BIMODAL KNOWLEDGE DISTILLATION OF BAYESIAN DEEP NEURAL NETWORKS WITH MIXTURES OF GAUSSIAN PROCESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Bayesian neural networks provide a probabilistic framework for uncertainty quantification through a posterior over model parameters. In this work, we study the predictive structure induced by this posterior and examine how this structure can be used for prediction and uncertainty quantification. From SGLD posterior samples, we observe that, for some inputs, predictive logits form two separated modes that support different classes. This structure identifies the class pair underlying the predictive ambiguity and indicates whether the uncertainty comes from disagreement between these two classes rather than broad ambiguity across classes. Posterior distillation enables efficient inference by compressing posterior samples, but a unimodal approximation can lose this mode structure. To preserve this structure, we propose MoDGF, a posterior distillation method based on a mixture of two Deep Gaussian Factor (DGF) components. Using this component-wise structure, we propose component-guided pairwise specialist inference and within-component epistemic OOD detection. Extensive experiments show that MoDGF preserves SGLD predictive performance across image and language tasks, while its component-wise information improves specialist inference and OOD detection.

1 INTRODUCTION

Deep neural networks (DNNs) are widely used in modern AI systems, including vision, language, and various predictive tasks. However, these models are usually overconfident (Guo et al., 2017; Ovadia et al., 2019). Accordingly, trustworthy AI requires improving not only accuracy but also uncertainty quantification to identify when and why predictions may be unreliable. Reliable uncertainty quantification is particularly important for risk-aware decision making in high-stakes applications.

Bayesian inference is a promising approach for uncertainty quantification because it performs probabilistic inference rather than a deterministic one. In deep learning, Bayesian neural networks (BNNs) (Blundell et al., 2015; Gal & Ghahramani, 2016; Wilson & Izmailov, 2020; Izmailov et al., 2021) place a prior distribution over network parameters and use data to obtain the posterior distribution. In this work, we focus on stochastic gradient Langevin dynamics (SGLD), a popularly used MCMC algorithm for BNNs (Welling & Teh, 2011), which uses mini-batch gradients to generate posterior samples. However, posterior prediction with SGLD requires storing many samples and evaluating multiple networks, leading to substantial memory and inference costs. To reduce these costs, several works have studied posterior and ensemble distillation, which compress multiple models into a student model (Korattikara Balan et al., 2015; Vadera et al., 2020; Wang et al., 2018; Malinin et al., 2019; Tran et al., 2020; Ryabinin et al., 2021; Park et al., 2025b).

Several posterior and ensemble distillation methods use a single unimodal representation, for example, a Dirichlet distribution, or a Gaussian process (Malinin et al., 2019; Park et al., 2025b). However, previous studies have examined multimodality in the parameter posterior of BNNs (Zhang et al., 2020; Izmailov et al., 2021; Laurent et al., 2024), while predictive multimodality has also been studied under a discretized prior setting (Fisher & Marzouk, 2025). In this work, we examine predictive bimodality induced by SGLD samples and observe that, for some inputs, the predictive logits form two separated modes that support different classes.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Predictive multimodality can provide information beyond overall predictive variability. When different predictive modes support different classes, the mode structure reveals which pair of classes is involved in the predictive disagreement. Moreover, it indicates whether the overall predictive variability comes from uncertainty within the modes or from the separation between them. These observations motivate preserving the predictive mode structure during posterior distillation.

Based on this motivation, we consider posterior distillation while preserving this predictive structure. The DGF model, introduced for ensemble distillation by Park et al. (2025b), approximates the teacher distribution by a single Gaussian process and thus cannot explicitly represent bimodal predictive structure. We therefore propose MoDGF, which uses a mixture of two DGF components as the student model to represent bimodal predictive structure in the distilled distribution.

Using this component-wise structure, we further propose two methods for prediction and uncertainty quantification. First, when the two components support different classes, they identify the class pair associated with the predictive ambiguity. Previous specialist methods address class-level confusion using specialists assigned to predefined class pairs or subsets (Hinton et al., 2015; Yan et al., 2015; Šuch et al., 2025; Jung et al., 2019; Chowdhury & Zhao, 2025). However, this selection does not directly account for whether the current input is ambiguous between those classes. In this paper, we propose component-guided pairwise specialist inference, which uses the class pair identified by the two components to select the corresponding pairwise specialist.

Second, we decompose epistemic uncertainty into between-component uncertainty associated with the bimodal structure and within-component uncertainty remaining within each component. Existing Bayesian OOD detection methods commonly use overall epistemic uncertainty as an OOD score (Mucsányi et al., 2024), but the between-component term can be large even for in-distribution inputs with bimodal predictions. We therefore propose Within-Component Epistemic (WCE) OOD Detection, which uses only within-component epistemic uncertainty as the OOD score to reduce false alarms caused by predictive bimodality.

Our main contributions are summarized as follows:

- We examine predictive bimodality induced by SGLD posterior samples. To preserve this predictive mode structure during posterior distillation, we propose MoDGF, a posterior distillation method based on a mixture of two DGF components.
- We use the preserved component-wise information for prediction and uncertainty quantification. Specifically, we use the class pair identified by the two components for pairwise specialist selection and within-component epistemic uncertainty for OOD detection.
- We empirically demonstrate that MoDGF achieves predictive performance comparable to SGLD across image and language tasks while improving over Single DGF. We also show that the component-wise information improves specialist inference and OOD detection.

2 Preliminaries

Let $\boldsymbol{x} \in \mathcal{X}$ and $y \in \{1, \dots, c\}$ denote an input and its class label, respectively. Let $(\boldsymbol{x}_i, y_i), i = 1, \dots, n$ be training data and write $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$. A DNN with parameters θ defines a logit function $f_\theta : \mathcal{X} \to \mathbb{R}^c$ and the likelihood $p(y = k \mid \boldsymbol{x}, \theta) = \mathrm{softmax}(f_\theta(\boldsymbol{x}))_k$.

2.1 Posterior Predictive Distribution and Uncertainty

Given a prior $p(\theta)$, the posterior distribution of the parameters is given as

$$p(\theta \mid \mathcal{D}) \propto p(\theta) \prod_{i=1}^{n} p(y_i \mid \boldsymbol{x}_i, \theta). \tag{1}$$

The posterior distribution is used for various downstream tasks. First of all, the predictive distribution for a given new input $\boldsymbol{x}$ is obtained by

$$p(y \mid \boldsymbol{x}, \mathcal{D}) = \int p(y \mid \boldsymbol{x}, \theta) p(\theta \mid \mathcal{D}) \, d\theta, \tag{2}$$

which is usually approximated by

$$p(y = k \mid \boldsymbol{x}, \mathcal{D}) \approx \frac{1}{N} \sum_{j=1}^{N} \text{softmax}\left(f_{\theta_j}(\boldsymbol{x})\right)_k, \tag{3}$$

where $\{\theta_j\}_{j=1}^N$ are posterior samples.

Secondly, the predictive uncertainty can be decomposed into the total, aleatoric, and epistemic uncertainties (Gal, 2016) as follows:

$$U_{\text{total}}(\boldsymbol{x}) = \text{H}\left[\mathbb{E}_{\theta|\mathcal{D}}\left[\text{softmax}(f_\theta(\boldsymbol{x}))\right]\right], \tag{4}$$

$$U_{\text{aleatoric}}(\boldsymbol{x}) = \mathbb{E}_{\theta|\mathcal{D}}\left[\text{H}\left[\text{softmax}(f_\theta(\boldsymbol{x}))\right]\right], \tag{5}$$

$$U_{\text{epistemic}}(\boldsymbol{x}) = U_{\text{total}}(\boldsymbol{x}) - U_{\text{aleatoric}}(\boldsymbol{x}) \tag{6}$$

where $\text{H}(p) = -\sum_{k=1}^{c} p_k \log p_k$ for $p = (p_1, \cdots, p_c)^\top$ is the entropy. There are various applications of this uncertainty decomposition including OOD detection, active learning, and reinforcement learning (Kendall & Gal, 2017; Gal et al., 2017; Depeweg et al., 2018).

2.2 Stochastic Gradient Langevin Dynamics (SGLD)

SGLD is an MCMC method that injects Gaussian noise into a stochastic gradient to generate posterior samples (Welling & Teh, 2011; Li et al., 2016). To be more specific, at iteration t, let $\mathcal{B}_t \subset \{1, \ldots, n\}$ be a mini-batch of size b. SGLD updates the parameter as

$$g_t = \nabla_\theta \log p(\theta_t) + \frac{n}{b} \sum_{i \in \mathcal{B}_t} \nabla_\theta \log p(y_i \mid \boldsymbol{x}_i, \theta_t), \tag{7}$$

$$\theta_{t+1} = \theta_t + \frac{\eta_t}{2} g_t + \sqrt{\eta_t}\, \boldsymbol{\xi}_t, \qquad \boldsymbol{\xi}_t \sim \mathcal{N}(\mathbf{0}, \mathbb{I}), \tag{8}$$

where η_t is the step size. After burn-in, we retain N samples $\{\theta_j\}_{j=1}^N$ of the parameter and the corresponding functions $\{f_{\theta_j}\}_{j=1}^N$. These functions serve as the teacher posterior distribution in our posterior distillation framework.

2.3 Deep Gaussian Factor Model (DGF)

The Deep Gaussian Factor (DGF) model is a Gaussian process with latent Gaussian factors. This model was introduced in Park et al. (2025b) under the name of the Deep Latent Factor (DLF) model. In this paper, we use the term DGF to emphasize that the latent factors follow Gaussian distributions. Concretely, DGF parameterizes a Gaussian process through latent Gaussian variables, with neural networks defining the mean function and factor loadings.

A DGF with latent factor dimension r models a Gaussian process as

$$f(\boldsymbol{x}) = \mu_\kappa(\boldsymbol{x}) + L\boldsymbol{Z}\Phi_\kappa(\boldsymbol{x}) + \boldsymbol{\epsilon}, \tag{9}$$

where $\mu_\kappa : \mathcal{X} \to \mathbb{R}^c$ is the mean function, $\Phi_\kappa : \mathcal{X} \to \mathbb{R}^r$ is the factor-loading function, $L \in \mathbb{R}^{c \times c}$ is a learnable lower-triangular matrix, $\boldsymbol{Z} \sim \text{MN}_{c,r}(\mathbf{0}, \mathbb{I}_c, \mathbb{I}_r)$ is a matrix-variate Gaussian latent factor, and $\boldsymbol{\epsilon} \sim \mathcal{N}_c(\mathbf{0}, \sigma_f^2 \mathbb{I}_c)$ is a Gaussian noise term. We denote the collection of DGF parameters by $\psi = (\kappa, L, \sigma_f^2)$ and let $\Lambda = LL^\top$.

Given the points $\{\boldsymbol{x}_m\}_{m=1}^M$, let $\boldsymbol{f} = (f(\boldsymbol{x}_m),\, m = 1, \ldots, M)^\top \in \mathbb{R}^{M \times c}$, $\boldsymbol{\mu}_\kappa = (\mu_\kappa(\boldsymbol{x}_m),\, m = 1, \ldots, M)^\top \in \mathbb{R}^{M \times c}$, and $\boldsymbol{\Phi}_\kappa = (\Phi_\kappa(\boldsymbol{x}_m),\, m = 1, \ldots, M)^\top \in \mathbb{R}^{M \times r}$. Then,

$$\text{vec}(\boldsymbol{f}) \sim \mathcal{N}\left(\text{vec}(\boldsymbol{\mu}_\kappa), \Lambda \otimes \boldsymbol{\Phi}_\kappa \boldsymbol{\Phi}_\kappa^\top + \sigma_f^2 \mathbb{I}_{Mc}\right), \tag{10}$$

where $\text{vec}(\cdot)$ denotes the column-wise vectorization of a matrix. As shown by Park et al. (2025b), DGF can approximate a broad class of Gaussian processes by choosing the architecture of the neural networks for the mean and factor loading functions accordingly.

Park et al. (2025b) approximate the distribution of f_θ induced by a teacher ensemble by DGF. Though DGF can approximate the posterior distribution well while preserving dependence between **inputs**, the approximated posterior $f_\theta(\boldsymbol{x})$ for a given $\boldsymbol{x}$ is Gaussian and hence always unimodal. Thus, DGF cannot capture useful information in the bimodality of the posterior distribution. In this paper, we consider a mixture of DGFs to capture the bimodality also.

3 PREDICTIVE BIMODALITY IN SGLD POSTERIOR SAMPLES

For a given input $\boldsymbol{x}$, we say that a class y has predictive bimodality if the posterior distribution of $f_{\theta,y}(\boldsymbol{x})$ is bimodal. If there exists a class y which has predictive bimodality, we say that $\boldsymbol{x}$ has predictive bimodality. In this section, we illustrate how often predictive bimodality emerges and discuss how the predictive bimodality can be utilized. For each input $\boldsymbol{x}$, we identify predictive bimodality using SGLD samples and the criterion described in Appendix C.

3.1 EMPIRICAL OBSERVATION OF PREDICTIVE BIMODALITY

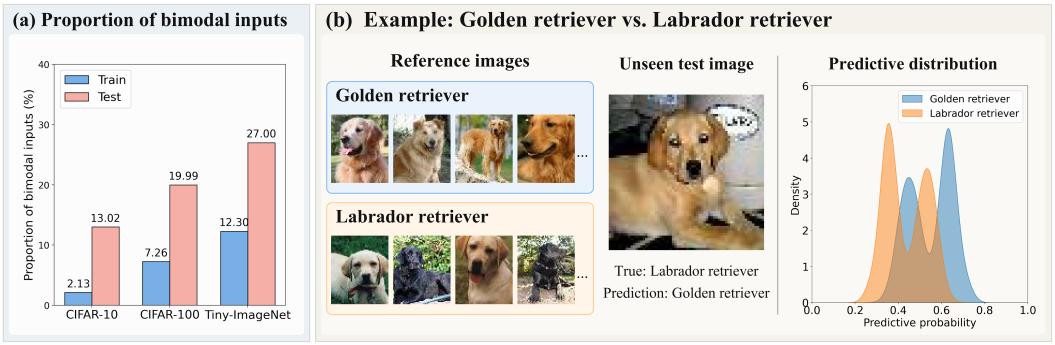

Figure 1: Predictive bimodality induced by SGLD posterior samples. (a) Proportion of inputs with bimodal predictive logit distributions in the training and test sets of three datasets. (b) A Tiny-ImageNet example with an unseen Labrador retriever input and the predictive probability distributions for Golden retriever and Labrador retriever.

As shown in Figure 1(a), predictive bimodality is frequently observed (more than 10% in the test data) across CIFAR-10, CIFAR-100, and Tiny-ImageNet. An interesting feature is that the proportion of inputs with predictive bimodality is higher in the test data than in the training data. We also found that predictive distributions with three or more modes were rarely observed in the SGLD samples. We therefore focus on bimodal predictive structure in this work.

Figure 1(b) visualizes the predictive probabilities for a predictive-bimodal input selected from Tiny-ImageNet. For the unseen Labrador retriever input, some SGLD posterior samples support Golden retriever, whereas others support Labrador retriever, resulting in two distinct predictive modes.

3.2 INFORMATION IN PREDICTIVE BIMODALITY

We next discuss two types of information provided by predictive bimodality, which motivate us to try to preserve predictive bimodality during posterior distillation. First, when the two predictive modes support two different classes, they identify the pair of classes involved in the predictive disagreement. In Section 5.1, we use this class-pair information to select the corresponding pairwise specialist. Empirical results indicate that the pairwise specialist selected by predictive bimodality improves prediction accuracy much.

Second, predictive bimodality provides a way of decomposing epistemic uncertainty. When predictive bimodality exists, the disagreement between the two modes affects epistemic uncertainty much. Thus, when we use epistemic uncertainty for OOD detection, we need to remove uncertainty due to predictive bimodality. In Section 5.2, we decompose the epistemic uncertainty as the sum of the between epistemic uncertainty (i.e. uncertainty due to the disagreement between two modes) and the within epistemic uncertainty (i.e. uncertainty around each mode), and use the within epistemic uncertainty for OOD detection.

4 THE PROPOSED DISTILLATION METHOD FOR PREDICTIVE BIMODALITY

In this section, we develop a posterior distillation method which preserves predictive bimodality. Posterior distillation using a single Gaussian process has received much attention (Park et al., 2025b). However, a single Gaussian process cannot capture predictive bimodality. Preserving predictive

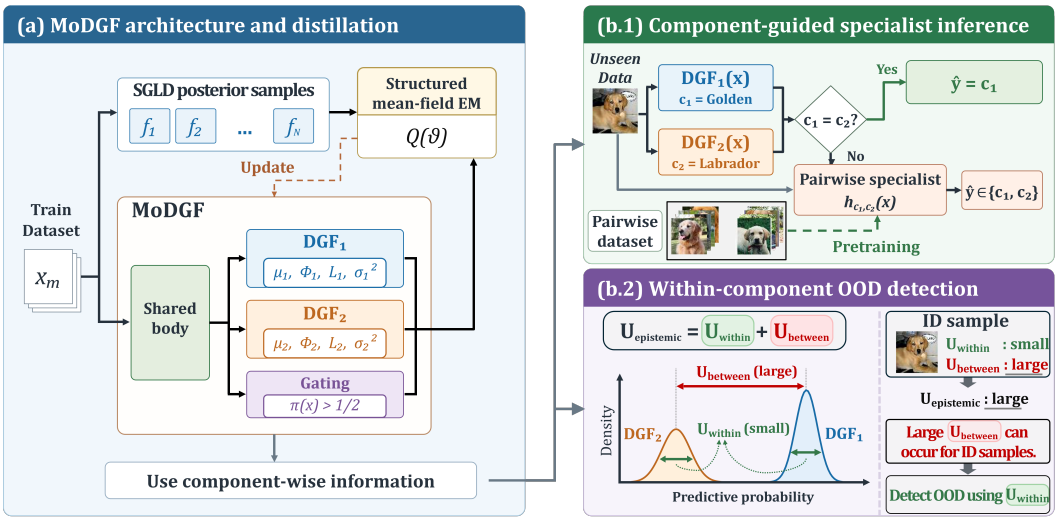

Figure 2: Overview of MoDGF and its applications to component-guided specialist inference and within-component epistemic OOD detection.

bimodality can also be important for uncertainty quantification. In Appendix A, we show that, for a class of sufficiently separated bimodal logit distributions, a single-Gaussian approximation can underestimate epistemic uncertainty. Such underestimation can affect downstream decisions that use epistemic uncertainty, such as sample selection in active learning. To preserve predictive bimodality after the posterior distillation, we propose MoDGF, a mixture of two DGF components.

4.1 Mixture of Deep Gaussian Factors (MoDGF)

We define MoDGF as a mixture of two DGF components. The two components represent predictive bimodality, while each component approximates the posterior distribution around each mode. Figure 2 (a) provides an overview of the posterior distillation with MoDGF and Figure 2 (b.1) and (b.2) illustrate the two applications of predictive bimodality that are explained in detail in Section 5.

Let $\mathcal{D}_{\text{design}} = \{\boldsymbol{x}_m\}_{m=1}^M$ denote a given set of design points where f_θ is evaluated for posterior distillation. That is, for given SGLD posterior samples $\theta_1, \ldots, \theta_N$, we use their logit outputs at the design points,

$$\boldsymbol{f}_{jm} = f_{\theta_j}(\boldsymbol{x}_m) \in \mathbb{R}^c, \qquad j = 1, \ldots, N, \quad m = 1, \ldots, M, \tag{11}$$

as observations for the posterior distillation.

Let $S(\boldsymbol{x}) \in \{1, 2\}$ denote the indicator of latent component (called the component indicator) at input $\boldsymbol{x}$. We consider

$$f(\boldsymbol{x}) = \sum_{s=1}^{2} \mathbb{I}\left(S(\boldsymbol{x}) = s\right) f^{(s)}(\boldsymbol{x}), \tag{12}$$

where $f^{(1)}$ and $f^{(2)}$ denote two functions for each component (called the component functions). Since the chance of predictive bimodality varies across inputs, we let the distribution of the component indicator depend on $\boldsymbol{x}$:

$$\mathbb{P}\left[S(\boldsymbol{x}) = s \mid \boldsymbol{x}\right] = \pi_s(\boldsymbol{x}; \gamma), \qquad s \in \{1, 2\}. \tag{13}$$

In turn, each component function is modeled by a DGF in Section 2.3. Specifically, for $s \in \{1, 2\}$,

$$f^{(s)}(\cdot) \sim \mathrm{GP}\left(\mu_s(\cdot), \Sigma_s(\cdot, \cdot)\right), \tag{14}$$

where $\mu_s = \mu_{\kappa_s}$ and $\Sigma_s(\boldsymbol{x}, \boldsymbol{x}') = \Phi_s(\boldsymbol{x})^\top \Phi_s(\boldsymbol{x}') \Lambda_s + \sigma_s^2 \mathbb{I}_c \mathbb{I}(\boldsymbol{x} = \boldsymbol{x}')$, with $\Phi_s = \Phi_{\kappa_s}$ and $\Lambda_s = L_s L_s^\top$. We denote the parameters of the s-th DGF component by $\psi_s = (\kappa_s, L_s, \sigma_s^2)$. As illustrated in Figure 2, we parameterize the two DGF components using a shared body network, followed by component-specific heads for μ_s and Φ_s.

The input-dependent mixture probabilities are parameterized as

$$\pi_1(\boldsymbol{x}; \gamma) = \frac{1}{2} + \frac{1}{2} \operatorname{sigmoid}\left(g_\gamma(\boldsymbol{x})\right), \qquad \pi_2(\boldsymbol{x}; \gamma) = 1 - \pi_1(\boldsymbol{x}; \gamma), \tag{15}$$

where $g_\gamma : \mathcal{X} \to \mathbb{R}$ is a scalar gating function applied to the same shared representation. We impose $\pi_1(\boldsymbol{x}; \gamma) > 1/2$ as an identifiability constraint to fix the component labeling during estimation. Accordingly, the first component is defined to have the larger mixture probability at each input.

For a given input $\boldsymbol{x}$, let $q_{\psi_s}(\boldsymbol{x})$ denote the distribution of $f^{(s)}(\boldsymbol{x})$ induced by the s-th DGF component:

$$q_{\psi_s}(\boldsymbol{x}) = \mathcal{N}_c\left(\mu_s(\boldsymbol{x}), \Sigma_s(\boldsymbol{x}, \boldsymbol{x})\right). \tag{16}$$

Let $\vartheta = (\psi_1, \psi_2, \gamma)$. Then, the resulting MoDGF distribution at $\boldsymbol{x}$ is given as

$$q_\vartheta(\boldsymbol{x}) = \sum_{s=1}^{2} \pi_s(\boldsymbol{x}; \gamma) q_{\psi_s}(\boldsymbol{x}). \tag{17}$$

4.2 Posterior Distillation via Structured Mean-Field EM

We estimate the parameters in MoDGF by maximizing the likelihood while treating $\boldsymbol{f}_{jm}, j = 1, \ldots, N, m = 1, \ldots, M$ as random observations from MoDGF as is done by Park et al. (2025b), and we resort to the EM algorithm for maximization. However, the standard EM is not directly applicable due to the mixture structure. To be more specific, for the jth posterior sample θ_j, let S_{jm} denote the component indicator at the mth design point $\boldsymbol{x}_m$, and let $\boldsymbol{Z}_j^{(s)}$ denote the DGF latent factor for component s, shared across design points. Unlike a single DGF, however, MoDGF includes the latent component indicators S_{jm} in addition to the DGF latent factors $\boldsymbol{Z}_j^{(s)}$. For each posterior sample j, the component-specific latent factor $\boldsymbol{Z}_j^{(s)}$ is shared across all of the design points. Therefore, the latent component indicators at different design points are not independent and so cannot be inferred separately, which makes the standard E-step to consider 2^M possible assignments of the latent component indicators. A detailed explanation is provided in Appendix D.2.

Though this exact computation for the E-step is infeasible for large M, we can use an approximated E-step. One possible approach is Monte Carlo EM (MC-EM) (Wei & Tanner, 1990; Sherman et al., 1999), which approximates the E-step using samples from the conditional distribution of the latent variables. An alternative is the structured mean-field EM, which approximates the conditional distribution of the latent variables by a product of simpler variational distributions (Saul & Jordan, 1996; Yuan & Neubauer, 2008). Among MC-EM and the structured mean-field EM, we use the latter one since it is computationally lighter while the accuracies are similar. Comparison of the MC-EM and structured mean-field EM are provided in Appendix D.6.

At iteration t, given the parameter $\vartheta^{(t)}$, the variational distribution for each posterior sample j is

$$r_j^{(t)} = \prod_{m=1}^{M} r_{jm}^{\mathrm{cat},(t)}(S_{jm}) \prod_{s=1}^{2} r_{js}^{\mathrm{lat},(t)}(\boldsymbol{Z}_j^{(s)}), \tag{18}$$

where $r_{jm}^{\mathrm{cat},(t)}$ is a categorical distribution for S_{jm} and $r_{js}^{\mathrm{lat},(t)}$ is a Gaussian distribution for $\boldsymbol{Z}_j^{(s)}$. We define the responsibility as $\tau_{jms}^{(t)} = r_{jm}^{\mathrm{cat},(t)}\{S_{jm} = s\}$. In the E-step, we alternately update the responsibilities $\tau_{jms}^{(t)}$ and the mean and covariance of $r_{js}^{\mathrm{lat},(t)}$ for $K = 20$ iterations. These alternating updates correspond to the coordinate ascent on the evidence lower bound (ELBO), where each set of variational factors is updated while the others are fixed (Blei et al., 2017).

Given the variational distributions from the E-step, the generalized M-step updates the model parameters by maximizing the following Q-function, ignoring terms independent of ϑ:

$$Q(\vartheta \mid \vartheta^{(t)}) = \sum_{j=1}^{N} \sum_{m=1}^{M} \sum_{s=1}^{2} \tau_{jms}^{(t)} \left[\log \pi_s(\boldsymbol{x}_m; \gamma) - \frac{c}{2} \log \sigma_s^2 - \frac{1}{2\sigma_s^2} R_{jms}^{(t)}(\psi_s)\right], \tag{19}$$

where

$$R_{jms}^{(t)}(\psi_s) = \mathbb{E}_{r_{js}^{\mathrm{lat},(t)}}\left[\left\| \boldsymbol{f}_{jm} - \mu_s(\boldsymbol{x}_m) - L_s \boldsymbol{Z}_j^{(s)} \Phi_s(\boldsymbol{x}_m) \right\|_2^2\right]. \tag{20}$$

We update $\vartheta^{(t+1)}$ by applying gradient ascent to $Q(\vartheta \mid \vartheta^{(t)})$. Details of the initialization, complete log-likelihood, and structured mean-field EM updates are provided in Appendix D.

5 APPLICATIONS OF THE COMPONENT-WISE POSTERIOR DISTRIBUTION

We can think that MoDGF decomposes the posterior distribution into a weighted sum of two distributions, and thus the two Gaussian processes involved in MoDGF are considered to be the component-wise posterior distribution. In this section, we explain how the component-wise posterior distributions are amply used for two applications - the specialist inference and OOD detection illustrated in Figure 2(b.1–b.2).

For an input $\boldsymbol{x}$, we define the component-wise predictive distribution as

$$\boldsymbol{p}_s(\boldsymbol{x}) = \mathbb{E}_{q_{\psi_s}(\boldsymbol{x})}\left[\mathrm{softmax}(f(\boldsymbol{x}))\right], \qquad s \in \{1, 2\}. \tag{21}$$

5.1 COMPONENT-GUIDED PAIRWISE SPECIALIST INFERENCE

Suppose that the two component-wise predictive distributions support different classes, revealing the class pair underlying their predictive disagreement. In this case, we can use a pairwise specialist to make the final prediction that selects one of the two classes.

Let $\widehat{y}_{a,b} : \mathcal{X} \to \{a, b\}$ be a specialist for $\{a, b\}$, which is a classifier only for two class labels a and b. Suppose that the specialists for all pairs of $\{a, b\}$ are given. For our case, we use a shared backbone with $\binom{c}{2}$ binary heads, one for each pair $\{a, b\}$. Each head $h_{a,b}$ is trained on data only from classes a and b. Below, we explain how we use these specialists with the component-wise predictive distributions for improving prediction accuracy further.

For component $s = 1, 2$, let its preferred class as $c_s(\boldsymbol{x}) = \arg\max_{k\in\{1,\ldots,c\}} \boldsymbol{p}_s(\boldsymbol{x})_k$. If $c_1(\boldsymbol{x}) = c_2(\boldsymbol{x})$, we let the predictive class $\widehat{y}(\boldsymbol{x}) = c_1(\boldsymbol{x})$. If $c_1(\boldsymbol{x}) \neq c_2(\boldsymbol{x})$, we use the specialist $\widehat{y}_{c_1(\boldsymbol{x}),c_2(\boldsymbol{x})}(\boldsymbol{x})$ to make the final decision. That is,

$$\widehat{y}(\boldsymbol{x}) = \begin{cases} c_1(\boldsymbol{x}), & c_1(\boldsymbol{x}) = c_2(\boldsymbol{x}), \\ \widehat{y}_{c_1(\boldsymbol{x}),c_2(\boldsymbol{x})}(\boldsymbol{x}), & c_1(\boldsymbol{x}) \neq c_2(\boldsymbol{x}). \end{cases} \tag{22}$$

5.2 WITHIN-COMPONENT EPISTEMIC OOD DETECTION

To use the component-wise posterior distributions for OOD detection, we propose a decomposition of the epistemic uncertainty of the MoDGF into within and between component epistemic uncertainty.

For the s-th component, we define its epistemic uncertainty as

$$U^{(s)}_{\mathrm{epistemic}}(\boldsymbol{x}) = \mathrm{H}\left[\boldsymbol{p}_s(\boldsymbol{x})\right] - \mathbb{E}_{q_{\psi_s}(\boldsymbol{x})}\left[\mathrm{H}\left[\mathrm{softmax}(f(\boldsymbol{x}))\right]\right]. \tag{23}$$

The following theorem gives the decomposition.

Theorem 1 (Epistemic uncertainty decomposition)**.** *The epistemic uncertainty of MoDGF satisfies*

$$U_{\mathrm{epistemic}}(\boldsymbol{x}) = U_{\mathrm{within}}(\boldsymbol{x}) + U_{\mathrm{between}}(\boldsymbol{x}), \tag{24}$$

$$U_{\mathrm{within}}(\boldsymbol{x}) = \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma) U^{(s)}_{\mathrm{epistemic}}(\boldsymbol{x}), \tag{25}$$

$$U_{\mathrm{between}}(\boldsymbol{x}) = \mathrm{JSD}_{\boldsymbol{\pi}(\boldsymbol{x})}\left(\boldsymbol{p}_1(\boldsymbol{x}), \boldsymbol{p}_2(\boldsymbol{x})\right). \tag{26}$$

where $\mathrm{JSD}_{\boldsymbol{\pi}(\boldsymbol{x})}$ *denotes the weighted Jensen–Shannon divergence with* $\boldsymbol{\pi}(\boldsymbol{x}) = (\pi_1(\boldsymbol{x};\gamma), \pi_2(\boldsymbol{x};\gamma))$.

Here, $U_{\mathrm{within}}(\boldsymbol{x})$ is the weighted average of the epistemic uncertainties of the two component-wise posterior distributions, whereas $U_{\mathrm{between}}(\boldsymbol{x})$ measures their predictive disagreement. The proof and the definition of the weighted Jensen–Shannon divergence (Lin, 1991) are provided in Appendix B.

Theorem 1 implies that large between-component epistemic uncertainty results in large epistemic uncertainty even for in-distribution inputs. For example, when the preferred classes of each component-wise posterior strongly differ (e.g. Golden retriever vs Labrador retriever), epistemic uncertainty becomes large even when an input is in-distribution. Thus, detecting OOD by simply looking at epistemic uncertainty may cause unnecessary false alarms. To reduce false alarms, we propose to use only $U_{\mathrm{within}}(\boldsymbol{x})$ as the **Within-Component Epistemic (WCE)** score for OOD detection. Empirical results in Section 6.3 confirm that this proposal really works.

6 EXPERIMENTS

We evaluate the proposed method from three perspectives. First, we evaluate whether MoDGF preserves the classification performance of SGLD on image classification and language reasoning tasks using accuracy, ECE, and NLL. Second, we evaluate whether component-guided pairwise specialist inference improves classification performance. Third, we examine whether within-component epistemic uncertainty provides better signals for OOD detection. Unless otherwise stated, all experiments are repeated with five different random seeds, and we report the mean and standard error.

6.1 PERFORMANCE ON CLASSIFICATION TASKS

We consider CIFAR-10, CIFAR-100, and Tiny-ImageNet for image classification, and ARC-Challenge (ARC-C), ARC-Easy (ARC-E), and OpenBookQA (OBQA) for language reasoning. We compare MoDGF with several distillation baselines, including BDK, Hydra, EnD^2, EDFM, and Single DGF (Korattikara Balan et al., 2015; Tran et al., 2020; Ryabinin et al., 2021; Park et al., 2025a;b). All distillation methods are trained using the same set of $N = 500$ SGLD posterior samples. For MoDGF, we use a latent factor dimension of $r = 16$ in each component in the main experiments. Further details on implementation and inference procedures are provided in Appendix E.

Table 1: Image classification results on CIFAR-10, CIFAR-100, and Tiny-ImageNet

	CIFAR-10			CIFAR-100			Tiny-ImageNet		
Method	Acc. (%) ↑	ECE (%) ↓	NLL ↓	Acc. (%) ↑	ECE (%) ↓	NLL ↓	Acc. (%) ↑	ECE (%) ↓	NLL ↓
SGLD	92.86 (0.12)	2.47 (0.09)	0.23 (0.01)	77.99 (0.19)	3.48 (0.19)	0.82 (0.01)	64.41 (0.58)	5.15 (0.87)	1.47 (0.01)
BDK	92.41 (0.18)	3.46 (0.11)	0.26 (0.01)	76.51 (0.17)	5.52 (0.39)	0.92 (0.02)	64.11 (0.30)	5.68 (0.41)	1.52 (0.03)
Hydra	92.41 (0.07)	2.91 (0.14)	0.28 (0.01)	73.89 (0.38)	5.11 (0.33)	1.13 (0.02)	62.14 (0.10)	6.19 (0.93)	1.65 (0.01)
EnD^2	92.50 (0.19)	3.10 (0.16)	0.25 (0.01)	76.54 (0.15)	5.01 (0.27)	0.92 (0.01)	63.51 (0.04)	6.23 (0.82)	1.53 (0.00)
EDFM	91.97 (0.11)	2.71 (0.18)	0.28 (0.01)	75.48 (0.11)	7.00 (0.07)	0.96 (0.02)	61.56 (0.07)	5.55 (0.23)	1.54 (0.01)
Single DGF	92.65 (0.17)	2.91 (0.09)	0.25 (0.01)	76.96 (0.35)	4.85 (0.35)	0.92 (0.01)	63.97 (0.02)	5.75 (0.77)	1.61 (0.00)
MoDGF (Ours)	**92.84 (0.12)**	**2.55 (0.10)**	**0.23 (0.00)**	**77.84 (0.14)**	**3.68 (0.26)**	**0.85 (0.01)**	**64.36 (0.03)**	**5.33 (0.36)**	**1.49 (0.00)**

Table 2: Language reasoning results on ARC-C, ARC-E, and OBQA.

	ARC-C			ARC-E			OBQA		
Method	Acc. (%) ↑	ECE (%) ↓	NLL ↓	Acc. (%) ↑	ECE (%) ↓	NLL ↓	Acc. (%) ↑	ECE (%) ↓	NLL ↓
SGLD	90.08 (0.83)	8.49 (0.38)	0.36 (0.09)	97.02 (0.43)	2.83 (0.29)	0.12 (0.02)	90.00 (0.16)	3.17 (0.18)	0.33 (0.06)
BDK	87.96 (0.83)	11.86 (0.81)	1.05 (0.27)	96.67 (0.35)	3.35 (0.23)	0.72 (0.08)	87.80 (0.21)	10.59 (2.13)	1.05 (0.11)
Hydra	89.97 (0.82)	9.31 (0.54)	0.77 (0.19)	96.96 (0.30)	2.98 (0.18)	0.24 (0.02)	89.48 (0.41)	5.82 (1.93)	0.38 (0.04)
EnD^2	89.41 (0.16)	9.91 (0.69)	0.78 (0.20)	96.55 (0.41)	3.23 (0.17)	0.23 (0.01)	89.20 (0.82)	8.06 (0.77)	0.45 (0.05)
EDFM	85.62 (0.42)	11.27 (1.03)	0.90 (0.04)	95.79 (0.21)	4.26 (0.18)	0.28 (0.02)	83.30 (0.10)	11.87 (0.42)	0.83 (0.01)
Single DGF	87.51 (1.51)	9.69 (0.83)	0.45 (0.02)	97.13 (0.20)	2.75 (0.35)	0.12 (0.01)	88.68 (0.81)	3.64 (0.74)	0.38 (0.04)
MoDGF (Ours)	**90.02 (0.51)**	**8.84 (1.37)**	**0.37 (0.03)**	**97.17 (0.14)**	**2.29 (0.57)**	**0.11 (0.01)**	**89.92 (0.12)**	**3.61 (0.46)**	**0.36 (0.03)**

Results. Tables 1 and 2 report the results on the image classification and language reasoning tasks, respectively. Across the six datasets, MoDGF achieves the highest mean accuracy and the lowest mean ECE and NLL among the compared posterior distillation methods, with predictive performance comparable to SGLD. We further examine whether MoDGF preserves the predictive distribution and mode structure of SGLD in Appendix F.

6.2 PAIRWISE SPECIALIST INFERENCE PERFORMANCE

We next evaluate whether the class pair identified by the two MoDGF components provides useful information for specialist inference. We compare against MoDGF without specialist inference and two existing specialist-based methods, Jung et al. and MS-NET-v2 (Jung et al., 2019; Chowdhury & Zhao, 2025). Our method selects a pairwise specialist according to the preferred classes of the two MoDGF components, whereas Jung et al. (2019) construct specialists for confusing class pairs identified from the base classifier, and MS-NET-v2 selects experts based on the top-2 predicted classes of the router network. Details of the baseline methods are provided in Appendix E.5.

Results. As shown in Table 3, component-guided specialist inference improves accuracy by 2.69 and 3.46 percentage points on CIFAR-10 and CIFAR-100, respectively. These improvements are larger than those obtained by Jung et al. and MS-NET-v2 on both datasets. These results suggest that the class pair identified by the two components is involved in the predictive ambiguity of the current input and provides useful information for selecting a pairwise specialist.

Table 3: Classification accuracy (%) and improvement over Base (No Specialist). Results are averaged over five random seeds, with standard errors for the improvements shown in parentheses.

	CIFAR-10		CIFAR-100	
Method	Acc. ↑	ΔAcc. ↑	Acc. ↑	ΔAcc. ↑
MoDGF (No Specialist)	92.84	-	77.84	-
Jung et al. (Jung et al., 2019)	94.83	+1.99 (0.20)	80.22	+2.38 (0.56)
MS-NET-v2 (Chowdhury & Zhao, 2025)	94.96	+2.12 (0.22)	80.07	+2.23 (0.29)
Component-guided Specialist (Ours)	**95.53**	**+2.69 (0.29)**	**81.30**	**+3.46 (0.36)**

6.3 OOD Detection

Finally, we evaluate whether within-component epistemic uncertainty provides a useful score for OOD detection. We use CIFAR-10 and CIFAR-100 as ID datasets and Tiny-ImageNet and SVHN (Netzer et al., 2011) as OOD datasets. We use overall epistemic uncertainty as the OOD score for the baseline methods, and for MoDGF, we additionally evaluate the proposed within-component epistemic uncertainty $U_{\text{within}}(\boldsymbol{x})$. Further experimental details are provided in Appendix E.6.

Table 4: OOD detection performance measured by AUROC (%). CIFAR-10 and CIFAR-100 are used as ID datasets, and Tiny-ImageNet and SVHN are used as OOD datasets.

	Tiny-ImageNet		SVHN	
Method	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100
SGLD	87.06 (0.13)	80.90 (0.81)	89.18 (0.95)	82.13 (2.78)
Hydra	81.71 (1.08)	73.44 (4.79)	88.55 (0.32)	77.05 (5.66)
EnD^2	86.81 (0.62)	77.97 (0.90)	87.35 (2.62)	80.88 (1.07)
EDFM	85.53 (0.31)	78.90 (0.62)	87.80 (2.93)	79.51 (5.12)
Single DGF	86.42 (0.23)	76.63 (0.92)	87.87 (1.35)	79.04 (5.13)
MoDGF (Epistemic)	86.76 (0.21)	80.90 (1.16)	88.61 (0.74)	81.77 (2.83)
MoDGF (WCE)	**87.30 (0.21)**	**82.40 (1.22)**	**90.80 (0.92)**	**83.50 (2.55)**

Results. As shown in Table 4, using overall epistemic uncertainty, MoDGF achieves AUROC comparable to SGLD across the four ID–OOD dataset pairs. Using WCE consistently improves the OOD detection performance of MoDGF across all four pairs and achieves the highest AUROC among the compared methods in every setting. These results suggest that excluding between-component disagreement can provide a more useful OOD score than overall epistemic uncertainty.

6.4 Additional Experiments

We further examine the sensitivity of MoDGF to its main design choices through ablation studies on the latent factor dimension, the number of mean-field iterations, and the choice of posterior sampling method. We also analyze the computational costs of MoDGF in terms of training time and FLOPs. Finally, we consider active learning as an additional application of the uncertainty decomposition, using overall, within-component, and between-component epistemic uncertainty as acquisition scores. The results are provided in Appendices G, I, and H, respectively.

7 Conclusion

We examined predictive bimodality induced by SGLD posterior samples and proposed MoDGF, a posterior distillation method that preserves this structure using two DGF components. These component-wise posterior distributions provide information that we use for component-guided pairwise specialist inference and within-component epistemic OOD detection. Extensive experiments show that MoDGF achieves predictive performance comparable to SGLD while improving over Single DGF. Moreover, the class pair identified by the two components improves specialist inference, while using within-component epistemic uncertainty as the OOD score improves OOD detection. These results suggest that preserving predictive mode structure during posterior distillation provides useful information for both prediction and uncertainty quantification.

In this work, we focused on bimodal predictive structure induced by SGLD posterior samples. It would be valuable to extend MoDGF to more general multimodal predictive distributions and to examine whether similar predictive mode structure arises under other posterior sampling methods. This could provide a broader understanding of how predictive mode structure can be used for prediction and uncertainty quantification in Bayesian neural networks.

AI USE STATEMENT

In this work, we used generative AI tools to assist with writing and reviewing the manuscript, synthesizing and organizing experimental results, and searching for relevant literature and research materials. We reviewed and verified all AI-assisted content before including it in the paper. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the details needed to reproduce our method and experiments in the Appendix. Appendix D provides the details of MoDGF, including its initialization, structured mean-field E-step, generalized M-step, and overall training algorithm. Appendix E provides the experimental settings, including the datasets and evaluation metrics, model architectures, SGLD sampling procedure, baseline implementations, specialist inference methods, and OOD detection setup. Additional analyses of computational costs and the main design choices of MoDGF are provided in Appendices I and G, respectively.

REFERENCES

Keith M. Ashman, Christina M. Bird, and Stephen E. Zepf. Detecting bimodality in astronomical datasets. *The Astronomical Journal*, 108:2348–2361, 1994. doi: 10.1086/117248.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017. doi: 10.1080/01621459.2017.1285773. URL https://doi.org/10.1080/01621459.2017.1285773. arXiv:1601.00670.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Md Intisar Chowdhury and Qiangfu Zhao. MS-NET-v2: Modular selective network optimized by systematic generation of expert modules. *International Journal of Machine Learning and Cybernetics*, 16(7–8):5131–5160, 2025. doi: 10.1007/s13042-025-02563-7.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux - effortless bayesian deep learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 20089–20103. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/a7c9585703d275249f30a088cebba0ad-Paper.pdf.

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 *Proceedings of the 35th International Conference on Machine Learning*, pp. 1184–1193, 2018.

Katharine Fisher and Youssef Marzouk. Can bayesian neural networks make confident predictions? *arXiv preprint arXiv:2501.11773*, 2025. doi: 10.48550/arXiv.2501.11773. URL https://arxiv.org/abs/2501.11773. arXiv:2501.11773; Mathematics of Modern Machine Learning Workshop at NeurIPS 2024.

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

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1183–1192, 2017.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330, 2017. URL https://arxiv.org/abs/1706.04599. arXiv:1706.04599.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531. arXiv:1503.02531.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Pavel Izmailov, Sharad Vikram, Matthew D. Hoffman, and Andrew Gordon Wilson. What are bayesian neural network posteriors really like? In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4629–4640. PMLR, 2021.

Jee-weon Jung, HeeSoo Heo, Hye-jin Shim, and Ha-Jin Yu. Distilling the knowledge of specialist deep neural networks in acoustic scene classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pp. 114–118, New York University, NY, USA, October 2019.

Robert E. Kass and Adrian E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995. doi: 10.1080/01621459.1995.10476572.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Anoop Korattikara Balan, Vivek Rathod, Kevin P. Murphy, and Max Welling. Bayesian dark knowledge. In *Advances in Neural Information Processing Systems*, volume 28, 2015. URL https://arxiv.org/abs/1506.04416. arXiv:1506.04416.

Ranganath Krishnan, Pi Esposito, and Mahesh Subedar. Bayesian-torch: Bayesian neural network layers for uncertainty estimation, January 2022. URL https://doi.org/10.5281/zenodo.5908307.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Olivier Laurent, Emanuel Aldea, and Gianni Franchi. A symmetry-aware exploration of bayesian neural network posteriors. In *International Conference on Learning Representations*, 2024.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. CS231N course project report, Stanford University, 2015. URL https://cs231n.stanford.edu/reports/2015/pdfs/yle_project.pdf.

Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016. doi: 10.1609/aaai.v30i1.10200. URL https://doi.org/10.1609/aaai.v30i1.10200. arXiv:1512.07666.

Jianhua Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. Ensemble distribution distillation. *arXiv preprint arXiv:1905.00076*, 2019. URL https://arxiv.org/abs/1905.00076. arXiv:1905.00076.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Bálint Mucsányi, Michael Kirchhof, and Seong Joon Oh. Benchmarking uncertainty disentanglement: Specialized uncertainties for specialized tasks. In *Advances in Neural Information Processing Systems: Datasets and Benchmarks Track*, volume 37, 2024. doi: 10.52202/079017-1614. URL https://doi.org/10.52202/079017-1614.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, pp. 2901–2907, 2015.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1906.02530. arXiv:1906.02530.

Jonggeon Park, Giung Nam, Hyunsu Kim, Jongmin Yoon, and Juho Lee. Ensemble distribution distillation via flow matching. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48170–48191. PMLR, 2025a.

Sehyun Park, Jongjin Lee, Yunseop Shin, Ilsang Ohn, and Yongdai Kim. Knowledge distillation of uncertainty using deep latent factor model. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. URL https://arxiv.org/abs/2510.19290. arXiv:2510.19290.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *International conference on learning representations*, 2018.

Max Ryabinin, Andrey Malinin, and Mark Gales. Scaling ensemble distribution distillation to many classes with proxy targets. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://arxiv.org/abs/2105.06987. arXiv:2105.06987.

Lawrence K. Saul and Michael I. Jordan. Exploiting tractable substructures in intractable networks. In *Advances in Neural Information Processing Systems*, volume 8, pp. 486–492. MIT Press, 1996.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978. doi: 10.1214/aos/1176344136.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Robert P. Sherman, Yu-Yun K. Ho, and Siddhartha R. Dalal. Conditions for convergence of Monte Carlo EM sequences with an application to product diffusion modeling. *The Econometrics Journal*, 2(2):248–267, 1999. doi: 10.1111/1368-423X.00032.

Ondrej Šuch, Peter Tarábek, Katarína Bachratá, and Andrea Tinajová. Pairwise coupling of convolutional neural networks for the better explainability of classification systems. *Applied Sciences*, 15(22):12007, 2025. doi: 10.3390/app152212007. URL https://doi.org/10.3390/app152212007.

Linh Tran, Bastiaan S. Veeling, Kevin Roth, Jakub Swiatkowski, Joshua V. Dillon, Jasper Snoek, Stephan Mandt, Tim Salimans, Sebastian Nowozin, and Rodolphe Jenatton. Hydra: Preserving ensemble diversity for model distillation. *arXiv preprint arXiv:2001.04694*, 2020. URL https://arxiv.org/abs/2001.04694. arXiv:2001.04694.

Meet Vadera, Brian Jalaian, and Benjamin Marlin. Generalized bayesian posterior expectation distillation for deep neural networks. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, volume 124 of *Proceedings of Machine Learning Research*, pp. 719–728, 2020. URL https://arxiv.org/abs/2005.08110. arXiv:2005.08110.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008. doi: 10.1561/2200000001.

Kuan-Chieh Wang, Paul Vicol, James Lucas, Li Gu, Roger Grosse, and Richard Zemel. Adversarial distillation of bayesian neural network posteriors. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 5190–5199, 2018. URL https://arxiv.org/abs/1806.10317. arXiv:1806.10317.

Greg C. G. Wei and Martin A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85 (411):699–704, 1990. doi: 10.1080/01621459.1990.10474930.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 681–688, 2011. URL https://icml.cc/2011/papers/398_icmlpaper.pdf.

Andrew G. Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 4697–4708, 2020.

Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. HD-CNN: Hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2740–2748, 2015. URL https://openaccess.thecvf.com/content_iccv_2015/html/Yan_HD-CNN_Hierarchical_Deep_ICCV_2015_paper.html.

Chao Yuan and Claus Neubauer. Variational mixture of gaussian process experts. In *Advances in Neural Information Processing Systems*, volume 21, 2008. URL https://proceedings.neurips.cc/paper/2008/hash/f4b9ec30ad9f68f89b29639786cb62ef-Abstract.html. NeurIPS paper hash:f4b9ec30ad9f68f89b29639786cb62ef.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient MCMC for bayesian deep learning. In *International Conference on Learning Representations*, 2020.

APPENDIX

A A LIMITATION OF A SINGLE-GAUSSIAN APPROXIMATION

To examine the effect of a single Gaussian approximation, we consider binary classification and let P denote the distribution of the scalar logit $f(\boldsymbol{x})$, where $p(y = 1 \mid \boldsymbol{x}, f) = \mathrm{sigmoid}(f(\boldsymbol{x}))$. For a given P, let $U_{\mathrm{epistemic}}(\boldsymbol{x}; P)$ denote the epistemic uncertainty in equation (6) under P, with the sigmoid replacing the softmax. The following theorem shows that a single Gaussian can underestimate epistemic uncertainty for a class of bimodal logit distributions.

Theorem 2. *Consider binary classification at a fixed input $\boldsymbol{x}$. At this input, suppose the distribution of $f(\boldsymbol{x})$ is*

$$P_{a,\sigma,\pi} = \pi\mathcal{N}(-a, \sigma^2) + (1-\pi)\mathcal{N}(a, \sigma^2), \qquad a > 0, \quad \pi \in (0,1), \quad \sigma > 0. \tag{27}$$

Let $\mathcal{G}$ denote the family of univariate Gaussian distributions and define

$$Q^\star_{a,\sigma,\pi} \in \arg\min_{Q \in \mathcal{G}} \mathrm{KL}\left(P_{a,\sigma,\pi} \,\|\, Q\right). \tag{28}$$

For any fixed $\sigma > 0$ and sufficiently large a, there exists $\epsilon_{a,\sigma} > 0$ such that, whenever $\left|\pi - \frac{1}{2}\right| < \epsilon_{a,\sigma}$, we have

$$U_{\mathrm{epistemic}}\left(\boldsymbol{x}; P_{a,\sigma,\pi}\right) > U_{\mathrm{epistemic}}\left(\boldsymbol{x}; Q^\star_{a,\sigma,\pi}\right). \tag{29}$$

Proof. For a distribution R of $f(\boldsymbol{x})$, the epistemic uncertainty is

$$U_{\mathrm{epistemic}}(\boldsymbol{x}; R) = \mathrm{H}\left(\mathbb{E}_R[\mathrm{sigmoid}(f(\boldsymbol{x}))]\right) - \mathbb{E}_R\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right]. \tag{30}$$

We first identify the Gaussian approximation $Q^\star_{a,\sigma,\pi}$. By the standard moment-matching property of the KL projection onto the Gaussian family (Wainwright & Jordan, 2008), $Q^\star_{a,\sigma,\pi}$ has the same mean and variance as $P_{a,\sigma,\pi}$. Under $P_{a,\sigma,\pi}$, $\mathbb{E}[f(\boldsymbol{x})] = (1-2\pi)a$ and $\mathrm{Var}[f(\boldsymbol{x})] = \sigma^2 + 4a^2\pi(1-\pi)$. Therefore,

$$Q^\star_{a,\sigma,\pi} = \mathcal{N}\left((1-2\pi)a, \sigma^2 + 4a^2\pi(1-\pi)\right). \tag{31}$$

We prove the result in two steps. First, we establish the strict inequality for the symmetric case $\pi = 1/2$. This case is useful because both $P_{a,\sigma,1/2}$ and $Q^\star_{a,\sigma,1/2}$ are symmetric around zero, which makes their predictive mean probabilities identical. We then use continuity to show that there exists $\epsilon_{a,\sigma} > 0$ such that

$$U_{\mathrm{epistemic}}(\boldsymbol{x}; P_{a,\sigma,\pi}) > U_{\mathrm{epistemic}}(\boldsymbol{x}; Q^\star_{a,\sigma,\pi}) \tag{32}$$

for all π satisfying $|\pi - 1/2| < \epsilon_{a,\sigma}$.

For $\pi = 1/2$, we have

$$P_{a,\sigma,1/2} = \frac{1}{2}\mathcal{N}(-a, \sigma^2) + \frac{1}{2}\mathcal{N}(a, \sigma^2), \quad \text{and} \quad Q^\star_{a,\sigma,1/2} = \mathcal{N}(0, a^2 + \sigma^2). \tag{33}$$

Both distributions are symmetric around zero. Therefore, the predictive probabilities are symmetric around $1/2$, and

$$\mathbb{E}_{P_{a,\sigma,1/2}}[\mathrm{sigmoid}(f(\boldsymbol{x}))] = \mathbb{E}_{Q^\star_{a,\sigma,1/2}}[\mathrm{sigmoid}(f(\boldsymbol{x}))] = \frac{1}{2}. \tag{34}$$

Thus,

$$\begin{aligned} U_{\mathrm{epistemic}}(\boldsymbol{x}; P_{a,\sigma,1/2}) - U_{\mathrm{epistemic}}(\boldsymbol{x}; Q^\star_{a,\sigma,1/2}) = \mathbb{E}_{Q^\star_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] \\ - \mathbb{E}_{P_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right]. \end{aligned} \tag{35}$$

We now compare the two entropy terms in equation (35). The function $\mathrm{H}(\mathrm{sigmoid}(t))$ is even and strictly decreasing for $t \geq 0$. For $t \geq 0$,

$$\mathrm{H}(\mathrm{sigmoid}(t)) = \log(1 + e^{-t}) + \frac{t}{1 + e^t} \leq (1+t)e^{-t}. \tag{36}$$

We first consider the entropy term under $P_{a,\sigma,1/2}$. Let $Z \sim \mathcal{N}(0,1)$. Since $\mathrm{H}(\mathrm{sigmoid}(t))$ is even,

$$\mathbb{E}_{P_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] = \mathbb{E}\left[\mathrm{H}\left(\mathrm{sigmoid}(a+\sigma Z)\right)\right]. \tag{37}$$

When $Z \geq -a/(2\sigma)$, we have $a + \sigma Z \geq a/2$, whereas on the complementary event the entropy is bounded above by $\log 2$. Therefore,

$$\begin{aligned}\mathbb{E}_{P_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] &\leq \mathrm{H}\left(\mathrm{sigmoid}(a/2)\right) + (\log 2)\mathbb{P}\left[Z < -\frac{a}{2\sigma}\right] \\ &\leq \left(1+\frac{a}{2}\right)e^{-a/2} + (\log 2)\exp\left(-\frac{a^2}{8\sigma^2}\right). \end{aligned} \tag{38}$$

For fixed $\sigma > 0$, this gives $\mathbb{E}_{P_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] = o(a^{-1})$ as $a \to \infty$.

We next consider the entropy term under $Q^\star_{a,\sigma,1/2}$. Let $v_a = a^2 + \sigma^2$. Under $Q^\star_{a,\sigma,1/2}$, we can write $f(\boldsymbol{x}) = \sqrt{v_a}Z$, where $Z \sim \mathcal{N}(0,1)$. Since $\mathrm{H}(\mathrm{sigmoid}(t)) \geq \mathrm{H}(\mathrm{sigmoid}(1))$ for $|t| \leq 1$,

$$\begin{aligned}\mathbb{E}_{Q^\star_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] &= \mathbb{E}\left[\mathrm{H}\left(\mathrm{sigmoid}(\sqrt{v_a}Z)\right)\right] \\ &\geq \mathbb{E}\left[\mathrm{H}\left(\mathrm{sigmoid}(\sqrt{v_a}Z)\right)\mathbb{I}\left(|Z| \leq \frac{1}{\sqrt{v_a}}\right)\right] \\ &\geq \mathrm{H}(\mathrm{sigmoid}(1))\mathbb{P}\left[|Z| \leq \frac{1}{\sqrt{v_a}}\right] \qquad (39) \\ &\geq \frac{2\mathrm{H}(\mathrm{sigmoid}(1))e^{-1/2}}{\sqrt{2\pi v_a}}, \qquad (40)\end{aligned}$$

where the last inequality follows since, for $v_a \geq 1$, the standard Gaussian density is bounded below by $\phi(1) = e^{-1/2}/\sqrt{2\pi}$ on $[-1/\sqrt{v_a}, 1/\sqrt{v_a}]$. Since $v_a = a^2 + \sigma^2$ and σ is fixed, $\sqrt{v_a}/a = \sqrt{1+\sigma^2/a^2} \to 1$ as $a \to \infty$. Therefore, $\mathbb{E}_{Q^\star_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] = \Omega(a^{-1})$.

Thus, for sufficiently large a,

$$\mathbb{E}_{Q^\star_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right] > \mathbb{E}_{P_{a,\sigma,1/2}}\left[\mathrm{H}\left(\mathrm{sigmoid}(f(\boldsymbol{x}))\right)\right]. \tag{41}$$

By equation (35),

$$U_{\mathrm{epistemic}}(\boldsymbol{x}; P_{a,\sigma,1/2}) > U_{\mathrm{epistemic}}(\boldsymbol{x}; Q^\star_{a,\sigma,1/2}). \tag{42}$$

Finally, we extend the result to a neighborhood of $\pi = 1/2$. Define

$$D_{a,\sigma}(\pi) = U_{\mathrm{epistemic}}(\boldsymbol{x}; P_{a,\sigma,\pi}) - U_{\mathrm{epistemic}}(\boldsymbol{x}; Q^\star_{a,\sigma,\pi}). \tag{43}$$

The function $D_{a,\sigma}(\pi)$ is continuous in π, since the mixture weights and the mean and variance of $Q^\star_{a,\sigma,\pi}$ are continuous in π, and the sigmoid function and binary entropy are bounded and continuous. For sufficiently large a, equation (42) gives $D_{a,\sigma}(1/2) > 0$. Therefore, there exists $\epsilon_{a,\sigma} > 0$ such that

$$|D_{a,\sigma}(\pi) - D_{a,\sigma}(1/2)| < \frac{1}{2}D_{a,\sigma}(1/2) \tag{44}$$

whenever $|\pi - 1/2| < \epsilon_{a,\sigma}$. It follows that

$$D_{a,\sigma}(\pi) > \frac{1}{2}D_{a,\sigma}(1/2) > 0, \tag{45}$$

which proves the theorem.

□

B Derivation of the Epistemic Uncertainty Decomposition

We provide the derivation of Theorem 1. For a fixed input $\boldsymbol{x}$, define the mixture-averaged predictive probability vector as

$$\overline{\boldsymbol{p}}(\boldsymbol{x}) = \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)\boldsymbol{p}_s(\boldsymbol{x}). \tag{46}$$

Then, the epistemic uncertainty of the MoDGF predictive distribution is

$$U_{\text{epistemic}}(\boldsymbol{x}) = \mathrm{H}\left[\overline{\boldsymbol{p}}(\boldsymbol{x})\right] - \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)\mathbb{E}_{q_{\psi_s}(\boldsymbol{x})}\left[\mathrm{H}\left[\mathrm{softmax}(f(\boldsymbol{x}))\right]\right]. \tag{47}$$

Adding and subtracting $\sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)\mathrm{H}[\boldsymbol{p}_s(\boldsymbol{x})]$ gives

$$\begin{aligned} U_{\text{epistemic}}(\boldsymbol{x}) = &\sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)\left\{\mathrm{H}\left[\boldsymbol{p}_s(\boldsymbol{x})\right] - \mathbb{E}_{q_{\psi_s}(\boldsymbol{x})}\left[\mathrm{H}\left[\mathrm{softmax}(f(\boldsymbol{x}))\right]\right]\right\} \\ &+ \mathrm{H}\left[\overline{\boldsymbol{p}}(\boldsymbol{x})\right] - \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)\mathrm{H}\left[\boldsymbol{p}_s(\boldsymbol{x})\right]. \end{aligned} \tag{48}$$

By equation (23), the first term is

$$\sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)U^{(s)}_{\text{epistemic}}(\boldsymbol{x}) = U_{\text{within}}(\boldsymbol{x}). \tag{49}$$

We next consider the second term. Let $\boldsymbol{\pi}(\boldsymbol{x}) = (\pi_1(\boldsymbol{x};\gamma), \pi_2(\boldsymbol{x};\gamma))$. The weighted Jensen–Shannon divergence (Lin, 1991) between the two component-wise predictive probability vectors is defined as

$$\mathrm{JSD}_{\boldsymbol{\pi}(\boldsymbol{x})}\left(\boldsymbol{p}_1(\boldsymbol{x}), \boldsymbol{p}_2(\boldsymbol{x})\right) = \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)D_{\mathrm{KL}}\left(\boldsymbol{p}_s(\boldsymbol{x}) \| \overline{\boldsymbol{p}}(\boldsymbol{x})\right), \tag{50}$$

where, for categorical probability vectors $\boldsymbol{p}$ and $\boldsymbol{q}$,

$$D_{\mathrm{KL}}(\boldsymbol{p}\|\boldsymbol{q}) = \sum_{k=1}^{c} p_k \log \frac{p_k}{q_k}. \tag{51}$$

Using the definition of the KL divergence, we have

$$\begin{aligned} \mathrm{JSD}_{\boldsymbol{\pi}(\boldsymbol{x})}\left(\boldsymbol{p}_1(\boldsymbol{x}), \boldsymbol{p}_2(\boldsymbol{x})\right) &= \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma) \sum_{k=1}^{c} p_{s,k}(\boldsymbol{x}) \log \frac{p_{s,k}(\boldsymbol{x})}{\overline{p}_k(\boldsymbol{x})} \\ &= \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma) \sum_{k=1}^{c} p_{s,k}(\boldsymbol{x}) \log p_{s,k}(\boldsymbol{x}) - \sum_{k=1}^{c} \overline{p}_k(\boldsymbol{x}) \log \overline{p}_k(\boldsymbol{x}) \\ &= \mathrm{H}\left[\overline{\boldsymbol{p}}(\boldsymbol{x})\right] - \sum_{s=1}^{2} \pi_s(\boldsymbol{x};\gamma)\mathrm{H}\left[\boldsymbol{p}_s(\boldsymbol{x})\right]. \end{aligned} \tag{52}$$

Therefore, the second term in equation (48) is $U_{\text{between}}(\boldsymbol{x})$. Combining the two terms gives

$$U_{\text{epistemic}}(\boldsymbol{x}) = U_{\text{within}}(\boldsymbol{x}) + U_{\text{between}}(\boldsymbol{x}), \tag{53}$$

which proves Theorem 1.

C IDENTIFYING PREDICTIVE BIMODALITY

We identify predictive bimodality by examining the raw logits of each class across stochastic gradient Langevin dynamics (SGLD) posterior samples. For each input $\boldsymbol{x}$ and class $y \in \{1, \ldots, c\}$, we consider

$$\left\{ f_{\theta_j, y}(\boldsymbol{x}) \right\}_{j=1}^{N}. \tag{54}$$

We fit a one-component Gaussian model and a two-component Gaussian mixture model separately to the logit samples of each class.

Let $\mathrm{BIC}_{1,y}(\boldsymbol{x})$ and $\mathrm{BIC}_{2,y}(\boldsymbol{x})$ denote the corresponding BIC values (Schwarz, 1978), and define

$$\Delta\mathrm{BIC}_y(\boldsymbol{x}) = \mathrm{BIC}_{1,y}(\boldsymbol{x}) - \mathrm{BIC}_{2,y}(\boldsymbol{x}). \tag{55}$$

We use $\Delta\mathrm{BIC}_y(\boldsymbol{x}) \geq 10$ as evidence in support of the two-component model (Kass & Raftery, 1995).

We additionally require the two fitted components to be sufficiently separated. For a fixed input $\boldsymbol{x}$ and class y, let $\mu_{y,1}$ and $\mu_{y,2}$ denote their means and $\sigma_{y,1}^2$ and $\sigma_{y,2}^2$ their variances. Following the normalized separation criterion of Ashman et al. (1994), we define

$$D_y(\boldsymbol{x}) = \frac{|\mu_{y,1} - \mu_{y,2}|}{\sqrt{(\sigma_{y,1}^2 + \sigma_{y,2}^2)/2}}. \tag{56}$$

We also let $\pi_{y,1}(\boldsymbol{x})$ and $\pi_{y,2}(\boldsymbol{x})$ denote the fitted mixture weights and require

$$\max_{s \in \{1,2\}} \pi_{y,s}(\boldsymbol{x}) \leq 0.95. \tag{57}$$

This condition excludes cases in which one fitted component accounts for almost all posterior samples.

The BIC condition favors a two-component fit over a single Gaussian, the separation condition requires the fitted components to be sufficiently separated, and the mixture-weight condition excludes cases with a negligible fitted component. Consistent with the definition in Section 3, we classify an input $\boldsymbol{x}$ as exhibiting predictive bimodality if at least one class satisfies all three conditions:

$$\exists\, y \in \{1, \ldots, c\} \quad \text{such that} \quad \Delta\mathrm{BIC}_y(\boldsymbol{x}) \geq 10, \quad D_y(\boldsymbol{x}) > 2, \quad \text{and} \quad \max_{s \in \{1,2\}} \pi_{y,s}(\boldsymbol{x}) \leq 0.95. \tag{58}$$

Three-component structure. We additionally examine whether the predictive logit distributions exhibit evidence of more than two components. For each input $\boldsymbol{x}$ and class y, we fit a three-component Gaussian mixture model and compare it with the two-component model using BIC. Let $\mathrm{BIC}_{3,y}(\boldsymbol{x})$ denote the BIC of the three-component model, and define

$$\Delta\mathrm{BIC}_{3,y}(\boldsymbol{x}) = \mathrm{BIC}_{2,y}(\boldsymbol{x}) - \mathrm{BIC}_{3,y}(\boldsymbol{x}). \tag{59}$$

We use $\Delta\mathrm{BIC}_{3,y}(\boldsymbol{x}) \geq 10$ as evidence in support of the three-component model. Using this criterion, we find that cases favoring the three-component model are rarely observed.

D Details of the Structured Mean-Field EM algorithm for MoDGF

This section provides details of the structured mean-field EM algorithm used to estimate the MoDGF parameters. We first derive the complete log-likelihood by introducing the latent component indicators and DGF latent factors. We then describe the structured mean-field E-step and the generalized M-step used to update the latent-variable distributions and model parameters, respectively. Finally, we describe the initialization procedure and summarize the overall MoDGF training algorithm.

D.1 Complete Log-Likelihood of MoDGF

For each posterior sample j, let $\boldsymbol{F}_j = (\boldsymbol{f}_{j1}, \ldots, \boldsymbol{f}_{jM})$ denote its outputs evaluated at the M design points. Let $S_{jm} \in \{1, 2\}$ denote the latent component indicator at design point $\boldsymbol{x}_m$, and let $\boldsymbol{Z}_j^{(s)}$ denote the DGF latent factor for component s, shared across design points. We assume

$$\boldsymbol{Z}_j^{(s)} \sim \mathrm{MN}_{c,r}\left(\boldsymbol{0}, \mathbb{I}_c, \mathbb{I}_r\right), \qquad j = 1, \ldots, N, \quad s \in \{1, 2\}. \tag{60}$$

Conditional on the latent component indicator and the corresponding latent factor, the output at each design point follows

$$\boldsymbol{f}_{jm} \mid S_{jm} = s, \boldsymbol{Z}_j^{(s)} \sim \mathcal{N}_c\left(\mu_s(\boldsymbol{x}_m) + L_s \boldsymbol{Z}_j^{(s)} \Phi_s(\boldsymbol{x}_m), \sigma_s^2 \mathbb{I}_c\right). \tag{61}$$

Since the same latent factor $\boldsymbol{Z}_j^{(s)}$ is shared across design points, the outputs from the same posterior sample preserve their dependence across inputs.

Let $\boldsymbol{F}_{1:N} = \{\boldsymbol{F}_j\}_{j=1}^N$, $\mathcal{S} = \{S_{jm} : j = 1, \ldots, N,\ m = 1, \ldots, M\}$, and $\mathcal{Z} = \{\boldsymbol{Z}_j^{(s)} : j = 1, \ldots, N,\ s = 1, 2\}$. The complete log-likelihood of the observed SGLD outputs and the latent variables is

$$\begin{aligned}\ell_{\mathrm{com}}\left(\vartheta; \boldsymbol{F}_{1:N}, \mathcal{S}, \mathcal{Z}\right) = \sum_{j=1}^{N} \sum_{m=1}^{M} \sum_{s=1}^{2} \mathbb{I}(S_{jm} = s) \Bigg[&\log \pi_s(\boldsymbol{x}_m; \gamma) - \frac{c}{2} \log(2\pi\sigma_s^2) \\ &- \frac{1}{2\sigma_s^2} \left\| \boldsymbol{f}_{jm} - \mu_s(\boldsymbol{x}_m) - L_s \boldsymbol{Z}_j^{(s)} \Phi_s(\boldsymbol{x}_m) \right\|_2^2 \Bigg] \\ - \frac{1}{2} \sum_{j=1}^{N} \sum_{s=1}^{2} &\left[\left\| \boldsymbol{Z}_j^{(s)} \right\|_F^2 + cr \log(2\pi) \right].\end{aligned} \tag{62}$$

For a single DGF, the conditional distribution of the DGF latent factor is available in closed form. In MoDGF, however, the component-specific latent factors are shared across design points, so the latent component assignments cannot be inferred independently. As a result, the exact E-step requires considering 2^M possible assignments of the latent component indicators for each posterior sample. We first explain this intractability and then introduce a structured mean-field approximation (Saul & Jordan, 1996).

D.2 Intractability of the Exact E-Step for MoDGF

For each posterior sample j, let

$$\boldsymbol{S}_j = (S_{j1}, \ldots, S_{jM}) \in \{1, 2\}^M \tag{63}$$

denote the component assignments over the M design points. The exact E-step requires the joint conditional distribution of $\boldsymbol{S}_j$, $\boldsymbol{Z}_j^{(1)}$, and $\boldsymbol{Z}_j^{(2)}$ given $\boldsymbol{F}_j$.

At iteration t, the exact Q-function is

$$Q_{\mathrm{exact}}\left(\vartheta \mid \vartheta^{(t)}\right) = \sum_{j=1}^{N} \mathbb{E}_{\boldsymbol{S}_j, \boldsymbol{Z}_j^{(1)}, \boldsymbol{Z}_j^{(2)} \mid \boldsymbol{F}_j, \vartheta^{(t)}} \left[\ell_{\mathrm{com},j}(\vartheta)\right], \tag{64}$$

where $\ell_{\mathrm{com},j}$ denotes the complete log-likelihood for the jth posterior sample. Expanding the expectation over $\boldsymbol{S}_j$ gives

$$Q_{\mathrm{exact}}\left(\vartheta \mid \vartheta^{(t)}\right) = \sum_{j=1}^{N} \sum_{\boldsymbol{s}\in\{1,2\}^M} \mathbb{P}\left[\boldsymbol{S}_j = \boldsymbol{s} \mid \boldsymbol{F}_j, \vartheta^{(t)}\right] \times \mathbb{E}_{\boldsymbol{Z}_j^{(1)},\boldsymbol{Z}_j^{(2)}\mid \boldsymbol{S}_j=\boldsymbol{s},\boldsymbol{F}_j,\vartheta^{(t)}}\left[\ell_{\mathrm{com},j}(\vartheta)\right]. \tag{65}$$

For compact notation, let

$$\widetilde{\boldsymbol{Z}}_j^{(s)} = \mathrm{vec}(\boldsymbol{Z}_j^{(s)}), \qquad A_{sm} = \Phi_s(\boldsymbol{x}_m)^\top \otimes L_s. \tag{66}$$

Then, $L_s \boldsymbol{Z}_j^{(s)} \Phi_s(\boldsymbol{x}_m) = A_{sm}\widetilde{\boldsymbol{Z}}_j^{(s)}$ and $\widetilde{\boldsymbol{Z}}_j^{(s)} \sim \mathcal{N}_{cr}(\boldsymbol{0}, \mathbb{I}_{cr})$.

We first consider a fixed assignment $\boldsymbol{s} = (s_1, \ldots, s_M)$. Let

$$\mathcal{I}_q(\boldsymbol{s}) = \{m : s_m = q\}, \qquad q \in \{1, 2\}, \tag{67}$$

denote the set of design points assigned to component q.

We collect the design points in $\mathcal{I}_q(\boldsymbol{s})$ as

$$\boldsymbol{X}_q = (\boldsymbol{x}_m : m \in \mathcal{I}_q(\boldsymbol{s})). \tag{68}$$

For notational simplicity, we suppress the dependence of $\boldsymbol{X}_q$ on $\boldsymbol{s}$. Similarly, let

$$\boldsymbol{F}_j^{(q)} = (\boldsymbol{f}_{jm} : m \in \mathcal{I}_q(\boldsymbol{s})) \tag{69}$$

denote the stacked outputs assigned to component q. We define the corresponding stacked mean and factor-loading matrix as

$$\boldsymbol{\mu}_q(\boldsymbol{X}_q) = (\mu_q(\boldsymbol{x}_m) : m \in \mathcal{I}_q(\boldsymbol{s})) \tag{70}$$

and

$$\Phi_q(\boldsymbol{X}_q) = \left(\Phi_q(\boldsymbol{x}_m)^\top : m \in \mathcal{I}_q(\boldsymbol{s})\right)^\top. \tag{71}$$

Let

$$\mathcal{A}_q = \Phi_q(\boldsymbol{X}_q) \otimes L_q. \tag{72}$$

Using $\widetilde{\boldsymbol{Z}}_j^{(q)} = \mathrm{vec}(\boldsymbol{Z}_j^{(q)})$, the outputs assigned to component q satisfy

$$\boldsymbol{F}_j^{(q)} \mid \widetilde{\boldsymbol{Z}}_j^{(q)}, \boldsymbol{S}_j = \boldsymbol{s}, \vartheta^{(t)} \sim \mathcal{N}\left(\boldsymbol{\mu}_q^{(t)}(\boldsymbol{X}_q) + \mathcal{A}_q^{(t)}\widetilde{\boldsymbol{Z}}_j^{(q)}, \sigma_q^{2(t)}\mathbb{I}\right). \tag{73}$$

Since $\widetilde{\boldsymbol{Z}}_j^{(q)} \sim \mathcal{N}_{cr}(\boldsymbol{0}, \mathbb{I}_{cr})$, integrating out the latent factor gives

$$\boldsymbol{F}_j^{(q)} \mid \boldsymbol{S}_j = \boldsymbol{s}, \vartheta^{(t)} \sim \mathcal{N}\left(\boldsymbol{\mu}_q^{(t)}(\boldsymbol{X}_q), \mathcal{A}_q^{(t)}\mathcal{A}_q^{(t)\top} + \sigma_q^{2(t)}\mathbb{I}\right). \tag{74}$$

Therefore, for a fixed assignment $\boldsymbol{s}$,

$$p\left(\boldsymbol{F}_j \mid \boldsymbol{S}_j = \boldsymbol{s}, \vartheta^{(t)}\right) = \prod_{q=1}^{2} \mathcal{N}\left(\boldsymbol{F}_j^{(q)}; \boldsymbol{\mu}_q^{(t)}(\boldsymbol{X}_q), \mathcal{A}_q^{(t)}\mathcal{A}_q^{(t)\top} + \sigma_q^{2(t)}\mathbb{I}\right). \tag{75}$$

The posterior probability of the assignment is then

$$\mathbb{P}\left[\boldsymbol{S}_j = \boldsymbol{s} \mid \boldsymbol{F}_j, \vartheta^{(t)}\right] = \frac{p\left(\boldsymbol{F}_j \mid \boldsymbol{S}_j = \boldsymbol{s}, \vartheta^{(t)}\right) \prod_{m=1}^{M} \pi_{s_m}\left(\boldsymbol{x}_m; \gamma^{(t)}\right)}{\sum_{\boldsymbol{s}'\in\{1,2\}^M} p\left(\boldsymbol{F}_j \mid \boldsymbol{S}_j = \boldsymbol{s}', \vartheta^{(t)}\right) \prod_{m=1}^{M} \pi_{s'_m}\left(\boldsymbol{x}_m; \gamma^{(t)}\right)}. \tag{76}$$

For a fixed assignment, the marginal likelihood in equation (75) can be evaluated in closed form. However, the same latent factor $\boldsymbol{Z}_j^{(q)}$ is shared across all design points in $\mathcal{I}_q(\boldsymbol{s})$. As a result, the covariance $\mathcal{A}_q\mathcal{A}_q^\top$ couples the outputs assigned to the same component, and the likelihood depends jointly on the full assignment vector $\boldsymbol{s}$. Therefore, the component indicators $S_{j1}, \ldots, S_{jM}$ cannot be inferred separately.

Computing the normalizing constant in equation (76) thus requires evaluating all $\boldsymbol{s} \in \{1, 2\}^M$. Since there are 2^M possible assignments, the exact E-step requires considering 2^M configurations for each posterior sample j, which becomes computationally infeasible for large M. We therefore use the structured mean-field approximation described in the next section.

D.3 Structured Mean-Field E-Step and Generalized M-Step

At iteration t, given the current parameter $\vartheta^{(t)}$, for each posterior sample j, we approximate the conditional distribution of the latent variables by

$$r_j^{(t)} = \prod_{m=1}^{M} r_{jm}^{\mathrm{cat},(t)}(S_{jm}) \prod_{s=1}^{2} r_{js}^{\mathrm{lat},(t)}(\boldsymbol{Z}_j^{(s)}), \tag{77}$$

where $r_{jm}^{\mathrm{cat},(t)}$ is a categorical distribution for the latent component indicator and $r_{js}^{\mathrm{lat},(t)}$ is the distribution of the component-specific latent factor. We define the responsibility as

$$\tau_{jms}^{(t)} = r_{jm}^{\mathrm{cat},(t)}(S_{jm} = s). \tag{78}$$

The E-step alternately updates the latent-factor distributions and the responsibilities while holding the model parameters $\vartheta^{(t)}$ fixed.

Latent-factor update. Given the current responsibilities, the structured mean-field update for the latent factor is Gaussian:

$$r_{js}^{\mathrm{lat},(t)}(\widetilde{\boldsymbol{Z}}_j^{(s)}) = \mathcal{N}_{cr}\left(\overline{\boldsymbol{z}}_{js}^{(t)}, V_{js}^{(t)}\right), \tag{79}$$

where

$$V_{js}^{(t)} = \left[\mathbb{I}_{cr} + \frac{1}{\sigma_s^{2(t)}} \sum_{m=1}^{M} \tau_{jms}^{(t)} A_{sm}^{(t)\top} A_{sm}^{(t)}\right]^{-1}, \tag{80}$$

and

$$\overline{\boldsymbol{z}}_{js}^{(t)} = V_{js}^{(t)} \left[\frac{1}{\sigma_s^{2(t)}} \sum_{m=1}^{M} \tau_{jms}^{(t)} A_{sm}^{(t)\top} \left(\boldsymbol{f}_{jm} - \mu_s^{(t)}(\boldsymbol{x}_m)\right)\right]. \tag{81}$$

Thus, the latent factor for each component is estimated using the outputs at all design points, weighted by their current responsibilities.

Responsibility update. Given the current latent-factor distribution, define the expected reconstruction error as

$$\Delta_{jms}^{(t)} = \left\|\boldsymbol{f}_{jm} - \mu_s^{(t)}(\boldsymbol{x}_m) - A_{sm}^{(t)} \overline{\boldsymbol{z}}_{js}^{(t)}\right\|_2^2 + \mathrm{tr}\left(A_{sm}^{(t)} V_{js}^{(t)} A_{sm}^{(t)\top}\right). \tag{82}$$

The responsibility is then updated as

$$\tau_{jms}^{(t)} = \frac{\pi_s(\boldsymbol{x}_m; \gamma^{(t)})(\sigma_s^{2(t)})^{-c/2} \exp\left(-\Delta_{jms}^{(t)}/(2\sigma_s^{2(t)})\right)}{\sum_{\ell=1}^{2} \pi_\ell(\boldsymbol{x}_m; \gamma^{(t)})(\sigma_\ell^{2(t)})^{-c/2} \exp\left(-\Delta_{jm\ell}^{(t)}/(2\sigma_\ell^{2(t)})\right)}. \tag{83}$$

We alternately update the latent-factor distributions and responsibilities 20 times in each E-step. The resulting $\tau_{jms}^{(t)}$, $\overline{\boldsymbol{z}}_{js}^{(t)}$, and $V_{js}^{(t)}$ are then fixed in the generalized M-step.

Generalized M-step. Given the variational distributions obtained from the E-step, we define the Q-function as the expectation of the complete log-likelihood with respect to these distributions. Ignoring terms independent of ϑ, the Q-function is

$$Q(\vartheta \mid \vartheta^{(t)}) = \sum_{j=1}^{N} \sum_{m=1}^{M} \sum_{s=1}^{2} \tau_{jms}^{(t)} \left[\log \pi_s(\boldsymbol{x}_m; \gamma) - \frac{c}{2} \log \sigma_s^2 - \frac{1}{2\sigma_s^2} R_{jms}^{(t)}(\psi_s)\right], \tag{84}$$

where

$$R_{jms}^{(t)}(\psi_s) = \left\|\boldsymbol{f}_{jm} - \mu_s(\boldsymbol{x}_m) - A_{sm} \overline{\boldsymbol{z}}_{js}^{(t)}\right\|_2^2 + \mathrm{tr}\left(A_{sm} V_{js}^{(t)} A_{sm}^{\top}\right). \tag{85}$$

Here, the responsibilities and latent-factor moments are fixed at the values obtained from the E-step, while μ_s, A_{sm}, and σ_s^2 are functions of the parameters to be updated. We use a generalized M-step and jointly update ψ_1, ψ_2, and γ by gradient ascent on $Q(\vartheta \mid \vartheta^{(t)})$ to obtain $\vartheta^{(t+1)}$.

D.4 Initialization of MoDGF

We initialize MoDGF using the initialization procedure for the DGF model before running the structured mean-field EM algorithm. We first initialize the first DGF component using the two-stage procedure described in Park et al. (2025b). The mean function μ_1 is first pretrained to approximate the average teacher outputs at the design points. We then initialize the factor-loading function and latent factors by fitting the teacher outputs while encouraging the empirical distribution of the latent factors to follow their Gaussian prior.

After this initialization, the parameters of the first DGF component are copied to the second component. In this work, we add a small Gaussian perturbation to the copied parameters so that the two components do not start from exactly the same parameter values. The gating parameter γ is initialized separately without pretraining.

D.5 Overall MoDGF Training Algorithm

The above updates use all design points, which can be computationally expensive when M is large. In this work, we use a mini-batch approximation to reduce the computational cost per iteration. To be more specific, at iteration t, we calculate the E-step updates and the Q-function using only the design points indexed by $\mathcal{B}_t$. The overall MoDGF training procedure is summarized in Algorithm 1. Let K denote the number of alternating updates of the latent-factor distributions and responsibilities within each E-step. Unless otherwise stated, we set $K = 20$.

Algorithm 1 Training Procedure for MoDGF

Input: Design points $\mathcal{D}_{\text{design}} = \{\boldsymbol{x}_m\}_{m=1}^{M}$, SGLD outputs $\{\boldsymbol{f}_{jm}\}_{j=1,m=1}^{N,M}$, mini-batch size b, learning rate $\eta > 0$, number of EM iterations T, number of E-step updates K

Initialize $\vartheta^{(0)}$ as described in Appendix D.4
Initialize $\tau_{jms} \leftarrow \pi_s(\boldsymbol{x}_m; \gamma^{(0)})$ for all j, m, and s
for $t = 0, 1, \ldots, T-1$ **do**
    Sample a mini-batch $\mathcal{B}_t \subset \{1, \ldots, M\}$ with $|\mathcal{B}_t| = b$
    for $k = 1, 2, \ldots, K$ **do**
        Update V_{js} and $\overline{\boldsymbol{z}}_{js}$ for all j and s using equations (80) and (81) on $\mathcal{B}_t$
        Update τ_{jms} for all j, $m \in \mathcal{B}_t$, and s using equation (83)
    end for
    Construct $Q(\vartheta)$ in equation (84) using τ_{jms}, $\overline{\boldsymbol{z}}_{js}$, and V_{js} on $\mathcal{B}_t$
    Update $\vartheta^{(t+1)} \leftarrow \vartheta^{(t)} + \eta \nabla_{\vartheta} Q(\vartheta)\big|_{\vartheta=\vartheta^{(t)}}$
end for

$\widehat{\vartheta} \leftarrow \vartheta^{(T)}$
Output: Estimated MoDGF parameter $\widehat{\vartheta}$

D.6 Comparison with Monte Carlo EM (MC-EM)

In this section, we compare the structured mean-field EM used in our method with Monte Carlo EM (MC-EM) (Wei & Tanner, 1990; Sherman et al., 1999). While structured mean-field EM approximates the conditional distribution of the latent variables using the variational distribution in equation (77), MC-EM approximates the E-step using samples from the conditional distribution of the latent variables. At each EM iteration, we generate Gibbs samples using a burn-in of 100 iterations, followed by 200 samples with a thinning interval of 1.

We first compare the quantities obtained from the E-step of structured mean-field EM and MC-EM. Figure 3 compares the Q-function value, the mean component assignment, and the Frobenius norms $\|\boldsymbol{Z}^{(1)}\|_F$ and $\|\boldsymbol{Z}^{(2)}\|_F$ of the two component-specific latent factors. We report the Q-function value normalized by NM, the mean responsibility for component 1 across posterior samples and design points, and the mean Frobenius norm of the posterior mean latent factor for each component across posterior samples. We also verified that the Gibbs sampling results are stable under different initializations.

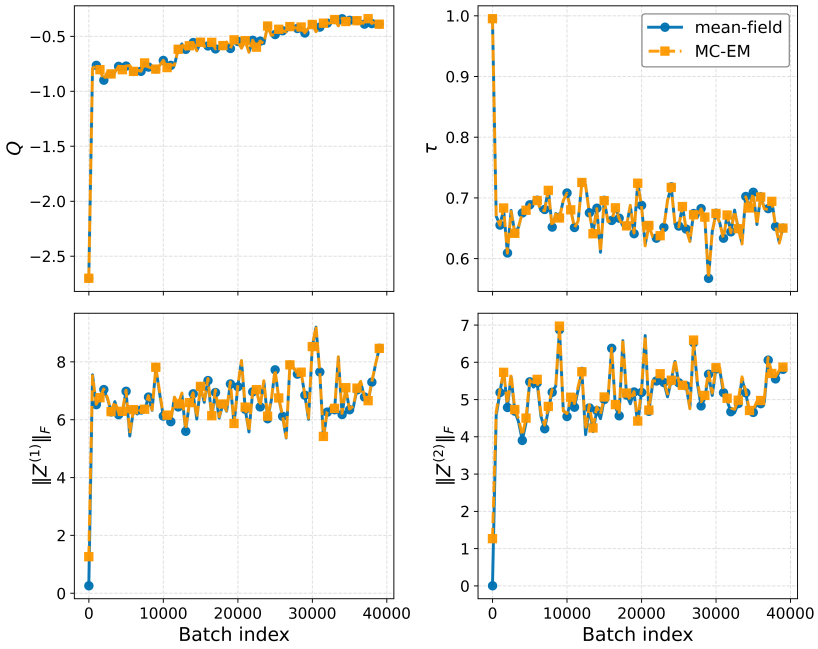

Figure 3: Comparison of structured mean-field EM and MC-EM in terms of the Q-function value, mean component assignment, and the Frobenius norms $\|\boldsymbol{Z}^{(1)}\|_F$ and $\|\boldsymbol{Z}^{(2)}\|_F$.

Figure 3 shows similar values for the reported summary statistics under structured mean-field EM and MC-EM.

We further compare the two methods after training MoDGF on CIFAR-10. Both methods use the same model architecture, SGLD posterior samples, and training settings, and differ only in the approximation used for the E-step. We report accuracy, ECE, NLL, and one-epoch training time.

Table 5: Comparison of structured mean-field EM and MC-EM on CIFAR-10. Predictive performance is averaged over five random seeds, and the training time is the one-epoch training time in seconds.

Method	Acc. (%) ↑	ECE (%) ↓	NLL ↓	Time (s) ↓
Structured mean-field EM	92.84	2.55	0.23	5.27
MC-EM	92.09	2.88	0.27	21.77

Table 5 shows that structured mean-field EM achieves higher mean accuracy and lower mean ECE and NLL than MC-EM, while requiring less training time per epoch (5.27 s vs. 21.77 s). These results support our use of structured mean-field EM for training MoDGF.

E Experimental Details

E.1 Datasets and Evaluation Metrics

Image classification datasets. We consider three image classification benchmarks: CIFAR-10, CIFAR-100 (Krizhevsky, 2009), and Tiny-ImageNet (Le & Yang, 2015). CIFAR-10 and CIFAR-100 each contain 50,000 training images and 10,000 test images of natural scenes at 32×32 pixel resolution, with 10 and 100 classes, respectively. Tiny-ImageNet is a subset of ImageNet with 200 classes, containing 100,000 training images and 10,000 validation images at 64×64 pixel resolution. We use the official validation split as the test set in our experiments. Table 6 summarizes the datasets.

Table 6: Image classification datasets.

Dataset	Train size	Test size	Classes
CIFAR-10	50,000	10,000	10
CIFAR-100	50,000	10,000	100
Tiny-ImageNet	100,000	10,000	200

Language reasoning datasets. We consider three language reasoning benchmarks: ARC-Challenge (ARC-C) and ARC-Easy (ARC-E) from the AI2 Reasoning Challenge (Clark et al., 2018), and OpenBookQA (OBQA) (Mihaylov et al., 2018). ARC-C and ARC-E are multiple-choice science question datasets with 3-5 answer choices per question. For both datasets, we use only questions with 4 answer choices to form a common 4-class classification setting. OBQA is a multiple-choice question answering benchmark with 4 answer choices that requires multi-step reasoning and common knowledge. Table 7 summarizes the dataset statistics. For each dataset, we use an 80/20 stratified split of the training set for training and validation, and evaluate on the standard validation set as the test set.

Table 7: Language reasoning dataset statistics.

Dataset	Training samples	Classes
ARC-C	1,117	4
ARC-E	2,241	4
OBQA	4,957	4

Evaluation metrics. Let $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n_{\text{test}}}$ denote the test set. We evaluate classification performance using the following metrics.

Accuracy (Acc.) is defined as

$$\text{Acc.} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \mathbf{1}(y_i = \widehat{y}_i), \tag{86}$$

where $\widehat{y}_i = \arg\max_k p(y = k \mid \boldsymbol{x}_i, \mathcal{D})$ and the posterior predictive probability $p(y = k \mid \boldsymbol{x}_i, \mathcal{D})$ is approximated as in equation (3).

Negative log-likelihood (NLL) is defined as

$$\text{NLL} = -\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \sum_{k=1}^{c} \mathbf{1}(y_i = k) \log p(y = k \mid \boldsymbol{x}_i, \mathcal{D}). \tag{87}$$

Expected calibration error (ECE; Naeini et al., 2015) is defined as

$$\text{ECE} = \sum_{h=1}^{H} \frac{|B_h|}{n_{\text{test}}} \left| \frac{1}{|B_h|} \sum_{(\boldsymbol{x}_i, y_i) \in B_h} \mathbf{1}(y_i = \widehat{y}_i) - \frac{1}{|B_h|} \sum_{(\boldsymbol{x}_i, y_i) \in B_h} p(\widehat{y}_i \mid \boldsymbol{x}_i, \mathcal{D}) \right|, \tag{88}$$

where $\{B_1, \ldots, B_H\}$ is a partition of the test set into H bins based on the predicted confidence $p(\widehat{y}_i \mid \boldsymbol{x}_i, \mathcal{D})$. We set $H = 15$ in our experiments.

AUROC. For OOD detection, we use the area under the receiver operating characteristic curve (AUROC). We assign label 0 to ID data and label 1 to OOD data, and compute AUROC using the corresponding uncertainty scores. For methods using overall epistemic uncertainty, the score is $U_{\text{epistemic}}(\boldsymbol{x})$ defined in equation (6). For MoDGF with WCE, the score is $U_{\text{within}}(\boldsymbol{x})$ defined in equation (25).

E.2 Model Architectures

Image classification. For image classification, we adopt Wide Residual Networks (WRN) (Zagoruyko & Komodakis, 2016) as the backbone architecture. Specifically, we use WRN-28-1 for CIFAR-10 and WRN-28-4 for CIFAR-100 and Tiny-ImageNet. This choice is applied consistently across all methods, including SGLD, all Bayesian inference baselines, all distillation baselines, and MoDGF.

Language reasoning. For language reasoning tasks, we use Qwen3-8B-Base (Qwen Team, 2025) as the pretrained language model. We perform parameter-efficient fine-tuning using LoRA (Low-Rank Adaptation; Hu et al., 2022), rather than updating all model parameters. The LoRA rank is set to $r = 8$, the scaling parameter is set to $\alpha = 16$, and the dropout rate is set to 0. The maximum input sequence length is set to 300. These settings are applied consistently across all methods.

E.3 Details of the SGLD Algorithm

We describe the SGLD posterior sampling procedure used to generate the teacher distribution for MoDGF. The procedure consists of a burn-in phase, followed by Langevin sampling that collects $N = 500$ posterior samples.

Image classification. All image experiments use SGD with weight decay 5×10^{-4} and a mini-batch size of 128 for the burn-in phase, with an initial learning rate of 0.1. For CIFAR-10 and CIFAR-100, the burn-in runs for 160 epochs with a multi-step schedule that decays the learning rate by a factor of 0.2 at epochs 60, 120, and 160. For Tiny-ImageNet, the burn-in runs for 80 epochs with a cosine annealing schedule with warm restarts (Loshchilov & Hutter, 2016), decaying the learning rate from 0.1 to 10^{-3}.

After burn-in, we switch to the Langevin sampling following equation (8), retaining the same mini-batch size of 128. The step size is initialized to $\eta_0 = 10^{-3}$ ($0.01\times$ the burn-in learning rate) and decayed by a polynomial schedule $\eta_t = a(b + t)^{-0.55}$, where a and b are chosen so that η_t decreases from η_0 to $0.1\eta_0$ over the sampling phase. We collect $N = 500$ posterior samples with a thinning interval of 10 epochs. For each collected sample θ_j, we evaluate the logit function f_{θ_j} on the training, validation, and test sets to obtain the logit outputs used for posterior distillation and evaluation.

Language reasoning. For language reasoning tasks, we use a warm-start procedure followed by SGLD sampling. In the burn-in phase, we fine-tune the pretrained Qwen3-8B-Base model with LoRA adapters for 5,000 steps using the AdamW optimizer with a learning rate of 10^{-4}, batch size 4, and a linear learning rate schedule with a warmup ratio of 0.06. The loss function is the standard cross-entropy loss scaled by the training set size n.

After burn-in, the SGLD sampling phase collects $N = 500$ posterior samples. The step size is initialized to match the final warm-start learning rate and follows the same polynomial decay schedule as in the image classification case. The SGLD update uses an isotropic Gaussian prior with unit variance, a posterior temperature of $T = 1.0$, and a thinning interval of 10 mini-batch steps. For each posterior sample, we evaluate the model on the training, validation, and test sets to obtain the answer-token logits, which are stored for subsequent posterior distillation and evaluation.

E.4 Details of the Distillation Baselines

All distillation methods use the same $N = 500$ SGLD posterior samples described in Appendix E.3 and the backbone architectures described in Appendix E.2. For CIFAR-10 and CIFAR-100, we train the student models for 200 epochs using Adam and decay the learning rate at epochs 60, 120, and 160. For Tiny-ImageNet, we train the models for 120 epochs using Adam with cosine annealing

and warm restarts. For all image classification tasks, we use a batch size of 128. For the language reasoning tasks, we use AdamW with a learning rate of 10^{-4}, a batch size of 4, and a linear learning rate schedule with a warmup ratio of 0.06.

Let $\boldsymbol{f}_{jm} = \boldsymbol{f}_j(\boldsymbol{x}_m) \in \mathbb{R}^c$ denote the logits from the jth SGLD posterior sample at design point $\boldsymbol{x}_m$, and let

$$\boldsymbol{\pi}_{jm} = \mathrm{softmax}\left(\boldsymbol{f}_{jm}\right) \tag{89}$$

denote the corresponding predictive probability vector.

Bayesian Dark Knowledge. BDK (Korattikara Balan et al., 2015) distills the posterior predictive distribution into a deterministic student model. Since we use a fixed set of SGLD posterior samples, we adapt BDK to an offline setting. For each design point, we first compute

$$\overline{\boldsymbol{\pi}}_m = \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{\pi}_{jm}. \tag{90}$$

The student $\boldsymbol{p}_\theta(\boldsymbol{x})$ is then trained by minimizing

$$\mathcal{L}_{\mathrm{BDK}} = \sum_{m=1}^{M} \mathrm{KL}\left(\overline{\boldsymbol{\pi}}_m \,\|\, \boldsymbol{p}_\theta(\boldsymbol{x}_m)\right). \tag{91}$$

At inference, the deterministic student produces a single predictive probability vector.

Hydra. Hydra (Tran et al., 2020) performs one-to-one distillation using a shared backbone with a separate classification head for each teacher member. In our implementation, we use $N = 500$ MLP heads, each with one hidden layer. Let $\boldsymbol{p}_{\theta,j}(\boldsymbol{x})$ denote the predictive probability from the jth head. The student is trained by minimizing

$$\mathcal{L}_{\mathrm{Hydra}} = \sum_{m=1}^{M}\sum_{j=1}^{N} \mathrm{KL}\left(\boldsymbol{\pi}_{jm} \,\|\, \boldsymbol{p}_{\theta,j}(\boldsymbol{x}_m)\right). \tag{92}$$

At inference, the shared backbone is evaluated once and the 500 heads produce member-specific predictive probabilities. Their average is used for prediction, while the individual predictions are used for uncertainty estimation.

EnD^2 **(Proxy-Dirichlet Distillation).** We use the Proxy-Dirichlet Distillation method of Ryabinin et al. (2021). For class $l \in \{1, \ldots, c\}$, the proxy distribution is constructed from the SGLD predictive probabilities as

$$\beta_{ml} = \frac{1}{N}\sum_{j=1}^{N} \pi_{jml}, \tag{93}$$

and

$$\alpha_{ml} = \frac{\beta_{ml}(c-1)}{2\sum_{k=1}^{c}\beta_{mk}\left(\log\beta_{mk} - \frac{1}{N}\sum_{j=1}^{N}\log\pi_{jmk}\right)}. \tag{94}$$

Let $\boldsymbol{\alpha}_m = (\alpha_{m1}, \ldots, \alpha_{mc})^\top$. The student network $\boldsymbol{\Psi}_\theta$ is trained by minimizing

$$\mathcal{L}_{\mathrm{EnD}^2} = \sum_{m=1}^{M} \mathrm{KL}\left(\mathrm{Dir}\left(\exp\left(\boldsymbol{\Psi}_\theta(\boldsymbol{x}_m)\right) + \mathbf{1}\right) \parallel \mathrm{Dir}\left(\boldsymbol{\alpha}_m + \mathbf{1}\right)\right). \tag{95}$$

At inference, we draw 500 predictive probability vectors from the learned Dirichlet distribution for prediction and uncertainty estimation.

Ensemble Distribution Distillation via Flow Matching (EDFM). EDFM (Park et al., 2025a) models the conditional distribution of teacher logits using flow matching. Unlike the probability-based baselines above, EDFM uses the teacher logits $\boldsymbol{f}_{jm}$ rather than the predictive probability vectors $\boldsymbol{\pi}_{jm}$ as the distillation targets.

For each design point $\boldsymbol{x}_m$, we sample a teacher index j uniformly from $\{1, \ldots, N\}$ and set

$$\boldsymbol{h}_1 = \boldsymbol{f}_{jm}. \tag{96}$$

We also sample an initial noise vector

$$\boldsymbol{h}_0 \sim \mathcal{N}_c\left(\boldsymbol{0}, \sigma_0^2 \boldsymbol{I}_c\right), \tag{97}$$

and define the interpolation

$$\boldsymbol{h}_t = (1-t)\boldsymbol{h}_0 + t\boldsymbol{h}_1. \tag{98}$$

Thus, $\boldsymbol{h}_t$ interpolates between the initial Gaussian sample $\boldsymbol{h}_0$ and the sampled teacher logit $\boldsymbol{h}_1$.

Let $\boldsymbol{v}_\theta(t, \boldsymbol{h}, \boldsymbol{x}) \in \mathbb{R}^c$ denote the vector field parameterized by the student network. The student is trained to predict the velocity $\boldsymbol{h}_1 - \boldsymbol{h}_0$ by minimizing

$$\mathcal{L}_{\mathrm{EDFM}} = \sum_{m=1}^{M} \mathbb{E}\left[\lambda(t) \left\| \boldsymbol{v}_\theta\left(t, \boldsymbol{h}_t, \boldsymbol{x}_m\right) - \left(\boldsymbol{h}_1 - \boldsymbol{h}_0\right) \right\|_2^2\right], \tag{99}$$

where we follow Park et al. (2025a) for the time sampling and weighting function $\lambda(t)$.

At inference, following the sampling procedure of EDFM, we generate 500 logit samples from the learned conditional distribution and apply the softmax function to obtain predictive probability vectors for prediction and uncertainty estimation.

Single DGF. Single DGF (Park et al., 2025b) models the SGLD logits with a single multivariate DGF. For the jth SGLD posterior sample, we use

$$\boldsymbol{f}_j(\boldsymbol{x}) = \boldsymbol{\mu}(\boldsymbol{x}) + \boldsymbol{L}\boldsymbol{Z}_j\boldsymbol{\Phi}(\boldsymbol{x}) + \boldsymbol{\epsilon}, \tag{100}$$

where $\boldsymbol{Z}_j \sim \mathrm{MN}_{c,r}(\boldsymbol{0}, \boldsymbol{I}_c, \boldsymbol{I}_r)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}_c(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_c)$. The mean function $\boldsymbol{\mu}$ and factor-loading function $\boldsymbol{\Phi}$ are parameterized by separate one-hidden-layer MLP heads on top of the shared backbone. The model is trained using the EM procedure of Park et al. (2025b).

For Single DGF, we consider latent factor dimensions $r \in \{4, 8, 12, 16\}$ and use $r = 8$ in the main experiments. At inference, we generate 500 logit samples from the learned DGF, transform them by the softmax function, and use the resulting predictive probability vectors for prediction and uncertainty estimation.

E.5 Details of the Specialist Inference Methods

We provide the implementation details of the three specialist methods used in Section 6.2. The methods differ in how the specialist classes are determined and how the specialists are used at inference.

Table 8: Comparison of the specialist methods used in the experiments.

Method	Specialist construction	Inference
Jung et al.	Confusing pairs from confusion matrix	Distilled student
MS-NET-v2	Confusing subsets from router predictions	Router top-2 and selected experts
Component-guided	All class pairs	Component preferred classes

Jung et al. For Jung et al. (2019), we compute the confusion matrix of the base classifier and select the two most confusing class pairs. A specialist is initialized from the base classifier for each selected pair. During training, 50% of each mini-batch is sampled from the corresponding class pair and the remaining 50% from the other classes. Each specialist retains the full c-class output. Following Jung et al. (2019), the predictions of the base classifier and the two specialists are distilled into a single student model. At inference, only the distilled student is used, without selecting a specialist for each input.

MS-NET-v2. For MS-NET-v2 (Chowdhury & Zhao, 2025), we construct the inter-class correlation matrix from the top-2 predictions of the router on the validation set. The cutoff parameter O determines the confusing class pairs selected from this matrix, and overlapping pairs are merged into multi-class subsets. We use $O = 20$, $O = 3$, and $O = 2$ for CIFAR-10, CIFAR-100, and Tiny-ImageNet, respectively. For CIFAR-10, the selected pairs remain as binary class subsets. Each expert is randomly initialized and trained independently using the same backbone architecture as the router, following the training procedure of Chowdhury & Zhao (2025). At inference, we follow their multi-expert inference procedure based on the top-2 predictions of the router.

Component-guided specialist. We use a shared backbone with a binary classification head for each unordered class pair, giving $\binom{c}{2}$ pairwise specialists. For each pair $\{a, b\}$, the corresponding head is trained on samples from classes a and b using cross-entropy loss. The shared backbone and all pair-specific heads are trained jointly.

At inference, we follow the rule in equation (22). We obtain the preferred classes $c_1(\boldsymbol{x})$ and $c_2(\boldsymbol{x})$ from the two MoDGF components. If $c_1(\boldsymbol{x}) \neq c_2(\boldsymbol{x})$, we use the specialist corresponding to $\{c_1(\boldsymbol{x}), c_2(\boldsymbol{x})\}$ to make the final prediction. Otherwise, we use $c_1(\boldsymbol{x})$ as the final prediction.

The above rule does not consider the mixture probabilities, so a specialist may be selected even when one component has a very small mixture probability. One possible extension is to require the smaller mixture probability to exceed a threshold before using a specialist, thereby avoiding specialist selection based on a negligible component.

Another consideration is that the number of pairwise specialists grows as $\binom{c}{2}$ with the number of classes. When this number becomes large, one possible approach is to construct specialists only for class pairs that are frequently confused on a validation set and use a specialist only when the corresponding pair is available.

E.6 Details of OOD Detection

For OOD detection, we use CIFAR-10 and CIFAR-100 as ID datasets and Tiny-ImageNet and SVHN as OOD datasets, as described in Section 6.3. For each ID dataset, we use all 10,000 test images and randomly select 10,000 images from the SVHN test set.

We compare SGLD (Welling & Teh, 2011) and the posterior distillation methods considered in Section 6.3: Hydra, EnD^2, EDFM, Single DGF, and MoDGF. We use epistemic uncertainty defined in equation (6) as the uncertainty score. Given L predictive probability vectors $\{\boldsymbol{p}^{(\ell)}(\boldsymbol{x})\}_{\ell=1}^{L}$, let

$$\overline{\boldsymbol{p}}(\boldsymbol{x}) = \frac{1}{L}\sum_{\ell=1}^{L} \boldsymbol{p}^{(\ell)}(\boldsymbol{x}). \tag{101}$$

We estimate the epistemic uncertainty as

$$\widehat{U}_{\mathrm{epistemic}}(\boldsymbol{x}) = \mathrm{H}\left[\overline{\boldsymbol{p}}(\boldsymbol{x})\right] - \frac{1}{L}\sum_{\ell=1}^{L} \mathrm{H}\left[\boldsymbol{p}^{(\ell)}(\boldsymbol{x})\right], \tag{102}$$

where H denotes the categorical entropy. We use $L = 500$ for all methods.

For SGLD and Hydra, the 500 predictive probability vectors are obtained from the posterior samples and member-specific heads, respectively. For EnD^2, EDFM, and Single DGF, we generate 500 samples from the learned predictive distributions. BDK is excluded because its deterministic student does not retain the sample-wise predictive variation required to estimate epistemic uncertainty.

For all baseline methods, we use $\widehat{U}_{\mathrm{epistemic}}(\boldsymbol{x})$ as the OOD score. For MoDGF, we evaluate both the overall epistemic uncertainty and the within-component epistemic uncertainty $U_{\mathrm{within}}(\boldsymbol{x})$ defined in equation (25). The OOD detection performance is evaluated using AUROC as described in Appendix E.1.

Extension to other Bayesian inference methods. To further evaluate OOD detection beyond SGLD and the posterior distillation methods, we additionally compare BBB (Blundell et al., 2015; Krishnan et al., 2022), Laplace approximation (Ritter et al., 2018; Daxberger et al., 2021), and

Table 9: OOD detection performance measured by AUROC (%). CIFAR-10 and CIFAR-100 are used as ID datasets, and SVHN and Tiny-ImageNet are used as OOD datasets.

		SVHN		Tiny-ImageNet	
	Method	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100
Bayesian Inference	SGLD	89.18 (0.95)	82.13 (2.78)	87.06 (0.13)	80.90 (0.81)
	BBB	86.80 (1.13)	80.16 (3.43)	83.49 (0.38)	76.38 (0.33)
	Laplace	88.71 (3.66)	78.33 (3.24)	85.02 (0.24)	77.88 (0.20)
	MC-Dropout	85.79 (2.74)	69.10 (1.08)	84.11 (0.22)	75.51 (0.33)
Posterior Distillation	Hydra	88.55 (0.32)	77.05 (5.66)	81.71 (1.08)	73.44 (4.79)
	EnD^2	87.35 (2.62)	80.88 (1.07)	86.81 (0.62)	77.97 (0.90)
	EDFM	87.80 (2.93)	79.51 (5.12)	85.53 (0.31)	78.90 (0.62)
	Single DGF	87.87 (1.35)	79.04 (5.13)	86.42 (0.23)	76.63 (0.92)
	MoDGF (Epistemic)	88.61 (0.74)	81.77 (2.83)	86.16 (0.21)	80.90 (1.16)
	MoDGF (WCE)	**90.80 (0.92)**	**83.50 (2.55)**	**87.30 (0.21)**	**82.40 (1.22)**

MC-Dropout (Gal & Ghahramani, 2016). For BBB, Laplace, and MC-Dropout, we generate 500 stochastic predictions and estimate epistemic uncertainty using equation (102).

As shown in Table 9, using overall epistemic uncertainty, MoDGF achieves AUROC comparable to SGLD across all four ID–OOD dataset pairs. Using WCE further improves AUROC by 1.14–2.19 percentage points and achieves the highest AUROC among all compared methods, suggesting that excluding between-component disagreement provides a more useful OOD score than overall epistemic uncertainty.

Analysis of epistemic uncertainty decomposition. We further analyze the epistemic uncertainty decomposition of MoDGF for ID and OOD inputs. Recall that the overall epistemic uncertainty is decomposed into the within-component epistemic uncertainty U_{within} and the between-component epistemic uncertainty U_{between}. For this analysis, we divide the CIFAR-10 ID inputs and SVHN OOD inputs into bins according to their overall epistemic uncertainty $U_{\text{epistemic}} = U_{\text{within}} + U_{\text{between}}$. For each bin, we calculate the mean values of U_{within} and U_{between} separately for the ID and OOD inputs.

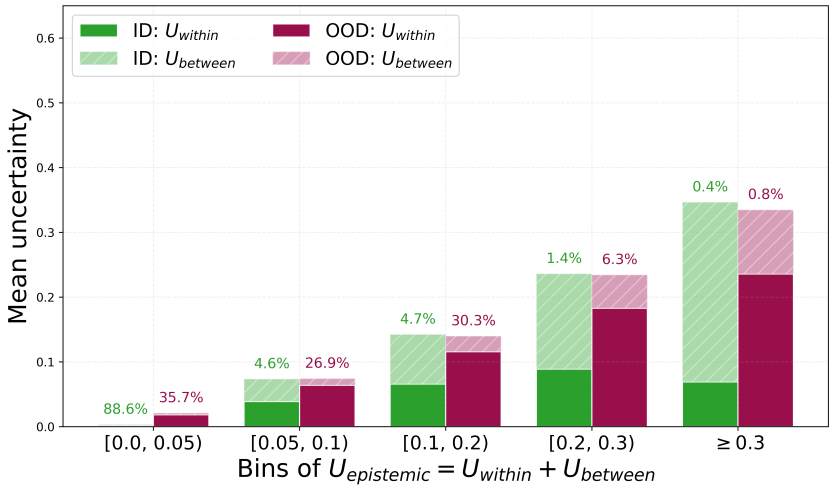

Figure 4: Decomposition of MoDGF epistemic uncertainty for ID and OOD inputs, grouped into bins of $U_{\text{epistemic}} = U_{\text{within}} + U_{\text{between}}$. Solid and hatched segments show the mean U_{within} and U_{between} in each bin, respectively. Percentages above the bars indicate the proportion of each ID or OOD dataset falling in the corresponding bin.

Figure 4 shows the proportions of ID and OOD inputs in each bin of $U_{\text{epistemic}}$, together with its decomposition into U_{within} and U_{between}. ID inputs are more concentrated in lower $U_{\text{epistemic}}$ bins, although some ID inputs also appear in the higher bins. However, in most bins, ID inputs have larger average U_{between} than OOD inputs, suggesting that high epistemic uncertainty for ID inputs can

partly arise from between-component disagreement. This observation motivates using U_{within}, which excludes this disagreement, as the OOD score. This is consistent with the OOD detection results in Table 9.

F Approximation of the SGLD Predictive Mode Structure

We further examine whether MoDGF preserves the predictive mode structure observed in the SGLD posterior samples. Figure 5 shows unimodal and bimodal SGLD predictive distributions and their MoDGF approximations. For the unimodal examples in Figure 5(a), the two DGF components are either close to each other or one component has a negligibly small mixture probability, with the dominant component closely approximating the unimodal SGLD distribution. For the bimodal examples in Figure 5(b), the two DGF components capture the two separated modes of the SGLD predictive distribution. These examples illustrate how MoDGF approximates both unimodal and bimodal predictive distributions.

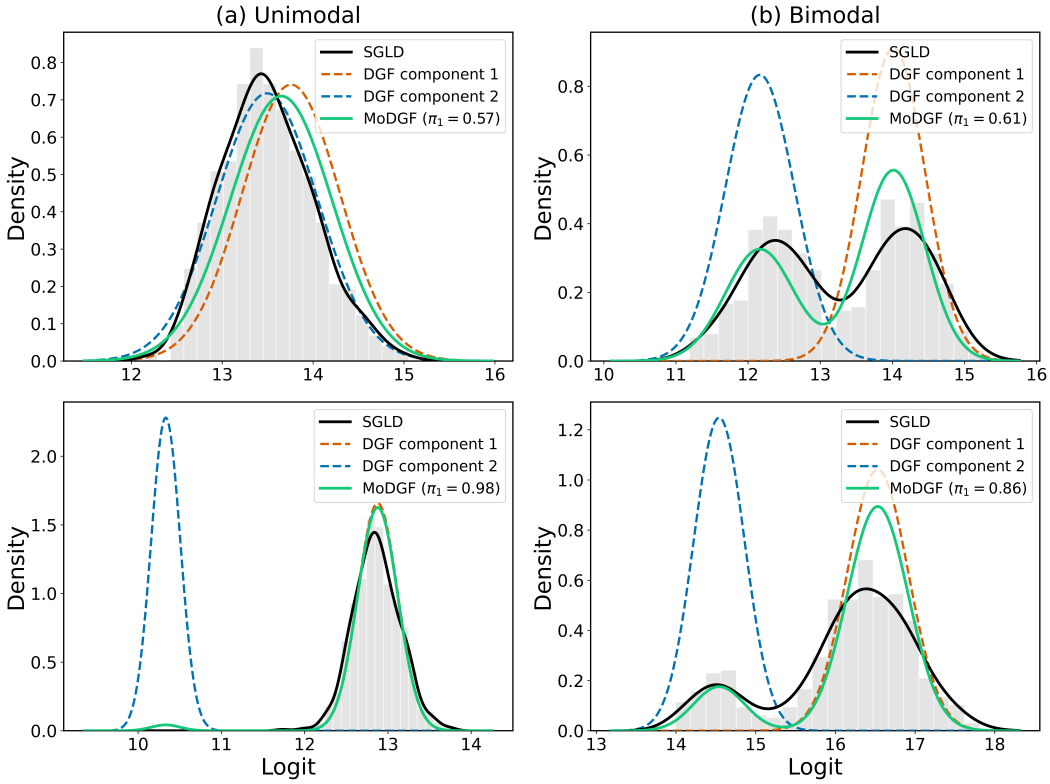

Figure 5: Examples of SGLD predictive logit distributions for the predicted class $c^\star(\boldsymbol{x})$ and the corresponding MoDGF approximations. (a) Unimodal examples. (b) Bimodal examples.

We also examine whether the two MoDGF components support the same classes as the SGLD predictive modes. We first classify each input as unimodal or bimodal using the criterion in Appendix C. For a unimodal input, we define an agreement when both MoDGF components have the same preferred class as the SGLD predictive distribution. For a bimodal input, we define an agreement when the preferred classes of the two MoDGF components match the two classes supported by the SGLD predictive modes, regardless of the component ordering. We report the proportion of test inputs satisfying these conditions as the class agreement.
As shown in Table 10, the class agreement is 89.57% on CIFAR-10, 76.03% on CIFAR-100, and 70.46% on Tiny-ImageNet. These results indicate that the two MoDGF components support the same classes as the SGLD predictive modes for a large proportion of test inputs.

Table 10: Agreement between the predictive mode structure of SGLD and the two MoDGF components. Results are reported as mean and standard deviation over five random seeds.

Dataset	Mode agreement (%) ↑
CIFAR-10	89.57 (0.21)
CIFAR-100	76.03 (0.20)
Tiny-ImageNet	70.46 (0.44)

G Ablation Studies

We conduct ablation studies on CIFAR-10 to examine the sensitivity of MoDGF to its main hyperparameters. Unless otherwise stated, the ablation studies use a latent factor dimension of $r = 16$ in each MoDGF component, $N = 500$ SGLD samples, and 20 mean-field iterations. Each configuration is evaluated over five random seeds. The solid line and shaded band indicate the mean and min–max range, respectively.

G.1 Latent Factor Dimension

The latent factor dimension r controls the expressiveness of the low-rank covariance approximation in each DGF component. To examine this trade-off between model capacity and overfitting, we vary $r \in \{4, 8, 12, 16, 20, 24\}$ while keeping the other hyperparameters fixed.

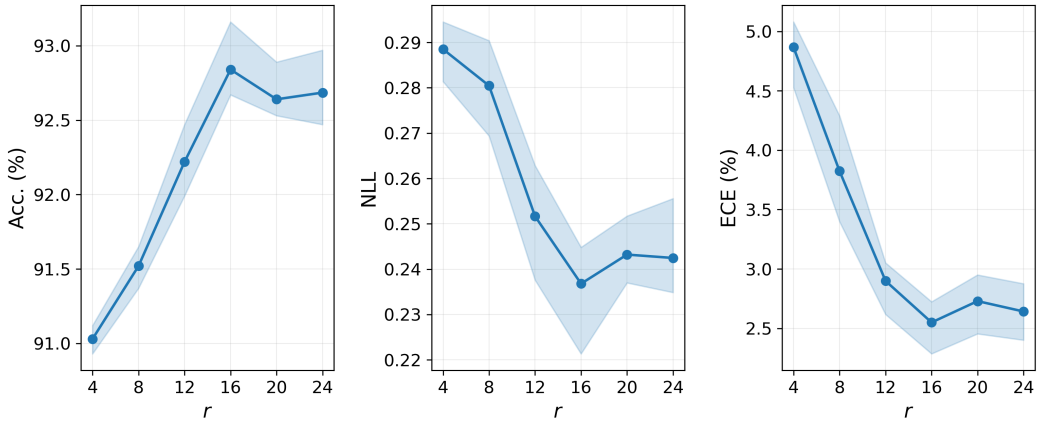

Figure 6: Accuracy, NLL, and ECE versus the latent factor dimension r on CIFAR-10.

Results. Figure 6 shows that all three metrics improve as r increases from 4 to 16. Beyond $r = 16$, accuracy slightly decreases while NLL and ECE remain stable, indicating that the additional latent dimensions do not capture further useful covariance structure.

G.2 Choice of the Number of Mean-Field Iterations

Since structured mean-field EM approximates the E-step through iterative updates of the variational factors, the number of mean-field iterations K is an important practical choice. We therefore vary $K \in \{10, 20, 30, 40, 50\}$ to examine its effect on predictive performance.

Results. Figure 7 shows that accuracy, NLL, and ECE improve substantially from $K = 10$ to $K = 20$ and remain relatively stable for larger values of K. Based on this result, we use $K = 20$ as the default setting, since further mean-field updates provide little additional improvement while increasing the computational cost.

G.3 MoDGF with Different Posterior Sampling Methods

Although our main experiments use SGLD, MoDGF only requires posterior samples for distillation and is not restricted to a particular posterior sampling method. To examine robustness to the choice

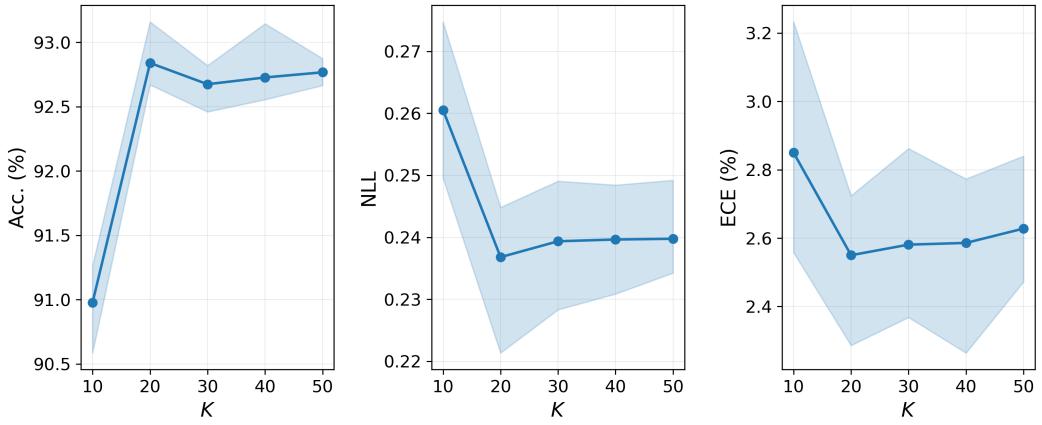

Figure 7: Accuracy, NLL, and ECE versus the number of mean-field iterations on CIFAR-10.

of posterior sampling method, we additionally consider cyclical SGHMC (cSGHMC; Zhang et al., 2020) and preconditioned SGLD (pSGLD; Li et al., 2016). For each sampling method, we compare MoDGF with the corresponding teacher.

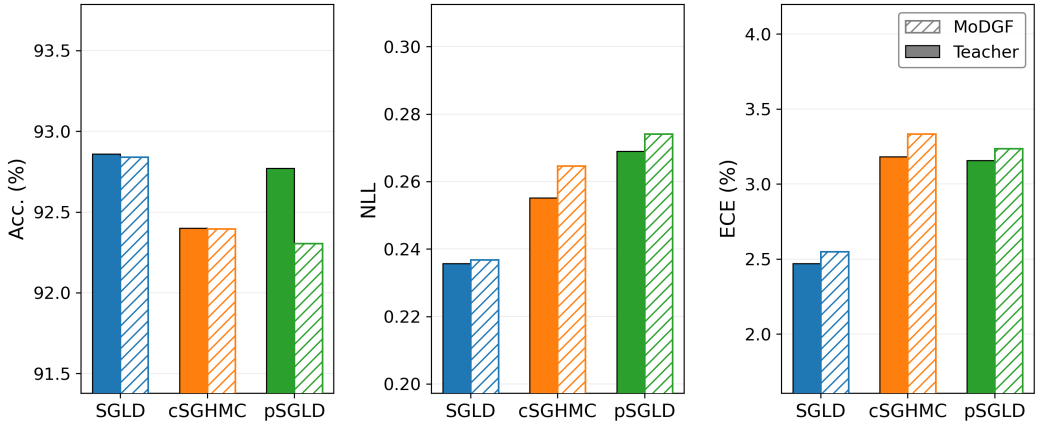

Figure 8: Accuracy, NLL, and ECE of the teacher and MoDGF for different posterior sampling methods on CIFAR-10. Only mean values over five random seeds are shown.

Results. Figure 8 shows that MoDGF achieves predictive performance close to that of the corresponding teacher across SGLD, cSGHMC, and pSGLD. Although the predictive performance varies across posterior sampling methods, the performance gap between MoDGF and its teacher remains relatively small. These results indicate that MoDGF can be applied to different posterior sampling methods without being specific to SGLD.

H Active Learning

In this section, we use active learning to further examine the epistemic uncertainty preserved by posterior distillation. Active learning iteratively selects informative samples from an unlabeled pool and adds their labels to the training set (Settles, 2009). Since epistemic uncertainty measures uncertainty about the model parameters, it provides a natural acquisition score for Bayesian active learning (Gal et al., 2017).

We compare SGLD, Single DGF, and MoDGF using their epistemic uncertainty as the acquisition score. For MoDGF, we additionally consider within-component epistemic uncertainty and between-component epistemic uncertainty to examine how the two terms in the decomposition contribute to sample selection. We also include random sampling and predictive entropy as baselines. Random sampling selects samples uniformly from the unlabeled pool without using model predictions.

Predictive entropy selects samples with the largest entropy of the posterior predictive distribution,

$$U_{\text{total}}(\boldsymbol{x}) = \mathrm{H}\left[p(y \mid \boldsymbol{x}, \mathcal{D})\right]. \tag{103}$$

We conduct experiments on CIFAR-10. We start with 1,000 randomly selected labeled training samples and add 1,000 samples at each acquisition round until the number of labeled samples reaches 10,000. At each round, the acquisition score is calculated for the remaining unlabeled samples, and the 1,000 samples with the largest scores are added to the labeled set. For random sampling, 1,000 samples are selected uniformly at random. We evaluate the classification accuracy on the CIFAR-10 test set after each acquisition round.

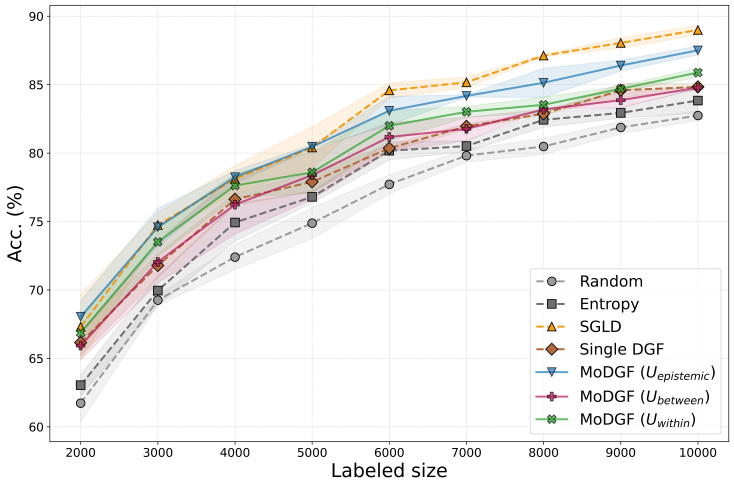

Figure 9: Active learning results on CIFAR-10. Test accuracy is reported as the number of labeled training samples increases from 1,000 to 10,000.

Results. Figure 9 shows that SGLD achieves the highest accuracy across most acquisition rounds. In particular, MoDGF using overall epistemic uncertainty consistently performs closer to SGLD than Single DGF. This result provides additional evidence that MoDGF better preserves the epistemic uncertainty of SGLD after posterior distillation. Within-component epistemic uncertainty also provides effective sample selection, but performs below the overall epistemic uncertainty of MoDGF. In contrast, between-component epistemic uncertainty performs similarly to Single DGF and predictive entropy. These results suggest that two components of epistemic uncertainty can play different roles in uncertainty-based sample selection.

I Computational Costs

In this section, we compare the computational costs of MoDGF with the posterior distillation baselines. We first report the training time and then analyze the FLOPs required during training.

I.1 Training Time

We compare the one-epoch training time of BDK, Hydra, EnD^2, EDFM, Single DGF, and MoDGF on CIFAR-10, CIFAR-100, and Tiny-ImageNet. For a fair comparison, all methods use the same model architectures and the same set of $N = 500$ SGLD posterior samples as in the main experiments. For each method and dataset, we measure the training time for one epoch over 20 runs and report the mean and standard deviation. All experiments were conducted on NVIDIA RTX PRO 6000 Blackwell GPUs with 96 GB of memory, using PyTorch 2.9.1 and CUDA 12.8. Training times are reported in seconds.

I.2 Floating-Point Operations

We further compare the computational cost in terms of FLOPs per training step. Let B denote the mini-batch size, c the number of classes, N the number of SGLD posterior samples, M the

Table 11: Comparison of one-epoch training time (seconds) on CIFAR-10, CIFAR-100, and Tiny-ImageNet. The results are reported as mean and standard deviation over 20 runs.

Method	CIFAR-10	CIFAR-100	Tiny-ImageNet
BDK	3.21 (0.03)	9.21 (0.08)	72.37 (0.09)
Hydra	3.94 (0.09)	10.21 (0.09)	77.70 (0.13)
EnD^2	3.95 (0.09)	10.27 (0.09)	77.94 (0.13)
EDFM	3.65 (0.08)	10.74 (0.08)	72.06 (0.11)
Single DGF	4.92 (0.03)	13.48 (0.03)	72.86 (0.04)
MoDGF	5.27 (0.09)	16.94 (0.23)	86.71 (0.32)

number of design points, and r the latent factor dimension. Let d denote the dimension of the feature representation produced by the shared backbone. We denote the FLOPs for one forward pass through the backbone by F_{body} and those of an output head with input dimension d and output dimension o by $F_{\mathrm{head}}(d, o)$. For EDFM, let F_{vector} denote the FLOPs for one forward evaluation of the vector-field network for a single input, excluding the backbone computation. Finally, α accounts for the forward and backward computations.

BDK and EnD^2 use a single output head, while Hydra uses H heads. Their FLOPs are

$$\mathrm{FLOPs}_{\mathrm{BDK}} = \mathrm{FLOPs}_{\mathrm{EnD}^2} = \alpha B \left[F_{\mathrm{body}} + F_{\mathrm{head}}(d, c)\right], \tag{104}$$

$$\mathrm{FLOPs}_{\mathrm{Hydra}} = \alpha B \left[F_{\mathrm{body}} + H F_{\mathrm{head}}(d, c)\right]. \tag{105}$$

For EDFM, the vector-field network takes the backbone features, the interpolated logits, and time as inputs. With one vector-field evaluation per input during training, its neural-network computation is

$$\mathrm{FLOPs}_{\mathrm{EDFM}} \approx \alpha B \left[F_{\mathrm{body}} + F_{\mathrm{vector}}\right]. \tag{106}$$

For Single DGF, the mean and factor-loading functions are produced by heads with output dimensions c and r, respectively. Thus, its neural-network computation is

$$F_{\mathrm{NN}}^{\text{Single DGF}} = \alpha B \left[F_{\mathrm{body}} + F_{\mathrm{head}}(d, c) + F_{\mathrm{head}}(d, r)\right]. \tag{107}$$

Single DGF additionally requires the closed-form latent-factor update in the E-step.

MoDGF shares the backbone between the two DGF components and uses an additional gating head. Its neural-network computation is

$$F_{\mathrm{NN}}^{\mathrm{MoDGF}} = \alpha B \left[F_{\mathrm{body}} + 2F_{\mathrm{head}}(d, c) + 2F_{\mathrm{head}}(d, r) + F_{\mathrm{head}}(d, 1)\right]. \tag{108}$$

MoDGF additionally performs the structured mean-field E-step. Let $F_{\mathrm{lat}}(M, c, r)$ and $F_{\mathrm{resp}}(M, c, r)$ denote the FLOPs for one latent-factor update and one responsibility update, respectively. Since these updates are repeated K times for N posterior samples, the E-step requires

$$F_{\mathrm{E}}^{\mathrm{MoDGF}} = KN \left[2F_{\mathrm{lat}}(M, c, r) + F_{\mathrm{resp}}(M, c, r)\right]. \tag{109}$$

Unless otherwise stated, we set $K = 20$.

The total training cost of MoDGF is therefore

$$\mathrm{FLOPs}_{\mathrm{MoDGF}} = F_{\mathrm{NN}}^{\mathrm{MoDGF}} + F_{\mathrm{E}}^{\mathrm{MoDGF}} + F_{\mathrm{Q}}^{\mathrm{MoDGF}}, \tag{110}$$

where $F_{\mathrm{Q}}^{\mathrm{MoDGF}}$ denotes the additional computation for the Q-function in the generalized M-step.

The E-step and Q-function computations operate on the model outputs and latent variables without additional evaluations of the shared backbone. Therefore, for a sufficiently large backbone, these additional computations are relatively small compared with the forward and backward computations of the backbone.