Collaborative Uncertainty in Multi-Agent Trajectory Forecasting

Bohan Tang[1] **Yiqi Zhong**[2] **Ulrich Neumann**[2] **Gang Wang**[3] **Ya Zhang**[1] **Siheng Chen**[1*]
[1]Shanghai Jiao Tong University [2]University of Southern California [3]Beijing Institute of Technology
[1]`tangbohan@alumni.sjtu.edu.cn` [1]`{sihengc, ya_zhang}@sjtu.edu.cn`
[2]`{yiqizhon, uneumann}@usc.edu` [3]`gangwang@bit.edu.cn`

Abstract

Uncertainty modeling is critical in trajectory forecasting systems for both interpretation and safety reasons. To better predict the future trajectories of multiple agents, recent works have introduced interaction modules to capture interactions among agents. This approach leads to correlations among the predicted trajectories. However, the uncertainty brought by such correlations is neglected. To fill this gap, we propose a novel concept, *collaborative uncertainty* (CU), which models the uncertainty resulting from the interaction module. We build a general CU-based framework to make a prediction model learn the future trajectory and the corresponding uncertainty. The CU-based framework is integrated as a plugin module to current state-of-the-art (SOTA) systems and deployed in two special cases based on multivariate Gaussian and Laplace distributions. In each case, we conduct extensive experiments on two synthetic datasets and two public, large-scale benchmarks of trajectory forecasting. The results are promising: 1) The results of synthetic datasets show that CU-based framework allows the model to appropriately approximate the ground-truth distribution. 2) The results of trajectory forecasting benchmarks demonstrate that the CU-based framework steadily helps SOTA systems improve their performances. Specially, the proposed CU-based framework helps VectorNet improve by 57 cm regarding Final Displacement Error on nuScenes dataset. 3) The visualization results of CU illustrate that the value of CU is highly related to the amount of the interactive information among agents.

1 Introduction

A multi-agent trajectory forecasting system aims to predict future trajectories of multiple agents based on their observed trajectories and surroundings [1, 2]. Precise trajectory prediction provides essential information for decision making and safety in numerous intelligent systems, including autonomous vehicles [3, 4, 5, 6], drones [7], and industrial robotics [8, 9].

The rapid development of deep learning has enabled a number of deep-learning-based algorithms to handle multi-agent trajectory forecasting [3, 4, 5, 6, 10, 11, 12, 13, 14, 15]. These methods exhibit state-of-the-art performances, with some having been integrated into real-world systems. However, deep-learning-based forecasting is not always reliable or interpretable [16, 17, 18]. In circumstances when noises from the environment are overwhelmingly distracting, or when the situation has never been encountered before, a deep-learning-based algorithm could provide baffling predictions, which might cause terrible tragedies. A fundamental challenge is to know when we could rely on those deep-learning-based forecasting algorithms. To tackle this problem, one solution is to report the uncertainty of each prediction. Finding ways to best conceptualize and measure the prediction uncertainty of deep-learning-based algorithms becomes an imperative, which motivates this work.

[*]The corresponding author is Siheng Chen.

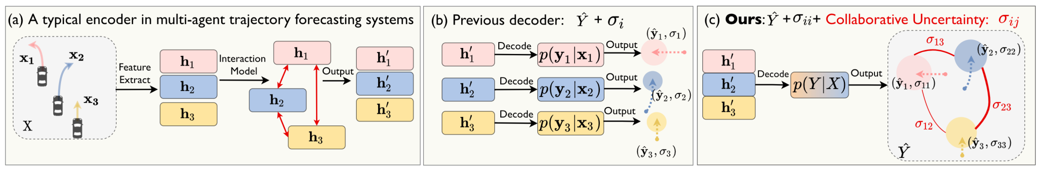

(a) A typical encoder in multi-agent trajectory forecasting systems
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
X
Feature Extract
$\mathbf{h}_1$
$\mathbf{h}_2$
$\mathbf{h}_3$
Interaction Model
Output
$\mathbf{h}'_1$
$\mathbf{h}'_2$
$\mathbf{h}'_3$
(b) Previous decoder: $\hat{Y}$ + σ_i
Decode
$p(\mathbf{y}_1|\mathbf{x}_1)$
$p(\mathbf{y}_2|\mathbf{x}_2)$
$p(\mathbf{y}_3|\mathbf{x}_3)$
Output
$(\hat{\mathbf{y}}_1, \sigma_1)$
$(\hat{\mathbf{y}}_2, \sigma_2)$
$(\hat{\mathbf{y}}_3, \sigma_3)$
(c) Ours: $\hat{Y}$ + σ_{ii} + Collaborative Uncertainty: σ_{ij}
Decode
$p(Y|X)$
Output
$(\hat{\mathbf{y}}_1, \sigma_{11})$
$(\hat{\mathbf{y}}_2, \sigma_{22})$
$(\hat{\mathbf{y}}_3, \sigma_{33})$
σ_{13}
σ_{23}
σ_{12}
$\hat{Y}$

Figure 1: **Uncertainty modeling in multi-agent trajectory forecasting.** (a) a typical pipeline of an encoder in multi-agent trajectory forecasting systems. (b) and (c) illustrate the decoder pipeline of previous methods and our method respectively. Previous methods output the predicted trajectory $\hat{Y}$ and individual uncertainty σ_i, and our method additionally outputs *collaborative uncertainty* σ_{ij}.

There are two main types of uncertainty to model in deep-learning-based algorithms [19]: 1) aleatoric uncertainty, regarding information aside from statistic models, which data cannot explain; 2) epistemic uncertainty, the uncertainty inside a model, when the model lacks knowledge of the system/process being modeled (e.g., due to limited training data). As it is most effective to model aleatoric uncertainty in big data regimes such as those common to deep learning with image data [17], this work focuses on aleatoric uncertainty. In the following passage, we use the term "uncertainty" to represent aleatoric uncertainty. [16] uses the predictive variance to approximate uncertainty in the Bayesian deep learning model, which has been widely adapted in many works [20, 21, 22, 11] for uncertainty modeling in multi-agent trajectory forecasting. However, the predictive variance of a single agent alone may not suffice to reflect the complete landscape of uncertainty, especially when agent-wise interaction is present. Recent works that attempt to exploit the interaction among agents have impressively boosted the prediction precision, which further highlights the need to better measure uncertainty in multi-agent trajectory forecasting. We seek to build a more sophisticated and robust measurement for capturing the previously neglected uncertainty brought by correlated predictions.

In this paper, we coin a concept *individual uncertainty* (IU) to describe the uncertainty that can be approximated by the predictive variance of a single agent. Relatively, we propose a new concept, *collaborative uncertainty* (CU) to estimate the uncertainty resulting from the usage of interaction modules in prediction models. We further introduce an original probabilistic CU-based framework to measure both individual and collaborative uncertainty in the multi-agent trajectory forecasting task. We apply this framework to two special cases: multivariate Gaussian distribution and multivariate Laplace distribution. In each case, our CU-based framework allows our model to simultaneously learn the mappings that are from input data to 1) accurate prediction, 2) individual uncertainty, and 3) collaborative uncertainty; see Figure 1 for model illustration. Extensive experiments demonstrate that CU modeling yields significantly larger performance gains in prediction models equipped with interaction modules (See Figure 4), confirming that CU is highly related to the existence of the interaction modeling procedure, and adding CU modeling benefits accurate predictions.

The contributions of this work are summarized as follows:

• We propose, analyze, and visualize a novel concept, *collaborative uncertainty* (CU), to model the uncertainty brought by the interaction modules in multi-agent trajectory forecasting.

• We design a general CU-based framework to empower a prediction model to generate a probabilistic output, where the mean is the future trajectory and the covariance reflects the corresponding uncertainty. Under this framework, we show two special cases based on multivariate Gaussian and Laplace distributions respectively.

• We conduct extensive experiments to validate the CU-empowered prediction model on both synthetic datasets and two large-scale real-world datasets. On self-generated synthetic datasets, we validate the proposed method is able to closely reconstruct the ground-truth distribution. On the public benchmarks, the CU-empowered prediction model consistently outperforms the corresponding one without CU. Specially, by leveraging the proposed CU, VectorNet improves by 57 cm regarding Final Displacement Error (FDE) on nuScenes dataset!

2 Related Works

Aleatoric uncertainty modeling in deep learning. Recent efforts are rising as to improve the measurement of aleatoric uncertainty in deep learning models. One seminal work is [17]. It proposes a unified Bayesian deep learning framework to explicitly represent aleatoric uncertainty using predictive variance for generic regression and classification tasks. Many existing works [23, 24, 25, 26, 27, 28] follow this idea and formulate uncertainty as learned loss attenuation. For example, to make predictive-variance-based aleatoric uncertainty measurements more efficient, [24] adds data augmentation during the test-time. But, these works only pay attention to individual uncertainty.

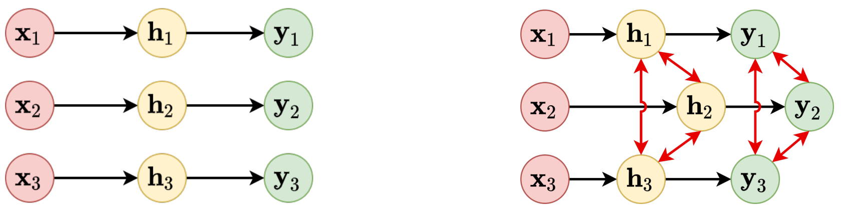
Other recent works attend to the uncertainty measurement for correlated predictive distributions. For example, [29] and [30] measure spatially correlated uncertainty in a generative model for respectively image reconstruction and pixel-wise classification, and [31] captures joint uncertainty as discrete variables in the field of reinforcement learning. Despite these three works, our work is the first to conceptualize and measure collaborative uncertainty in the multi-agent trajectory forecasting task. To the best of our knowledge, there are only two papers [32, 33] close to our track. [32] and [33] model the joint uncertainty in the pose estimation task and multi-agent trajectory forecasting task respectively. However, they present several limitations: 1) They only examined the circumstance where the model's output follows Gaussian distribution; 2) They did not provide a theoretical conceptualization or definition for the uncertainty due to correlated predictive distributions, and they did not analyze the causes of such uncertainty. These are essential problems to tackle. In this work, we not only formulate a general framework that works for both Gaussian and Laplace distributions, but we also theoretically conceptualize *collaborative uncertainty* and analyze its causes.

Multi-agent trajectory forecasting. This task takes the observed trajectories from multiple agents as the inputs, and outputs the predicted trajectory for each agent. Like many other sequence prediction tasks, this task used to use a recurrent architecture to process the inputs [34, 35, 36]. Later, however, the graph neural networks become a more common approach as they can significantly assist trajectory forecasting by capturing the interactions among agents [3, 4, 5, 6, 12, 13, 14, 15]. For safety reasons, it is necessary to report the uncertainty of each predicted trajectory. Works to date about uncertainty measurement [11, 21, 22, 37, 38] have appropriately modeled the interaction among multi-agent trajectories for boosting performances, but they overlook the uncertainty resulting from the correlation in predicted trajectories. We seek to fill this gap by introducing and modeling *collaborative uncertainty*.

3 Methodology

3.1 Problem Formulation

Consider m agents in a data sample, and let $\mathrm{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$, $\mathrm{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_m\}$ be the past observed and the future trajectories of all agents, where $\mathbf{x}_i \in \mathbb{R}^{2T_-}$ and $\mathbf{y}_i \in \mathbb{R}^{2T_+}$ are the past observed and the future trajectories of the i-th agent. Each $\mathbf{x}_i/\mathbf{y}_i$ consists of two-dimensional coordinates at different timestamps of T_-/T_+. We assume that a training dataset $\mathcal{D}$ consists of N individual and identically distributed data samples $\{(\mathrm{X}^i, \mathrm{Y}^i)\}_{i=1}^N$. For predicting future trajectories of multiple agents and modeling the uncertainty over the predictions, we seek to use a probabilistic framework to model the predictive distribution $p(\mathrm{Y} \mid \mathrm{X})$ of multiple agents' future trajectories based on the training dataset $\mathcal{D}$. Previous works in uncertainty modeling [16, 17, 39] use Gaussian distribution to approximate $p(\mathrm{Y} \mid \mathrm{X})$.



(a) Individual Model (b) Collaborative Model

Figure 2: **Graphical model for deep learning networks in the three-agent trajectory forecasting setting**: (a) represents the model that predicts the trajectory of each agent independently; (b) shows the model that explicitly captures the interaction among multiple agents. $\mathbf{x}_i$ is the observed trajectory of the i-th agent; $\mathbf{h}_i$ and $\mathbf{y}_i$ are its corresponding hidden feature and future trajectory respectively.

The assumption behind this approach is that $p(\mathbf{y}_i|\mathbf{x}_i)$ is independent for every $i \in \{1, 2, 3, ..., m\}$. Mathematically, they set the covariance matrix of $p(\mathrm{Y} \mid \mathrm{X})$ as a diagonal matrix. This assumption is valid for the regression task that uses the model shown in Figure 2(a). We refer the uncertainty under the independence assumption as *individual uncertainty* in this paper. However, Figure 2(b) considers a prediction model that includes interaction modeling among multiple agents: $\mathbf{y}_i$ is no longer dependent solely on $\mathbf{x}_i$, but also on other agents $\mathbf{x}_j$ where $j \neq i$ in the scene. We call the uncertainty brought by this interaction *collaborative uncertainty*. The existence of collaborative uncertainty turns $p(\mathrm{Y}\,|\,\mathrm{X})$ from the individual distribution into the joint distribution of multiple agents.

Contrary to existing methods, we consider collaborative uncertainty and model $p(\mathrm{Y}\,|\,\mathrm{X})$ more accurately by making the covariance matrix a full matrix without imposing any restrictions on its form. In the following subsection, we will introduce an approach to modeling both individual uncertainty and collaborative uncertainty using a unified CU-based framework.

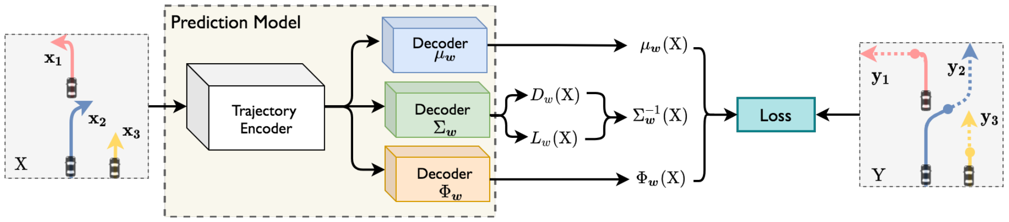


Figure 3: **Proposed uncertainty estimation framework.** The encoder may contain a module that exploits agent-wise interaction. Decoders output the mean $\mu_{\boldsymbol{w}}(\mathrm{X})$, covariance $\Sigma_{\boldsymbol{w}}(\mathrm{X})$ containing individual and collaborative uncertainty, and auxiliary parameters $\Phi_{\boldsymbol{w}}(\mathrm{X})$. The outputs formulate the training loss with the ground truth Y. $\Phi_{\boldsymbol{w}}(\cdot)$ is only used in Laplace collaborative uncertainty.

3.2 General Formulation of Collaborative Uncertainty

In this work, to model collaborative uncertainty, we abandon the independence assumption held by previous works [16, 17, 39], setting $p(\mathrm{Y}\,|\,\mathrm{X})$ as a joint multivariate distribution, whose mean is $\mu \in \mathbb{R}^{m\times 2T_+}$ and covariance $\Sigma \in \mathbb{R}^{m\times m\times 2T_+}$. Element $\mu_{i,t}$ is the expected position of the i-th agent at timestamp t. As the diagonal elements of Σ are considered individual uncertainty [16, 17, 39], we further let *off-diagonal* elements describe collaborative uncertainty. Diagonal element $\Sigma_{i,i,t}$ models the variance of the i-th agent at timestamp t; off-diagonal element $\Sigma_{i,j,t}$ models the covariance between the i-th and j-th agents at timestamp t. Therefore, we can simultaneously obtain individual and collaborative uncertainty by estimating the covariance Σ of $p(\mathrm{Y}\,|\,\mathrm{X})$. Accordingly, we propose a CU-based comprehensive uncertainty estimation framework (see Figure 3) with the following steps:

Step 1: *Choose a probability density function,* $p(\mathrm{Y}\,|\,\mathrm{X};\mu,\Sigma,\Phi)$, for the predictive distribution, which includes a mean $\mu\in\mathbb{R}^{m\times 2T_+}$ used to approximate the future trajectories, a covariance $\Sigma\in\mathbb{R}^{m\times m\times 2T_+}$ used to quantify individual uncertainty and collaborative uncertainty, and some auxiliary parameters Φ used to describe the predictive distribution. Further, we set covariance matrix Σ_t, which represents the covariance matrix at timestamp t, as a full matrix instead of an identity or diagonal matrix.

Step 2: *Design a prediction model,* $\mathcal{F}[\mu_{\boldsymbol{w}}(\mathrm{X}),\Sigma_{\boldsymbol{w}}(\mathrm{X}),\Phi_{\boldsymbol{w}}(\mathrm{X})]$, where $\mu_{\boldsymbol{w}}(\mathrm{X})$, $\Sigma_{\boldsymbol{w}}(\mathrm{X})$ and $\Phi_{\boldsymbol{w}}(\mathrm{X})$ are three neural networks, which approximate values of mean μ, covariance Σ and auxiliary parameters Φ respectively. Note that $\boldsymbol{w}$ only indicates the parameters of these neural networks are trainable, and does not mean they share same parameters.

Step 3: *Derive a loss function* from $p(\mathrm{Y}\,|\,\mathrm{X};\mu,\Sigma,\Phi)$ via maximum likelihood estimation: $\mathcal{L}(\boldsymbol{w}) = -\sum_{i=1}^{N}\log p(\mathrm{Y}^i\,|\,\mathrm{X}^i;\mu_{\boldsymbol{w}}(\mathrm{X}^i),\Sigma_{\boldsymbol{w}}(\mathrm{X}^i),\Phi_{\boldsymbol{w}}(\mathrm{X}^i))$ minimized to update trainable parameters in $\mu_{\boldsymbol{w}}(\cdot)$, $\Sigma_{\boldsymbol{w}}(\cdot)$ and $\Phi_{\boldsymbol{w}}(\cdot)$.

3.3 Two Special Cases

Table 1: Two special cases with various assumptions about covariance Σ. **DIA**: the diagonal matrix (individual uncertainty only). **FULL**: the full matrix (both individual and collaborative uncertainty).

Assumption	Loss Function of Two Special Cases	
	Gaussian Distribution	Laplace Distribution
DIA: $\begin{pmatrix}\sigma_{11} & 0 & \cdots & 0\\ 0 & \sigma_{22} & \cdots & 0\\ \vdots & \vdots & & \vdots\\ 0 & 0 & \cdots & \sigma_{mm}\end{pmatrix}$	$\frac{1}{2}\sum_{i=1}^{m}[\sigma_{ii}^{-2}\lVert\mathbf{y}_i-\mu_{\boldsymbol{w}}(\mathbf{x}_i)\rVert_2^2+\log\sigma_{ii}^2]$	$\sum_{i=1}^{m}[\sigma_{ii}^{-2}\lVert\mathbf{y}_i-\mu_{\boldsymbol{w}}(\mathbf{x}_i)\rVert_1+\log\sigma_{ii}^2]$
FULL: $\begin{pmatrix}\sigma_{11} & \sigma_{12} & \cdots & \sigma_{1m}\\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2m}\\ \vdots & \vdots & & \vdots\\ \sigma_{m1} & \sigma_{m2} & \cdots & \sigma_{mm}\end{pmatrix}$	$\frac{1}{2}[q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X})-\sum_{j=1}^{m}\log(d_{jj})]$	$\frac{1}{2}[\frac{q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X})}{\Phi_{\boldsymbol{w}}(\mathrm{X})}+m\log\Phi_{\boldsymbol{w}}(\mathrm{X})-\sum_{j=1}^{m}\log(d_{jj})]$

In multi-agent trajectory forecasting, based on Laplace and Gaussian distributions, the ℓ_1- and ℓ_2-based loss functions are commonly adopted to train prediction models [3, 4, 15, 40]. Here we apply the probabilistic framework proposed in Section 3.2 to model the individual and collaborative uncertainty based on multivariate Gaussian distribution and multivariate Laplace distribution respectively, which leads to two novel loss functions. Mathematically, the essential difference between our proposed loss functions and previous loss functions derived from Gaussian distribution and Laplace distribution for modeling individual uncertainty is that they have different assumptions about the covariance matrix; see a summary in Table 1. We regard the covariance as a full matrix.

3.3.1 Gaussian collaborative uncertainty

We start by the multivariate Gaussian distribution, as it has a simpler probability density function than the multivariate Laplace distribution.

Probability density function. We follow the framework proposed in Section 3.2 and choose the probability density function as the multivariate Gaussian distribution:

$$p(\mathrm{Y}\,|\,\mathrm{X};\mu,\Sigma,\Phi) = (2\pi)^{-\frac{m}{2}} \cdot \det[\Sigma]^{-\frac{1}{2}} \cdot e^{-\frac{1}{2}(\mathrm{Y}-\mu)\Sigma^{-1}(\mathrm{Y}-\mu)^T}, \tag{1}$$

where $\det[\Sigma]$ represents the determinant of covariance Σ.

Model design. Based on (1), we can approximate the value of mean μ via a neural network $\mu_{\boldsymbol{w}}(\cdot)$. When using the same way to approximate the value of covariance Σ, however, we face two challenges: 1) each covariance matrix Σ_t in covariance Σ needs to be inverted, which could lead to numerical instability; 2) it is computationally expensive and numerically unstable to compute the determinant of each covariance matrix Σ_t in covariance Σ directly given a large amount of trainable parameters.

For the first challenge, we use a neural network $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$ to directly approximate the inverse of covariance Σ. For the second challenge, similar to [29] and [32], we apply the square-root-free Cholesky decomposition to each $\Sigma_{t\boldsymbol{w}}^{-1}$ in $\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})$: $\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X}) = L_{\boldsymbol{w}}(\mathrm{X}) D_{\boldsymbol{w}}(\mathrm{X}) L_{\boldsymbol{w}}^T(\mathrm{X})$,

where $L_{\boldsymbol{w}}(\mathrm{X})$ is a lower unit triangular matrix and $D_{\boldsymbol{w}}(\mathrm{X})$ is a diagonal matrix. Then, the determinant of the inverse of covariance Σ^{-1} is obtained by $\prod_{j=1}^{m} d_{jj}$, where d_{jj} is the j-th diagonal element in $D_{\boldsymbol{w}}(\mathrm{X})$. We can thus get the parameterized form of (1) as: $\boxed{p(\mathrm{Y}|\mathrm{X};\boldsymbol{w}) = (2\pi)^{-\frac{m}{2}} (\prod_{j=1}^{m} d_{jj})^{\frac{1}{2}} e^{-\frac{q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X})}{2}}}$,

where $q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X}) = [\mathrm{Y}-\mu_{\boldsymbol{w}}(\mathrm{X})]\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})[\mathrm{Y}-\mu_{\boldsymbol{w}}(\mathrm{X})]^T$.

As there are no auxiliary parameters in the parameterized form of (1), we can get the prediction model $\mathcal{F}[\mu_{\boldsymbol{w}}(\mathrm{X}), \Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})]$, whose framework is illustrated in Figure 3. Once $\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})$ is fixed and given, individual and collaborative uncertainty are computed through the inversion.

Loss function. According to the square-root-free Cholesky decomposition and the parameterized form of (1), the Gaussian collaborative uncertainty loss function is then:

$$\mathcal{L}_{\text{Gau-cu}}(\boldsymbol{w}) = \frac{1}{2}\frac{1}{N}\sum_{i=1}^{N}[q_{\boldsymbol{w}}(\mathrm{Y}^i, \mathrm{X}^i) - \sum_{j=1}^{m}\log(d_{jj}^i)]. \tag{2}$$

We update the trainable parameters in $\mu_{\boldsymbol{w}}(\cdot)$ and $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$ through minimizing (2). Note that $q_{\boldsymbol{w}}(\cdot,\cdot)$ is related to $\mu_{\boldsymbol{w}}(\cdot)$ and $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$, and d_{jj}^i is related to $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$.

3.3.2 Laplace collaborative uncertainty

In multi-agent trajectory forecasting, previous methods [3, 4, 15] have found that the ℓ_1-based loss function derived from Laplace distribution usually leads to better prediction performances than the ℓ_2-based loss function from Gaussian distribution, because the former is more robust to outliers. It is thus important to consider multivariate Laplace distribution.

Probability density function. We follow the framework proposed in Section 3.2 and choose the probability density function as the multivariate Laplace distribution:

$$p(\mathrm{Y}\,|\,\mathrm{X};\mu,\Sigma,\Phi) = \frac{2\det[\Sigma]^{-\frac{1}{2}}}{(2\pi)^{\frac{m}{2}}\lambda} \cdot \frac{K_{(\frac{m}{2}-1)}(\sqrt{\frac{2}{\lambda}(\mathrm{Y}-\mu)\Sigma^{-1}(\mathrm{Y}-\mu)^T})}{(\sqrt{\frac{\lambda}{2}(\mathrm{Y}-\mu)\Sigma^{-1}(\mathrm{Y}-\mu)^T})^{\frac{m}{2}}}, \tag{3}$$

where $\det[\Sigma]$ denotes the determinant of covariance Σ, and $K_{(\frac{m}{2}-1)}(\cdot)$ denotes the modified Bessel function of the second kind with order $(\frac{m}{2}-1)$.

Model design. Similar to Section 3.3.1, we employ two neural networks $\mu_{\boldsymbol{w}}(\cdot)$ and $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$ to approximate the values of μ and Σ^{-1} respectively, and represent $\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})$ via its square-root-free Cholesky decomposition we used in the Gaussian collaborative uncertainty. Since the modified Bessel function is intractable for a neural network to work with, different from Section 3.3.1, we should simplify (3).

Inspired by [41], we simplify (3) by utilizing the multivariate Gaussian distribution to approximate the multivariate Laplace Distribution. We reformulate a multivariate Laplace distribution by introducing auxiliary variables. Let $\mathbf{z} \in \mathbb{R}^+$ be a random variable with the probability density function: $p(\mathbf{z}|\mathrm{X};\boldsymbol{w}) = \frac{1}{\lambda}e^{-\frac{\mathbf{z}}{\lambda}}$, then we can get: $p(\mathrm{Y}|\mathbf{z},\mathrm{X};\boldsymbol{w}) = \frac{\det[\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})]^{\frac{1}{2}}}{(2\pi\mathbf{z})^{\frac{m}{2}}}e^{-\frac{q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X})}{2\mathbf{z}}}$, where $q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X}) = [\mathrm{Y}-\mu_{\boldsymbol{w}}(\mathrm{X})]\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})[\mathrm{Y}-\mu_{\boldsymbol{w}}(\mathrm{X})]^T$. Further, if the value of $\mathbf{z}$ is given, $p(\mathrm{Y}|\mathbf{z},\mathrm{X};\boldsymbol{w})$ is a multivariate Gaussian distribution. In this work, instead of drawing a value for $\mathbf{z}$ from the exponential distribution, we use a neural network $\Phi_{\boldsymbol{w}}(\cdot)$ to directly output a value for $\mathbf{z}$. The intuition is that, in the training process of the prediction model, the value of $p(\mathrm{Y}|\mathrm{X};\boldsymbol{w})$ is the conditional expectation of $\mathbf{z}$ given X and Y, which makes $p(\mathrm{Y}|\mathbf{z},\mathrm{X};w)$ a function of $\mathbf{z}$ whose domain is $\mathbb{R}^+$. Thus, there should exist an appropriate $\mathbf{z}^* \in \mathbb{R}^+$ to make: $p(\mathrm{Y}|\mathrm{X};\boldsymbol{w}) = p(\mathrm{Y}|\mathbf{z}^*,\mathrm{X};\boldsymbol{w})$ (see proof in the appendix). To find such a $\mathbf{z}^*$, we use $\Phi_{\boldsymbol{w}}(\mathrm{X})$, which can employ its learning ability. Then, we can get the parameterized form of $p(\mathrm{Y}|\mathrm{X};\boldsymbol{w})$ as: $p(\mathrm{Y}|\mathrm{X};\boldsymbol{w}) = \frac{\det[\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})]^{\frac{1}{2}}}{(2\pi\Phi_{\boldsymbol{w}}(\mathrm{X}))^{\frac{m}{2}}}e^{-\frac{q_{\boldsymbol{w}}(\mathrm{Y},\mathrm{X})}{2\Phi_{\boldsymbol{w}}(\mathrm{X})}}$.

Finally, we can get the prediction model $\mathcal{F}[\mu_{\boldsymbol{w}}(\mathrm{X}),\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X}),\Phi_{\boldsymbol{w}}(\mathrm{X})]$, whose framework is illustrated in Figure 3. Individual and collaborative uncertainty are indirectly learned by the $\Sigma_{\boldsymbol{w}}^{-1}(\mathrm{X})$.

Loss function. On the basis of the square-root-free Cholesky decomposition and the parameterized form of $p(\mathrm{Y}|\mathrm{X};\boldsymbol{w})$, the Laplace collaborative uncertainty loss function is then:

$$\mathcal{L}_{\mathrm{Lap-cu}}(\boldsymbol{w}) = \frac{1}{2}\frac{1}{N}\sum_{i=1}^{N}[\frac{q_{\boldsymbol{w}}(\mathrm{Y}^i,\mathrm{X}^i)}{\Phi_{\boldsymbol{w}}(\mathrm{X}^i)} + m\log\Phi_{\boldsymbol{w}}(\mathrm{X}^i) - \sum_{j=1}^{m}\log(d_{jj}^i)]. \tag{4}$$

where d_{jj}^i is the j-th diagonal element in $D_{\boldsymbol{w}}(\mathrm{X}^i)$. The parameters of $\mu_{\boldsymbol{w}}(\cdot)$, $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$ and $\Phi_{\boldsymbol{w}}(\cdot)$ are updated by minimizing (4). And $q_{\boldsymbol{w}}(\cdot,\cdot)$ is related to $\mu_{\boldsymbol{w}}(\cdot)$ and $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$, and d_{jj}^i belongs to $\Sigma_{\boldsymbol{w}}^{-1}(\cdot)$.

3.4 Discussion

After presenting how to quantify collaborative uncertainty in Section 3.2 and Section 3.3, here we discuss the nature of collaborative uncertainty.

As mentioned in Section 3.2, we can divide a prediction model for multi-agent trajectory forecasting into two types: individual models and collaborative models. An individual model predicts the future trajectory and the corresponding uncertainty for each agent independently; a collaborative model leverages an interaction module to explicitly capture the interactions among multiple agents, which makes all the predicted trajectories correlated. Moreover, this interaction modeling procedure can bring extra uncertainty to the model; in other words, we consider that the interaction modeling in a prediction model leads to collaborative uncertainty (CU).

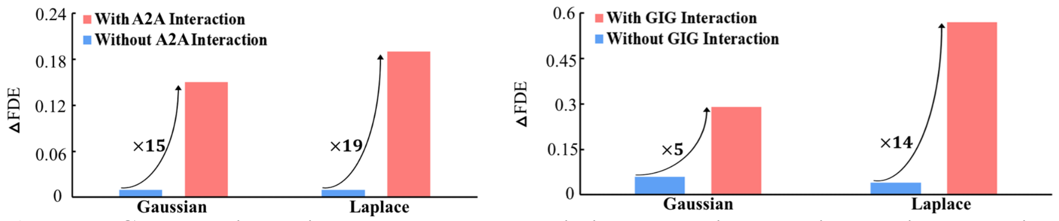


Figure 4: **CU modeling gains more when a prediction model includes interaction modeling.** Blue and red bars reflect the gains from CU modeling for a prediction model w/o an interaction module respectively. ΔFDE is the difference of FDE values between versions w/o CU modeling.

To validate this, we can empirically compare how much impact of collaborative uncertainty modeling would bring to an individual model and a collaborative model. In our experiment, we use two cutting-edge multi-agent trajectory forecasting models, LaneGCN [3] and VectorNet [4].[2] In LaneGCN/VectorNet, an agent-to-agent (A2A) module/a global-interaction-graph (GIG) module explicitly models the interactions among multiple agents. As illustrated in Figure 4, when we remove the A2A/GIG module from LaneGCN/VectorNet, collaborative uncertainty modeling brings much less gain to LaneGCN/VectorNet; see Section 4.2.3 for more details of this experiment. This reflects that the cause of collaborative uncertainty mainly comes from the interaction modeling.

4 Experiments

We first use two self-generated synthetic datasets with a limited number of agents as the toy version of the real world problem. We use the simplified datasets to test our method's ability of capturing distribution information of the input data that obeys a certain type of multivariate distribution. We then conduct extensive experiments on two published benchmarks to prove the value of our proposed method in solving real world problems. We introduce the experiments in Sec. 4.1 and Sec. 4.2.

4.1 Toy Problem

We define a toy problem to validate the capability of the proposed framework for accurate probability distribution estimation. The toy problem requires models to take the mutually correlated trajectories sampled from a given distribution as the input and output the mean and the covariance of the distribution.

As far as we know, in real-world datasets, we only have the ground truth for the predicted trajectory, which is the mean of the distribution while have no access to the ground truth of the uncertainty, which is the covariance matrix of the distribution. Therefore, we generate two synthetic datasets with the ground truth for both mean and covariance matrix given two distributions respectively.

Datasets. We generate two synthetic datasets for ternary Gaussian and ternary Laplace distribution respectively. Each dataset contains training, validation and test sets, which have 36000, 7000 and 7000 instances respectively. Each instance includes the trajectories of three agents, which consist of the two-dimensional point coordinates of the three agents at 50 different timestamp. In each instance, the trajectories of three agents are sampled from a ternary joint Gaussian/Laplace distribution. Generation details are provided in the appendix.

Implementation details. The network architecture used in the experiment contains an encoder and two decoders, all of which are four-layer MLPs. The neural network outputs the predicted mean and covariance matrix of the given ternary Gaussian/Laplace distribution. Although the ground truth covariance matrix is known for synthetic datasets, they are not used in training. We train the network using the previous uncertainty modeling method and our proposed method on each synthetic dataset.

Metric. Here, we adopt three metrics for evaluation: the ℓ_2 distances between the estimated mean and the ground truth mean, the ℓ_1 distances between the estimated covariance matrix and the ground truth covariance matrix and the KL divergence between the ground truth distribution and the estimated distribution. We provide metrics computing details in the appendix.

Evaluation results. The test set results of the synthetic dataset are shown in Table 2 and Figure 5. Since the previous uncertainty modeling method only models IU (individual uncertainty) but our method models both IU and CU (collborative uncertainty), our method allows the model to estimate a more accurate mean and covariance matrix on a given distribution, which leads to a much lower KL divergence between the ground truth distribution and the estimated distribution.

4.2 Real World Problem

The experiments results above show the superiority of our method for simplified problems. Here we further validate its ability on two real-world large-scale autonomous driving datasets in single future prediction. We also conduct ablation studies in this subsection to present the effectiveness of the method design.

4.2.1 Experiment setup

Datasets. *Argoverse* [42] and *nuScenes* [47] are two widely used multi-agent trajectory forecasting benchmarks. *Argoverse* has over 30K scenarios collected in Pittsburgh and Miami. Each scenario is a sequence of frames sampled at 10 Hz. The sequences are split as training, validation and test sets, which have 205942, 39472 and 78143 sequences respectively. *nuScenes* collects 1000 scenes in

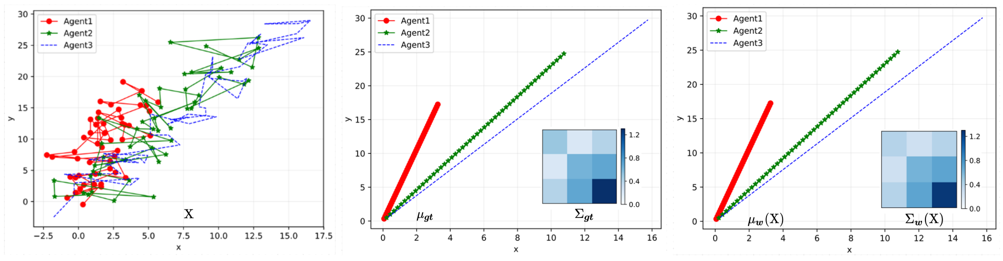
[2]To focus on agents' behaviors, we remove the map information and the related modules in this experiment.

Table 2: Comparison with the prior uncertainty modeling method on synthetic datasets under the two different assumptions of distribution. $\mu_{\boldsymbol{w}}(\mathrm{X})$: the estimated mean. μ_{gt}: the ground truth mean. $\Sigma_{\boldsymbol{w}}(\mathrm{X})$: the estimated covariance matrix. Σ_{gt}: the ground truth covariance matrix. KL: the KL divergence $D_{KL}(p_g(\mathrm{X})\|p_e(\mathrm{X}))$, where $p_e(\mathrm{X})$ is the estimated distribution and $p_g(\mathrm{X})$ is the ground truth distribution.

	Gaussian		Laplace	
	IU Only	IU + CU	IU Only	IU + CU
ℓ_2 of μ	0.68	**0.49**	0.42	**0.34**
ℓ_1 of Σ	1.98	**1.01**	1.96	**1.13**
KL	6.68	**0.40**	12.6	**1.65**

Table 3: Comparison with SOTA methods on Argoverse test set. CU boosts performances in single future prediction.

Method	ADE	FDE
Argo Baseline (NN) [42]	3.45	7.88
Argo Baseline [42]	2.96	6.81
uulm-mrm [43, 44]	1.90	4.19
VectorNet [4]	1.81	4.01
TNT [45]	1.78	3.91
Jean [46]	1.74	4.24
LaneGCN [3]	1.71	3.78
Ours:		
LaneGCN (Our Implementation)	1.76	3.84
Using $\mathcal{L}_{\text{Gau-cu}}$	1.73	3.83
Using $\mathcal{L}_{\text{Lap-cu}}$	**1.70**	**3.74**


(a) Gaussian Synthetic Dataset

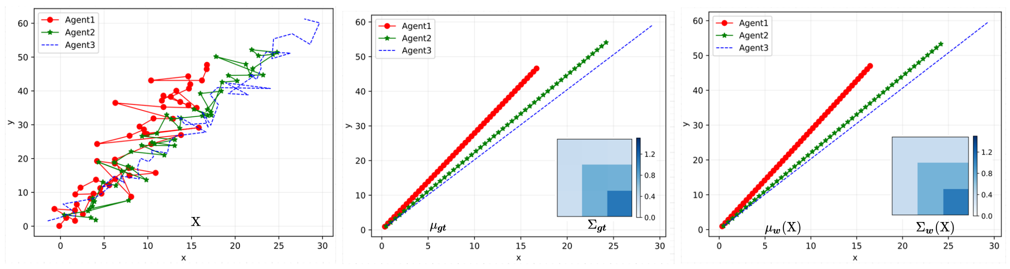

(b) Laplace Synthetic Dataset

Figure 5: **Sample visualization on synthetic dataset.** Our proposed CU-based framework allows the model to learn the mean and covariance matrix of ground truth distribution accurately. Each input instance X is the numerical sum of a ground truth mean μ_{gt} and a random variable ϵ with the information of the ground truth covariance matrix Σ_{gt}. Please find more details in Appendix.

Boston and Singapore. Each scene is annotated at 2 Hz and is 20s long. The prediction instances are split as training, validation and test sets, which have 32186, 8560 and 9041 instances respectively. For Argoverse, we forecast future trajectories for 3s based on the observed trajectories of 2s. For nuScenes, we forecast future trajectories for 6s based on the observed trajectories of 2s.

Metrics. We adopt two extensively used multi-agent trajectory forecasting metrics, Average Displacement Error (ADE) and Final Displacement Error (FDE). ADE is defined as the average of pointwise ℓ_2 distances between the predicted trajectory and ground truth. FDE is defined as the ℓ_2 distance between the final points of the prediction and ground truth (ADE and FDE are measured in meters).

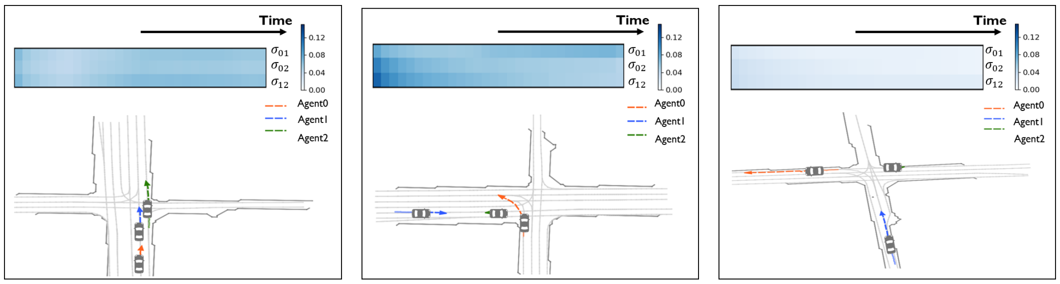
Implementation details. The proposed model is implemented follow the basis of LaneGCN [3] and VectorNet [4], two cutting-edge multi-agent trajectory forecasting models. We implement the encoder of LaneGCN/VectorNet as the trajectory encoder in our proposed model. We further use the four-layer multilayer perceptrons (MLPs) to respectively implement three decoders $\mu_{\boldsymbol{w}}(\cdot)$, $\Sigma_{\boldsymbol{w}}(\cdot)$ and $\Phi_{\boldsymbol{w}}(\cdot)$. Note that all experiments are based on our own implementation of LaneGCN/VectorNet, which may not perform as well as the official LaneGCN/VectorNet.

Table 4: Ablation on covariance Σ with chosen probability density function (**PDF**) and Interaction module (**INT.**). **DIA** denotes the diagonal matrix (individual uncertainty). **FULL** denotes the full matrix (individual and collaborative uncertainty). On Argoverse and nuScenes, both LaneGCN and VectorNet with individual and collaborative uncertainty surpasses the ones with individual uncertainty only. Collaborative uncertainty makes a larger impact for the model with an interaction module.

METHOD	DATASET	PDF TYPE	INT.	ADE		ΔADE	FDE		ΔFDE
				DIA	FULL		DIA	FULL	
LANEGCN	ARGOVERSE	GAUSSIAN	×	1.69	1.67	0.02	3.88	3.85	0.03
			√	1.45	1.42	**0.03**	3.19	3.14	**0.05**
		LAPLACE	×	1.67	1.67	0.00	3.82	3.82	0.00
			√	1.43	**1.41**	**0.02**	3.16	**3.11**	**0.05**
	NUSCENES	GAUSSIAN	×	4.60	4.58	0.02	11.02	11.01	0.01
			√	4.47	4.39	**0.08**	10.59	10.44	**0.15**
		LAPLACE	×	4.53	4.52	0.01	10.93	10.92	0.01
			√	4.34	**4.25**	**0.09**	10.34	**10.15**	**0.19**
VECTORNET	ARGOVERSE	GAUSSIAN	×	1.82	1.78	0.04	4.16	4.08	0.08
			√	1.63	1.57	**0.06**	3.60	3.46	**0.14**
		LAPLACE	×	1.78	1.76	0.02	4.06	4.02	0.04
			√	1.56	**1.52**	**0.04**	3.42	**3.34**	**0.08**
	NUSCENES	GAUSSIAN	×	4.25	4.23	0.02	10.35	10.29	0.06
			√	4.07	3.99	**0.08**	9.86	9.57	**0.29**
		LAPLACE	×	4.19	4.18	0.01	10.25	10.21	0.04
			√	4.02	**3.81**	**0.21**	9.79	**9.22**	**0.57**

4.2.2 Results

Evaluation results on benchmark datasets. In this subsection, we implement our proposed framework based on LaneGCN [3] as it is the SOTA model in multi-agent trajectory forecasting. We compare our proposed methods in the Argoverse trajectory forecasting benchmark with two official baselines, including Argoverse Baseline [42] and Argoverse Baseline (NN) [42], and five SOTA methods of this benchmark: LaneGCN [3], TNT [45], Jean [46], VectorNet [4] and uulm-mrm [43, 44]. Table 3 shows that although our implementation of LaneGCN appears less good than the official implementation of LaneGCN, our implementation of LaneGCN with the individual and collaborative uncertainty notably outperforms all of the other competing methods in both ADE and FDE. Therefore, our proposed collaborative uncertainty modeling enhances the SOTA prediction models.



(a) Scenario I (b) Scenario II (c) Scenario III

Figure 6: **Visualization of CU on Argoverse dataset**. (a) In scenario I, Agent0 and Agent1 are driving on the same road, which is next to the road where Agent2 is driving. This type of scenario might generate complicated interactive information making σ_{01}, σ_{02} and σ_{12} show a non-monotonic change over time. (b) In scenario II, Agent0 and Agent1 are moving towards each other, thus σ_{01} increases over time. Agent2 is parking at an intersection waiting for green light, as little new interactive information between Agent2 and the other two agents would be generated before the red light turns green, σ_{02} and σ_{12} decrease over time. (c) In scenario III, Agent0, Agent1 and Agent2 are located in completely different areas heading to different directions, σ_{01}, σ_{02} and σ_{12} are close to zero.

Visualization of collaborative uncertainty.[3] To visually understand which factor might influence the value of collaborative uncertainty in multi-agent trajectory forecasting, we present the visualization results generated by our model in Figure 6. We visualize 3 scenarios, and each scenario includes 3 agents (orange, blue and green lines) trajectories (solid lines are the past trajectories and dashed lines are the future trajectories) and their corresponding collaborative uncertainty values changing over the last 30 frames (the heatmap, where σ_{ij} denotes the collaborative uncertainty of agent i and agent j). These results show that the value of collaborative uncertainty is highly related to the amount of the interactive information among agents.

4.2.3 Ablation study

We study: 1) how different approaches of uncertainty modeling would affect the prediction model; 2) how the interaction module would influence the effectiveness of collaborative uncertainty modeling. In this part, we adopt LaneGCN/VectorNet as our trajectory encoder for proving that our proposed method can be used as a plug-in module to improve the performances of existing models in multi-agent trajectory forecasting. To focus on the agents' behaviors, we ignore map information and map-related modules in LaneGCN/VectorNet; we only use the agent encoder of LaneGCN/VectorNet, which extracts intermediate feature from agents' past trajectories and actor-to-actor (A2A) module/global-interaction-graph (GIG) module that exploits the interactions among agents. In this part, the experiments are conducted on the validate sets of Argoverse and nuScenes benchmarks.

Different approaches of uncertainty modeling. We consider two approaches of uncertainty modeling: assuming the covariance matrix is a diagonal matrix (DIA) to model individual uncertainty only, and assuming the covariance matrix is a full matrix (FULL) to model both individual and collaborative uncertainty. Based on the results in Table 4, we see that 1) modeling individual and collaborative uncertainty together (FULL) is consistently better than modeling individual uncertainty only (DIA); and 2) our proposed Laplace CU-based framework enables LaneGCN and VectorNet to achieve the best performances on ADE & FDE metrics on both Argoverse and nuScenes benchmarks. These results reflect that our proposed collaborative uncertainty modeling can work as a plugin module to significantly improve the prediction performance of existing models.

Effects of interaction module. A2A/GIG module is the interaction module in LaneGCN/VectorNet. We study the effects of the interaction module in collaborative uncertainty modeling via the ablation study on A2A/GIG module. ΔADE/FDE is the difference of ADE/FDE values between versions with and without collaborative uncertainty. Higher values reflect bigger gains brought by modeling collaborative uncertainty. From Table 4, gains from modeling collaborative uncertainty in a collaborative model (with A2A/GIG) are much greater than in an individual model (without A2A/GIG). Figure 4 visualizes the ΔFDE on nuScenes benchmark; see discussions in Section 3.4.

5 Conclusions

This work proposes a novel probabilistic collaborative uncertainty (CU)-based framework for multi-agent trajectory forecasting. Its key novelty is conceptualizing and modeling CU introduced by interaction modeling. The experimental results in this work demonstrate the ability of the proposed CU-based framework to boost the performance of SOTA prediction systems, especially collaborative-model-based systems. The proposed framework potentially lead to more reliable self-driving systems.

This work shows the promise of our method for single-future trajectory prediction, which predicts the single best trajectory for agents. With growing research interests in predicting multiple potential trajectories for agents (i.e. multi-future trajectory prediction), we leave the generalization of our method to multi-future trajectory prediction as an important direction for future research.

Acknowledgement

This work was supported in part by the National Key R&D Program of China under Grant 2021YFB1714800, the National Natural Science Foundation of China under Grant 6217010074, 62173034, and the Science and Technology Commission of Shanghai Municipal under Grant 21511100900.

[3]These visualization results are based on Gaussian CU. Please see more visualization results in the appendix.

References

[1] Patrick Stahl, Birsen Donmez, and Greg A. Jamieson. Anticipation in driving: The role of experience in the efficacy of pre-event conflict cues. *IEEE Trans. Hum. Mach. Syst.*, 44(5):603–613, 2014.

[2] Patrick Stahl, Birsen Donmez, and Greg A Jamieson. Supporting anticipation in driving through attentional and interpretational in-vehicle displays. *Accident Analysis and Prevention*, 91(jun.):103–113, 2016.

[3] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *ECCV*, 2020.

[4] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[5] Maosheng Ye, Tongyi Cao, and Qifeng Chen. Tpcn: Temporal point cloud networks for motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11318–11327, June 2021.

[6] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. Home: Heatmap output for future motion estimation, 2021.

[7] Ke Xiao, Jianyu Zhao, Yunhua He, and Shui Yu. Trajectory prediction of UAV in smart city using recurrent neural networks. In *2019 IEEE International Conference on Communications, ICC 2019, Shanghai, China, May 20-24, 2019*, pages 1–6. IEEE, 2019.

[8] Nikolay Jetchev and Marc Toussaint. Trajectory prediction: learning to map situations to robot trajectories. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 449–456. ACM, 2009.

[9] Christoph Rösmann, Malte Oeljeklaus, Frank Hoffmann, and Torsten Bertram. Online trajectory prediction and planning for social robot navigation. In *IEEE International Conference on Advanced Intelligent Mechatronics, AIM 2017, Munich, Germany, July 3-7, 2017*, pages 1255–1260. IEEE, 2017.

[10] Tianyang Zhao, Yifei Xu, Mathew Monfort, Wongun Choi, Chris Baker, Yibiao Zhao, Yizhou Wang, and Ying Nian Wu. Multi-agent tensor fusion for contextual trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[11] Chiho Choi and Behzad Dariush. Looking to relations for future trajectory forecast. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[12] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. *arXiv preprint arXiv:2001.03093*, 2020.

[13] Wenyuan Zeng, Ming Liang, Renjie Liao, and Raquel Urtasun. Lanercnn: Distributed representations for graph-centric motion forecasting. *arXiv preprint arXiv:2101.06653*, 2021.

[14] Jiachen Li, Fan Yang, Masayoshi Tomizuka, and Chiho Choi. Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning. *Proceedings of the Neural Information Processing Systems (NeurIPS)*, 2020.

[15] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian D. Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 137–146, 2019.

[16] Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

[17] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5574–5584. Curran Associates, Inc., 2017.

[18] Xujiang Zhao, Feng Chen, Shu Hu, and Jin-Hee Cho. Uncertainty aware semi-supervised learning on graph data, 2020.

[19] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112, 2009.

[20] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. Long-term on-board prediction of people in traffic scenes under uncertainty. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4194–4202. IEEE Computer Society, 2018.

[21] Ajay Jain, Sergio Casas, Renjie Liao, Yuwen Xiong, Song Feng, Sean Segal, and Raquel Urtasun. Discrete residual flow for probabilistic pedestrian behavior prediction. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*, volume 100 of *Proceedings of Machine Learning Research*, pages 407–419. PMLR, 2019.

[22] Joey Hong, Benjamin Sapp, and James Philbin. Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8454–8462. Computer Vision Foundation / IEEE, 2019.

[23] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7482–7491. IEEE Computer Society, 2018.

[24] Murat Seckin Ayhan and Philipp Berens. Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. 2018.

[25] Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1184–1193, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[26] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3266–3273. IEEE, 2018.

[27] Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 338:34–45, 2019.

[28] Lingkai Kong, Jimeng Sun, and Chao Zhang. SDE-net: Equipping deep neural networks with uncertainty estimates. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5405–5415. PMLR, 13–18 Jul 2020.

[29] G. Dorta, S. Vicente, L. Agapito, N. D. F. Campbell, and I. Simpson. Structured uncertainty prediction networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5477–5485, 2018.

[30] Miguel Monteiro, Loïc Le Folgoc, Daniel Coelho de Castro, Nick Pawlowski, Bernardo Marques, Konstantinos Kamnitsas, Mark van der Wilk, and Ben Glocker. Stochastic segmentation networks: Modelling spatially correlated aleatoric uncertainty. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[31] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models, 2021.

[32] Nitesh B. Gundavarapu, Divyansh Srivastava, Rahul Mitra, Abhishek Sharma, and Arjun Jain. Structured aleatoric uncertainty in human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 50–53. Computer Vision Foundation / IEEE, 2019.

[33] Roger Girgis, Florian Golemo, Felipe Codevilla, Jim Aldon D'Souza, Martin Weiss, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Autobots: Latent variable sequential set transformers, 2021.

[34] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Fei-Fei Li, and Silvio Savarese. Social LSTM: human trajectory prediction in crowded spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 961–971. IEEE Computer Society, 2016.

[35] Irtiza Hasan, Francesco Setti, Theodore Tsesmelis, Alessio Del Bue, Fabio Galasso, and Marco Cristani. MX-LSTM: mixing tracklets and vislets to jointly forecast trajectories and head poses. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6067–6076. IEEE Computer Society, 2018.

[36] Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G. Hauptmann, and Li Fei-Fei. Peeking into the future: Predicting future person activities and locations in videos. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5725–5734. Computer Vision Foundation / IEEE, 2019.

[37] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B. Choy, Philip H. S. Torr, and Manmohan Chandraker. DESIRE: distant future prediction in dynamic scenes with interacting agents. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2165–2174. IEEE Computer Society, 2017.

[38] Panna Felsen, Patrick Lucey, and Sujoy Ganguly. Where will they go? predicting fine-grained adversarial multi-agent motion using conditional variational autoencoders. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, volume 11215 of *Lecture Notes in Computer Science*, pages 761–776. Springer, 2018.

[39] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[40] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction, 2020.

[41] T. Eltoft, Taesu Kim, and Te-Won Lee. On the multivariate laplace distribution. *IEEE Signal Processing Letters*, 13(5):300–303, 2006.

[42] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[43] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pages 2090–2096. IEEE, 2019.

[44] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, and Jeff Schneider. Motion prediction of traffic actors for autonomous driving using deep convolutional networks. *CoRR*, abs/1808.05819, 2018.

[45] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, Congcong Li, and Dragomir Anguelov. TNT: target-driven trajectory prediction. *CoRR*, abs/2008.08294, 2020.

[46] Jean Mercat, Thomas Gilles, Nicole El Zoghby, Guillaume Sandou, Dominique Beauvois, and Guillermo Pita Gil. Multi-head attention for multi-modal joint vehicle motion forecasting. In *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*, pages 9638–9644. IEEE, 2020.

[47] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

[48] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Appendix

A Proof of Laplace Model Design

Proof. Consider the i-th data sample $(\mathrm{X}^i, \mathrm{Y}^i)$, as in the training process of the prediction model the values of X^i and Y^i are given, $p(\mathrm{Y}^i|\mathbf{z}^i, \mathrm{X}^i; \boldsymbol{w})$ is a function of $\mathbf{z}^i \in \mathbb{R}^+$ with the probability density function: $p(\mathbf{z}^i|\mathrm{X}^i) = \frac{1}{\lambda}e^{-\frac{\mathbf{z}^i}{\lambda}}$:

$$p(\mathrm{Y}^i|\mathbf{z}^i, \mathrm{X}^i; \boldsymbol{w}) = f_{\boldsymbol{w}}(\mathbf{z}^i) = \frac{1}{(\mathbf{z}^i)^{\frac{m}{2}}} e^{-\frac{g^i_{\boldsymbol{w}}}{\mathbf{z}^i}},$$

where $g^i_{\boldsymbol{w}} = \frac{1}{2}(\mathrm{Y}^i - \mu_{\boldsymbol{w}}(\mathrm{X}^i))[\Sigma^{-1}_{\boldsymbol{w}}(\mathrm{X}^i)](\mathrm{Y}^i - \mu_{\boldsymbol{w}}(\mathrm{X}^i))^T$ and $m \in \mathbb{N}^+$ is the number of the agents in the i-th data sample. We need to prove that there should exist a $(\mathbf{z}^i)^* \in \mathbb{R}^+$ to make:

$$\begin{aligned} p(\mathrm{Y}^i|\mathrm{X}^i; \boldsymbol{w}) &= \int_0^{+\infty} p(\mathrm{Y}^i|\mathbf{z}^i, \mathrm{X}^i; \boldsymbol{w}) p(\mathbf{z}^i|\mathrm{X}^i; \boldsymbol{w}) d\mathbf{z}^i \\ &= \int_0^{+\infty} f_{\boldsymbol{w}}(\mathbf{z}^i) p(\mathbf{z}^i|\mathrm{X}^i) d\mathbf{z}^i \\ &= E_{\mathbf{z}^i}[f_{\boldsymbol{w}}(\mathbf{z}^i)] \\ &= f_{\boldsymbol{w}}((\mathbf{z}^i)^*) \\ &= p(\mathrm{Y}^i|(\mathbf{z}^i)^*, \mathrm{X}^i; \boldsymbol{w}). \end{aligned}$$

And the existence of $(\mathbf{z}^i)^*$ can be proved by proving a fact that, when $\mathbf{z}^i \in \mathbb{R}^+$, $f(\mathbf{z}^i)$ is a continuous bounded function.

As the $g^i_{\boldsymbol{w}}$ can be reformulated as:

$$\begin{aligned} g^i_{\boldsymbol{w}} &= \frac{1}{2}(\mathrm{Y}^i - \mu_{\boldsymbol{w}}(\mathrm{X}^i))[L'_{\boldsymbol{w}}(\mathrm{X}) L'^T_{\boldsymbol{w}}(\mathrm{X})](\mathrm{Y}^i - \mu_{\boldsymbol{w}}(\mathrm{X}^i))^T \\ &= \frac{1}{2}[(\mathrm{Y}^i - \mu_{\boldsymbol{w}}(\mathrm{X}^i)) L'_{\boldsymbol{w}}(\mathrm{X})][(\mathrm{Y}^i - \mu_{\boldsymbol{w}}(\mathrm{X}^i)) L'_{\boldsymbol{w}}(\mathrm{X})]^T \\ &\geq 0, \end{aligned} \tag{5}$$

where $L'_{\boldsymbol{w}}(\mathrm{X})$ is a lower triangular matrix and the equal sign of (5) is only true when $\mu_{\boldsymbol{w}}(\mathrm{X}^i)$ is equal to Y^i, but in practice, $\mu_{\boldsymbol{w}}(\mathrm{X}^i)$ is hardly equal to Y^i, which means in the training process we have:

$$g^i_{\boldsymbol{w}} > 0. \tag{6}$$

Based on (6), let $\mathbf{s}^i = \frac{1}{\mathbf{z}^i}$, then as $\mathbf{z}^i \to 0^+$, we have $\mathbf{s}^i \to +\infty$, so for $\mathbf{z}^i \to 0^+$:

$$\begin{aligned} \lim_{\mathbf{z}^i \to 0^+} f_{\boldsymbol{w}}(\mathbf{z}^i) &= \lim_{\mathbf{s}^i \to +\infty} (\mathbf{s}^i)^{\frac{m}{2}} e^{-\mathbf{s}^i g^i_{\boldsymbol{w}}} \\ &= \lim_{\mathbf{s}^i \to +\infty} \frac{(\frac{m}{2})!}{(g^i_{\boldsymbol{w}})^{\frac{m}{2}} e^{\mathbf{s}^i g^i_{\boldsymbol{w}}}} \\ &= 0 \end{aligned} \tag{7}$$

For $\mathbf{z}^i \to +\infty$:

$$\lim_{\mathbf{z}^i \to +\infty} f_{\boldsymbol{w}}(\mathbf{z}^i) = 0 \tag{8}$$

Furthermore, as the derivative of $f_{\boldsymbol{w}}(\mathbf{z}^i)$ is then:

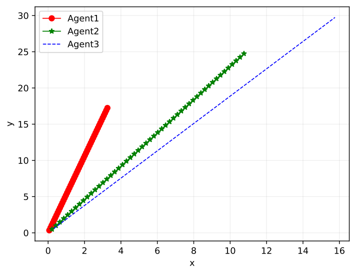
$$f'_{\boldsymbol{w}}(\mathbf{z}^i) = (\mathbf{z}^i)^{-\frac{m}{2}-2} \cdot e^{-(\mathbf{z}^i)^{-1} g^i_{\boldsymbol{w}}} \cdot (g^i_{\boldsymbol{w}} - \frac{m}{2}\mathbf{z}^i). \tag{9}$$

According to (6), (8), (7) and (9), when we set $f'_{\boldsymbol{w}}(\mathbf{z}^i) = 0$, we can get the maximum value of $f_{\boldsymbol{w}}(\mathbf{z}^i)$ is $f_{\boldsymbol{w}}(\frac{2g^i_{\boldsymbol{w}}}{m}) \in \mathbb{R}^+$.

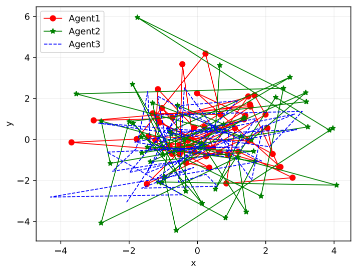
On the basis of above discussions, when $\mathbf{z}^i \in \mathbb{R}^+$, $f(\mathbf{z}^i)$ is a continuous bounded function, which means the $(\mathbf{z}^i)^*$ is existent. □

B Toy Problem

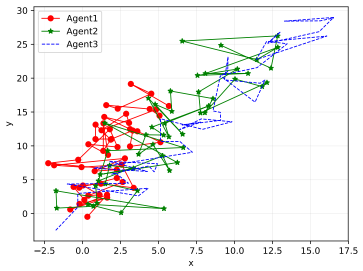
B.1 Generation Details of Synthetic Datasets



(a) μ_{gt}

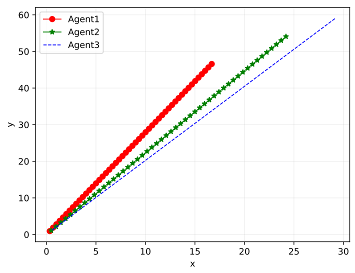


(b) ϵ

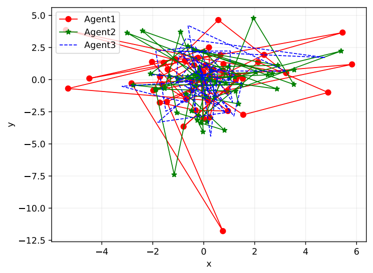


(c) $\mathbf{x}$

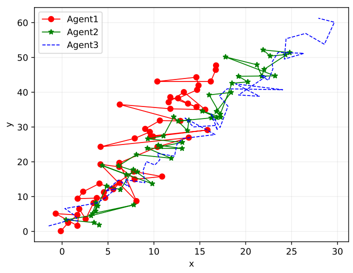
Figure 7: **Sample visualization of the generation of Gaussian synthetic dataset.** $\mathbf{x}$ is the sum of μ_{gt} and ϵ.



(a) μ_{gt}



(b) ϵ



(c) $\mathbf{x}$

Figure 8: **Sample visualization of the generation of Laplace synthetic dataset.** $\mathbf{x}$ is the sum of μ_{gt} and ϵ.

For the Gaussian synthetic dataset, since a random variable $\mathbf{x}$ that obeys a multivariate Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ can be formulated as the sum of the mean μ and a random variable ϵ: $x = \mu + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \Sigma)$. For generating the Gaussian synthetic dataset, we firstly generate 50 different two-dimensional coordinates of three agents that move in a uniform straight line. We denote this part of the data as μ_{gt}, and set it as the mean of the multivariate Gaussian distribution to which the trajectories belong. Subsequently, we sample a set of data ϵ from a multivariate Gaussian distribution $\mathcal{N}(0, \Sigma_{gt})$ (where $\Sigma_{gt} \in \mathbb{R}^{3\times3}$), obviously this set of data contains the information of the covariance matrix of its distribution. Finally, we add the data μ_{gt} representing the mean value of the distribution and the data ϵ representing the covariance matrix information of the distribution to get our final data $\mathbf{x}$, which is $\mathbf{x} = \mu_{gt} + \epsilon$. At this time, the data $\mathbf{x}$ we get is equivalent to the data sampled from the multivariate Gaussian distribution $\mathcal{N}(\mu_{gt}, \Sigma_{gt})$. Moreover, following similar steps, we can get the Laplace synthetic dataset.

B.2 Metric Computation Details

ℓ_2 of μ is the average of pointwise ℓ_2 distances between the estimated mean and the ground truth mean. ℓ_1 of Σ is the average of pointwise ℓ_1 distances between the estimated covariance matrix and the ground truth covariance.

KL is the KL divergence between the ground truth distribution and the estimated distribution $D_{KL}(p_g(X) \| p_e(X))$, where $p_e(X) \sim \mathcal{N}(\mu_{p_e}, \Sigma_{p_e})$ is the estimated distribution, $p_g(X) \sim \mathcal{N}(\mu_{p_g}, \Sigma_{p_g})$ is the ground truth distribution, $\Sigma_{p_e} \in \mathbb{R}^{k\times k}$ and $\Sigma_{p_g} \in \mathbb{R}^{k\times k}$. For multivariate

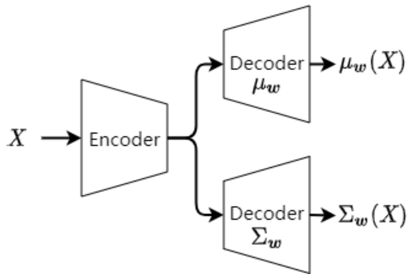
Gaussian distribution, we compute it by the following formula (10):

$$\begin{aligned} D_{KL}(p_g(X)||p_e(X)) &= p_g(X)\int_X [\log(p_g(X)) - \log(p_e(X))]dX \\ &= \frac{1}{2}[\log(\frac{|\Sigma_{p_e}|}{|\Sigma_{p_g}|}) - k + (\mu_{p_g} - \mu_{p_e})^T \Sigma_{p_e}^{-1}(\mu_{p_g} - \mu_{p_e}) + trace\{\Sigma_{p_e}^{-1}\Sigma_{p_g}\}]. \end{aligned} \tag{10}$$

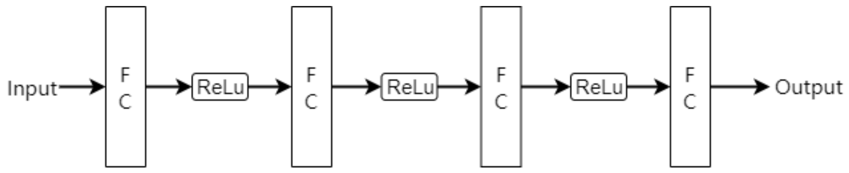
For multivariate Laplace distribution, as the probability density function of it is too complicated, when we compute the KL divergence, we firstly compute the value of $p_g(X)$ and $p_e(X)$ for each given data sample X respectively, and then we compute $D_{KL}(p_g(X)||p_e(X))$ by the following formula (11):

$$D_{KL}(p_g(X)||p_e(X)) = \sum_X p_g(X)[\log(p_g(X)) - \log(p_e(X))]. \tag{11}$$

B.3 Implementation Details


X
Encoder
Decoder $\mu_{\boldsymbol{w}}$
$\mu_{\boldsymbol{w}}(X)$
Decoder $\Sigma_{\boldsymbol{w}}$
$\Sigma_{\boldsymbol{w}}(X)$

(a) Framework



(b) Four-Layer MLP

Figure 9: **The network architecture used in synthetic datasets.** (a): The framework of the used network. (b): The four-layer multilayer perceptron used to form the encoder and decoders of the used network, where FC denotes the full connected layer and ReLu denotes the ReLu activation function.

Model Structure: For toy problem, the network architecture contains an encoder and two decoders, all of which are four-layer multilayer perceptrons (MLPs), which are shown in Figure 9.

Training Details: For toy problem, we train the model on 1 GTX 1080Ti GPU using a batch size of 72 with the Adam [48] optimizer with an initial learning rate of 5×10^{-3} and the training process finishes at 36 epochs.

C Real World Problem

C.1 Implementation Details

Model Structure: We implemented the LaneGCN according to the structure shown in the appendix of [3]. And the structure of our implemented VectorNet is shown in Figure 10.

Training Details: For LaneGCN in Argoverse, we train the model on 1 GTX 1080Ti GPU using a batch size of 64 with the Adam [48] optimizer with an initial learning rate of 2.5×10^{-4} and the training process finishes at 65 epochs. For LaneGCN in nuScenes, we train the model on 2 GTX 1080Ti GPUs using a batch size of 56 with the Adam [48] optimizer with an initial learning rate of 8×10^{-4} and the training process finishes at 90 epochs. For VectorNet in Argoverse, we train the model on 2 GTX 1080Ti GPUs using a batch size of 128 with the Adam [48] optimizer with an initial

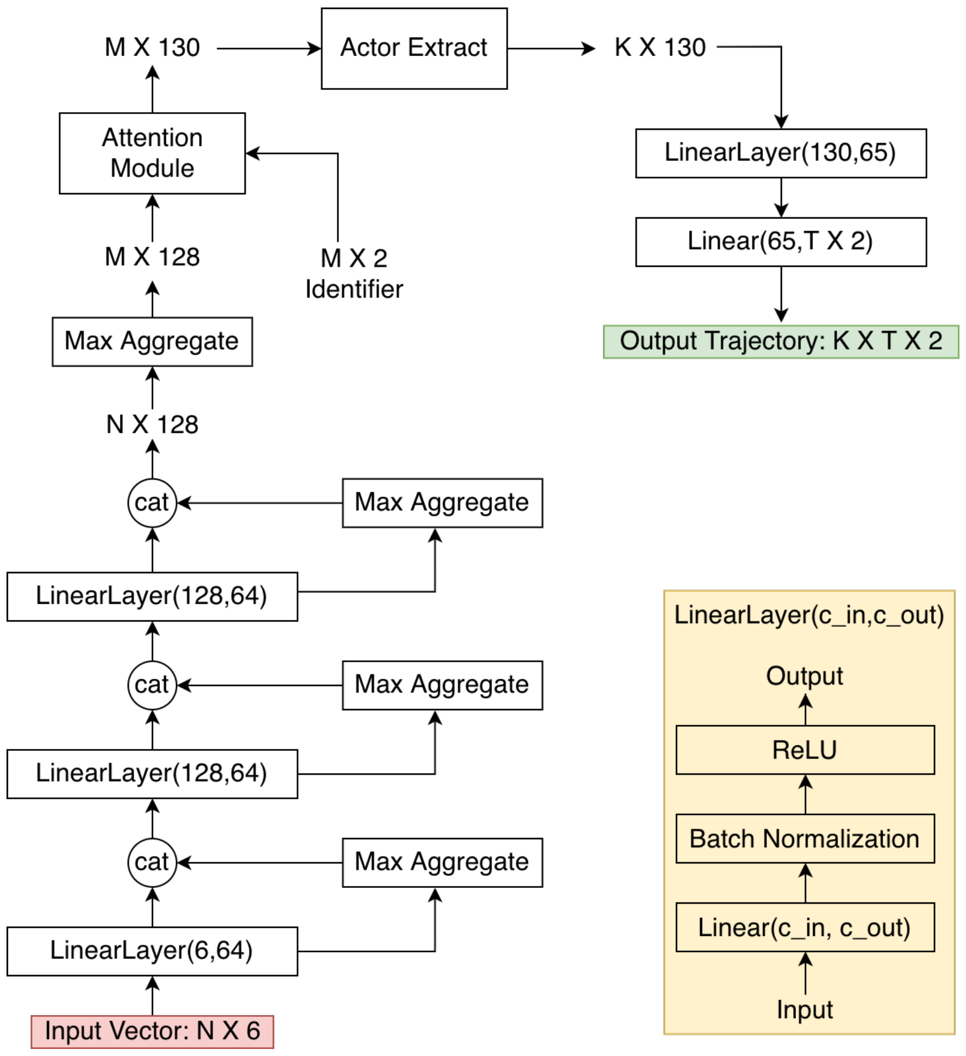


Figure 10: **Implementation structure of VectorNet.** N is the input vector number, M is the polygon number, K is the actor number.More details please refer to [4]

learning rate of 2×10^{-3} and the training process finishes at 105 epochs. For VectorNet in nuScenes, we train the model on 2 GTX 1080Ti GPUs using a batch size of 128 with the Adam [48] optimizer with an initial learning rate of 1×10^{-3} and the training process finishes at 500 epochs.

Table 5: Two special cases with various assumptions about covariance Σ. **ID** denotes the identity matrix (no uncertainty). **DIA** denotes the diagonal matrix (individual uncertainty). **FULL** denotes the full matrix (individual and collaborative uncertainty).

Assumption	Two Special Cases	
	Gaussian Distribution	Laplace Distribution
ID: $\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$	$\Vert \mathrm{Y} - \mu_{\boldsymbol{w}}(\mathrm{X}) \Vert_2^2$	$\Vert \mathrm{Y} - \mu_{\boldsymbol{w}}(\mathrm{X}) \Vert_1$
DIA: $\begin{pmatrix} \sigma_{11} & 0 & \cdots & 0 \\ 0 & \sigma_{22} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \sigma_{mm} \end{pmatrix}$	$\frac{1}{2} \sum_{i=1}^{m} [\sigma_{ii}^{-2} \Vert \mathbf{y}_i - \mu_{\boldsymbol{w}}(\mathbf{x}_i) \Vert_2^2 + \log \sigma_{ii}^2]$	$\sum_{i=1}^{m} [\sigma_{ii}^{-2} \Vert \mathbf{y}_i - \mu_{\boldsymbol{w}}(\mathbf{x}_i) \Vert_1 + \log \sigma_{ii}^2]$
FULL: $\begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2m} \\ \vdots & \vdots & & \vdots \\ \sigma_{m1} & \sigma_{m2} & \cdots & \sigma_{mm} \end{pmatrix}$	$\frac{1}{2} [q_{\boldsymbol{w}}(\mathrm{Y}, \mathrm{X}) - \sum_{j=1}^{m} \log(d_{jj})]$	$\frac{1}{2} [\frac{q_{\boldsymbol{w}}(\mathrm{Y}, \mathrm{X})}{\Phi_{\boldsymbol{w}}(\mathrm{X})} + m \log \Phi_{\boldsymbol{w}}(\mathrm{X}) - \sum_{j=1}^{m} \log(d_{jj})]$

Table 6: Ablation on assumptions about covariance Σ of chosen probability density functions (PDFs) in single future prediction. **ID** denotes the identity matrix (no uncertainty). **DIA** denotes the diagonal matrix (individual uncertainty). **FULL** denotes the full matrix (individual and collaborative uncertainty). On Argoverse and nuScenes, a model with individual uncertainty surpasses a model without uncertainty; a model with individual and collaborative uncertainty surpasses a model with individual uncertainty only.

Dataset	Method	Assumption about Σ	Type of Chosen PDF			
			Gaussian		Laplace	
			ADE	FDE	ADE	FDE
Argoverse	LaneGCN	ID	1.52	3.32	1.44	3.17
		DIA	1.45	3.19	1.43	3.16
		FULL	**1.42**	**3.14**	**1.41**	**3.11**
	VectorNet	ID	1.67	3.62	1.59	3.44
		DIA	1.63	3.60	1.56	3.42
		FULL	**1.57**	**3.46**	**1.52**	**3.34**
nuScenes	LaneGCN	ID	4.50	10.62	4.47	10.54
		DIA	4.47	10.59	4.34	10.34
		FULL	**4.39**	**10.44**	**4.25**	**10.15**
	VectorNet	ID	4.23	9.91	4.09	9.80
		DIA	4.07	9.86	4.02	9.79
		FULL	**3.99**	**9.57**	**3.81**	**9.22**

Table 7: Ablation on assumptions about covariance Σ of chosen probability density functions (PDFs) on the basis of the official version of LaneGCN in single future prediction. **ID** denotes the identity matrix (no uncertainty). **DIA** denotes the diagonal matrix (individual uncertainty). **FULL** denotes the full matrix (individual and collaborative uncertainty). On both of the validate set and the test set of Argoverse, a model with individual uncertainty surpasses a model without uncertainty; a model with individual and collaborative uncertainty surpasses a model with individual uncertainty only.

Dataset	Set	Assumption about Σ	Type of Chosen PDF			
			Gaussian		Laplace	
			ADE	FDE	ADE	FDE
Argoverse	Validate	ID	1.36	2.94	1.28	2.78
		DIA	1.32	2.90	1.27	2.76
		FULL	**1.31**	**2.89**	**1.26**	**2.75**
	Test	ID	1.69	3.72	1.64	3.61
		DIA	1.67	3.70	1.62	3.56
		FULL	**1.66**	**3.67**	**1.61**	**3.53**

C.2 Additional Results

We compare our proposed approach with the approach not modeling uncertainty, which assumes the covariance Σ is an identity matrix (ID). As the results illustrated in Table 6, our proposed Laplace CU-based framework still enables LaneGCN and VectorNet to achieve the best performances on ADE & FDE metrics on both Argoverse and nuScenes benchmarks in single future prediction.

After the paper is accepted in NeurIPS 2021, we also apply our proposed framework to the official version of LaneGCN (with map information) and test it on the Argoverse benchmark in single future prediction. As results shown in the Table 7, our proposed Laplace CU-based framework still enables LaneGCN to achieve the best performances on ADE & FDE metrics on both validate set and test set of Argoverse benchmarks in single future prediction.

C.3 Extra Visualization Results

Visualization of collaborative uncertainty between two agents. In Figure 11, there are 8 actor pairs (blue and orange lines) trajectories (solid lines are the past trajectories and dashed lines are the future trajectories) and their corresponding collaborative uncertainty values changing over the last 30 frames (the heatmap). Pair I to IV and Pair V to VIII are the ones w/o obvious interaction. These results show that the value of collaborative uncertainty is highly related to the amount of the interactive information among agents.

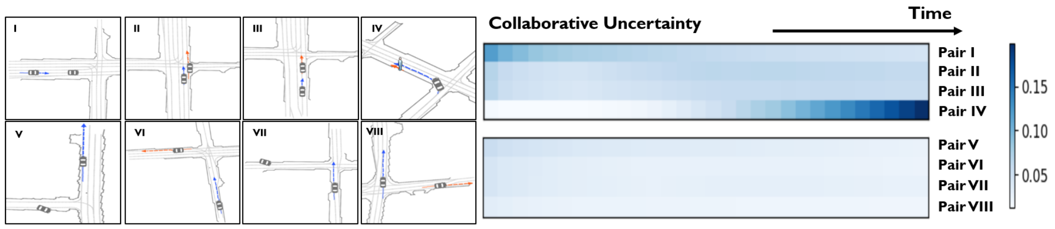


Figure 11: **Visualization of CU on Argoverse dataset**. Pair I: One agent approaching another agent parking at an intersection waiting for green light, as little new interactive information would be generated before the red light turns green, CU decreases over time. Pair II: Agents moving side by side, which might generate complicated interactive information making CU show a non-monotonic change over time. Pair III: Agents driving on the same road, which might generate complicated interactive information making CU show a non-monotonic change over time. Pair IV: Agents moving close to each other, CU increases over time. Pair V to Pair VIII: Agents located in completely different areas on the map, CUs are close to zero.