Mathias VIGOUROUX
mathias.vigouroux@ens.psl.eu

1 Week 1: Bayesian Linear Regression

The first started with a gentlte introduction to Robustness in Deep Learning through Bayesian Linear Regression.

1.1 Bayesian linear regression: Results

This class of models yields the following results for the predictive distribution on the synthetic dataset [Question 1.4]

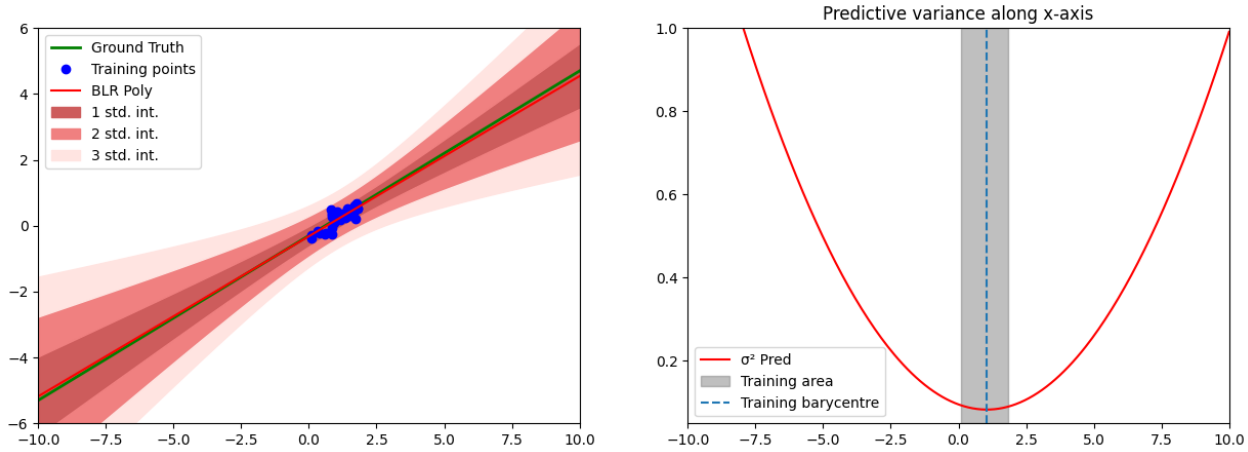

Figure 1: Bayesian Linear Regression

As expected, the model looks similar to a linear regression (first panel of 1). However, the model is able to fit a smaller standard deviation, i.e. a larger confidence, on its predictions when it is near from the train data set. Moreover, the variance is increasing polynomially along the x-axis (second panel of 1), i.e. when moving away from the data.
Now, let's do a theoretical analysis to explain the form of the distribution (simplified case $\alpha = 0$, $\beta = 1$) [Question 1.5]

1.2 Bayesian linear regression: Theory

1.2.1 Predictive Distribution

The expression for the predictive distribution of (y) given an input (x^*), dataset (D), and parameters (α, β) is as follows:
$[p(y|x^*, D, \alpha, \beta) = \mathcal{N}\left(y; \mu^T \Phi(x^*), \beta^{-1} + \Phi(x^*)^T \Sigma \Phi(x^*)\right)]$
In the specific case where $(\alpha = 0)$ and $(\beta = 1)$, the expression simplifies to:

$$[p(y|x^*, D, \alpha = 0, \beta = 1) = \mathcal{N}\left(y; \left((\Phi^T\Phi)^{-1}\Phi^T Y\right)^T \Phi(x^*), \Phi(x^*)^T(\Phi^T\Phi)^{-1}\Phi(x^*)\right)]$$

1.2.2 Predictive Variance

As it was said from the observations, the variance along (x) is characterized as a quadratic scalar function of (x^*):

$$[\sigma^2(x^*) = \Phi(x^*)^T(\Phi^T\Phi)^{-1}\Phi(x^*)]$$

This metric provides insight into the uncertainty or variability in the prediction at the input (x^*).

1.3 Non-linear regression

However, the data might not be well explained though a linear regression. It is particularly the case of periodic function. For instance, there could be sinusoidale perturbations (that might be of interest) on the data (for instance global temperature which both increase and is periodical).

1.3.1 Polynomial Basis function

A first approach to tackle this issue is to use polynomial basis functions (instead of just regressing of $(1, X)$ the data will be regressed against $(1, \Phi_1(X), ..., \Phi_n(X)$, where the (Φ_i) is a polynomial basis. The results are depicted in 2

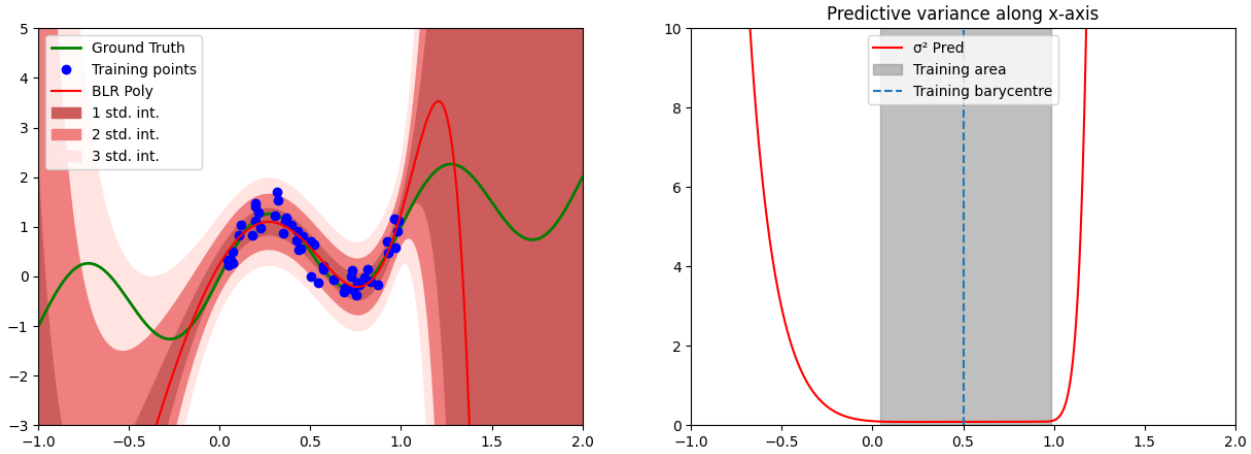

Figure 2: Bayesian Regression with Polynomial basis functions

However, the problem with such a modelling is that polynoms always diverge. This causes an explosion of the prediction (both in mean and variance).

1.3.2 Gaussian Basis function

To tackle this , using a Gaussian family seems a reasonable idea (since Gaussian are similar to polynoms in the center and tends to 0 in the extremes). As it is depicted in 3 , compared to a polynomial basis function, there isn't any explosion of the predictions.However, the predictions with Gaussian functions are less smooth.

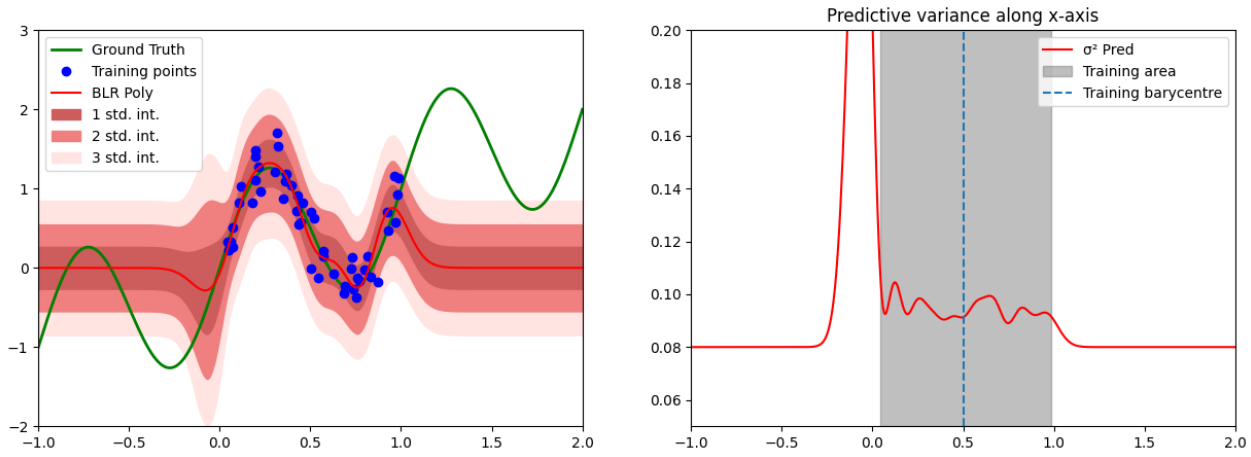

Figure 3: Gaussian Basis function

Theory [Question 2.5] The variance of the data is set to be 0.08 which seems to be the asymptotically value in the right panel of 3. This can be proven theoretically.
The predictive variance is explicitly represented as $(\frac{1}{\beta} + \Phi(x^*)^T \Sigma \Phi(x^*))$. Consequently, as data points move away from the distribution, particularly those distant from the Gaussian centers defined by the kernel, the individual components of $(\Phi(x^*))$ quickly approach 0. Compared to previously, when Φ was polynomial and was not tending to zero, Φ is Gausian, thus it tends to zero at both extremes.

This behavior causes the predictive variance to tend towards (β^{-1}). This observation is consistent with the numerical values, where $(\beta = \frac{1}{2\sigma^2})$ and $(\sigma = 0.2)$. This yiels $\beta = 1/0.08$, resulting in $var = 0.08$, as observed.
analysis of the Gaussian basis feature mapsGaussian basis feature maps results [Question 2.4/2.5]

2 Week 2: Approximate Inference in Classification

In the realm of classification tasks, even a basic Logistic Regression doesn't grant us access to a closed form of the posterior $p(\boldsymbol{w}|\mathcal{D})$. Unlike the scenario in Linear Regression, the likelihood here doesn't align with a Gaussian prior, rendering approximation necessary.
Throughout this session, we will navigate and juxtapose different approaches to approximate inference using 2D binary classification datasets. Explored methodologies encompass Laplacian approximation, variational inference employing mean-field approximation, and the utilization of Monte Carlo dropout.

2.1 Bayesian Logistic Regression

To begin with the classification of our 2D classification task, let's start with some logistic regression done using different type of Perceptron.

2.1.1 Maximum-A-Posteriori Estimate

The first type of perceptron used is the Maxium-A-Posteriori Estimate which convey a BAyesian approach to the weight of the perceptron.
Given by $\boldsymbol{w}_{\text{MAP}} = arg\max_{\boldsymbol{w}} \prod_{n=1}^{N} p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) p(\boldsymbol{w}$ this yields, in the case of a Gaussian prior,

$$\boldsymbol{w}_{\text{MAP}} = arg\min_{\boldsymbol{w}} \sum_{n=1}^{N} \big(-y_n \log \sigma(\boldsymbol{w}^T \boldsymbol{x}_n + b) - (1 - y_n) \log(1 - \sigma(\boldsymbol{w}^T \boldsymbol{x}_n + b)) + \frac{1}{2\Sigma_0^2} ||\boldsymbol{w}||_2^2 \big)$$

By using a perceptron with such weights the results of the classification are depicted in 4. In the intermediary region between the two data clouds, the confidence over the prediction is low. However, this level of low confidence are organized lineary since the model is a linear separator . Thus, even thought the data points are organized in a circle, the model is not able to depicted the fact that there are regions behind the data cloud in which it hasn't seen any data and where it should have a low confidence.

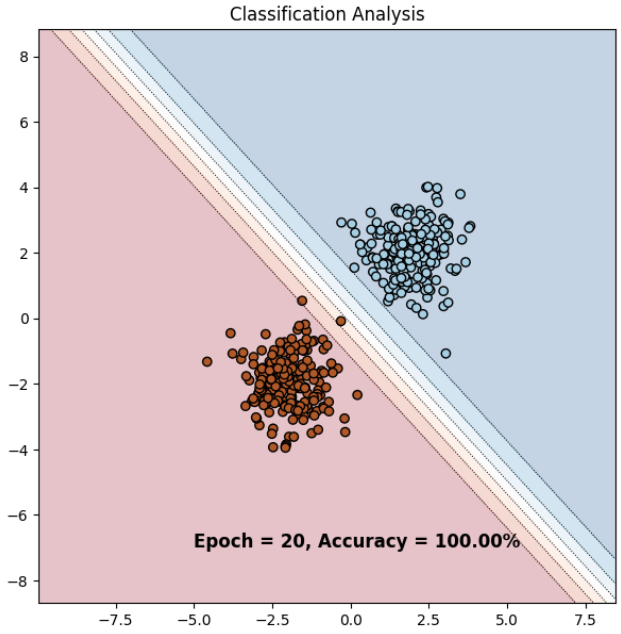

Figure 4: Maximum a Posteriori Estimate

However, just as in the case of a Bayesian Linear Regression, this linear organisation is not sufficient.

2.1.2 Laplace Approximation

One possibility is to use Laplace approximation to estimate the intractable posterior $p(\boldsymbol{w}|\mathcal{D})$. More specifically, $p(\boldsymbol{w}|\mathcal{D})$ is approximated with a normal distribution $\mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}_{lap}, \Sigma_{lap}^2)$. These results are presented in 5.

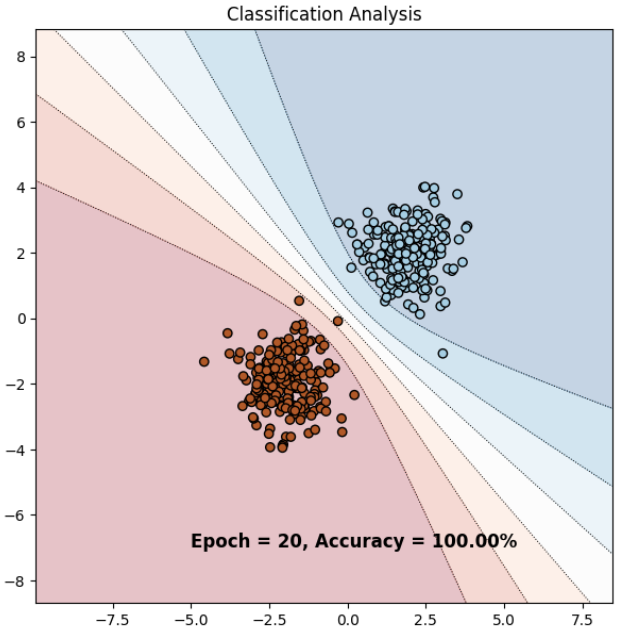

Figure 5: Laplace Approximation for Bayesian Logistic Regression

Once again, in the region between the two data cloud, the confidence of the classifier is low. Compared to the MAP estimator, the confidence levels are not just ordered linearly trhought space but are following a more polynomial shape. This allows the models to capture the circular distribution of the data points.

2.1.3 Variational Inference

A Linear Variational Layer is a layer that infers the mean and variance of its weights and bias. To do so, it computes model parameters by generating Gaussian vectors for the weights and bias. Yet, the prior values of parameters are known, and the layer generates samples of posterior parameters. Then, it computes regularization in the loss, ensuring that the posterior distribution aligns with the prior. More precisely, the regularization is a technique to prevent overfitting and ensure that learned parameters stay close to their prior values. In variational inference, regularization terms are added to the loss function to enforce a similarity between the posterior and prior distributions of model parameters. This helps in obtaining more robust and generalizable models. These models are depicted in 6

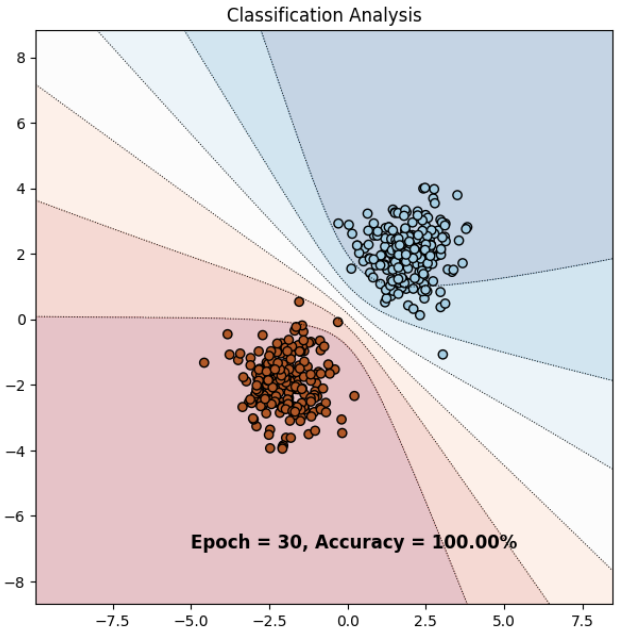

Figure 6: Variational Inference for Logistic Regression

Comparison with the Laplace Method
Graphically, the output do not differ very much. However, There are theoretical differences. Indeed, the Linear Variational Layer incorporates probabilistic modeling by inferring parameters with Gaussian variational distributions, and the regularization term ensures that the learned posterior distributions align with prior expectations. The distinction between Laplace's Method and Variational Inference lies in how they handle the posterior distribution, with Laplace's Method using a fixed mean and VI generating parameters randomly during training based on the learned posterior distribution.
More precisely, on one hand, the Laplace's Method,uses optimal (learned) deterministic parameters obtained after training as the mean. Thus, it assumes a Gaussian posterior distribution with a fixed mean. In a nutshell, this method provides a deterministic approximation around the mode of the posterior distribution.
On the other hand, the Variational Inference (VI), Generates model parameters randomly during training based on the learned posterior distribution. Then, the posterior distribution is not assumed to be fixed; instead, it is learned during the training process. Thus, VI provides a more flexible approach, allowing for exploration of the entire posterior distribution.

2.2 Bayesian Neural Networks

Let's now move to non linearly separable dataset (such as Half Moon data).
The natural way to create a richer classes of function (more specifically to augment the VC-dimension of our classifier) is it to use rather than a perceptron a multi-layer perceptron (MLP).

2.2.1 Multi-Layer Variational Inference

By stacking Variational Inference Layers defined previously, it is possible to have indeed VI for non linearly spearable probelms just as it is depicted in 7

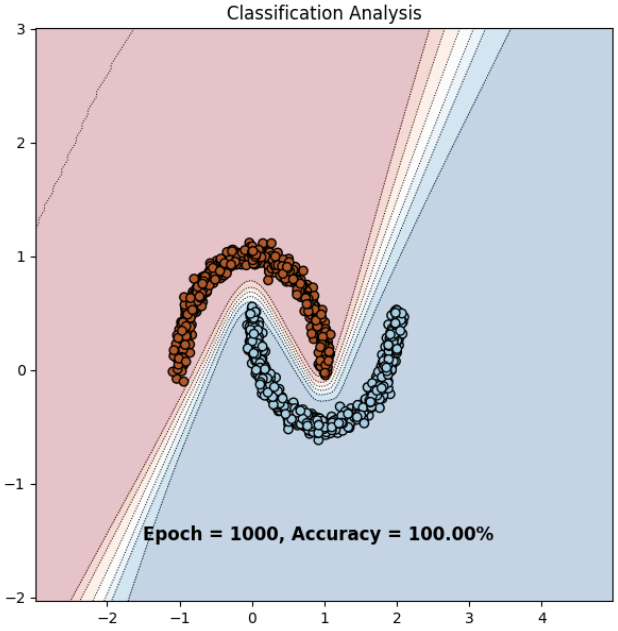

Figure 7: Multi-Layer Variational Inference

2.2.2 Monte-Carlo Dropout

One other possibility to have confidence is to introduce noise in the data trhough MC dorpout. The aim of this technique s to randomy remove some weights. This yiels the following results 8.

2.2.3 Comparison of VI and MC Dropout

Lets recall the two technics : **MC Dropout (Monte Carlo Dropout):**

- **Easy to Compute:** MC Dropout provides a computationally straightforward and intuitive method for estimating model uncertainty. This is because dropout is a regularization technique that involves randomly dropping out (setting to zero) some neurons during training. During inference, instead of

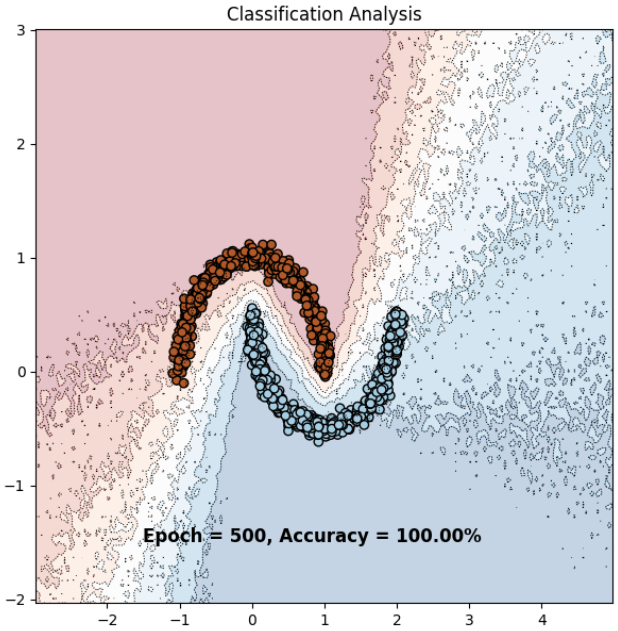

Figure 8: Multi-Layer MC Dropout

using a single forward pass, multiple stochastic forward passes are performed with dropout enabled. This process effectively turns the model into an ensemble of different sub-models.

- **Higher and Less Smooth Uncertainty:** The uncertainty estimates provided by MC Dropout tend to be higher and less smooth, especially in regions far from the training data. This means that the model expresses a higher degree of uncertainty about its predictions in these regions, and the uncertainty varies more abruptly.
- **Intuitive and Desired Uncertainty:** The statement suggests that the uncertainty characteristics produced by MC Dropout align more closely with what is desired. This could mean that the higher and less smooth uncertainty aligns better with the true uncertainty or variability in the data, providing a more realistic representation of the model's lack of confidence in certain predictions.

Variational Inference (VI):

- **Complex Inference Process:**VI, on the other hand, involves learning a distribution over the model parameters during training. This allows for a more complex representation of uncertainty, but it may involve more computation during both training and inference.
- **Smoothing Effect:** The uncertainty estimates provided by VI tend to be smoother, meaning that they exhibit less variability or abrupt changes. This could be advantageous in some situations, but it might not capture certain types of uncertainties as effectively.

In summary, the comparison suggests that MC Dropout offers a simpler and more intuitive approach for estimating model uncertainty. The uncertainty it provides is characterized by being higher, less smooth, and potentially more in line with what is desired, especially in regions far from the training data. This simplicity and alignment with expectations make MC Dropout an attractive choice for certain applications where interpretability and realism of uncertainty are important.

3 Week 3: Uncertainty Applications

In this final week, attention will be directed towards applications centered on uncertainty estimation. The initial focus will involve utilizing MC Dropout variational inference to qualitatively evaluate images with the highest uncertainty in classification. Following that, the application of this uncertainty will be explored in the context of predicting failures, an important use case.

3.1 MC Dropout for classification on MNIST

To begin with the first type of application is on the MNIST, the dataset of reference for computer vision.

Following the previous part which showed that MC-Dropout seems the best type of algorithm for classification with uncertainty, a classical Convolution Network will be trained on MNIST (of type LeNet [2]) with MC Dropout.
Since the class is multivariate the previous definition of uncertainty would not work. Therefore, in this part, uncertainty will be defined as the variation ratio :

$$variation - ratio[\mathbf{x}] = 1 - \frac{f_x^{c^*}}{T}$$

where $f_x^{c^*}$ is the number of occurences in histogram corresponding to the majority class, (*i.e.* the mode) c^*.
Following this method, predictions of the network can be ranked as more or less confident.
Therefore 9 shows numbers on which LeNet has high confidence, and indeed all numbers look very ”nice”. On the other hand, **??** depicted images on which the network had low confidence. Visually, it is possible to see that this time the number are not easily readable, even by a human.

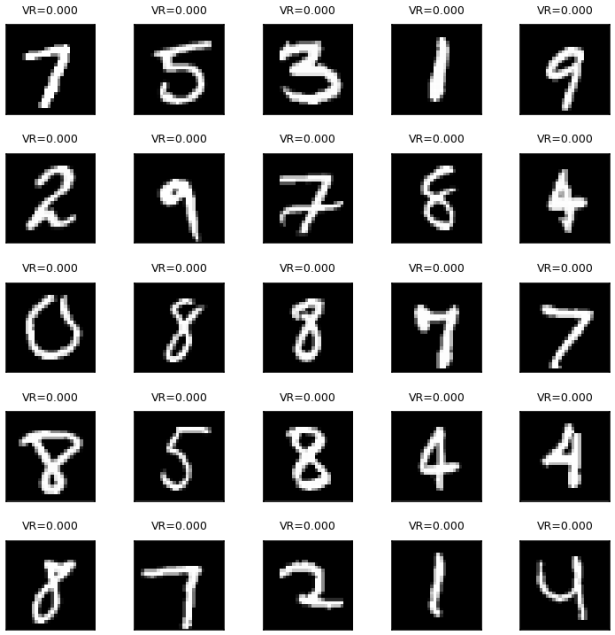

Figure 9: Certain Classification on the MNIST dataset

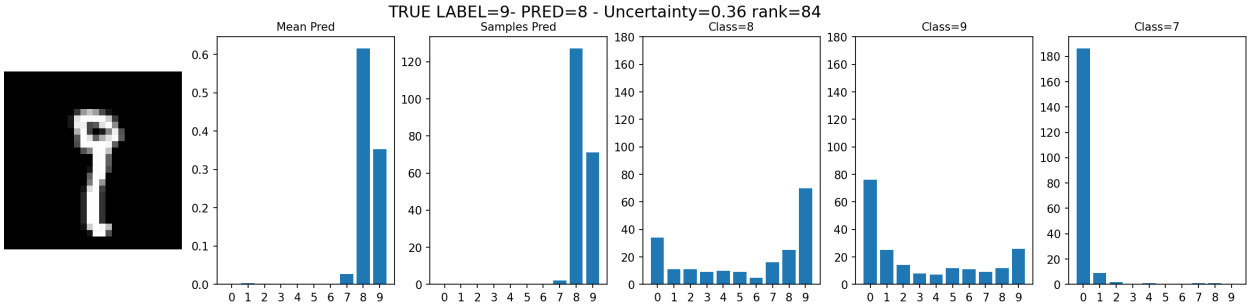

Figure 10: Uncertain Classification on the MNIST dataset: focus on one prediction

Finally, for a given number which was hard to predict, it possible to see which where the other labels that the network was predicting as it is shown in 10.

3.2 Failure Prediction

3.2.1 LeNetConfiNet

The previous model was not able to detect the wrong classification, only the one with low or high prediction. Event though, confidence and performance are likely to be closely related, LeNetConfiNet is a network that explicitly tries to do failure prediction.

To do so , it incorporates a Classifier block for image classification and a Confidence block specifically trained to estimate the true class probability. The classifier block is responsible for classifying images. Thus, during training, it is trained by maximizing the cross-entropy loss. This involves optimizing the alignment between the predicted classes vector and the real classes vector.
In parallel, the Confidence block is specifically dedicated to estimating the true class probability (TCP) of the input image.Thus, during training, this block is trained by maximizing the mean squared error. The optimization process involves minimizing the difference between the predicted true class probability and the real true class probability.
Finally, The model employs the TCP value obtained from the Confidence block to measure the uncertainty associated with the model's predictions. This uncertainty serves as a confidence measure, allowing for the distinction between correct and incorrect predictions.

4 Comparing MCP, MCDropout and ConfidNet

To compare these different models the precision recall curve will be used. As a reminder :

- Precision = TP / (TP + FP).
- Recall = TP / (TP + FN).

The aim of the precision-recall curve is to shows how precision and recall change with varying thresholds. A higher area under the curve (AUC-PR) indicates better overall performance, especially in situations where the positive class is rare. Thus, the precision-recall curve is particularly useful tool to compare models on imbalanced datasets (and we can hopefully assert that failures are more rare than functioning predictions).
The results can be be seen in 11.

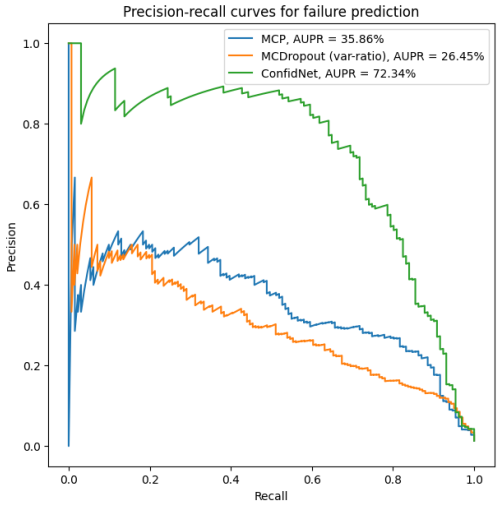

Figure 11: Comparing different confidence models

It is possible to see that ConfidNet performs better than the two others models ! However, all models have a non nul AUC-PR above chance thus they are all able to detect failure.

4.1 Out-of-distribution detection

To finish, the last application was Out-of-distribution Image Detection in Neural Networks (ODIN) [3].
This framework is made out of 4 key methods.

- **Temperature Scaling:** ODIN introduces a temperature parameter to the softmax output of the neural network during inference. The softmax function converts the model's raw output scores into probabilities. The temperature scaling process involves dividing the logits (the raw output scores before softmax) by a temperature parameter. This temperature scaling makes the probabilities more confident or peaky.
- **Input Perturbation:** Before applying temperature scaling, ODIN perturbs the input data slightly. This perturbation is done to increase the difference between in-distribution and out-of-distribution samples. By adding a small perturbation to the input data, ODIN aims to make the model's predictions more sensitive to variations in the input, making it more likely to produce confident predictions for in-distribution samples.

- **Decision on Out-of-Distribution:** After temperature scaling and input perturbation, ODIN compares the confidence scores (probability of the predicted class) of the original and perturbed samples. If the confidence score decreases after perturbation, the sample is considered more likely to be out-of-distribution.
- **Thresholding:** A threshold is applied to the confidence score difference, and samples exceeding this threshold are classified as out-of-distribution.

This models can be trained on a Kanji dataset (japanese character) that were injected in the MNIST dataset 12. In my case, the three models perform similarly, in particular MCP and ODIN. However, it is MC-Dropout which in my run, perform slightly better. Indeed, the differences are not as large as in the case of failure detection or confidence in general. This highlight the difficulty of OOD.

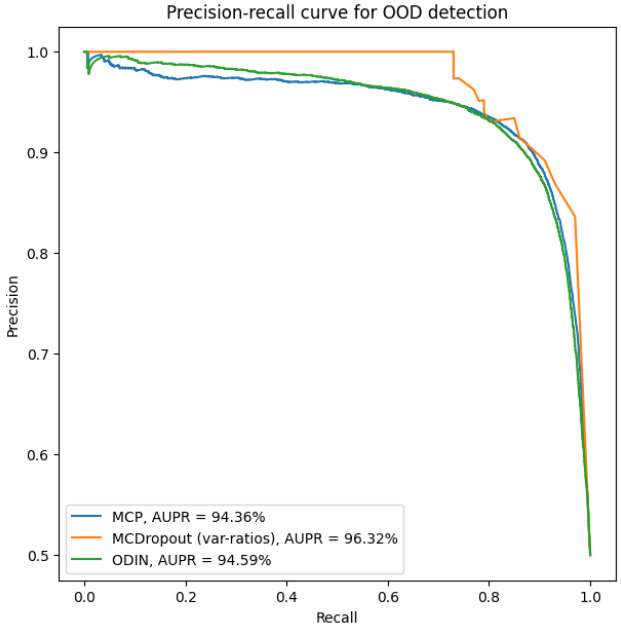

Figure 12: Comparing different out of distribution models

5 Conclusion

In my opinion, OOD is the most challenging problem that robustness has yet to overcome.
Indeed, by definition out of distribution data points are rare events, defining what is rare or far from the distrbituion is thus a very question and for the moment it is manly done via finding "à la main" a good threshold or by having hypotheses on the shape of the data should have. Focusing only on the data seems to be not enough. A recent discussion that I had with Frédéric Chazal gave me a new opening on the subject. One should dive into the activation of the netork and maybe look at topological changes in it.
In particular his team worked on Covariate Data Set Shift Detection via Activation Graphs of Deep Neural Networks [1].

References

[1] Felix Hensel, Charles Arnal, Mathieu Carrière, Théo Lacombe, Hiroaki Kurihara, Yuichi Ike, and Frédéric Chazal. Magdiff: Covariate data set shift detection via activation graphs of deep neural networks, 2023.

[2] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[3] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks, 2020.