Gaussian Mixture Models and Bayesian Neural Networks: Theoretical Background

Abdul Samad Khan

May 2025

Abstract

This document provides a comprehensive theoretical foundation for Gaussian Mixture Models (GMMs) and their application in Bayesian Neural Networks (BNNs). Written as a supplementary resource for a BNN implementation on GitHub, it covers essential probabilistic concepts, including Bayes' rule, normal distributions, the Central Limit Theorem, Multivariate Normal distributions, Maximum Likelihood Estimation, Gaussian Discriminant Analysis, Quadratic Discriminant Analysis, and joint Gaussian inference. The discussion culminates in connecting these concepts to BNNs, emphasizing their role in modeling uncertainty in neural network parameters. This is aimed at graduate students or researchers seeking a clear, in-depth understanding of the probabilistic underpinnings of BNNs.

1 Introduction

As part of our Bayesian Neural Network (BNN) implementation, this document serves as a theoretical companion to the codebase. BNNs extend traditional neural networks by treating weights as random variables, enabling uncertainty quantification in predictions. To ground this approach, we explore Gaussian Mixture Models (GMMs) and related probabilistic tools, which provide a framework for modeling complex distributions and inference in BNNs. We start with foundational concepts and build toward their application in BNNs, assuming a graduate-level understanding of probability and linear algebra.

2 Bayes' Rule

Bayes' rule is the foundation of probabilistic inference, allowing us to update beliefs based on new evidence. For two events A and B, with $\mathbb{P}(B) > 0$, Bayes' rule states:

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\mathbb{P}(B)}, \tag{1}$$

where $\mathbb{P}(A|B)$ is the posterior probability, $\mathbb{P}(B|A)$ is the likelihood, $\mathbb{P}(A)$ is the prior, and $\mathbb{P}(B)$ is the marginal likelihood. In the context of BNNs, Bayes' rule enables us to compute the posterior distribution over network weights given observed data, a process central to Bayesian inference.

3 Normal Distribution

Definition 1 (Normal Distribution)**.** A random variable X follows a normal (Gaussian) distribution, denoted $X \sim \mathcal{N}(\mu, \sigma^2)$, if its probability density function (PDF) is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{2}$$

where μ is the mean and σ^2 is the variance.

The normal distribution is fundamental due to its analytical tractability and prevalence in natural phenomena, as justified by the Central Limit Theorem. In BNNs, weights are often modeled as normally distributed to capture uncertainty.

4 Central Limit Theorem

Theorem 1 (Central Limit Theorem)**.** *Let $X_1, X_2, \ldots, X_n$ be independent and identically distributed random variables with mean μ and variance σ^2. The sum $S_n = X_1 + \cdots + X_n$, when normalized, converges in distribution to a normal distribution as $n \to \infty$:*

$$\frac{S_n - n\mu}{\sqrt{n\sigma^2}} \xrightarrow{d} \mathcal{N}(0,1). \tag{3}$$

The CLT explains why normal distributions appear in aggregated effects, such as stock prices influenced by numerous small factors (e.g., demand, supply disruptions). In BNNs, the CLT supports the use of Gaussian priors for weights, as their combined effect often approximates normality.

5 Multivariate Normal Distribution

Definition 2 (Multivariate Normal Distribution)**.** A random vector $\mathbf{X} = (X_1, \ldots, X_d)^\top \in \mathbb{R}^d$ follows a multivariate normal (MVN) distribution, denoted $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$, if its PDF is:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right), \tag{4}$$

where $\mu \in \mathbb{R}^d$ is the mean vector, and $\Sigma \in \mathbb{R}^{d\times d}$ is the positive-definite covariance matrix.

The MVN models correlated variables, with the covariance matrix Σ defining the elliptical shape of the distribution. In BNNs, MVNs are used to model joint distributions over weights, capturing dependencies across layers.

The geometry of an MVN is described by the eigenvectors and eigenvalues of Σ. The eigenvectors define the principal axes of the elliptical contours, while the eigenvalues determine their lengths. For any 2D projection of an MVN, the contours remain elliptical, reflecting the Gaussian nature of all linear combinations.

6 Maximum Likelihood Estimation for MVNs

Given a dataset $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, where $\mathbf{x}_i \sim \mathcal{N}(\mu, \Sigma)$, the log-likelihood is:

$$\ell(\mu, \Sigma) = -\frac{n}{2} \log |2\pi\Sigma| - \frac{1}{2} \sum_{i=1}^{n} (\mathbf{x}_i - \mu)^\top \Sigma^{-1} (\mathbf{x}_i - \mu). \tag{5}$$

Maximizing this yields the MLEs:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i, \tag{6}$$

$$\hat{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^\top. \tag{7}$$

These estimators are intuitive: the sample mean and covariance. In BNNs, MLE can provide point estimates for weight distributions, though Bayesian methods incorporate priors for uncertainty quantification.

7 Gaussian Discriminant Analysis and Quadratic Discriminant Analysis

Definition 3 (Gaussian Discriminant Analysis)**.** Gaussian Discriminant Analysis (GDA) is a classification method that assumes each class k follows an MVN distribution, $\mathbf{X}|Y = k \sim \mathcal{N}(\mu_k, \Sigma)$, with a shared covariance matrix across classes. The posterior probability of class k given $\mathbf{x}$ is computed via Bayes' rule:

$$\mathbb{P}(Y = k|\mathbf{X} = \mathbf{x}) \propto \mathbb{P}(\mathbf{X} = \mathbf{x}|Y = k)\mathbb{P}(Y = k). \tag{8}$$

The decision boundary is linear due to the shared covariance, leading to Linear Discriminant Analysis (LDA).

Definition 4 (Quadratic Discriminant Analysis)**.** Quadratic Discriminant Analysis (QDA) extends GDA by allowing each class to have its own covariance matrix, $\mathbf{X}|Y = k \sim \mathcal{N}(\mu_k, \Sigma_k)$. This results in quadratic decision boundaries.

In BNNs, GDA/QDA-inspired techniques can model class-conditional distributions, with weights treated as Gaussian random variables, aligning with the Bayesian framework.

8 Joint Gaussian Inference

Consider a partitioned MVN:

$$\begin{bmatrix} \mathbf{X}_a \\ \mathbf{X}_b \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_a \\ \mu_b \end{bmatrix}, \begin{bmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{bmatrix} \right). \tag{9}$$

The conditional distribution $\mathbf{X}_a|\mathbf{X}_b = \mathbf{x}_b$ is also Gaussian:

$$\mathbf{X}_a|\mathbf{X}_b = \mathbf{x}_b \sim \mathcal{N}(\mu_{a|b}, \Sigma_{a|b}), \tag{10}$$

$$\mu_{a|b} = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(\mathbf{x}_b - \mu_b), \tag{11}$$

$$\Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}. \tag{12}$$

This property, used in Kalman filters, is crucial for BNNs, where we condition weight distributions on observed data to obtain posterior distributions.

9 Bayesian Neural Networks: Background and Connection to GMMs

Traditional neural networks produce point estimates for weights, lacking uncertainty quantification. BNNs treat weights as random variables, typically with Gaussian priors, $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$. Given data $\mathscr{D} = \{(\mathbf{x}_i, y_i)\}$, the posterior $p(\mathbf{w}|\mathscr{D})$ is computed using Bayes' rule:

$$p(\mathbf{w}|\mathscr{D}) = \frac{p(\mathscr{D}|\mathbf{w})p(\mathbf{w})}{p(\mathscr{D})}. \tag{13}$$

Exact computation is intractable due to the non-linear likelihood, so approximations like variational inference or MCMC are used, often assuming Gaussian posteriors.

GMMs relate to BNNs by modeling data distributions as mixtures of Gaussians, $p(\mathbf{x}) = \sum_k \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$. In BNNs, GMMs can represent complex data distributions, while the Gaussian priors and posteriors on weights align with the MVN framework. Joint Gaussian inference supports posterior updates, and GDA/QDA provide classification analogs. The CLT justifies Gaussian assumptions for aggregated effects, making this framework robust for uncertainty-aware modeling in our BNN implementation.

10 Conclusion

This document has outlined the probabilistic foundations for our BNN implementation, from Bayes' rule to GMMs and joint Gaussian inference. These concepts provide a rigorous basis for modeling uncertainty in neural networks, as implemented in our GitHub codebase. For graduate students, this serves as a clear entry point to the interplay of probability and machine learning, with practical implications for robust predictive modeling.