Student Name: Essakine Amer
Student ID: amer.essakine@ens-paris-saclay.fr

Foundations of deep learning
TP Answer report

TP1 : Bayesian Models and Uncertainty

Question 1.4 : Linear Bayesian regression

For linear Bayesian regression, the predictive distribution follows a Gaussian distribution :

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

Where

$$\Sigma^{-1} = \alpha I + \beta\Phi^T\Phi \text{ and } \mu = \beta\Sigma\Phi^T Y$$

The prediction gives the following results : The uncertainty is measured by the standard deviation

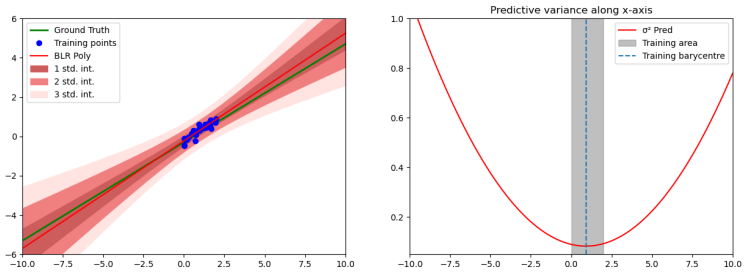


Figure 1: *On the left* The model prediction with uncertainty. *On the right* The predictive variance along x-axis

of the posterior distribution. On the left figure, we can see that the shaded figure (uncertainty) increases significantly as we move away from the training points. Similarly, the predictive variance increases symmetrically as we move away from the training data.
Let's consider the case when we train on data with point cloud data with a whole in the middle : We see that the training points span a slightly larger range, reducing predictive uncertainty in the

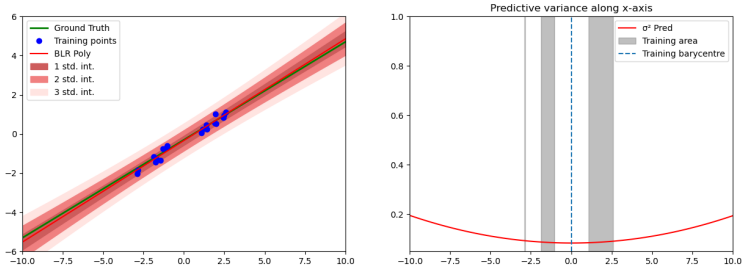


Figure 2: Prediction result on point cloud data with a whole in the middle

middle region compared to the previous plot.

Question 1.5 : Theoretical analysis to explain the form of the distribution

We assume that $\alpha = 0$ which means we have no prior on the distribuition, we also assume that $\beta = 1$. The posterior distribution is a Gaussian distribution : For linear Bayesian regression, the predictive distribution follows a Gaussian distribution :

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

Where

$$\Sigma^{-1} = \Phi^T\Phi \text{ and } \mu = \Sigma\Phi^T Y$$

We can find an explicit formula of :

$$\Phi^T\Phi = \begin{pmatrix} n & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 \end{pmatrix}$$

Hence

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

The denominator $n\sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2$ is proportional to the variance of the data points x_i. Therefore, if the data points are spread widely (have a large variance), the value of Σ decreases, leading to reduced uncertainty. This reflects greater confidence in the parameter estimates, as the data covers a larger portion of the space. This result is consistent with our findings in Exercise 1.4, where the second example had a larger variance, leading to a smaller posterior variance for the parameters.

Question 2.4/2.5 : Non-linear regression

We can remark that within the model the training points are followed reasonably well but strug-

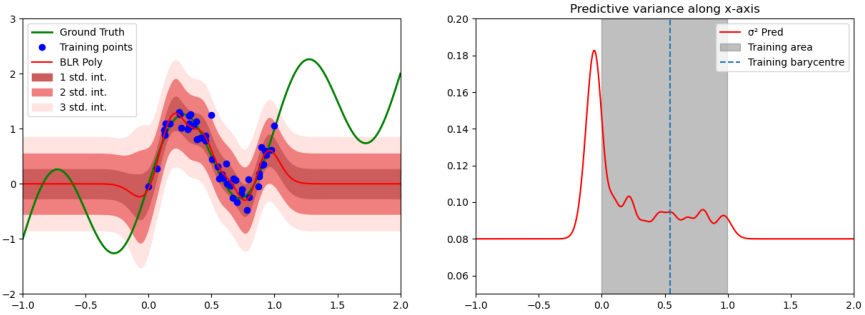


Figure 3: Prediction result on sinus data using radial basis

gle to generalize outside the training area. we conclude that the Bayesian regression shows low uncertainty in regions where training data is dense and well spread. However, it cannot generalize well beyond the training range. Similarly, The predictive variance increases significantly near the boundaries of the training area before converging to a fixed value far outside the training range.

The variance converges outside the training range to a fixed value due to the localized nature of the Gaussian basis used, and this result is general for any localized function. This is because

$$lim_{x\infty}\phi(x) = 0$$

Which imply that :

$$\phi(x^*)^T\Sigma\phi(x^*) \to 0$$

When x^* goes to infinity. And since the predictive variance is given by :

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

Hence the predictive variance takes the value of σ^2 far outside the training area.

TP2 : Bayesian Neural Networks

Question 1.2 : Laplace approximation results

The Laplace method approximates the posterior with the normal distribution with mean w_{map} and variance given by the Hessian of the posterior at w_{map}. Its idea is simple : improves the logistic regression (where we fully trust the weights found w_{map}) by adding a bit of uncertainity represented by the Hessian

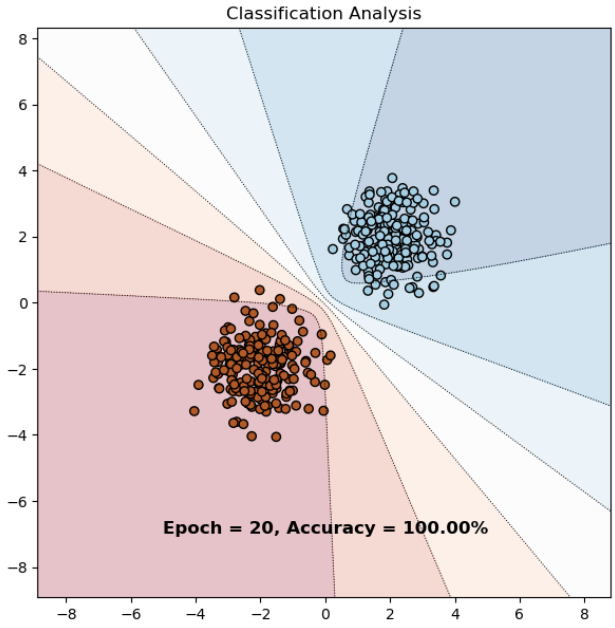


Figure 4: Laplace approximation

We can see a significant improvement on logistic regression, as the figure shows clear and sharp decision boundaries, indicating high confidence in predictions. Compared to logistic regression, this method results in more confident and precise separation.

Question 1.3 : Variational inference

The LinearVariational class approximates the posterior distribution in a Bayesian framework. It is structured similarly to a linear layer in neural networks but differs in that the weights are sampled during each forward pass using the reparameterization trick. Additionally, compute the KL divergence, which will be used in the loss function.

What is the main difference between Laplaces and VIs approximations?

- The Laplace approximation the posterior locally as a Gaussian centered at the MAP estimate and use the Hessian to locally approximate the variance and account for uncertanity.
- In contrast, VI optimizes a parametric posterior approximation globally, allowing it to capture more complex distributions due to the global properties.

Also

- Laplace method require to compute the Hessian matrix which could be expensive in high dimensions.
- VI relies on stochastic gradient descent which is scalable to large datasets and high-dimensions.

Question 2.1 : MC dropout

As we can see from the figure, both approaches yield similar results, though MC Dropout is

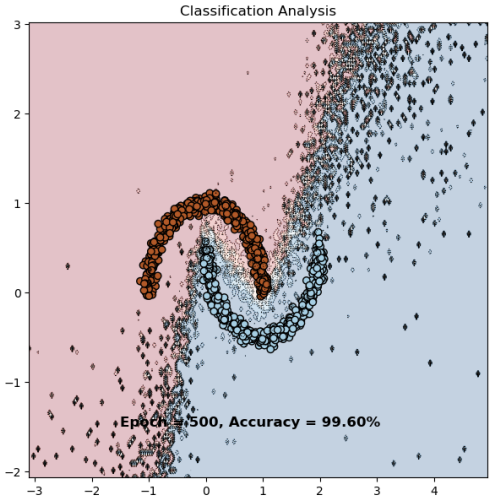


Figure 5: Classification using MLP with dropout

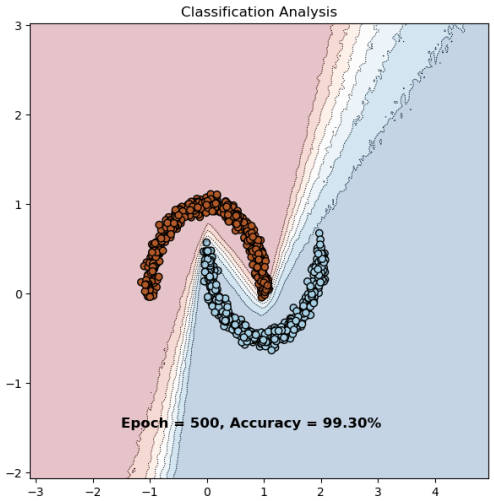


Figure 6: Classification using VI

slightly noisier. MC Dropout simply adds a dropout layer to an MLP, achieving results comparable to VI while being simpler to implement and train.

1 TP3 : Applications of Deep Learning Robustness

Comment results for investigating most uncertain vs confident samples

Let's plot the most uncertain and confident samples : The model is confient on clean, well drawn images but struggles with noisy and ambiguous inputs,reflected by higher uncertainty.

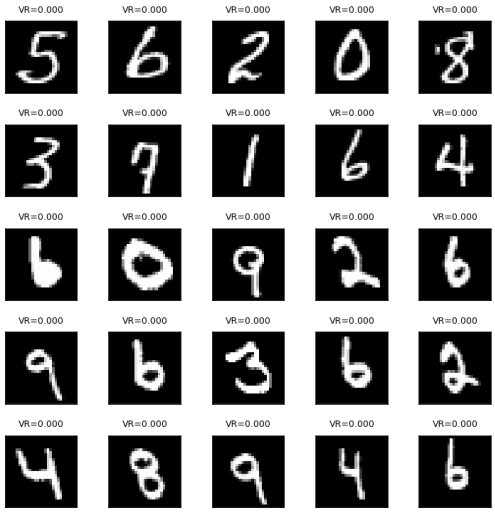


Figure 7: Samples the model is most confident on

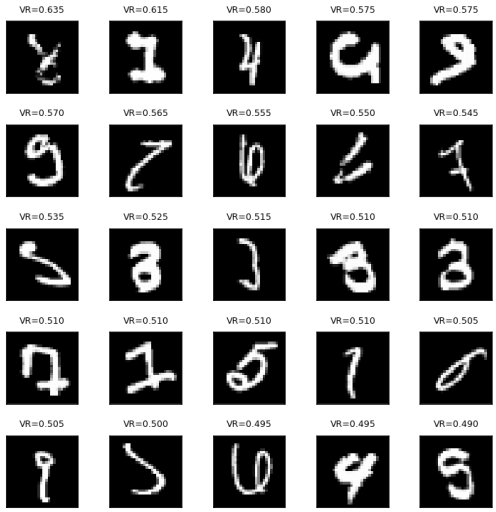
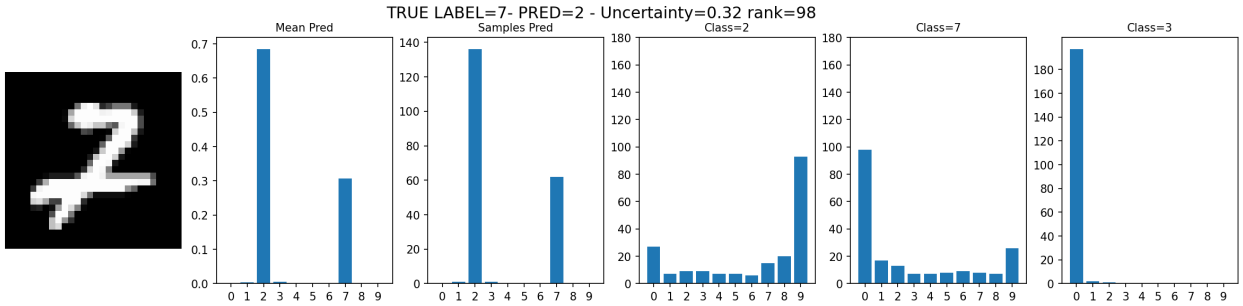


Figure 8: Samples the model is most confused about

Let's investigate more why the model is confused on some examples :



This is an example where the model misclassified the true class. The image exhibits relatively low uncertainty, as the model was fairly confident in predicting it as a "2" rather than a "7" (indicated by a higher occurrence of predictions for class "2"). However, a human can clearly identify the digit as a "7," highlighting the model's failure to recognize the subtle features distinguishing the two classes.

Question 2.1/2.2 : Failure precition:

The goal of failure prediction is to identify moments where the model is likely to produce an incorrect predictions. It is very essential to the user of the model especially in medical applications where an error can have a grave mistake with no one to take accountability (as the result was generated by a model). Estimating the model's uncertainty helps identify cases where the model may be wrong and where expert guidance is needed.

Important objectives of the failure predicition is :

- **Improving Reliability** : detecting failure instances ensures that they can be handled (by an expert guidance or a call-back)
- **Identifying Limitations** : failure prediction helps identify weaknesses in the model such as insufficient training data or flawed data samples
- **Increase trust** : If the model can identify where it fails, users can trust the model's predictions, allowing the use of models in medical applications for example

The code of LeNetConfidNet is very similar to the code LeNet. It includes a CNN of two convolutional layers each followed by a max pooling and then a 2 linear layers, this is the classifier block of LeNetConfideNet. Then there is a 4 layer MLP used to estimate the model's uncertainity.

Analyze results between MCP, MCDropout and ConfidNet [II.2]

The MCP method is the most effective for failure prediction, achieving the highest AUPR and

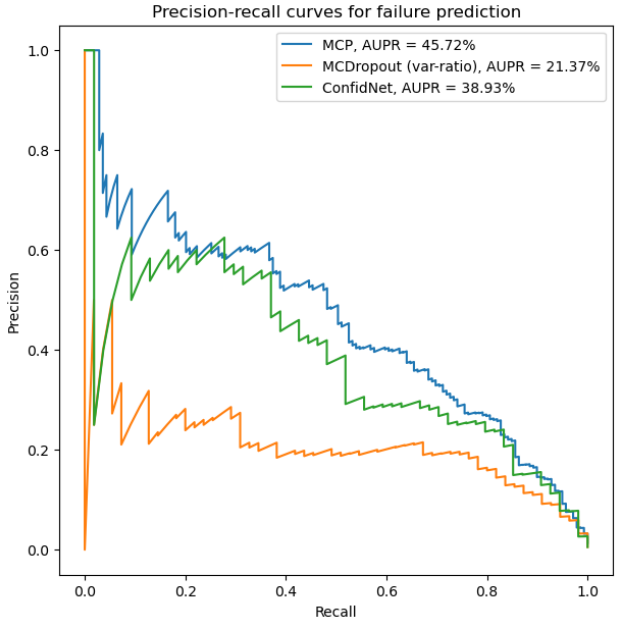


Figure 9: Cmparaison results between MCP, MCDropout and ConfidNet

maintaining strong precision across recall values. ConfidNet performs similarly, while MC Dropout underperforms, likely due to noisy uncertainty estimates.

Analyze results and explain the difference between the 3 methods [III.1]]

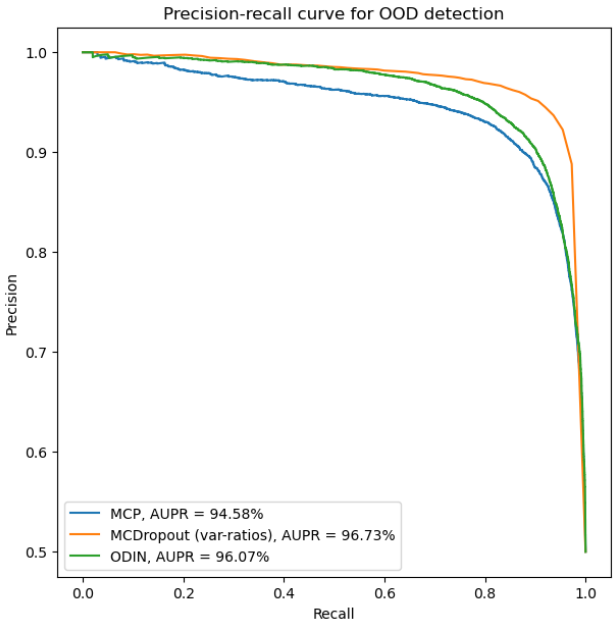


Figure 10: Comparaison of the three methods for OOD detection

From the figure, we can see that ODIN performs better than MCP but still performs slightly worse than MCDropout (should it be the opposite ?)