features as a series of encoded space vectors which the decoder uses for estimating outputs. Suppose, \(\{x_t\}_T\) represents the x-position up to T time steps as \(\{x_1, x_2, \ldots, x_T\}\), then, the encoder embeds the data into an encoded space using non-linear function as \(e = g(x)\). The decoder then uses the encoded features to construct \(F\) forward time steps, \(\{x_{T+1}, x_{T+2}, \ldots, x_{T+F}\}\). The current architecture has two LSTM cells for the encoder and one for the decoder (Figure 3). Through an ablation study, we found encoding both position and velocity of the pedestrian was beneficial for accurate prediction rather than encoding only position information. Hence, the input data for LSTM layers is a multivariate time series with four features \(\{x, y, u, v\}\) corresponding to x and y position and velocity respectively. The output of the neural network has similar features and the predicted \(\hat{x}, \hat{y}\) show pedestrian’s future position. For regularization, dropout of weights with a certain probability, \(p\) followed by \(\tanh\) activation has been implemented. The steps are repeated for each LSTM layer to create the encoded vector space which carries the essential features of the training data. The decoder then uses the latent encoded vector to predict the future motion. The decoder architecture has a single LSTM layer that takes the encoded vector space as input. The LSTM layer is then followed by a dropout layer with linear activation to estimate the output.

### B. CONVOLUTIONAL MODEL

Recently, convolutional networks like 1D CNNs have been used for time series analysis. The CNN model is a simple sequence-to-sequence architecture that uses convolutional layers to handle temporal representations. In the current 1D CNN model, we represent the past trajectory as a one-dimensional channel with four features, \(\{x, y, u, v\}\). We constantly pad the input at each convolutional layer so that the number of features in both input and output remains same. We have used “causal” padding for modeling temporal data which eliminates the curse of auto correlation such that the output[t] does not depend upon input[t + 1]. In total, three 1D Convolutional layers with 128, 64, 64 filters respectively have been used to extract features (Figure 4). We use a kernel size of 5 which showed better root mean-squared error (RMSE) in an ablation study with other odd kernel sizes \{3,7\}. A single dropout layer with probability, \(p\) is applied after the first two convolutional layers to prevent overfitting. Further, ‘ReLU’ activation function has been applied to all the convolutional layers. We observed a 1D Max Pool layer followed by upsampling performed better than global average pooling. Finally, for many-to-many predictions, time distributed dense layer has been used to...
predict multiple time steps simultaneously generating a single trajectory. Further, MC dropout can be applied during inference to generate a distribution of trajectories.

C. CNN-LSTM
A CNN-LSTM architecture is a hybrid network that uses both convolution and LSTM layers as an end-to-end model for time-series forecasting. Unlike the LSTM encoder-decoder model, one-dimensional convolutional layers are used for feature extraction to create an embedding rather than LSTM layers. The current model has two one-dimensional convolution layers with filters 128 and 64 respectively. Each convolution layer is followed by a dropout layer where weights are dropped with a certain probability, p to prevent overfitting. In order to process the data into the format required by the LSTM, a Flatten layer is connected after convolution. Further, the LSTM layer acts as decoder and utilises the features for prediction. Similar to previous models, a time-distributed layer at the end enables multi-step forecasting.

D. MC DROPOUT
Epistemic uncertainty during forecasting problems can be estimated from the mean and variance of the marginal distribution, \( p(y^*|x^*, X, Y) \). Recall that \( X \) and \( Y \) represent the training input and output samples while \( y^* \) represents the predicted states for some new test sample, \( x^* \). The marginal distribution is sampled from posterior, \( p(\theta|X, Y) \) based on test input, \( x^* \). The sampling process can be challenging and computationally expensive due to non-linearity in BNNs. Therefore, Monte Carlo dropout can be used as a variational approximation to such Bayesian Neural Networks [24] without modifying the existing network architecture much. Hence, model uncertainty known as epistemic uncertainty can be estimated with ease without any additional computational cost, unlike other inference methods.

As discussed, Monte Carlo dropout (MC dropout) has been applied to each network architecture during inference to quantify uncertainty (Figure 3, 4). The MC dropout model applies stochastic dropout during test time to the neural network. Thus, for any new set of input, \( x^* \), we compute the inference by random dropout at each layer of the model. The probability of dropout is set as \( p \) and the inference model is run \( N \) times to obtain a set of outputs \( \{y^*_1, y^*_2, \ldots, y^*_N\} \). We can then estimate the mean, \( \bar{y}^* \) and variance, \( \Sigma_{y^*} \) of the marginal distribution where the variance indicates model uncertainty (Equation (3)).

\[
\text{Mean, } \bar{y}^* = \frac{1}{N} \sum_{n=1}^{N} y^*_n(n) \\
\text{Var, } \Sigma_{y^*} = \frac{1}{N} \sum_{n=1}^{N} (y^*_n - \bar{y}^*)^2 \tag{3}
\]

III. EXPERIMENTS
In this section, we discuss the datasets, data augmentation, implementation details for each network and the performance metrics. Following common practice from literature [12], we trained our models on publicly available pedestrian datasets. Two most popular datasets are the ETH dataset [28] which contains the ETH and HOTEL scene while the UCY dataset [29] which contains the UNIV, ZARA1 and ZARA2 scenes. In order to draw parallelism with past works [13], we studied 8 (3.2 secs) historical steps to predict 12 (4.8 secs) steps into the future. Further, we extended our study to predict multiple time horizons with long-term forecasts up to 8 seconds into future.

A. DATA AUGMENTATION
Initially, we trained our model on the ETH dataset only which contains approximately 420 pedestrian trajectories under varied crowd settings. However, a small number of trajectories is insufficient for training. Therefore, we performed data augmentation using Taken’s Embedding theorem [30]. We used a sliding window of \( T = 1 \) step to generate multiple small trajectories out of a single large trajectory. For instance, a pedestrian’s trajectory of 29 steps will result in 10 small \( \{x, y\} \) trajectory pairs of 20 steps each if past trajectory information of 8 steps is used for predicting 12 steps into future. In total, we constructed 1597 multivariate time series sequences which we split into 1260 training and 337 testing sequences for the ETH hotel dataset. Further, in Section IV-D, we train on the ZARA1, ZARA2 datasets too to provide a comprehensive comparison of uncertainty quantification across all the models.

B. IMPLEMENTATION DETAILS
All the neural networks are trained end-to-end using Tensorflow. Adam optimizer with a learning rate of \( 1e^{-3} \).
was used to compute the mean-squared error (MSE) loss. Each model was trained for 100 epochs with a batch size of 32. The LSTM encoder-decoder model was trained with ‘tanh’ activation while the other two models had ‘ReLU’ activation. 10% of the training data was used for validation. The MSE error was monitored on the validation loss with callback functions like EarlyStopping and ReduceLROnPlateau. The model was compiled and fit using train and test data.

C. PERFORMANCE METRICS
The trained model is then used to predict the future position of the pedestrian. By default, the model predicts deterministic future states. However, probabilistic predictions can be inferred using Monte Carlo dropout. We can run the stochastic inference using MC dropout repeatedly to generate a distribution of trajectories (Figure 5). The mean of the distribution represents the predicted path while the associated variance quantifies uncertainty. We adopt the widely used performance metrics [12] namely average displacement error (ADE) and final displacement error (FDE) for prediction comparison between the deterministic and probabilistic models.

(a) Average Displacement Error (ADE): It refers to the mean of Euclidean distance over all estimated points of every trajectory with the corresponding points in ground truth.

\[ \text{ADE} = \frac{1}{T} \sum_{t=0}^{T} \left\| \hat{Y}_t - Y_t \right\| \]  

(4)

(b) Final Displacement Error (FDE): The mean of Euclidean distance between the predicted final destination and true final destination across all trajectories.

\[ \text{FDE} = \left\| \hat{Y}_f - Y_f \right\| \]  

(5)

where \( \hat{Y}_t \) is the predicted location at timestamp \( t \) and \( Y_t \) is the ground truth position.

IV. RESULTS AND DISCUSSION
In this section, we discuss about the uncertainty estimation and performance metrics of pedestrian trajectory prediction. Initially, we quantify uncertainty in prediction using the probabilistic models based on the ETH dataset. We also provide a confidence estimation of our predictions with respect to the ground truth. Later, we have shown the effect of varying forecast horizon and dropout on performance metrics for all the deterministic and probabilistic models. Finally, we provide a comprehensive comparison of performance metrics among all the models across multiple datasets.

A. UNCERTAINTY ESTIMATION
For predicting uncertainty and evaluating its trustworthiness, we generated a distribution of trajectories using MC dropout. We have shown the uncertainty estimation of a single pedestrian trajectory from the ETH dataset using the 1D CNN network (Figure 5). For the current scenario, we predict 12 steps or 4.8 seconds into the future based on 3.2 seconds of past trajectory data. A single trajectory (red) is generated by deterministic prediction and provides point estimates of future states. However, we generate probabilistic predictions by applying MC dropout to each neural network model. For instance, the 1D CNN with MC dropout model was sampled \( N = 30 \) times during inference with a stochastic dropout, \( p = 0.2 \) (20% of the weights are randomly dropped) to generate a distribution of \( N \) different trajectories. The mean and variance of the distribution quantifies the uncertainty during trajectory prediction.

The predicted trajectory distribution (Figure 5) shows the pedestrian’s motion along both x and y direction with predominant motion along x. Therefore, we need to quantify the mean and variance along both the directions and treat the predicted trajectory cluster as a bivariate distribution. At each prediction step, the trajectories can be represented as a cluster of \( N \) points distributed on the x-y domain. Assuming Gaussian distribution for each point cluster, we can then estimate the mean and covariance showing the associated uncertainty at each step (Figure 6). The Gaussian representation of the predicted states thus shows the future states with associated uncertainty. Further, we can compute the mean and covariance of this bivariate Gaussian distribution as:

\[ \begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{bmatrix} \right) \]  

(6)

Here, \( \{\mu_x, \mu_y\} \) and \( \{\Sigma_{xx}, \Sigma_{xy}, \Sigma_{yy}\} \) represent the mean and covariance of pedestrian’s movement along x-y domain respectively. Each covariance term shows the correlation of motion along one direction with respect to another. Our results indicate that the covariance, \( \Sigma_{xy} \) is significantly higher compared to covariance along other directions (Figure 7). Similar observation was seen across all the trajectories of
FIGURE 7. Variance in pedestrian motion along x, y and xy.

FIGURE 8. Uncertainty quantification of pedestrian motion on a single trajectory. This shows that uncertainty in motion is more along x followed by y and xy. As the covariance $\Sigma_{xy}$ is negligible, only the standard deviation, $\sigma = \sqrt{\Sigma}$ along x and y were considered to quantify the uncertainty during trajectory prediction.

We compared the estimated uncertainty during prediction of each probabilistic model with the ground truth. ▲ (Fig 8). We have shown the mean predicted path with two standard deviations ($2\sigma$) along x and y direction to quantify uncertainty. The model takes 8 input states (●, green dot) to predict 12 states into future. One can visualise the mean of the predicted distribution (●, blue dot) alongwith the standard deviations (+) along x and y for a single trajectory chosen from the ETH dataset (Figure 8). On inspection, it seems the uncertainty predicted by the CNN-LSTM model grows with prediction horizon while both LSTM and 1D-CNN models provide conservative probabilistic estimates that neither decrease nor increase with time. Further, it appears that the final displacement error (FDE) is minimum for 1D CNN while it is maximum for LSTM. However, to obtain conclusive evidence on predictive accuracy of each model, we need to consider the performance metrics of all possible trajectories as well as estimate whether the ground truth lies within the 95% ($2\sigma$) confidence interval of our predictions.

VELOCITY CORRELATION

In this section, we show the accuracy of current probabilistic model for predicting velocity. We have shown the predicted and actual velocity along x and y using CNN-LSTM for the above trajectory (Figure 9). Every velocity correlation plot contains 20 steps, 8 for observation and 12 for prediction. The predicted velocity at each step can be used to compute its deviation from the actual velocity. Visually, results for the single trajectory show that the predicted velocity along x, $V_x$ is highly correlated with the actual. The x-velocity strongly influences the x-y trajectory plot for CNN-LSTM (Figure 8) as it can be observed that the predicted and actual location for the pedestrian are very close along x. However, the predicted velocity along y, $V_y$ is not correlated and the pedestrian deviates from the actual path laterally. Presumably, the model puts less importance on minimising the MSE loss for the velocity features {u,v} while training as compared to the position features {x,y}. Moreover, the velocity data is more random compared to position and sometimes the model fails to precisely capture the randomness of motion. The importance of velocity correlation has been mentioned in details in the failure cases section.
We define a parameter known as confidence score \( CS \) to check whether the ground truth \( \{x_{true}, y_{true}\} \) lie within two standard deviation \((2\sigma)\) of our predicted distribution.

\[
CS_x = \frac{|x_{true} - \mu_x|_{i=1...F} < 2\sigma_x}{F} \times 100
\]

\[
CS_y = \frac{|y_{true} - \mu_y|_{i=1...F} < 2\sigma_y}{F} \times 100
\]

Here, \( F \) represents the number of predicted states into future. We predict the confidence score along x and y for each test trajectory and then take the mean across all trajectories to obtain a single confidence score for that prediction horizon. Further, we also show the variation of confidence score with prediction time horizon, \( T_F = 3.2, 4.8, 6.4, 8 \) seconds into future.

The LSTM model with MC dropout (gray) provides a high mean confidence score over other models for forecasts upto 4.8 seconds. For instance at \( T_F = 3.2 \) seconds, the LSTM model has a mean \( CS_x \approx 90\% \) which signifies the percentage of ground truth that lies within 2\( \sigma \) confidence interval of the predictions (Figure 10a). However, for long-term forecasts beyond 4.8 seconds, the CNN-LSTM model (light maroon) outperforms the other two models with a mean \( CS_x \approx 80\% \). Similar trend can be observed for \( CS_y \) for both short-term and long-term forecasts (Figure 10b). However, the confidence score along y is less than x for each model. This shows the model is more confident and captures uncertainty effectively along the direction of predominant motion and suitable for long-term linear motions. However, when the motion is significantly small along any direction, the model is less confident and uncertainty is not captured accurately. Among all models, the 1D-CNN probabilistic model (blue) has the lowest confidence score along both x and y across all time horizons. Our results thus indicate that the LSTM model (gray) performs better for short-term probabilistic predictions while CNN-LSTM model (light maroon) has better long-term probabilistic accuracy. Further, the confidence score along y gradually decreased with prediction horizon for each model. This shows the positional accuracy along y for predicted states becomes more uncertain with increase in prediction horizon. To understand how prediction horizon affects uncertainty estimates, we studied its effect on performance metrics like ADE and FDE. Further, we compared each probabilistic model against its deterministic prediction. We also studied the effect of stochastic dropout probability, \( p \) on performance metrics.

### B. DROPOUT

Stochastic dropout during inference is critical for inducing uncertainty into the model. The network weights are randomly dropped with certain probability, \( p \) at every inference generating a new trajectory. The whole inference process was repeated multiple times generating a distribution of trajectories (Figure 5). For implementing probabilistic inference, we considered dropout layers in each neural network architecture. Meanwhile the deterministic predictions are obtained using the same neural network architecture that uses dropout layers only during training but not for inference. For the experiments, each probabilistic model was studied with dropout probabilities, \( p = \{0.2, 0.3, 0.4, 0.5\} \). Our main aim is to quantify uncertainty and understand the effect of dropout probability, \( p \) on performance metrics for each probabilistic model and compare that to deterministic prediction.

In Figure 11, we compare the uncertainty estimation for dropout probability, \( p = 0.2 \) and 0.4 using probabilistic CNN-LSTM model for a particular trajectory. The results indicate that with increase in dropout probability, the uncertainty in pedestrian state along x increased significantly. Since, more weights are randomly dropped, the variance associated with prediction also increases and thus the model becomes less certain during prediction. However, no significant difference in uncertainty was observed along y. Further, the mean predicted path is farther away from the ground truth at \( p = 0.4 \) than at \( p = 0.2 \) for the current trajectory. A more detailed analysis on variation of performance metrics with stochastic dropout has been provided below.

In Figure 12, minimum average displacement error, \( ADE = 0.543 \) (less is better) was obtained at \( p = 0.2 \). Further, the ADE increased with dropout probability till \( p = 0.4 \). It is evident as a higher dropout implies more weights are randomly dropped from the architecture so the variance in trajectories would increase. This increase in randomness across the predicted distribution somehow generates a mean predicted path that has more ADE with the ground truth. Yet, further increase in dropout probability, \( p = 0.5 \) resulted
in smaller ADE. Meanwhile, no significant variation was observed in final displacement error with change in dropout probability for the LSTM model.

However, probabilistic predictions with MC dropout on 1D CNN and CNN-LSTM showed significant improvement in ADE and FDE over deterministic predictions across all dropout values. For instance, at $p = 0.4$, 1D CNN model with LSTM dropout has an ADE = 0.819 which shows an improvement of 13.8% over vanilla 1D-CNN model with ADE = 0.95 (Figure 13). The performance improvement was even higher for the probabilistic CNN-LSTM model with at least 20% improvement in ADE over vanilla CNN-LSTM model (Figure 14) across all dropout values. Similar trend was also seen for the final displacement error where the FDE increased with dropout probability till $p = 0.4$ and then decreased across all models.

Overall, the probabilistic models 1D-CNN and CNN-LSTM showed significant improvement in both ADE and FDE over deterministic predictions across all dropout probability values. Further, the ADE and FDE increased till $p = 0.4$ across all models which shows dropout induces uncertainty into the model as higher dropout rates should
lead to more variance in trajectory distribution. However, further increase in dropout, $p = 0.5$ resulted in smaller ADE and FDE (less is better) which seems counter-intuitive.

C. TIME HORIZON

We compare the probabilistic and deterministic models for multiple time horizons into future. Based on a past trajectory of $T = 3.2$ seconds, we predicted the future states along with their associated uncertainty at $T_f = 3.2, 4.8, 6.4, 8$ seconds. A constant dropout probability, $p = 0.2$ was considered for all experiments as it had the minimum average displacement error and final displacement error (Section IV-B). Our results indicate that minimum ADE and FDE occur for the smallest time horizon at $T_f = 3.2$ seconds. Further, both ADE and FDE increased with prediction horizon across all models. It shows that irrespective of the models, error in prediction increases with the increase in prediction horizon.

In Figure 15, we compare the uncertainty in predicted future states at $T_f = 3.2$ and 4.8 seconds respectively. Both plots indicate that the uncertainty grows with prediction horizon. Especially, at $T_f = 4.8$ seconds, the variance along both $x$ and $y$ grow significantly with each predicted step when compared with the predicted trajectory at $T_f = 3.2$ seconds. Further, we have compared the performance metrics based on the mean of predicted trajectories from probabilistic prediction with its deterministic forecast. For both probabilistic and deterministic models, both ADE and FDE increased with prediction horizon.

In Figure 16, LSTM with MC dropout model shows no improvement in performance metrics over vanilla LSTM. However, both 1D CNN and CNN-LSTM with MC dropout produce mean predictions that have lower ADE and FDE when compared with the standard deterministic architectures without MC dropout (Figure 17, 18). Especially, the probabilistic CNN-LSTM model has the lowest ADE and FDE among all probabilistic models across each prediction horizon. Further, the CNN-LSTM model with MC dropout produced at least $15\%$ and $35\%$ improvement in ADE and FDE respectively over its deterministic prediction.

Overall, uncertainty increased with both dropout and prediction horizon. Especially, with increase in dropout, the uncertainty increased along the predominant direction of motion, i.e., $x$. Meanwhile, increase in prediction horizon
D. QUANTITATIVE EVALUATION

In this section, we evaluate the current probabilistic methods and compare them to state-of-the-art pedestrian forecasting models across ETH [28] and UCY [29] dataset. Evaluation of error metrics, ADE and FDE for the state-of-the-art models is based upon the best-of-20 protocol from a list of twenty trajectories. As a preferred practice in literature, we observed 8 steps to predict 12 steps into future. In Table 1, we summarize the performance metrics, ADE/FDE based on our predictions for the current models. In total, there are three probabilistic models trained with MC dropout and three standard neural network architectures which generate single deterministic predictions. All the probabilistic models with MC dropout are inferred using a dropout probability, $p = 0.2$.

**Baselines:** We compare our model against the following state-of-the-art models:

1. S-LSTM [12]: Each person is modeled using LSTM with social pooling layer.
2. SGAN [32]: A generative model that uses social information.
3. Sophie [33]: A generative model that uses both social and image information.
4. Social-BiGAT [34]: Attention based generative network for multi-modal trajectory forecasting.

The current results indicate that the novel CNN-LSTM model with MC dropout has the lowest error among the considered probabilistic and deterministic networks for the ETH and HOTEL scene. The mean predicted path has a minimum ADE/FDE (0.3/0.48) for the HOTEL scene. We speculate that the CNN captures feature more efficiently than standard LSTM encoder. Moreover, the probabilistic CNN-LSTM model outperforms every other generative baseline model across the ETH and HOTEL scene which shows inclusion of social information like agent interaction may not necessarily produce better performance metrics. As suggested in literature [14], the simple linear velocity model [32] outperforms many state-of-the-art architectures. Moreover, the consideration of pedestrian velocity information during training improved the performance of current models without the consideration of social information.

Although, probabilistic CNN-LSTM achieves state-of-the-art performance for ETH and HOTEL scenes, the same is not true across ZARA scenes. In ETH and HOTEL scene, the pedestrian density is low and the scenario is simple for which the inclusion of only position and velocity information produced better predictions. Meanwhile, the pedestrian density is more with significant pedestrian interactions for the ZARA scenes and the current proposed model performs worse than generative models. Nonetheless, the overall average error for probabilistic LSTM and CNN-LSTM networks was minimum across all datasets. Although, the social-BiGAT had a comparable ADE, the final FDE was considerably large compared to current probabilistic models. Interestingly, the probabilistic models had a better final displacement error over all state-of-the-art generative models. It can be due to
the averaging of multiple trajectories generated from probabilistic models. Overall, the current probabilistic models outperform the state-of-the-art generative models across ETH dataset while perform worse for the ZARA scenes. It shows it is not always necessary to consider contextual or social information to accurately predict trajectories. Furthermore, as the current model does not rely upon social interaction or contextual cues, it is much faster and easier for real-time implementation.

E. FAILURE CASES

Exploration of velocity correlation plots highlighted few failure cases in motion of the pedestrian. We term these deviations as (a) Moving fast/slow: the predicted pedestrian motion is either faster or slower than actual and (b) Turning left/right: pedestrian deviates from actual path laterally.

1) MOVING FAST/SLOW

Figure 19 predicts the pedestrian moving faster than actual. The predicted steps into the future are ahead of the actual states. This happens when the velocity is uncorrelated along the x axis. It can be observed in the 12 predicted steps that the predicted velocity, $V_x$ is higher than than actual velocity by some margin. This offset in velocity is reflected onto the future positions and the pedestrian either moves fast or slow depending upon the $V_x$.

2) TURNING LEFT/RIGHT

Similar observations can be made when the predicted path of the pedestrian deviates significantly form the actual in the lateral direction (Figure 20). The pedestrian motion along x-y reveals that the pedestrian has suddenly changed direction and turns right. However, the predicted future positions are unable to capture this motion and predicts a straight trajectory based on observed historical data. This can be addressed through the velocity correlation plots. The lateral velocity, $V_y$ plot shows that the pedestrian was initially not moving along y and then abruptly starts to move laterally (encircled). The trained model fails to capture this randomness and that is why the predicted trajectory is linear and is unable to follow the turning maneuver. Both failure cases reveal that it is necessary for the model to capture the randomness in velocity accurately to improve overall accuracy in trajectory prediction.

V. CONCLUSION AND FUTURE RESEARCH

In this paper, we presented a Bayesian approach using Monte Carlo dropout to quantify the uncertainty in pedestrian trajectory prediction. The method was evaluated on real-world pedestrian dataset to generate a distribution of trajectories instead of a single trajectory. The current results indicate that the mean predicted path of probabilistic model is better and closer to the ground truth than the predictions from deterministic models. The probabilistic models also had a lower average error than state-of-the-art methods for the ETH dataset. Further, the effect of varying dropout probability and time horizon showed that both ADE and FDE increased. It implies the probabilistic models become less certain in its predictions with increase in either prediction horizon or dropout probability. However, the performance metrics of probabilistic models were better than deterministic models.

In future, we plan to improve the probabilistic predictions such that the ground truth should always lie within the predicted trajectory distribution. We will explore other Bayesian methods for uncertainty quantification or change the current neural network architecture [31] for accurate long-term prediction.

APPENDIX

OTHER DATASETS

Apart from pedestrian datasets like ETH and UCY, the current probabilistic model was evaluated on mixed traffic dataset. The model was trained upon a single scene from the inD dataset [35] which contains naturalistic trajectories of both vehicles and vulnerable road users (VRUs). In total, the scene contains the position and velocity information of 355 vehicles and 30 vulnerable road users namely pedestrians and bicyclists (Figure 21). Unlike the ETH and UCY dataset
which are sampled at 2.5 Hz (0.4 secs), the inD data is collected at 25 Hz. Therefore, we decided to observe 25 historical states (1 sec) to predict 25 states into the future. In order to create a training set, every possible 50-length subset of each trajectory was taken. Further, we performed a skip of 5 samples while generating each new subset of 50 sample sub-trajectory. With this technique, the inD scene zero contains 31237 training samples and 6223 test samples to evaluate predictions.

The predictions are carried out using the LSTM network for both probabilistic and deterministic predictions. The architecture and parameters of the network were kept similar as before. We evaluate the accuracy of trajectory prediction for a single trajectory and compute the error metrics ADE and FDE across the test trajectories. It was observed that the model was able to accurately predict and capture faster motions as seen for the vehicle (Figure 22).

Meanwhile, for a pedestrian, the FDE seems large and the model mostly predicts fast/slow motion when compared to the actual trajectory. Our understanding is that since the model is predominantly trained upon vehicle trajectories with the velocity information. And, the vehicles move significantly faster compared to pedestrians, the model is unable to accurately predict slow moving objects like pedestrians in a mixed traffic scenario. Nonetheless, the average error, i.e., ADE/FDE for the probabilistic model was 0.95/1.29 which was significantly lower than deterministic prediction 1.51/1.53 across the inD dataset. This also shows that our model is generalizable over other datasets too.

REFERENCES

[1] C. Zhang and A. Eskandarian, “A quality index metric and method for online self-assessment of autonomous vehicles sensory perception,” 2022, arXiv:2203.02588.

[2] A. Nayak, A. Eskandarian, P. Ghorai, and Z. Doerzaph, “A comparative study on feature descriptors for relative pose estimation in connected vehicles,” in Proc. ASME Int. Mech. Eng. Congr. Expo., 2021, Art. no. V07BT07A022.

[3] A. Eskandarian, C. Wu, and C. Sun, “Research advances and challenges of autonomous and connected ground vehicles,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 2, pp. 683–711, Feb. 2021, doi: 10.1109/TITS.2019.2958352.

[4] H. Xue, D. Q. Huynh, and M. Reynolds, “Bi-prediction: Pedestrian trajectory prediction based on bidirectional LSTM classification,” in Proc. Int. Conf. Digit. Image Comput. Techn. Appl. (DICTA), 2017, pp. 1–8.

[5] H. Xue, D. Q. Huynh, and M. Reynolds, “SS-LSTM: A hierarchical LSTM model for pedestrian trajectory prediction,” in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), 2018, pp. 1186–1194.

[6] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, “Vehicle trajectory prediction based on motion model and maneuver recognition,” in Proc. IEEE/RSJ Int. Conf. Intell.Robots Syst., 2013, pp. 4363–4369, doi: 10.1109/IROS.2013.6696982.

[7] G. Kahn, A. Villaflo, V. Pong, P. Abbeel, and S. Levine, “Uncertainty-aware reinforcement learning for collision avoidance,” 2017, arXiv:1702.01182.

[8] Y. Suo, W. Chen, C. Clarumant, and S. Yang, “A ship trajectory prediction framework based on a recurrent neural network,” Sensors, vol. 20, no. 18, p. 5133, 2020.

[9] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, “Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture,” in Proc. IEEE Intell. Veh. Symp. (IV), 2018, pp. 1672–1678.

[10] F. Altché and A. de La Fortelle, “An LSTM network for highway trajectory prediction,” in Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC), 2017, pp. 353–359.

[11] H. Manh and G. Alaghband, “Scene-LSTM: A model for human trajectory prediction,” 2018, arXiv:1808.04018.

[12] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social LSTM: Human trajectory prediction in crowded spaces,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 961–971.
[13] N. Nikhil and B. T. Morris, “Convolutional neural network for trajectory prediction,” in Proc. Eur. Conf. Comput. Vis. Workshops (ECCV), 2018, pp. 186–196.

[14] S. Zamboni, Z. T. Kefato, S. Girdzijauskas, C. Norén, and L. Dal Col, “Pedestrian trajectory prediction with convolutional neural networks,” Pattern Recognit., vol. 121, Jan. 2022, Art. no. 108252.

[15] X. Wu, A. Nayak, and A. Eskandarian, “Motion planning of autonomous vehicles under dynamic traffic environment in intersections using probabilistic rapidly exploring random tree,” SAE Int. J. Connected Autom. Veh., vol. 4, no. 4, pp. 383–399, 2021.

[16] N. Nikhil and B. T. Morris, “Convolutional neural network for trajectory prediction,” in Proc. IEEE 5th Int. Conf. Intell. Comput. Commun. Process., 2009, pp. 417–422.

[17] D. Ellis, E. Sommerlade, and I. Reid, “Modelling pedestrian trajectory patterns with Gaussian processes,” in Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops, ICCV Workshops, 2009, pp. 1229–1234.

[18] I. Wiest, M. Höflken, U. Kredel, and K. Dietmayer, “Probabilistic trajectory prediction with Gaussian mixture models,” in Proc. IEEE Intell. Veh. Symp., 2012, pp. 141–146.

[19] C. Sun and A. Eskandarian, “A predictive frontal and oblique collision threat assessment,” in Proc. ASME Int. Mech. Eng. Congr. Expo., 2021, Art. no. V07BT07A026.

[20] M. Althoff, “Reachability analysis and its application to the safety assessment of autonomous cars,” Ph.D. dissertation, Fakultät Elektrotechnik Informationstechnik, Technische Universität, München, Germany, 2010.

[21] S. Zernetsch, H. Reichert, V. Kress, K. Doll, and B. Sick, “Trajectory forecasts with uncertainties of vulnerable road users by means of neural networks,” in Proc. IEEE Intell. Veh. Symp. (IV), 2019, pp. 810–815.

[22] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, “Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network,” in Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC), 2017, pp. 399–404.

[23] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in Proc. Int. Conf. Mach. Learn., 2016, pp. 1050–1059.

[24] X. Zhang and S. Mahadevan, “Bayesian neural networks for flight trajectory prediction and safety assessment,” Decis. Support Syst., vol. 131, Apr. 2020, Art. no. 113246.

[25] L. Zhu and N. Laptev, “Deep and confident prediction for time series at uber,” in Proc. IEEE Int. Conf. Data Mining. Workshops (ICDMW), 2017, hb0xp. pp. 103–110.

[26] L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun, “Hands-on Bayesian neural networks—A tutorial for deep learning users,” 2020, arXiv:2007.06823.

[27] S. Pellegrini, A. Ess, and L. Van Gool, “Improving data association by joint modeling of pedestrian trajectories and groupings,” in Proc. Eur. Conf. Comput. Vis., 2010, pp. 452–465.

[28] L. Leal-Taixé, M. Fienzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese, “Learning an image-based motion context for multiple people tracking,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2014, pp. 3542–3549, doi: 10.1109/CVPR.2014.453.

[29] F. Takens, “Detecting strange attractors in turbulence,” in Dynamical Systems and Turbulence, Warwick 1980. Heidelberg, Geramny: Springer, 1981, pp. 366–381.

[30] C. Zhang, A. Eskandarian, and X. Du, “Attention-based neural network for driving environment complexity perception,” in Proc. IEEE Intell. Transp. Syst. Conf. (ITSC), 2021, pp. 2781–2787.

[31] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, “Social GAN: Socially acceptable trajectories with generative adversarial networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 2255–2264.

[32] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, “Sophie: An attentive gan for predicting paths compliant to social and physical constraints,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 1349–1358.

[33] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, S. H. Rezatofighi, and S. Savarese, “Social-BiGAT: Multimodal trajectory forecasting using bicycle-GAN and graph attention networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019, p. 13.

[34] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, “The inD dataset: A drone dataset of naturalistic road user trajectories at German intersections,” in Proc. IEEE Intell. Veh. Symp. (IV), 2020, pp. 1929–1934.

[35] A. Nayak, A. Eskandarian, and X. Du, “Attention-based neural network trajectory forecasting using bicycle-GAN and graph attention networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019, p. 13.