RDFIA Part 3

Raouf Toukal 21213774
Keryan Chelouche 28607835

Janvier 2023

Contents

Summary

After initially exploring the core principles of deep learning in the realm of computer vision, particularly through Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), our exploration has expanded through hands-on experimentation with diverse applications like transfer learning and Generative Adversarial Networks (GANs). In this concluding session of our practical series, we will pivot our attention to the emerging field of Bayesian deep learning.

In machine learning, conventional approaches such as linear and logistic regression have long been employed to understand the connections between variables. Yet, with the field's evolution, encountering more intricate datasets and models, the constraints of these traditional methods have become more evident. Enter Bayesian deep learning, which combines the powerful capabilities of deep learning with the adaptability and clarity provided by Bayesian statistics.

The initial section of our report will focus on Bayesian linear regression. Despite its simplicity, this method is powerful in capturing the uncertainty within predictions. We will explore the use of linear, polynomial, and Gaussian basis functions in this framework, examining their unique contributions to representing uncertainty.

Although Bayesian linear regression is straightforward in principle, classification tasks require a deeper theoretical foundation. Herein lies the importance of approximate inference, a cornerstone of Bayesian deep learning essential for quantifying uncertainty in intricate models. In our hands-on sessions, we will delve into and juxtapose various methods of approximate inference, such as Laplace approximation, variational inference, and Markov Chain Monte Carlo (MCMC) techniques, employing straightforward classification datasets for demonstration.

Lastly, we will explore different uses of uncertainty quantification. Our first step will involve using Monte-Carlo Dropout within variational inference to pinpoint the predictions that our model is least certain about. Following this, we will extend our insights to two real-world applications where precise uncertainty measurement is crucial: predicting failures and detecting out-of-distribution (OOD) instances.

Bayesian Linear Regression

Bayesian approaches diverge from classic linear regression methods by considering parameters as random variables, thus focusing on estimating their probability distributions rather than pinpointing exact values. This shift brings us to the notion of predictive distribution, a key feature in Bayesian modeling that encapsulates the uncertainty in forecasts. In essence, Bayesian models provide a spectrum of potential outcomes, each associated with its probability distribution, rather than a singular, fixed prediction.

In this session, we'll explore Bayesian Linear Regression models, experimenting with different types of basis functions, including linear, polynomial, and Gaussian. We'll apply these models to 1D toy regression datasets that vary in complexity, from straightforward linear patterns to more intricate non-linear ones, like ascending sinusoidal curves.

The aim is to gain practical experience with basic Bayesian models, comprehend their mechanics, and acquire a deeper understanding of the predictive distribution.

2.1 Linear Basis function model

In this section, we begin by examining a linear dataset to explore how linear basis functions perform within the context of Bayesian Linear Regression.

2.1.1 Question 1

We implemented the linear basis function as follows :

```
def phi_linear(x):
    return np.vstack((np.ones(len(x)),x)).T
```

2.1.2 Question 2

The posterior distribution's closed form in the linear scenario can be expressed as the multiplication of the likelihood by the prior:

$$p(w|x_i, y_i) \propto p(y_i|x_i, w)p(w)$$

Given that the prior is Gaussian, : $p(w|\alpha) = \mathcal{N}(w; 0, \alpha^{-1}I)$ and the likelihood is also Gaussian, $p(y_i|x_i, w) = \mathcal{N}(\Phi_i^T w, \beta^{-1})$, it follows that the closed form of the posterior is also Gaussian. We can demonstrate that :

$$p(w|X, Y) = \mathcal{N}(w|\mu, \Sigma)$$

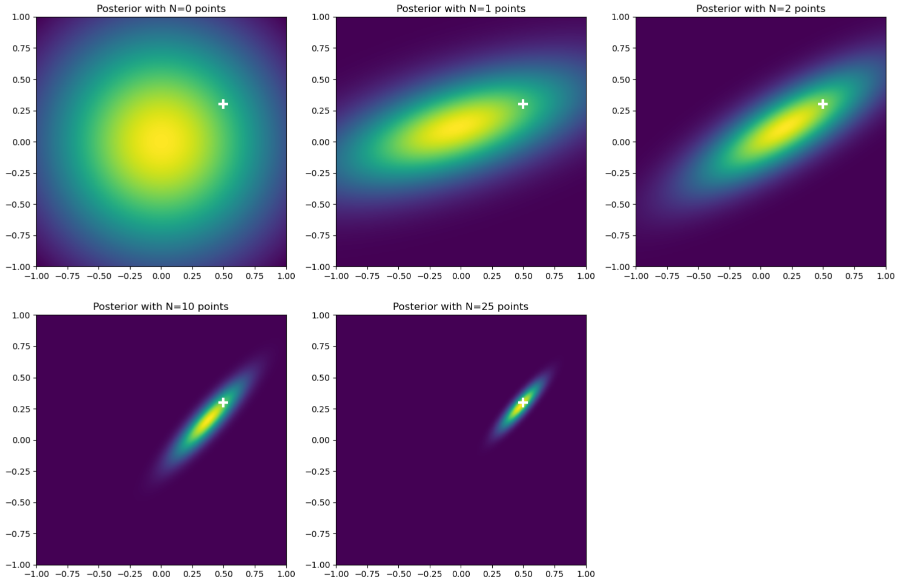


Figure 2.1: Figure representing posterior sampling for different points

where

$$\Sigma^{-1} = \alpha I + \beta \Phi^T \Phi$$

and

$$\mu = \beta \Sigma \Phi^T Y$$

With :
α and β are hyperparameters (β is the noise precision parameter)
I the identity matrix
Y the label matrix and ϕ is the basis function.

To examine how the quantity of data points affects the posterior's (epistemic) uncertainty, we illustrated the outcomes using a bivariate Gaussian distribution within a two-dimensional illustrative example. illustrated in 2.1, where the white cross denotes the optimal parameters.

As the number of data points (N) increases, we observe that the mean of the distribution progressively converges towards the white cross, accompanied by a reduction in variance. From this, we conclude that increasing the number of data points not only brings the posterior distribution (of the parameters w) closer to the ground truth, but also significantly diminishes posterior (epistemic) uncertainty.

2.1.3 Question 3

The predictive distribution is the output space for a new sample, it can be computed by marginalizing over w :

$$p(y|x^*, D, \alpha, \beta) = \int p(y|x^*, w, \beta)p(w|D, \alpha, \beta)\, dw$$

The predictive distribution, being a convolution of the likelihood and the posterior—both Gaussian—results in a Gaussian distribution itself.

$$p(y|x^*, D, \alpha, \beta) = \mathcal{N}\left(y; \mu^T \phi(x^*), \frac{1}{\beta} + \phi(x^*)^T \Sigma \phi(x^*)\right)$$

With :
- Mean of predictive distribution : $\mu^T \Phi(x^*)$
- Variance of predictive distribution :$\sigma^2_{\text{pred}}(x^*) = \frac{1}{\beta} + \Phi(x^*)^T \Sigma \Phi(x^*)$

and, $\Phi(x^*)^T \Sigma \Phi(x^*)$ represents the uncertainty over the parameters w (epistemic uncertainty).

2.1.4 Question 5

In this question, we aim to graphically represent and scrutinize the outcomes predicted by our Bayesian Linear Regression model. shown in the 2.2.

To the left, the prediction of model is illustrated in red, whereas the actual values are presented in green. One can observe that the uncertainty (indicated by the different intensities of red) grows as we depart from the region with data.

To deepen our comprehension, we have depicted the predictive variance (which signifies uncertainty) on the figure to the right. Upon examining the form of the uncertainty function in relation to x^*, we observe that it forms a 2D parabola, with its minima located at the center of the data points.

Additionally, it is observed that the predictive variance expands when distancing from the data distribution. This can be linked to the model's heightened certainty nearer to the dataset and, inversely, its escalating uncertainty as it diverges from the dataset.

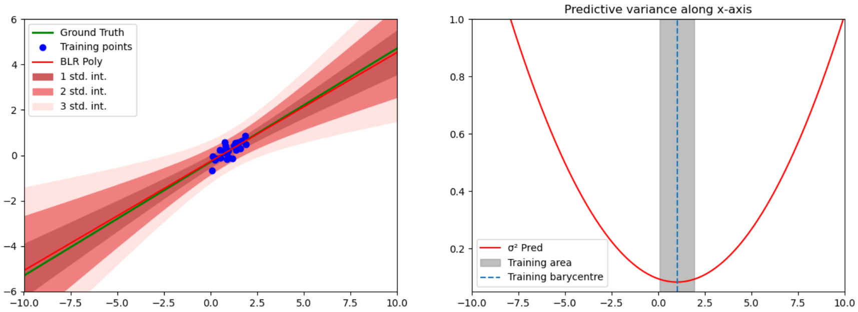


Figure 2.2: Figure representing the predictive distribution on the test set using a linear ϕ with Bayesian Linear Regression

We can analytically prove these results in the scenario where $\alpha = 0$ and $\beta = 1$. To compute, $\sigma^2_{pred}(x^*)$ the following steps can be undertaken :

$$\Sigma^{-1} = \alpha I + \beta \Phi^T \Phi = \Phi^T \Phi = \begin{pmatrix} N & \beta 1^T X \\ \beta 1^T X & \beta X^T X \end{pmatrix} = \begin{pmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix}$$

Σ^{-1} is a 2×2 matrix, which means that its inverse can be calculated as :

$$\Sigma = \frac{1}{\det \Sigma^{-1}} \begin{pmatrix} \sum x_i^2 & -\sum x_i \\ -\sum x_i & n \end{pmatrix} = \frac{1}{n \sum x_i^2 - (\sum x_i)^2} \begin{pmatrix} \sum x_i^2 & -\sum x_i \\ -\sum x_i & n \end{pmatrix}$$

We also know that the predictive variance can be written as :

$$\sigma^2_{pred}(x^*) = \frac{1}{\beta} + \phi(x^*)^T \Sigma \phi(x^*)$$

Since $\beta = 1$:

$$\sigma^2_{pred}(x^*) = \phi(x^*)^T \Sigma \phi(x^*) = (1 \quad x^*) \Sigma \begin{pmatrix} 1 \\ x^* \end{pmatrix}$$

By replacing Σ and $\phi(x^*)$ we get:

$$\sigma^2_{pred}(x^*) = \frac{\sum x_i^2 - x^* \sum x_i + x^*(-\sum x_i + n x^*)}{n \sum x_i^2 - (\sum x_i)^2} = \frac{\sum x_i^2 - 2x^* \sum x_i + n(x^*)^2}{n \sum x_i^2 - (\sum x_i)^2}$$

$$= \frac{n(\frac{\sum x_i^2}{n} - 2x^* \frac{\sum x_i}{n} + (x^*)^2)}{n^2(\frac{\sum x_i^2}{n} - (\frac{\sum x_i}{n})^2)} = \frac{\frac{\sum x_i^2}{n} - 2x^* \bar{x} + (x^*)^2}{n(\frac{\sum x_i^2}{n} - \bar{x}^2)}$$

$$= \frac{\frac{\sum x_i^2}{n} - 2x^* \bar{x} + (x^*)^2 + \bar{x}^2 - \bar{x}^2}{n \text{Var}(X)} = \frac{\text{Var}(X) + (x^* - \bar{x})^2}{n \text{Var}(X)}$$

To conclude, we have :

$$\sigma^2_{pred}(x^*) = \frac{1}{n} + \frac{1}{n \text{Var}(X)} (x^* - \bar{x})^2$$

From the equation for predictive variance, when $\alpha = 0$ and $\beta = 1$, we deduce that the minimum is attained at $x^* = \bar{(}x)$, which is the mean of the data points.

This observation aligns with our previous findings illustrated in the preceding figure: the predictive variance serves as an indicator of the model's uncertainty, which is reduced in areas close to the mean of the data.

2.1.5 Bonus

To assess how data impacts uncertainty, we will examine the same function's performance across different datasets. The key difference in this new dataset is the existence of a gap between clusters of training points, creating a more complex learning environment for the model. The outcomes are depicted in 2.3.

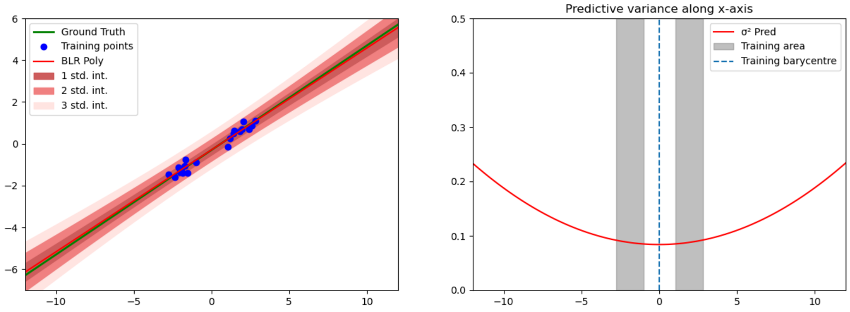


Figure 2.3: Figure representing the predictions on the test set of the hole dataset using a linear ϕ

Applying the same analytical approach as previously, we find that the lowest point of the uncertainty function is situated between the two point clusters, at the data's midpoint. This is unexpectedly counterintuitive; given the lack of points in the center, one might predict higher uncertainty there. Yet, the model exhibits the least uncertainty at the mean of the data, suggesting a possible shortfall in its performance.

This demonstrates that while linear regression can be effective, its capabilities are indeed limited and may not be suitable for complex datasets.

2.2 Non-Linear models

In this section, we will use a more complex toy dataset, which is an increasing sinusoidal curve, and extend the linear model into a non-linear regression.

The goal of this part is to get insight on the importance of the chosen basis function on the predictive variance behavior.

2.2.1 Polynomial basis functions

Question 1

We implemented the polynomial basis function as follows :

```
def phi_polynomial(x):

    D = 10
    phi = np.array([x_i**d for d in range(D) for x_i in x]).reshape(D, -1).T
    return phi
```

Question 2

In this instance, we employ a polynomial basis function. The model's results are presented in 2.4. Initially, we can observe in the left figure that the model exhibits robust performance in the vicinity of the training data points. This is corroborated by the visualization of the predictive variance in the right figure, where we note that the minimum uncertainty extends throughout the entire training data region, not just at the center.

Additionally, in both the left and right figure, it is clear that as one moves away from the data points — zones distant from the true values (highlighted in green in the left image) — uncertainty markedly increases. This feature is beneficial for the model since it signifies an ability to identify areas of reduced confidence.

To sum up, employing a polynomial basis for predictive variance provides a more nuanced portrayal, effectively capturing uncertainty in a manner that exceeds what is achievable with a linear basis.

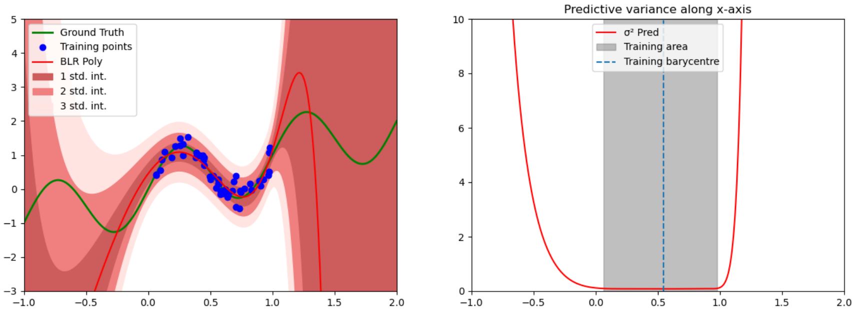


Figure 2.4: Figure representing the predictions on the test set using a polynomial Φ

2.2.2 Gaussian basis function

An additional intriguing choice is the Gaussian basis function. 2.5 illustrates the predictive variance obtained with this basis.

On the left, we observe once more that the model's forecast (illustrated in red) is closely in sync with the actual data (shown in green) in regions close to the training points.

The predictive function, illustrated on the right, takes on an unconventional form when contrasted with earlier outcomes. Unlike the polynomial function that demonstrates a reduction then a rise in uncertainty, the Gaussian basis function presents variability, especially near the training data region. Beyond this area, it stabilizes to a constant value. This behavior deviates from expectations, as uncertainty is typically presumed to increase with distance from the training data.

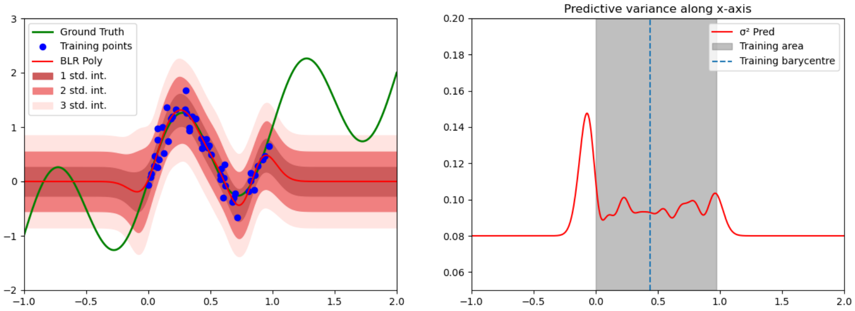


Figure 2.5: Figure representing the predictions on the test set using a gaussian Φ

Question 5

When employing localized basis functions such as Gaussians, the epistemic uncertainty $\phi(x^*)^T\Sigma\phi(x^*)$ diminishes to zero as we distance ourselves from our training data. It is important to recall that the predictive variance is expressed as $\sigma^2_{pred}(x^*) = \frac{1}{\beta} + \phi(x^*)^T\Sigma\phi(x^*)$. Consequently, when the epistemic uncertainty tends towards zero, the predictive variance $\sigma^2(x^*)$ simplifies to $\frac{1}{\beta}$, where β denotes the noise precision factor. Hence, the uncertainty converges to the value of $\frac{1}{\beta}$, which equates to 0.08 in our scenario.

Approximate inference

In tasks involving classification, acquiring the posterior probability $P(w|D)$ might be challenging, leading to the necessity of employing approximation strategies. This discussion will delve into exploring methods for approximate inference, including the Laplace approximation, variational inference employing mean-field approximation, and the Monte Carlo dropout technique, with a particular emphasis on their application to binary classification datasets in two dimensions.

Our goal is to comprehend how these methods are implemented and how effective they are on linear and non-linear data structures by applying them in practice.

3.1 Bayesian Logistic Regression

3.1.1 Question 1

Figure 3.1 Displays the uncertainty linked with the baseline in the linear case using the Maximum a Posteriori (MAP) estimate. It's significant to note that the uncertainty remains constant and does not increase as the distance from the training data grows.

considering $p(y = 1|x, w_{MAP})$,the uncertainty stays unchanged, even for points far from the training distribution. This indicates that the model's confidence does not vary with the distance from the training points, a characteristic usually considered undesirable. From this, we infer that a more accurate approximation method would be more suitable.

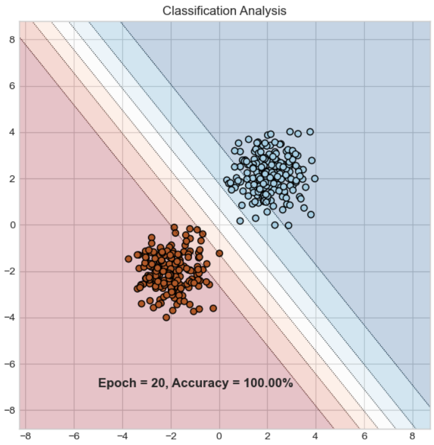


Figure 3.1: Figure representing the result of the baseline MAP for the Logistic Regression

3.1.2 Question 2

To approximate the posterior distribution, we utilize another commonly used approximation technique: the Laplace approximation. Figure 3.2 illustrates that the uncertainty grows as the distance from the training distribution increases. This increase in uncertainty is especially noticeable around the decision boundary, signifying epistemic uncertainty.

Compared to the Maximum a Posteriori (MAP) estimate, the predictive distribution from the Laplace approximation provides a closer and more accurate reflection of the model's certainty. The reason for this is that the MAP estimate approximates a single point, whereas the Laplace approximation encompasses a broader view of the posterior distribution.

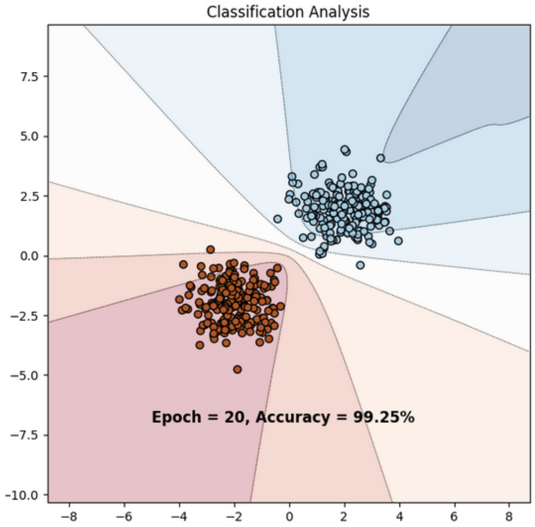


Figure 3.2: Figure representing the result of the Laplace approximation for the Bayesian Logistic Regression

3.1.3 Question 3

The weight decay hyperparameter is key in controlling the model's complexity. It works by adding a penalty to the loss function for larger weights, encouraging the model to favor smaller weight values, typically aiding in avoiding overfitting. This idea of regularization corresponds with the concept of precision (inverse variance) in the prior distribution over the weights from a Bayesian perspective. A higher weight decay value corresponds to a tighter prior, pulling the weights closer to zero unless compelling evidence from the data suggests otherwise. This regularization effect can significantly impact the predictive distribution.

A larger value for the weight decay hyperparameter equates to a more stringent prior, drawing the weights towards zero unless the data provides strong evidence to the contrary. This effect of regularization can have a marked influence on the predictive distribution.

As observed in Figures 3.3 and the previously mentioned text, varying the weight decay parameter results in distinct predictive behaviors.

Setting the weight decay too high results in heightened predictive uncertainty, which is depicted as broader shaded regions in graphical representations. On the other hand, lower values of weight decay lead to increased confidence, as shown by the more constricted shaded areas. Finding an optimal level of weight decay is crucial for the model's ability to generalize well, since values that are too high or too low may not be suitable for data the model has not encountered before.

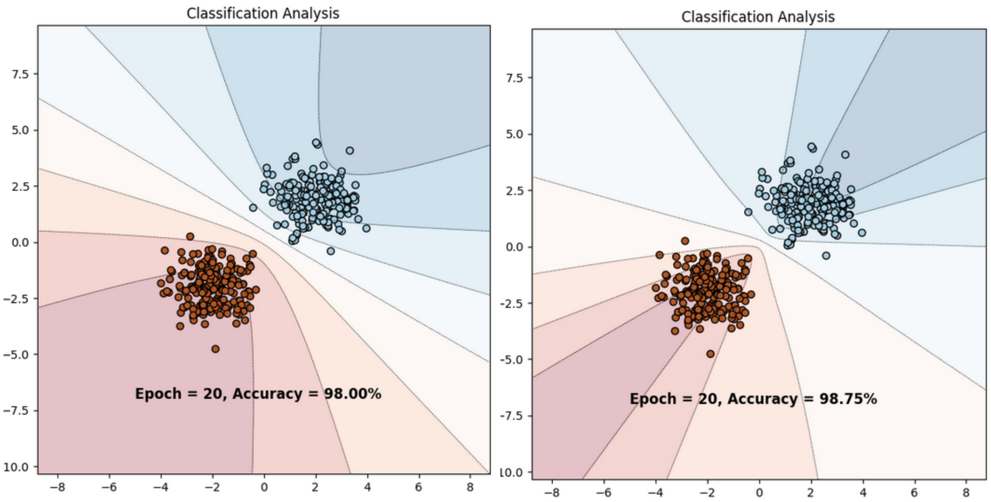


Figure 3.3: Figure representing the effect of WEIGHT_DECAY Regularization Hyperparameter: Left (0.5) vs. Right ($5e^{-10}$) Weight Decay

3.1.4 Variational Inference

The provided code snippet details the framework of two Python classes, VariationalLogisticRegression and LinearVariational. These classes are structured to integrate variational inference techniques for modeling uncertainty within logistic regression tasks. The `LinearVariational` class acts as a variational counterpart to a standard linear layer, characterized by variational parameters for weights (`'w_mu'`, `'w_rho'`) and biases (`'b_mu'`), alongside establishing a prior variance for these parameters. A key functionality within this class is the `sampling` method, leveraging the reparameterization trick to draw samples from the weights' variational posterior distribution, facilitating gradient-based optimization. The `kl_divergence` method is responsible for calculating the Kullback-Leibler divergence, providing a measure of the discrepancy between the variational posterior and the prior distributions of the weights. During the forward pass, implemented by the `forward` method, a linear transformation is executed using the sampled weights and biases' mean values.

The `VariationalLogisticRegression` class extends the functionality of the `LinearVariational` layer to facilitate a logistic regression model using variational inference techniques. It starts with initializing a linear variational layer and incorporates the sigmoid function within its `forward` method to execute the logistic regression's forward operation. To compute the loss, this class's `kl_divergence` method relies on the `LinearVariational` layer for assessing the KL divergence, thereby embedding the variational inference approach within the logistic regression framework.

This structure allows for the estimation of Bayesian uncertainty in logistic regression models by employing variational inference to approximate the weight posterior distributions. This provides a systematic way to measure prediction uncertainty.

Question 5

The training loop utilizes a conventional PyTorch setup, with the goal of optimizing the Evidence Lower Bound (ELBO). The ELBO is an essential criterion that balances the negative log-likelihood (NLL) of the data given the model and the Kullback-Leibler (KL) divergence between the variational distribution $q_\theta(w)$ and the prior distribution $p(w)$. The NLL, calculated via binary cross-entropy loss, evaluates how well the model fits the data, whereas the KL divergence acts as a regularization term, discouraging the variational distribution from straying too far from the prior and reducing the information loss when $q_\theta(w)$ approximates $p(w)$. In practice, training involves minimizing the negative ELBO, which integrates the NLL and KL terms into a unified loss function. Through gradient descent, the parameters θ of the variational distribution are adjusted to more closely resemble the true posterior. This procedure includes resetting gradients, computing the loss, backpropagating, updating parameters, and intermittently checking the model's predictive performance and visualizing its decision boundary.

Figure 3.4 demonstrates that while the uncertainty resembles that of the Laplace approximation, it becomes more pronounced near the decision boundary and maintains a desirable level of uncertainty for distant points. Despite using three different approximation methods, prediction accuracy remains consistent. However, the Laplace and Variational Inference methods excel by providing a nuanced view of model confidence, ensuring predictions are not uniformly certain across the input space, thereby enhancing decision-making in areas with sparse data.

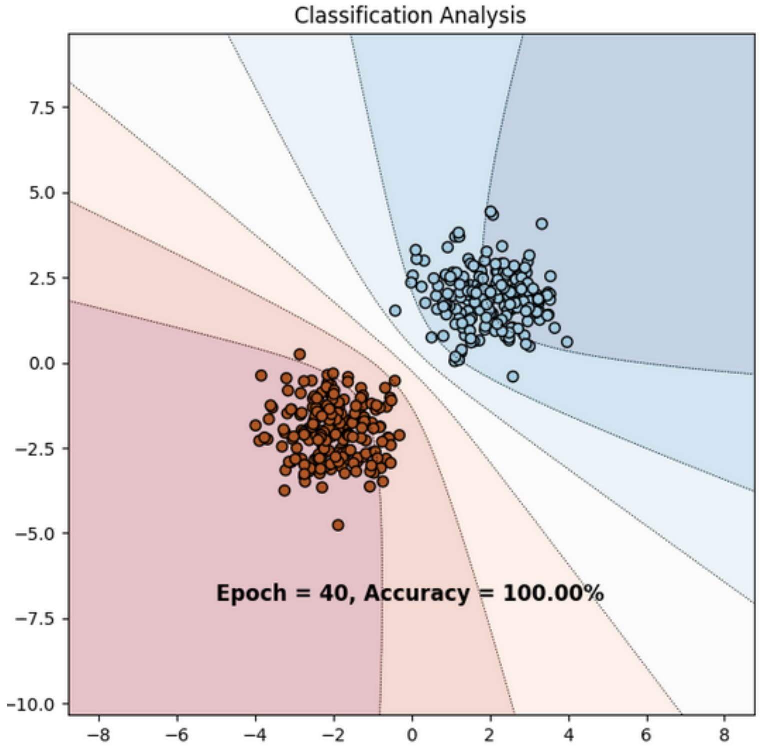


Figure 3.4: Figure representing the result of the Variational approximation with the Logistic Regression

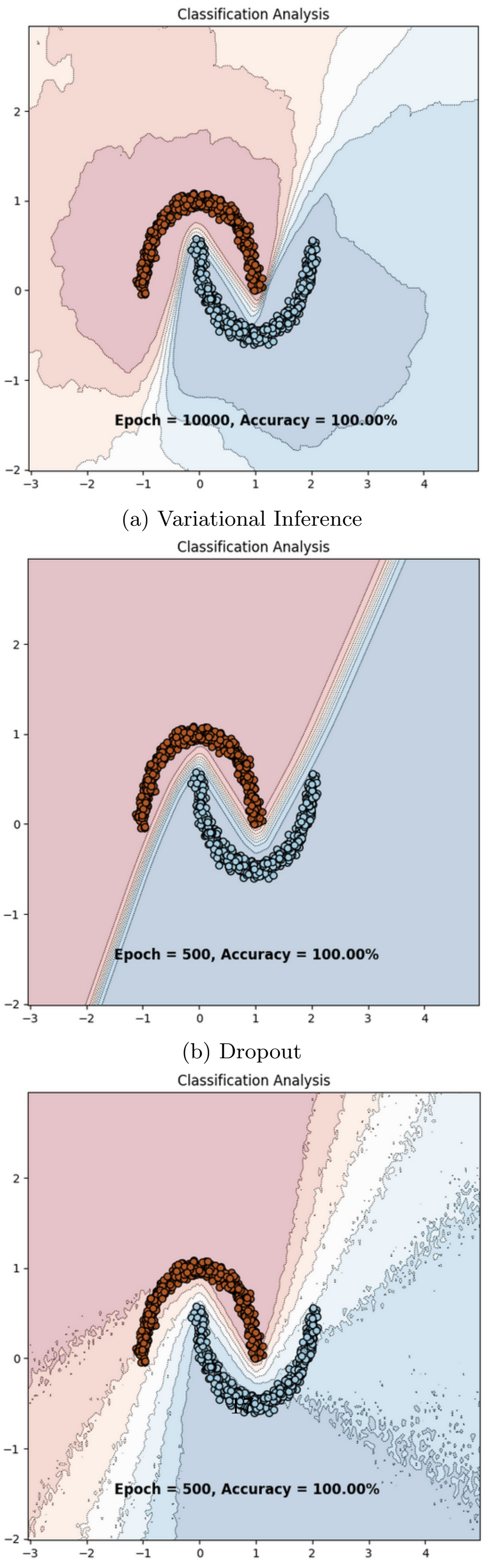
3.2 Bayesian Neural Networks

In this part, the investigation broadens to include Variational Inference (VI) in contexts involving non-linearity, and introduces Monte Carlo Dropout (MC dropout) as a new method of approximation, with results shown in Figure 3.5.

The initial image, depicting VI, verifies the expectation that predictive uncertainty increases as one moves away from the data points. There's a discernible decrease in areas of uncertainty, credited to the non-linear decision boundary which limits the model's choices, thus enhancing its confidence.

The application of classical dropout tightens uncertainty, yet transitioning to MC dropout yields superior results by significantly amplifying predictive variance away from the training data. The primary advantage of MC dropout over Bayesian Logistic Regression (BLR) lies in its enhanced precision in uncertainty estimation, coupled with its simplicity of implementation and reduced computational demands, making it a compelling choice for uncertainty quantification in complex models.

Using traditional dropout narrows the uncertainty, but switching to MC dropout brings about markedly better outcomes by greatly increasing the predictive variance distant from the training data. The key benefit of MC dropout compared to Bayesian Logistic Regression (BLR) is its improved accuracy in estimating uncertainty, along with its ease of implementation and lower computational needs, positioning it as an attractive option for assessing uncertainty in intricate models.



(a) Variational Inference

(b) Dropout

(c) MC Dropout

Applications of uncertainty

In this concluding lab session, our attention will be centered on applications revolving around uncertainty estimation.

To start, we'll employ MC Dropout variational inference to qualitatively assess the images that exhibit the highest uncertainty as per the mode. Following that, we'll delve into two scenarios where accurate uncertainty estimation is vital: predicting failures and detecting out-of-distribution instances.

The aim is to acquire practical experience in utilizing uncertainty estimation for predicting failures and identifying out-of-distribution instances.

4.1 Monte-Carlo Dropout on MNIST

In this part, our focus will be on implementing Monte-Carlo Dropout variational inference on the MNIST dataset. We intend to obtain a measure of uncertainty using the MC Dropout approach, assisting in pinpointing the most uncertain images within the dataset. The objective is to investigate the uncertainty levels in randomly selected images predicted with high confidence as well as those predicted with the least confidence.

4.1.1 Question 1

The results are shown in the figures 4.1 and 4.2. There is a noticeable difference between the randomly chosen examples and those identified as least confident. The randomly selected confident images are clearly legible and easily recognizable, typically marked by a concentrated probability distribution reflecting the model's strong conviction in its prediction. Conversely, the images deemed least confident tend to be ambiguous and can be challenging to interpret, even for humans, exhibiting probability distributions that are dispersed over several classes. Moreover, the histograms associated with the images offer insights into the instances of failure. For the confidently chosen random images, a pronounced peak in the histogram, aligning with the predicted class, distinctly shows the model's assurance. Yet, this pattern shifts markedly for the images with the highest uncertainty. In these cases, the histograms display several peaks, indicating a significant degree of uncertainty and a broader spread in the probability distribution. This mirrors the human response to ambiguous images, where the model displays indecision among various class labels, as evidenced by the multiple peaks in the histograms.
For instance, in the second image among the least certain ones, the model struggles to distinguish between '6' and '8'.

It's also noticeable that the variance ratio increases when dealing with the most uncertain samples, further highlighting the dispersion and ambiguity in their probability distributions.

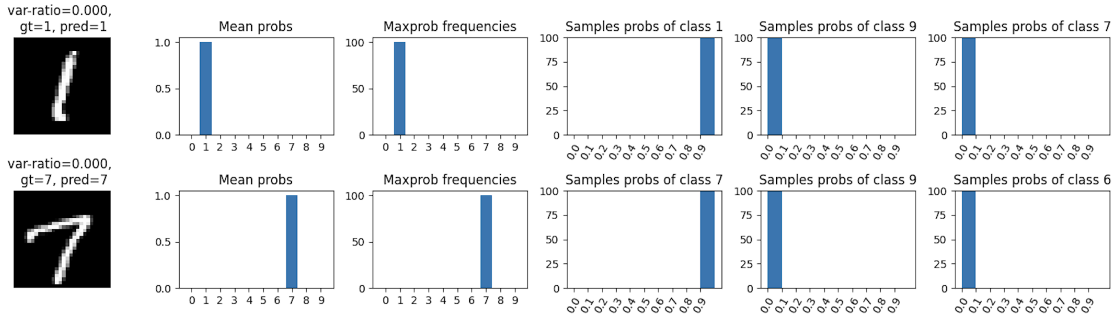


Figure 4.1: Figure representing the result of the confident samples of MC sampling

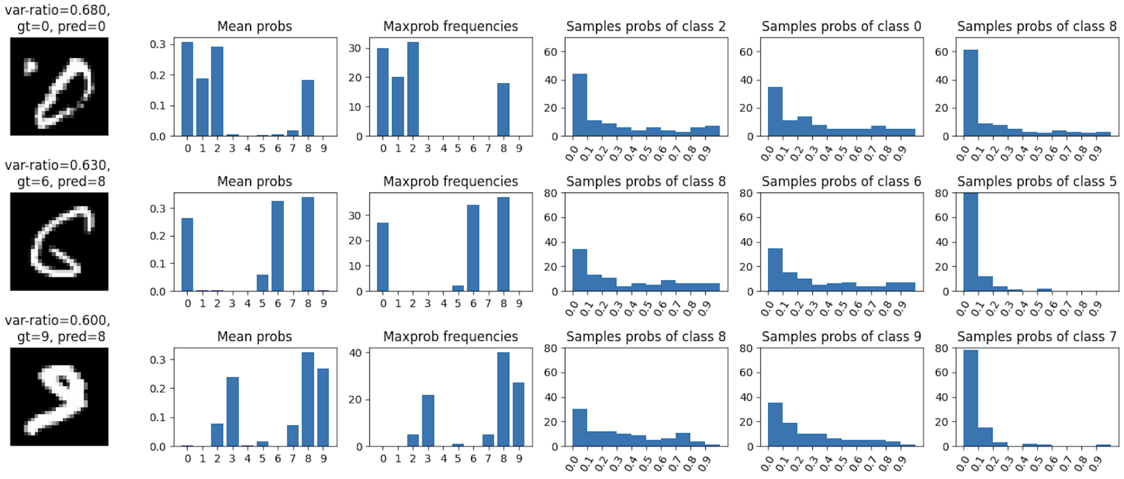


Figure 4.2: Figure representing the result of the most uncertain samples of MC sampling

4.2 Failure prediction

The objective of this segment is to establish dependable confidence metrics for model forecasts, enabling the differentiation between accurate and inaccurate predictions. By possessing a robust measure of confidence, a system can decide whether to trust its own prediction, defer to human oversight or a secondary system, or trigger an alert. We will investigate ConfidNet, a technique specifically designed for predicting failures, and assess its effectiveness in comparison to MCDropout by employing entropy and Maximum Class Probability (MCP) as benchmarks. We assess the effectiveness of ConfidNet in identifying failures relative to earlier benchmarks, such as Maximum Class Probability (MCP) and

MCDropout, using entropy as a measure. In our experiments, classification errors are treated as the positive class for detection purposes.

4.2.1 Question 1

Figure 4.3 showcases the performance indicators, including the precision-recall curve and AUPR (Area Under the Precision-Recall Curve), for ConfidNet versus the baseline approaches MCP and MC-Dropout. This figure effectively evaluates the reliability of the confidence measures by ranking test samples according to their respective uncertainty levels. Through this evaluation, ConfidNet is highlighted as the leading method, demonstrating the most advantageous precision-recall curve and achieving the highest AUPR at 50.71

It's crucial to acknowledge that, despite the comparative rankings, all models show less than ideal performance. In an optimal situation, both precision and recall should be elevated; however, in these instances, precision tends to diminish to nearly zero as recall peaks. This trend underscores a notable shortfall in the models' capacity to confidently and precisely pinpoint pertinent cases, especially in contexts where achieving high recall is imperative.

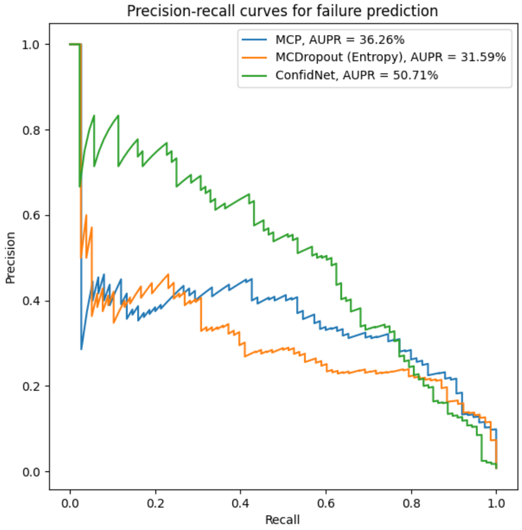


Figure 4.3: Figure representing the precision-recall curves of each method along with their AUPR

Noted that we used AUPR metric instead of standard AUROC because AUPR is more informative where there is a significant class imbalance. This is the case in failure prediction, where the number of failures is much smaller than the number of correct predictions.

4.3 Out-of-distribution detection

Assessing uncertainty for novel instances, particularly those that are out-of-distribution (OOD), is crucial in visual recognition tasks.

Figure 4.4 illustrates the precision-recall curves for different Out-Of-Distribution (OOD) detection techniques, along with their respective AUPR (Area Under the Precision-Recall Curve) scores. The precision-recall curves showcase notable performances for all three models under consideration. ODIN takes the lead with an AUPR of 98.89

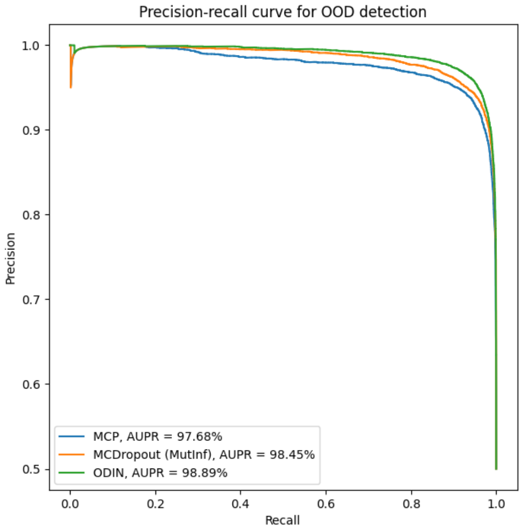


Figure 4.4: Figure representing the precision-recall curves of each OOD method along with their AUPR