Faculté des Sciences et Ingénierie - Sorbonne université

Master Informatique parcours - Données Apprentissage et Connaissances

RDFIA

Report on practical sessions

Section 3 : Bayesian deep learning

Done by:
BENHADDAD Sabrina
BENSIDHOUM Azzedine

February 2024

Contents

List of Figures

Summary

After initially studying the fundamental concepts of deep learning as applied to computer vision, such as Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), we have deepened our understanding through experimentation with various applications of these techniques, including transfer learning and Generative Adversarial Networks (GANs). In this third and final practical session block, our focus will shift to a new aspect : Bayesian deep learning.

In the field of machine learning, traditional methods like linear and logistic regression have been widely used to model relationships between variables. However, as the field has advanced with increasingly complex datasets and models, the limitations of these methods have become apparent. This is where Bayesian deep learning comes into play, merging the strengths of deep learning with the flexibility and interpretability of Bayesian statistics.

The first part of our report will concentrate on Bayesian linear regression. This approach, while straightforward, is potent in modeling uncertainty in predictions. We will investigate the utilization of linear, polynomial, and Gaussian basis functions in this context and how they distinctly represent uncertainty.

While Bayesian linear regression is conceptually simple, classification methods demand a more theoretical approach. This is where approximate inference becomes critical. Approximate inference is a pivotal component of Bayesian deep learning, representing a crucial aspect that allows us to quantify uncertainty in complex models. During our practical sessions, we will explore and compare different approximate inference approaches, including the Laplace approximation, variational inference, and Markov Chain Monte Carlo (MCMC) approximation, using simple classification datasets.

Finally, we will delve into various applications of uncertainty quantification. We will initially employ Monte-Carlo Dropout for variational inference to assess the predictions our model is most uncertain about. Subsequently, we will apply our learnings to two practical scenarios where accurate uncertainty estimation is vital : failure prediction and out-of-distribution (OOD) detection.

Chapter 1

Bayesian linear regression

Bayesian methods, in contrast to traditional linear regression techniques that provide point estimates for parameters, adopt a distinct approach. They treat parameters as random variables and estimate the probability distribution over these parameters. This leads us to the concept of the predictive distribution, which essentially gives an overview of how Bayesian models quantify uncertainty in their predictions. Essentially, instead of having a single prediction, we have a set of possible predictions, each with its own distribution.

During this session, we will be working with Bayesian Linear Regression models, varying the basis functions (linear, polynomial, and Gaussian). The datasets used are 1D toy regression samples, ranging from linear datasets to more complex non-linear datasets, such as increasing sinusoidal curves.

The goal is to get hands-on experience with simple Bayesian models, understand how they work, and gain finer insights into the predictive distribution.

1.1 Linear Basis function model

In this section, we will start with a linear dataset, where we will analyze the behavior of linear basis functions in the framework of Bayesian Linear Regression.

1.1.1 Question 1

We have implemented the linear basis function as follows :

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

1.1.2 Question 2

The closed form of the posterior distribution in linear case can be written as the product of the likelihood and the prior :

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

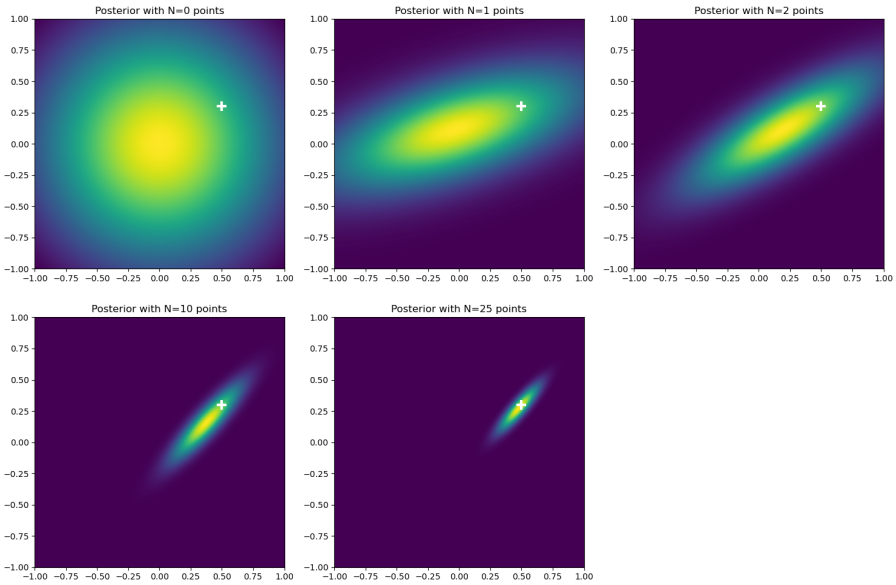


Figure 1.1: Figure representing posterior sampling for different points

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 :

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

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 is the identity matrix, Y is the label matrix and ϕ is the basis function.

In order to study the impact of the number of data points on posterior (epistemic) uncertainty, we visualized the resulting multivariate Gaussian distribution using a two-dimensional toy example. This is illustrated in 1.1, 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.

1.1.3 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 is a convolution of two Gaussians (the likelihood and the posterior), so it is a Gaussian as well. It can be written as :

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

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

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

1.1.4 Question 5

In this question, we will visualize and analyze the result of the predictions of our Bayesian Linear Regression model, shown in the 1.2.

On the left, the model's prediction is shown in red, while the ground truth is depicted in green. It is noticeable that the uncertainty (represented by varying shades of red) increases as we move away from the data.

To better understand this aspect, we have projected predictive variance (representing uncertainty) in the right-hand figure. Analyzing the uncertainty function's shape with respect to x^*, we observe that it forms a 2D parabola, with its minima located at the center of the data points.

Furthermore, we note that the predictive variance increases as we move away from the data distribution. This phenomenon can be attributed to the model's increasing certainty as it approaches the dataset, and conversely, its growing uncertainty when moving away from the dataset.

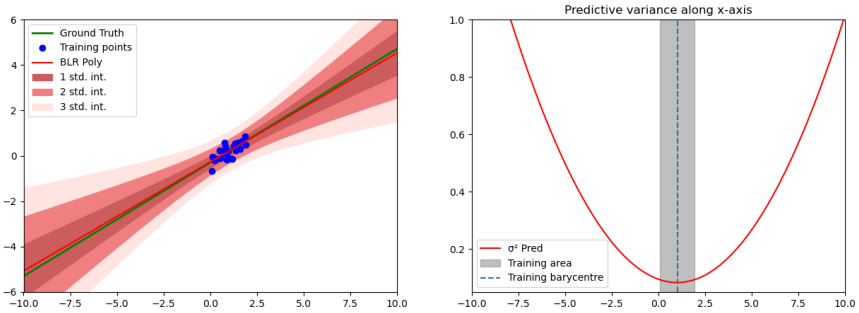


Figure 1.2: Figure representing the predictive distribution on the test set using a linear ϕ with Bayesian Linear Regression

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 $\beta = 1$:

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

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

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

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

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

To conclude, we have :

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

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 is consistent with what we have noted in the previous figure : the predictive variance signifies the model's uncertainty, and it is minimized in the vicinity of the data's mean.

1.1.5 Bonus

To evaluate the influence of data on uncertainty, we will compare the behavior of the same function across different datasets. The primary distinction with this new dataset is the presence of a gap between clusters of training points, which presents a more challenging learning scenario for the model. The outcomes are depicted in 1.3.

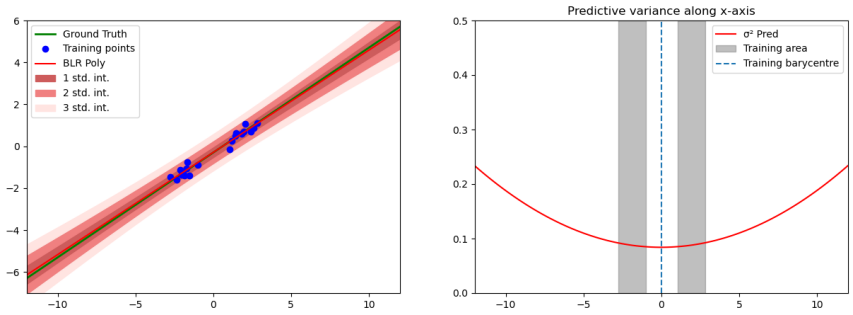


Figure 1.3: Figure representing the predictions on the test set of the hole dataset using a linear ϕ

Using the same analysis as before, we observe that the minimum of the uncertainty function lies between the two clusters of points, at the center of the data. Contrary to what was expected, this is counterintuitive; with an absence of points at the center, one would anticipate greater uncertainty in that region. Nevertheless, the model assigns the lowest uncertainty at the data's mean, which indicates a potential inadequacy in performance.

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

1.2 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.

1.2.1 Polynomial basis functions

Question 1

We have 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 1.4. 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.

Furthermore, in both the left and right figures, it is apparent that moving away from the data points — areas not in proximity to the ground truth (indicated in green in the left figure) — results in substantially increased uncertainty. This characteristic is advantageous for the model, as it indicates a capacity to recognize areas of low confidence.

In conclusion, the predictive variance when using a polynomial basis offers a more refined representation, accurately depicting uncertainty in a way that surpasses the linear basis.

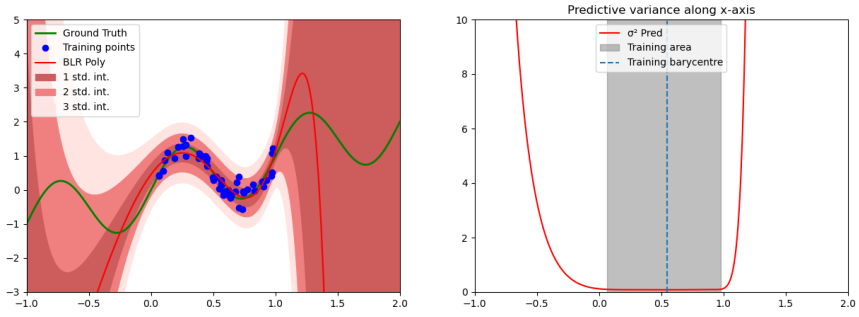


Figure 1.4: Figure representing the predictions on the test set using a polynomial Φ

1.2.2 Gaussian basis function

An additional intriguing choice is the Gaussian basis function. 1.5 illustrates the predictive variance obtained with this basis.

On the left, we once again see that the model's prediction (depicted in red) closely aligns with the ground truth (in green) in areas near the training data.

The predictive function, displayed on the right, adopts an atypical shape compared to previous results. Contrary to the polynomial function, which exhibits a decrease followed by an increase in uncertainty, the Gaussian basis function shows fluctuations, particularly around 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.

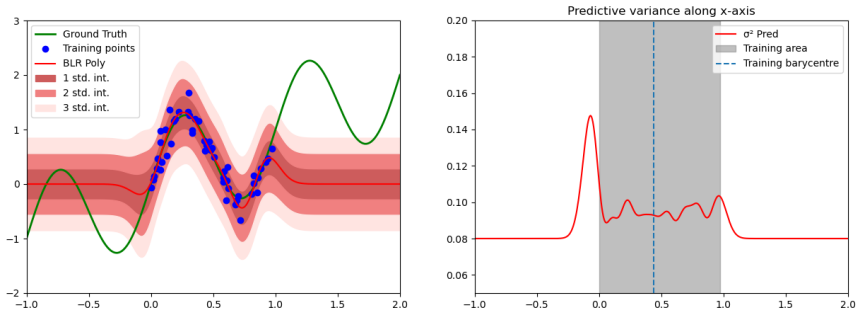


Figure 1.5: Figure representing the predictions on the test set using a gaussian Φ

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.

Chapter 2

Approximate inference

In classification tasks, obtaining the posterior $P(w|D)$ can be elusive, prompting the need for approximation methods. This session will focus on examining approximate inference techniques such as Laplacian approximation, variational inference with mean-field approximation, and Monte-Carlo dropout, specifically applied to 2D binary classification datasets.

Our aim is to gain an understanding of these methods' implementation and efficacy on both linear and non-linear data structures through practical application.

2.1 Bayesian Logistic Regression

2.1.1 Question 1 : Maximum-A-Posteriori Estimate

Figure 2.1 presents the uncertainty associated with the baseline in the linear scenario using the Maximum a Posteriori (MAP) estimate. It's notable that the uncertainty does not escalate with increasing distance from the training data.

When we consider $p(y = 1|x, w_{MAP})$, the uncertainty remains constant, even for points well away from the training distribution. This implies that the model's confidence remains unchanged irrespective of the distance from the training points—a trait that is not typically desired. This observation leads us to conclude that a more precise method of approximation would be preferable.

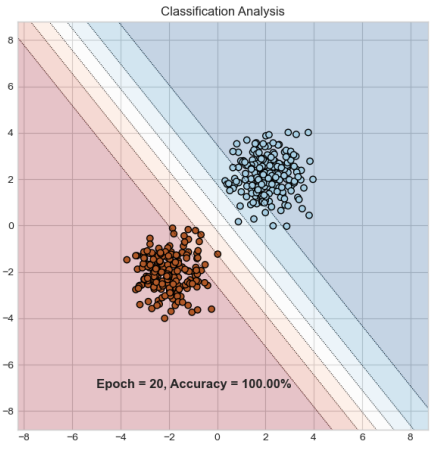


Figure 2.1: Figure representing the result of the baseline MAP for the Logistic Regression

2.1.2 Question 2 : Laplace Approximation

To estimate the posterior distribution, we employ an additional standard approximation method: Laplace approximation. Figure 2.2 demonstrates that the uncertainty increases as we move away from the training distribution. This uncertainty is particularly pronounced around the decision boundary, indicating epistemic uncertainty.

In comparison to the previous Maximum a Posteriori (MAP) estimate, the predictive distribution obtained through Laplace approximation provides a more detailed and accurate representation of the model's certainty level. This distinction arises because the MAP estimate is a point approximation, whereas Laplace approximation yields a comprehensive representation of the entire posterior distribution.

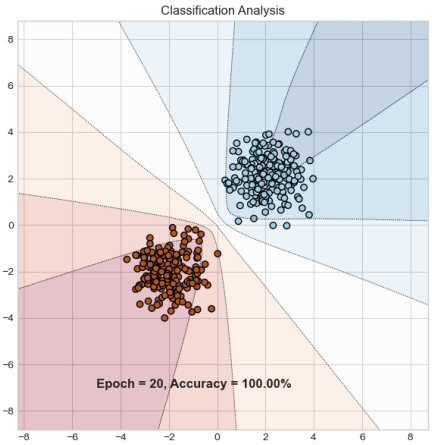


Figure 2.2: Figure representing the result of the Laplace approximation for the Bayesian Logistic Regression

2.1.3 Question 3

The weight decay hyperparameter plays a crucial role in regulating the model's complexity. It acts by imposing a penalty on the loss function for having large weights, thereby incentivizing the model to maintain smaller weight values, which usually help prevent overfitting. This regularization concept aligns with the precision (inverse variance) of the prior distribution over the weights in Bayesian terms.

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.

As observed in Figures 2.3 and the previously mentioned text, varying the weight decay parameter results in distinct predictive behaviors.

When weight decay is set too high, it leads to increased predictive uncertainty, represented

by wider shaded areas in the figures. Conversely, lower weight decay values result in higher confidence, illustrated by narrower shaded areas. Striking the right balance in weight decay is essential for model generalization, as excessively high or low values may not be justified for unseen data.

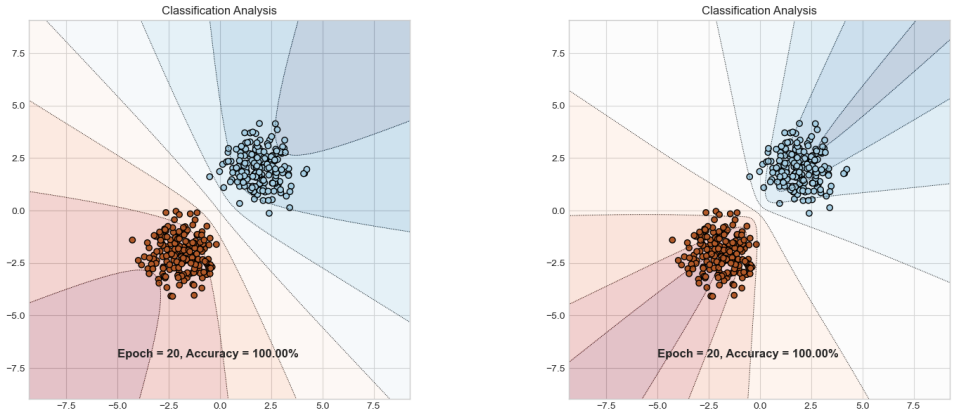


Figure 2.3: Figure representing the effect of WEIGHT_DECAY Regularization Hyperparameter: Left (0.5) vs. Right ($5e^{-10}$) Weight Decay

2.1.4 Variational Inference

The code snippet outlines the structure of two Python classes, `VariationalLogisticRegression` and `LinearVariational`, both designed to incorporate variational inference methods for uncertainty modeling in logistic regression.

The `LinearVariational` class is essentially a variational version of a linear layer, with its core defined by variational parameters for both weights ('w_mu', 'w_rho') and biases ('b_mu'). It also sets up a prior variance for these parameters. A notable feature of this class is its `sampling` method, which uses the reparameterization trick to draw samples from the variational posterior distribution of weights, enabling gradient-based optimization. Additionally, the `kl_divergence` method computes the Kullback-Leibler divergence, quantifying the difference between the variational posterior and the prior distributions of the weights. The forward pass in this class, defined by the `forward` method, applies a linear transformation using the sampled weights and the mean values of the biases.

On the other hand, the `VariationalLogisticRegression` class builds on the `LinearVariational` layer to implement a logistic regression model through variational inference. It initializes with a linear variational layer and employs the sigmoid function in its `forward` method to carry out the logistic regression's forward pass. For computing the loss, the `kl_divergence` method in this class defers to the `LinearVariational` layer to calcu-

late the KL divergence, integrating the variational inference framework into the logistic regression model.

This architecture enables Bayesian uncertainty estimation in logistic regression models by leveraging variational inference to approximate the posterior distributions of model weights, offering a principled approach to quantify uncertainty in predictions.

Question 5

The training loop employs a standard PyTorch framework, aiming to optimize the Evidence Lower Bound (ELBO). The ELBO is a crucial objective 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, computed using binary cross-entropy loss, assesses the model's fit to the data, while the KL divergence serves as a regularization term, penalizing deviations of the variational distribution from the prior and minimizing the information loss in using $q_\theta(w)$ to approximate $p(w)$.
In practice, the training minimizes the negative ELBO, combining the NLL and KL terms into a single loss function. Gradient descent updates the parameters θ of the variational distribution to better approximate the true posterior.
This process involves gradient resetting, loss computation, backpropagation, parameter updating, and periodic assessments of predictive accuracy alongside visualizations of the model's decision boundary.

Figure 2.4 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.

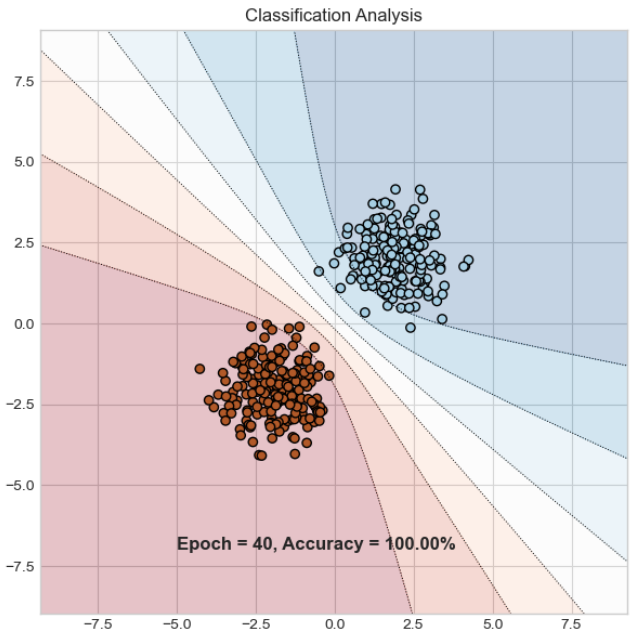
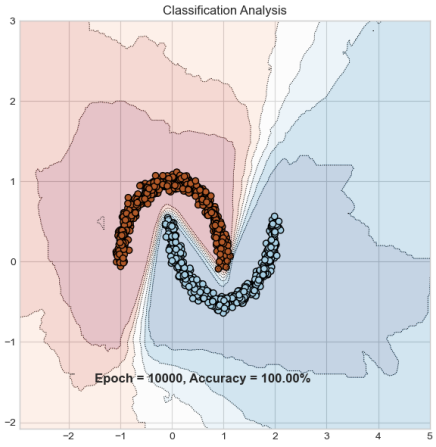


Figure 2.4: Figure representing the result of the Variational approximation with the Logistic Regression

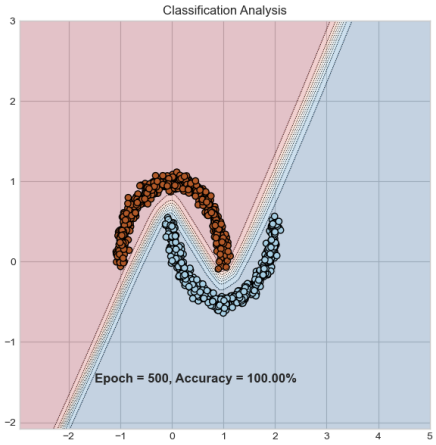
2.2 Bayesian Neural Networks

In this section, the exploration extends to Variational Inference (VI) in non-linear scenarios, alongside the introduction of Monte Carlo Dropout (MC dropout) as a novel approximation method, with outcomes depicted in Figure 2.5.
The first image, illustrating VI, confirms the anticipation that predictive variance escalates with distance from the data points. A noticeable reduction in uncertain regions is observed, attributable to the non-linear decision boundary that constrains the model's options, thereby boosting its certainty.

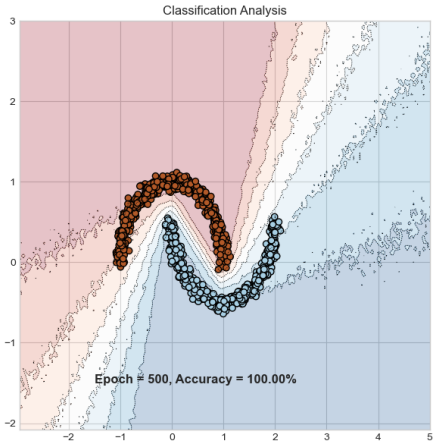
The application of classical dropout tightens uncertainty, yet transitioning to MC dropout yields superior results by significantly amplifying predictive variance away from the training data. The primary advantage of MC dropout over Bayesian Logistic Regression (BLR) lies in its enhanced precision in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



(a) Variational Inference



(b) Dropout



(c) MC Dropout

Figure 2.5: Results of a Bayesian neural network using different approximation methods.

Chapter 3

Applications of uncertainty

In this final lab session, we will focus on applications based on uncertainty estimation.

Initially, we will use MC Dropout variational inference to qualitatively evaluate the most uncertain images according to the mode. Then, we will move on to two examples where good uncertainty estimation is crucial : failure prediction and out-of-distribution detection.

The objective is to gain hands-on experience in applying uncertainty estimation for failure prediction and out-of-distribution detection.

3.1 Monte-Carlo Dropout on MNIST

In this section, we will concentrate on applying Monte-Carlo Dropout variational inference to the MNIST dataset. Our aim is to derive an uncertainty measure using the MC Dropout method, which helps identify the most uncertain images in the dataset. The goal is to explore the uncertainty for random images that are predicted with high confidence and images that are predicted with the lowest confidence.

3.1.1 Question 1

The results are shown in the figures 3.1 and 3.2. We observe a clear distinction between the randomly selected examples and those with the least confidence. The random confident examples are legibly written and easily identifiable, often characterized by a focused probability distribution that indicates the model's confidence in its prediction. In contrast, the least confident images appear ambiguous and are often difficult to discern even by the human eye, with their probability distributions spread across multiple classes.

Additionally, the histograms accompanying the images provide insight into the cases of failure. For the randomly selected confident images, a single spike in the histogram, corresponding to the predicted class, clearly demonstrates the model's certainty. However, this pattern changes significantly for the most uncertain images. Here, the histograms exhibit multiple spikes, signaling a high level of uncertainty and a more dispersed probability distribution. Similar to how a human would react when presented with these ambiguous images, the model exhibits confusion between different class labels, as reflected in the multiple spikes in the histograms.

For instance, in the second image among the least certain ones, the model struggles to distinguish between '6' and '8'.

We can also observe that the var-ratio gets higher when faced with the most uncertain samples, further indicating the spread and ambiguity in their probability distributions.

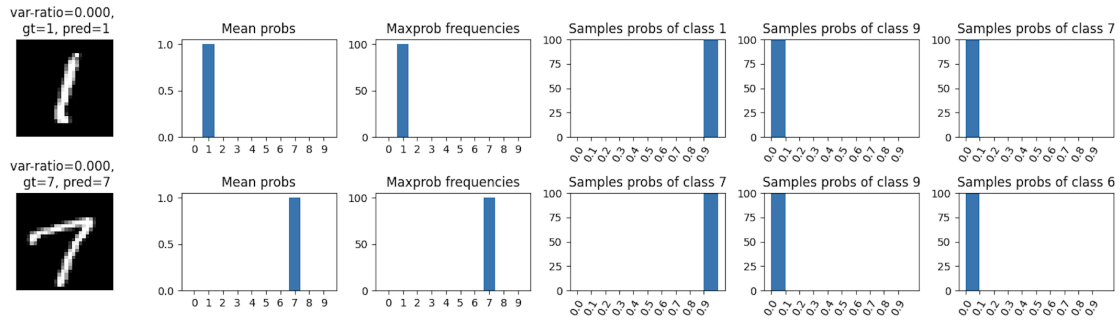


Figure 3.1: Figure representing the result of the confident samples of MC sampling

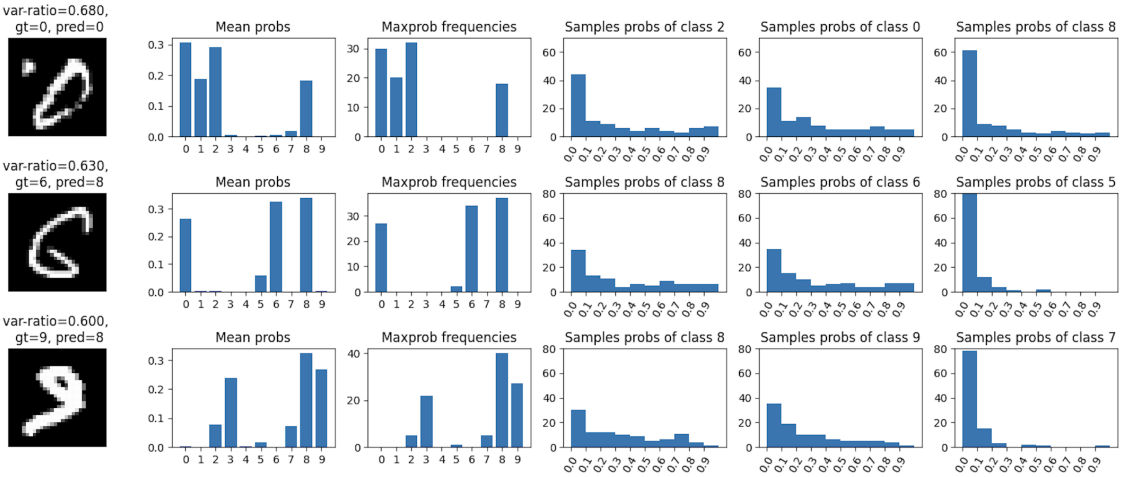


Figure 3.2: Figure representing the result of the most uncertain samples of MC sampling

3.2 Failure prediction

The goal of this section is to develop reliable confidence measures for model predictions, capable of distinguishing between correct and incorrect predictions. With an effective confidence measure, a system can determine whether to rely on its own prediction, escalate to human intervention or a backup system, or activate an alarm. We will explore ConfidNet, a method tailored for failure prediction, and compare its performance with MCDropout using entropy and Maximum Class Probability (MCP).

We evaluate ConfidNet's ability to detect failures against previous baselines, including Maximum Class Probability (MCP) and MCDropout with entropy.

In our experiments, classification errors are treated as the positive class for detection purposes.

3.2.1 Question 1

Figure 3.3 displays the performance metrics, including the precision-recall curve and AUPR (Area Under the Precision-Recall Curve), for ConfidNet in comparison to baseline methods MCP and MC-Dropout. The figure essentially benchmarks the quality of the confidence criteria by ordering test examples based on their associated uncertainty levels. From the analysis, ConfidNet emerges as the frontrunner, boasting the most favorable precision-recall curve and the highest AUPR at 50.71%. MCP comes next with an AUPR of 36.26%, while MC-Dropout trails at 31.59%.

However, it's important to note that despite their relative rankings, none of the models demonstrate particularly strong performance. Ideally, both precision and recall should be high, yet in these cases, precision tends to approach zero as recall reaches its maximum. This indicates a significant limitation in the models' ability to confidently and accurately identify relevant instances, particularly in scenarios where maximizing recall is critical.

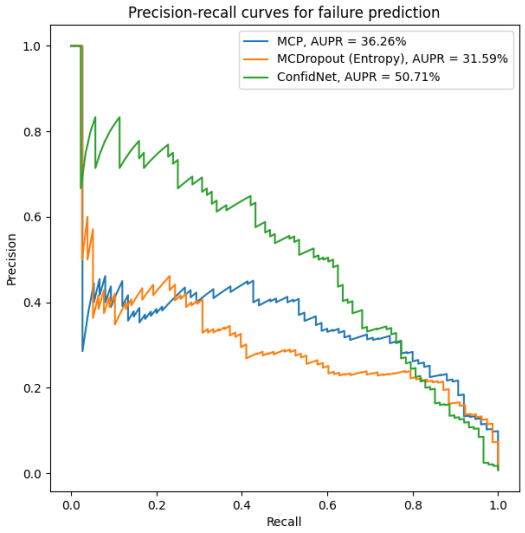


Figure 3.3: Figure representing the precision-recall curves of each method along with their AUPR

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.

3.3 Out-of-distribution detection

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

Figure 3.4 presents the precision-recall curves for various OOD detection methods, along with their corresponding AUPR (Area Under the Precision-Recall Curve) values. The precision-recall curves indicate commendable performance across all three evaluated models. ODIN leads with an AUPR of 98.89%, followed closely by MCDropout at 98.45%, and MCP at 97.68%. This ranking aligns with expectations, given ODIN's specialization in OOD detection. Its superior performance over MCDropout and MCP is attributed to its ability to more accurately identify OOD samples. ODIN's advantage stems from employing temperature scaling and leveraging adversarial attacks to ascertain model confidence, offering a more comprehensive measure of uncertainty compared to the techniques utilized by MCDropout and MCP.

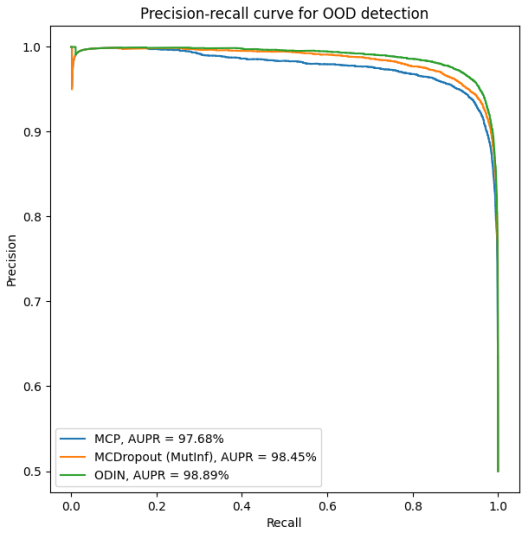


Figure 3.4: Figure representing the precision-recall curves of each OOD method along with their AUPR