Quantifying Uncertainty in Motion Prediction with Variational Bayesian Mixture

Supplementary Material

A. Implementation Details

We implement our model using PyTorch, trained for 20 epochs on the INTERACTION dataset with a batch size of 64 and 25 epochs on the Argoverse 2 dataset with a batch size of 64. With only 1.3M parameters, the model balances scalability and performance. We set $\alpha = 1$ and use the Adam optimization solver with a learning rate of 0.0001 and the learning rate decay schedule with a step size of 5 epochs and a rate of 0.3 to ensure efficient convergence. We train and evaluate our model using only a single NVIDIA GeForce RTX 3090 Ti.

B. Evaluation Metrics

B.1. Uncertainty Quantification

In our formulation, the random variables $\boldsymbol{s}_{\mathrm{f}}$ and $\boldsymbol{v}$ are both Gaussian variables, and z follows a categorical distribution. Therefore, we can compute the total uncertainty for a predicted distribution by its entropy. Given the generative model in equation 4, the total entropy can be estimated by the summation of three individual expected entropy:

$$\begin{aligned}
&\sum_{z} \int_{\boldsymbol{v}} \int_{\boldsymbol{s}_{\mathrm{f}}} p(\boldsymbol{s}_{\mathrm{f}}, \boldsymbol{v}, z|\boldsymbol{x}) \log p(\boldsymbol{s}_{\mathrm{f}}, \boldsymbol{v}, z|\boldsymbol{x}) d\boldsymbol{s}_{\mathrm{f}} d\boldsymbol{v} \\
=&\mathbb{E}_{\boldsymbol{v}, z\sim p(\boldsymbol{v},z|\boldsymbol{x})} \text{Entropy}(p(\boldsymbol{s}_{\mathrm{f}}|\boldsymbol{v}, x)) \\
&+ \mathbb{E}_{\boldsymbol{s}_{\mathrm{f}}\sim p(z\sim p(z)} \text{Entropy}\left(p(\boldsymbol{v}|\boldsymbol{x}, z)\right) \\
&+ \text{Entropy}\left(p(z)\right).
\end{aligned} \tag{17}$$

Since we have a fixed prior $p(z)$, the comparison of the total entropy reduces to comparing the sum of the first two terms. In our experiment, we use Monte-Carlo sampling to generate N_{mc} samples of $\boldsymbol{v}$ for entropy calculation.

B.2. Motion Prediction

For motion prediction, we use the standard Minimum Average Displacement Error (minADE), Minimum Final Displacement Error (minFDE), and Miss Rate (MR) to assess the accuracy and effectiveness of our approach. minADE and minFDE are distance-based metrics commonly used in multi-modal trajectory prediction (i.e., trajectory prediction with multiple possible outcomes) tasks. The minADE calculates the average Euclidean distance between predicted and ground truth trajectories at each time step, taking the minimum across all trajectories in the prediction set:

$$\text{minADE}(\hat{x}_n^k, x_n) = \frac{1}{NT} \sum_{n=1}^{N} \min_{k=1,\dots,K} \sum_{t=1}^{T} \left\| \hat{x}_{n,t}^k - x_{n,t} \right\|_2. \tag{18}$$

On the other hand, the minFDE measures the Euclidean distance between predicted and ground truth final positions, effectively assessing the long-term prediction performance of the model:

$$\text{minFDE}(\hat{x}_n^k, x_n) = \frac{1}{N} \sum_{n=1}^{N} \min_{k=1,\dots,K} \left\| \hat{x}_{n,T}^k - x_{n,T} \right\|_2. \tag{19}$$

MR represents the ratio of 'miss' cases over all cases. The definitions of MR are significantly different for the INTERACTION dataset and the Argoverse 2 dataset.

In the INTERACTION dataset, if its prediction at the final timestamp (T=30) is out of a given lateral or longitudinal threshold of the ground truth, it will be assumed as a 'miss.' In the INTERACTION dataset, we need to align both the ground truth and the prediction by rotating them based on the yaw angle of the ground truth at the final timestamp, ensuring that the x-axis represents the longitudinal direction and the y-axis corresponds to the lateral direction. The lateral threshold is established as 1 meter, while the longitudinal threshold is a piecewise function set as:

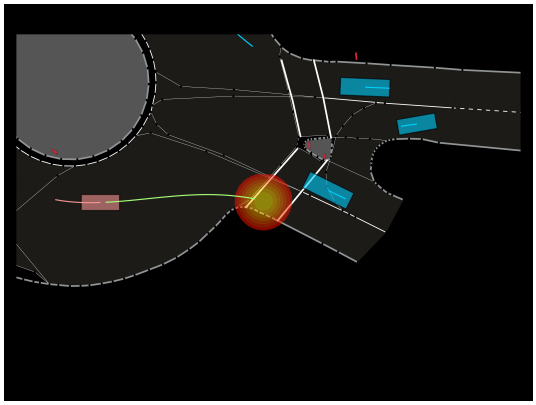
$$\text{Threshold}_{\text{lon}} = \begin{cases} 1 & v < 1.4m/s \\ 1 + \frac{v-1.4}{11-1.4} & 1.4m/s \leq v \leq 11m/s \\ 2 & v \geq 11m/s \end{cases} \tag{20}$$

For the Argoverse 2 dataset, the MR indicates the proportion of test samples where none of the predicted trajectories fall within a 2-meter range of the ground truth, as measured through the endpoint error measurement.

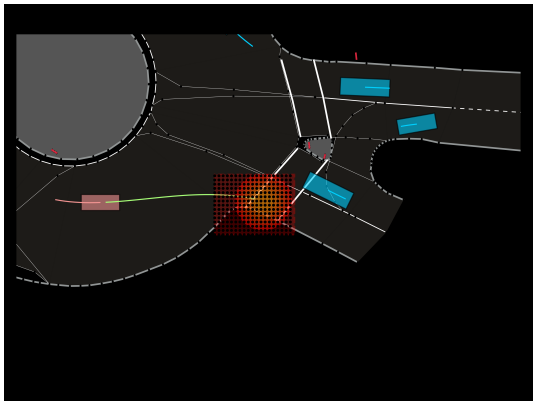
C. Methodology Details

C.1. Derivation of Evidence Lower Bound (ELBO)

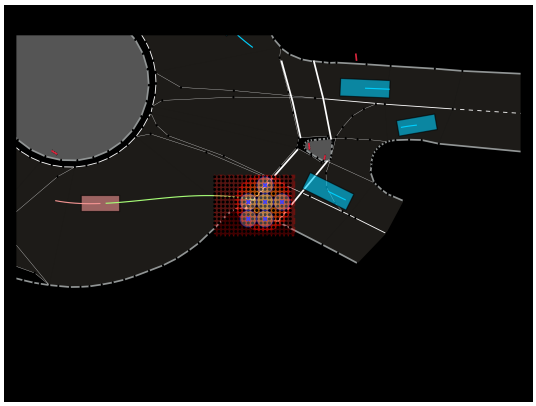
The standard and intuitive objective for training the probabilistic model in SeNeVA is to let the modeled conditional distribution $p(\boldsymbol{s}_{\mathrm{f}}|\boldsymbol{x})$ to match ground-truth data distribution through maximizing the likelihood. However, direct computation on the likelihood function is intractable since it involves calculating the integration given as $p(\boldsymbol{s}_{\mathrm{f}}|\boldsymbol{x}) = \sum_z \int_{\boldsymbol{v}} p(\boldsymbol{s}_{\mathrm{f}}, \boldsymbol{v}, z|\boldsymbol{x}) d\boldsymbol{v} dz$, which is hard to estimate and optimize. To address this issue, we follow the popular *variational inference* method and introduce a tractable, closed-form, and easy-sampling proxy posterior $q(\boldsymbol{v}, z|\boldsymbol{s}_{\mathrm{f}}, \boldsymbol{x})$ of the latent variables conditioned on the observed variables, and the lower bound of the log-likelihood can be derived with



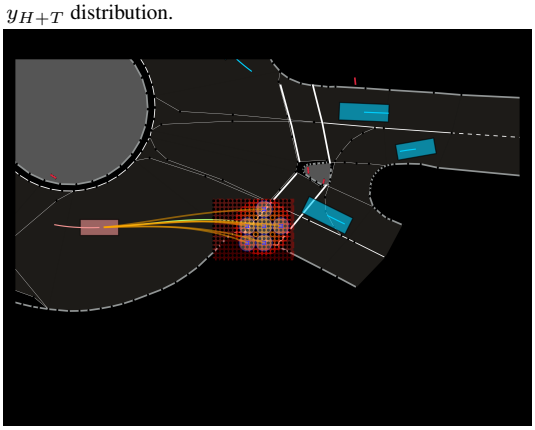
(a) **Distribution evaluation output.** The heatmap illustrate the distribution of y_{H+T} in this case quantified by the SeNeVA model.



(b) **Dense grid generation.** For sampling, we generate a grid of candidates that covers 2 standard deviation area of the y_{H+T} distribution.



(c) **NMS sampling results.** Following Algorithm 1, we can sample a set of top-M candidates (blue) regarding circular buffers defined by radius r and their IoU threshold γ.



(d) **Backward completion.** Starting from the last location, we assume homogeneous uncertainty over time and compute intermediate waypoints to obtain the final trajectories (orange).

Figure 6. **Example visualization of the backward sampling process.** The multi-modal trajectory prediction is generated through (a) evaluating distribution, (b) generating dense candidates, (c) applying NMS sampling, and finally (d) completing intermediate trajectories. To better illustrate the effectiveness of our method, we plot the history (red) and ground-truth future trajectory (green) of the target agent.

Jensen's Inequality:

$$\begin{aligned}\log p(\boldsymbol{s}_\mathrm{f}|\boldsymbol{x}) &= \log \int_z \int_{\boldsymbol{v}} p(\boldsymbol{s}_\mathrm{f}, \boldsymbol{v}, z|\boldsymbol{x}) d\boldsymbol{v} dz \\ &= \log \mathbb{E}_{q(\boldsymbol{v},z|\boldsymbol{s}_\mathrm{f},\boldsymbol{x})} \left[\frac{p(\boldsymbol{s}_\mathrm{f}, \boldsymbol{v}, z|\boldsymbol{x})}{q(\boldsymbol{v}, z|\boldsymbol{s}_\mathrm{f}, \boldsymbol{x})}\right] \\ &\geq \mathbb{E}_{q(\boldsymbol{v},z|\boldsymbol{s}_\mathrm{f},\boldsymbol{x})} \log \left[\frac{p(\boldsymbol{s}_\mathrm{f}, \boldsymbol{v}, z|\boldsymbol{x})}{q(\boldsymbol{v}, z|\boldsymbol{s}_\mathrm{f}, \boldsymbol{x})}\right].\end{aligned} \tag{21}$$

Since we factorize the joint distribution $p(\boldsymbol{s}_\mathrm{f}, \boldsymbol{v}, z|\boldsymbol{x})$ and the posterior $q(\boldsymbol{v}, z|\boldsymbol{s}_\mathrm{f}, \boldsymbol{x})$ in equation 4 and equation 6, respectively, we can leverage the factorization and expand the expectation term to compute the analytical solution of the lower bound. To simplify the following notation, we denote $p(\boldsymbol{s}_\mathrm{f}, \boldsymbol{v}, z|x) = p_{\boldsymbol{s}_\mathrm{f},\boldsymbol{v},z}$, $q(\boldsymbol{v}, z|\boldsymbol{s}_\mathrm{f}, \boldsymbol{x}) = q_{\boldsymbol{v},z}$, $p(\boldsymbol{s}_\mathrm{f}|\boldsymbol{v}, \boldsymbol{x}) = p_{\boldsymbol{s}_\mathrm{f}}$, $p(\boldsymbol{v}|\boldsymbol{x}, z) = p_{\boldsymbol{v}}$, $p(z) = p_z$, $q(\boldsymbol{v}|\boldsymbol{s}_\mathrm{f}, \boldsymbol{x}) = q_{\boldsymbol{v}}$, and $q(z|\boldsymbol{v}, \boldsymbol{x}) = q_z$. The expansion of the expectation term writes:

$$\begin{aligned}\mathbb{E}_{q_{\boldsymbol{v},z}} \log \left[\frac{p_{\boldsymbol{s}_\mathrm{f},\boldsymbol{v},z}}{q_{\boldsymbol{v},z}}\right] &= \int_z \int_{\boldsymbol{v}} \log \left[\frac{p_{\boldsymbol{s}_\mathrm{f}} p_{\boldsymbol{v}} p_z}{q_{\boldsymbol{v}} q_z}\right] \cdot q_{\boldsymbol{v},z} d\boldsymbol{v} dz \\ &= \mathbb{E}_{q_{\boldsymbol{v}}} \log p_{\boldsymbol{s}_\mathrm{f}} + \int_z q_z \int_{\boldsymbol{v}} q_{\boldsymbol{v}} \log \left[\frac{p_{\boldsymbol{v}}}{q_{\boldsymbol{v}}}\right] d\boldsymbol{v} dz \\ &\quad + \int_{\boldsymbol{v}} q_{\boldsymbol{v}} \int_z q_z \log \left[\frac{p_z}{q_z}\right] dz d\boldsymbol{v} \\ &= \mathbb{E}_{q_{\boldsymbol{v}}} \left(\log p_{\boldsymbol{s}_\mathrm{f}} - D_\mathrm{KL}(q_z \| p_z)\right) - \mathbb{E}_{q_z} D_\mathrm{KL}(q_{\boldsymbol{v}} \| p_{\boldsymbol{v}}).\end{aligned} \tag{22}$$

The expansion above is the ELBO objective we maximize during training equivalent to the formula given in equation 10. Therefore, maximizing the lower bound is equal to minimizing the KL divergence, driving the variational posterior $q(\boldsymbol{v}, z|\boldsymbol{s}_\mathrm{f}, \boldsymbol{x})$ towards the ground-truth posterior. As a result, maximizing the ELBO objective can effectively maximize the likelihood.

C.2. Derivation of Assignment Network Loss

The assignment network directly approximates $p(z|\boldsymbol{x})$ to avoid tedious sampling at inference time from the latent $\boldsymbol{v}$ space to estimate the posterior $q(z|\boldsymbol{x}) = \int_{\boldsymbol{v}} q(z|\boldsymbol{v}, \boldsymbol{x})d\boldsymbol{v}$. We can obtain the distribution over z given in equation 13 by applying Bayes' rule. Herein, we estimate the conditional distribution $p(\boldsymbol{s}_{\text{f}}|\boldsymbol{x}, z)$ by applying Monte-Carlo sampling over the latent $\boldsymbol{v}$ space at training:

$$\begin{aligned} p(\boldsymbol{s}_{\text{f}}|\boldsymbol{x}, z) &= \int_{\boldsymbol{v}} p(\boldsymbol{s}_{\text{f}}, \boldsymbol{v}|\boldsymbol{x}, z)d\boldsymbol{v} \\ &\approx \frac{1}{N_{\text{mc}}} \sum_{n=1}^{N_{\text{mc}}} p(\boldsymbol{s}_{\text{f}}, \boldsymbol{v}^{(n)}|\boldsymbol{x})p(\boldsymbol{v}^{(n)}, z|\boldsymbol{x}). \end{aligned} \tag{23}$$

C.3. Backward Sampling

As mentioned in section 4.5, we propose the backward sampling procedure to generate a collection of trajectories leveraging the distribution information learned by the model. The idea is first to sample the final location y_{H+T} that accounts for most uncertainty in the trajectory. The backward sampling procedure consists of three steps: Evaluation, Sampling, and Completion.

Evaluation At this stage, we leverage the output $\hat{\pi}$ from the assignment network to determine how we evaluate the distribution of y_{H+T}. One can use the component corresponding to $\hat{\pi}_{\text{max}}$. In our case, we promote multi-modality by computing the distribution as a mixture of top-6 components. For handling the latent space $\boldsymbol{v}$, one can apply Monte-Carlo sampling to approximate the integral. In our case, we choose to use the maximum likelihood samples (i.e., $\boldsymbol{v}^{\text{ml}} = \underset{\boldsymbol{v}}{\text{argmax}} p(\boldsymbol{v}, z|\boldsymbol{x})$) in equation 16 to evaluate the distribution, as shown in Figure 6a.

Sampling To allow full exploitation of the distribution information, we first generate a dense grid of candidates that covers the area within 2 standard deviations of the distribution mean. We adopt a rectangular grid with a resolution of 0.5 meters for simplicity, as illustrated in Figure 6b. One can quickly improve precision by choosing a smaller resolution or clipping the grid area. We then apply the NMS sampling given in Algorithm 1 to sample M candidates from the dense grid considering their circular buffers determined by hyperparameter r and the IoU threshold γ (see Figure 6c). Together, the two hyperparameters determine the density of selected candidates. In our practice, we choose $r = 1.4$ meters and $\gamma = 0\%$.

Completion The last step is to complete the intermediate trajectory from the target agent's current position to the sampled final locations. One can easily apply random sampling on each timestep to get the waypoints. Nevertheless, we find trajectories generated by this approach lack auto-consistency and can be non-smooth. To address the problem, we propose a strong assumption that displacement uncertainty is uniform over time. Hence, we can first parameterize an uncertainty distance parameter $u^{(m)}$ for each selected candidate and then use it for computing waypoints for all previous timesteps. Specifically, for sampled candidate $y_{H+T}^{(m)}$ from the distribution $\mathcal{N}(\mu_{H+T}, \Sigma_{H+T})$, we have

$$u^{(m)} = L_{H+T}^{-1}\left(y_{H+T}^{(m)} - \mu_{H+T}\right) : \Sigma_{H+T} = LL^{\intercal}, \tag{24}$$

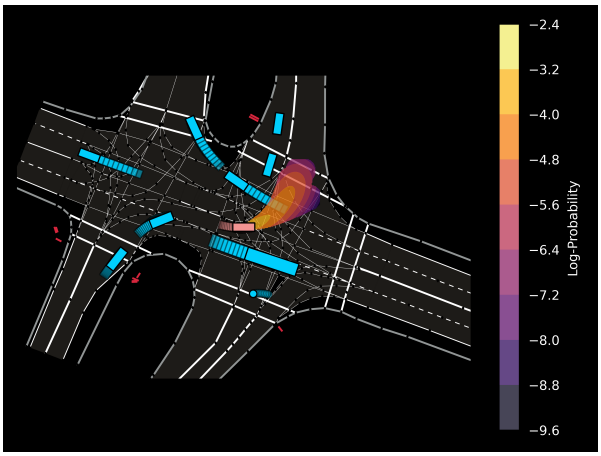
where L is the upper triangle Cholesky decomposition of the covariance. For each timestep $t = 1, \ldots, T-1$, we have

$$y_{H+t}^{(m)} = \mu_{H+t} + L_{H+t} \cdot u^{(m)}. \tag{25}$$

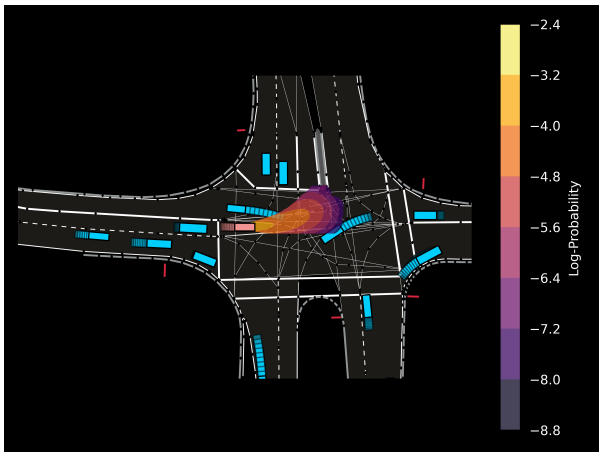
Finally, we connect intermediate waypoints with the sampled candidates to derive the required trajectory set. Figure 6d illustrates the final output from the backward sampling process.

D. Extensive Qualitative Results

We further visualize the quantified trajectory distributions on some representative cases selected from the INTERACTION dataset. Figure 7 illustrates two examples from unsignalized intersections, where SeNeVA successfully identifies the left-turn intention of the driver and quantifies the distribution of future trajectories that conform to the road geometry. In Figure 8, we visualize two cases in the expressway merging, where the SeNeVA model can anticipate the maneuver of the surrounding vehicles and predict distributions that avoid collisions.

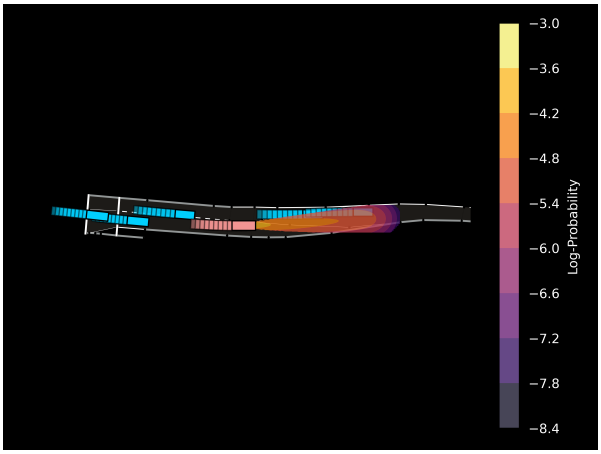


(a) Results on a case from DR_USA_Intersection_GL

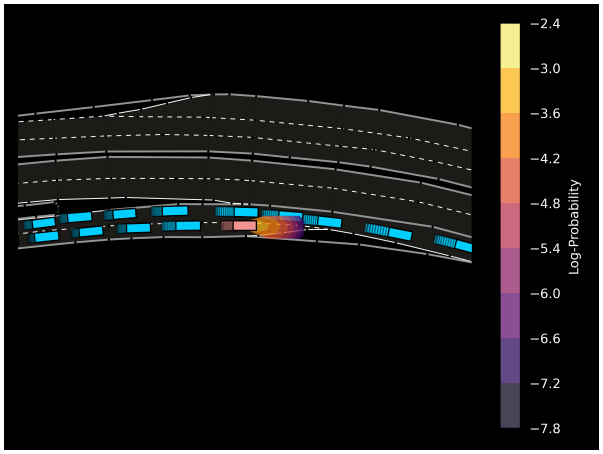


(b) Results on a case from DR_USA_Intersection_MA

Figure 7. **Representative example visualization of quantified uncertainty on intersections.** The heatmap generated by the SeNeVA model successfully identifies the left-turn intention of drivers in both cases. The predicted distributions conform to the road geometry.



(a) Results on a case from DR_DEU_Merging_MT



(b) Results on a case from DR_CHN_Merging_ZS0

Figure 8. **Representative example visualization of quantified uncertainty on intersections.** The model recognizes the existence of surrounding vehicles and predicts with higher certainty that a vehicle will stay hold to avoid collisions.