

### 3.1 Introduction

In this chapter, we embark on some comprehensive examinations of the performances and performances of Bayesian Deep Learning model in comparisons with traditional approach in the contexts of limited data prediction tasks. Our investigations cover empirical evaluations, comparative analyses, and insight obtained from stringent experimentation. The primary objectives of these chapters are to scrutinize the capability and limitation of Bayesian Deep Learning model for data scarcity and heterogeneity. We discuss in-depth different methodology and technique specifically suited for such scenarios, aimed at elucidating the practical implication and feasibility of implementing Applications in the real world of Bayesian frameworks.

### 3.2 Overview of the Experimental Setup and Objectives

We utilize three distinct datasets for our comparative analysis and experimental evaluation.

These goals are to test the effectiveness of Bayesian method on the data itself. They are some attempts to compare Bayesian and traditional method and not to compare the effectiveness of Bayesian method in all cases in the various types of data used. The experiments were deliberately conducted on small data with noise because the goal is to train and compare on limited data concepts and to test robustness. Performance evaluations based on metrics including Time taken, Training Accuracy, Validation Accuracy, Test Accuracy and with only 10 epochs.

The first datasets consist of **Brain MRIS Images** for Brain Tumors Detection, sourced from NAVONEELS CHAKRABARTY and last updated five years ago. These images totaling approximately 16MB in size, are classified into two classes: "yes," indicating the presence of some brain tumor, and "no," indicating

their absence. Despite this limited additional information available. Our goals with these datasets are to explore and compare the effectiveness of traditional Convolutional Neural Networks (CNN) algorithm and model against Bayesian method in the context of brain tumors detection.

The second datasets consist of 7023 MRIS image showcasing various brain tumor types. We exclusively used these datasets for 'no tumor' class images. It's some crucial components for our study's comprehensive analysis. Despite concern about glioma class categorization, we ensured data integrity by sourcing alternative images.

The third datasets comprise Chest CT-Scan image Datasets obtained from Kaggle. They consist of chest CT-scan image representing different types of chest cancer, including Adenocarcinoma, Large cell carcinoma, Squamous cell carcinoma, and normal cells images. These datasets are partitioned into training, testing, and validation set in proportion of 70%, 20%, and 10% respectively. By utilizing these dataset, we can comprehensively evaluate our models' predictive capability in the context of chest cancer detection.

### 3.2.1 Brief Recap of the Algorithms Used

In addition to the diverse datasets, we employ a suite of deep learning algorithms. I want to say that I only used Convolutional Neural Networks over the rest in this experience.

so we used Convolutional Neural Networks (CNNs) models such as Vgg16, Vgg19 2D CNN, VGG16, U-Net, and cnn inception v3 and ResNet50 models.

in the other hand we used these Bayesian Methods : Bayesian Approximation Dropout with L2, Bayes by Backprop, Variational Inference, Monte Carlo Dropout

again the goal is to compare Bayesian and traditional methods and not to compare the effectiveness of Bayesian methods in all cases of data used.

### 3.3 Comparaisons

In this section, we undertake an extensive comparison between traditional Convolutional Neural Network (CNN) algorithms and Bayesian CNN models, focusing specifically on their efficacy in tackling limited data prediction tasks.

#### 3.3.1 Results and Analysis

##### Brain mri images

We evaluated the performance of various deep learning models on the Brain MRI Imagesf or Brain Tumor Detection dataset. The models included Simple CNN, VGG16, 2D CNN, and U-Net, in contrast, we use Bayesian Neural Network (BNN) using the traditional approach dropout, and Bayes by backpropagation.

Models		Train	Time (s)	Test	Val
traditional CNN	Simple CNN	0.8533	680.51	0.796	0.825
	Vgg16	0.632	33.392	0.733	0.660
	2D CNN	0.614	11.85	0.786	0.663
	U-net	over	over	over	over
bayesian methods	Bayesian Dropout	0.952	11.70	0.947	0.952
	Bayes by backpropa	0.653	12.50	0.9066	0.8022
	MCD	0.60	0.64	0.9066	0.579

Figure 3.1: Performance of Deep Learning Model

**U-Net:** Although no specific training time is provided for U-Net and no value else, its training duration for just one epoch is noticeably extensive. This suggests potential inefficiency for this task due to its prolonged training process, hindering precise time measurements.

**Simple CNN:** This model took 680.51 seconds for training. Simpler architectures generally require more time to train due to fewer layers and parameters.

about vgg16 and 2d CNN, the training time was very short, but the data was not understood to the model and deal with it enough.

it hardly give a value greater than 60.0.%

We can clearly see the effectiveness of Bayesian methods here, but I want to note that training on the GPU is always faster. The values of this table were updated and getting better values were obtained than using the CPU.

The traditional methods suffered from Uncertainty and did not understand the data and train with it well, I also note The values do not contain overfit, and it has been verified that the last final validation loss was smaller than the last final train loss, and the same as if the last train accuracy is greater than the last validation accuracy. this measure is appropriate and safe, but there are of course other considerations.

Let's explore why higher training accuracy is advantageous, especially in limited data scenarios:

**Bayesian approach dropout :** This approach incorporates regularization techniques such as L2 regularization and dropout to improve model generalization and prevent overfitting. With a training accuracy of 0.952, the model effectively captures underlying patterns in the data. In limited data scenarios, higher training accuracy suggests better utilization of available information, leading to improved generalization and robustness. again The values do not contain overfit while final validation loss was higher then the final train loss value.

### **Bayes by backpropagation:**

this model achieves a training accuracy of 0.653, but in return achieves a higher test and validation accuracy, and that is what in fact we all care about.

This indicates that the model effectively learns from the available data, potentially leading to better generalization and performance on unseen data.

In contexts with limited data, higher training accuracy signifies the model's ability to extract meaningful representations from the dataset.

As for the methods that did not appear in the table, **Monte Carlo dropout**, it were not good enough and only added ambiguity to the data and just keep overfitting, unlike theory. Indeed, it is not as easy as it is said. Applying these methods requires deeper study and a lot of experimentation and is compatible with data. Different and with different characteristics as well.

### **brain tumor mri dataset**

#### **Test Accuracy:**

Bayes by Backpropagation achieves the highest test accuracy (81.6%).

This indicates that Bayes by Backpropagation perform better in generalizing to unseen and limited data compared to ResNet50 and Simple CNN which fell into overfit every time.

It is noteworthy that there is a difference in the time values, due to the use of the CPU and the GPU.

Simple cnn and resnet were not exempt from data limitation.

Although Resnet was learned, it was not trained as required and was affected by the fact that the data was small in size.

Models		Train	Time (s)	Test	Val
traditional CNN	Simple CNN	0.938	2507	0.973	0.934
	ResNet50	0.606	5252	0.683	0.684
bayesian methods	Bayes by backpropa	0.816	131.89	0.816	0.877
	MCD	0.91	89.93	0.954	0.91

**Figure 3.2: Performance of Deep Learning Model**

For bayes by backpropagation only, we had to adjust the dropout to smaller than 0.5 so that it would not be exposed to overfitting, but it was consuming a lot of GPU.

The monte carlo dropout method applied on Simple CNN was also not effective were it is marked in red.

ResNet model was very week and did not handle the data effectively.

**Training and Validation Accuracy:** Bayes by Backpropagation exhibits high training accuracy (99.5.%) but slightly lower validation accuracy (87.7.%) but it is not good enough for a good process.

## Chest ctscan images

Specifically here, in the third attempt, I added noise to the data so that it becomes very incomprehensible. This is with the aim of testing the ability of Bayesian methods to deal not only with limited data, but even with noise.

Although the previous data also contains its own noise, adding noise, such as flipping the images and adding more light to the images, makes it difficult to train.

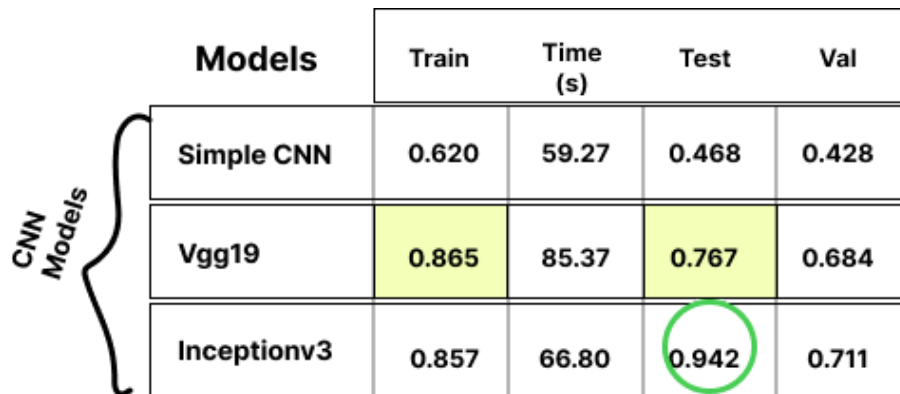
## Before Noise

**Simple CNN** achieved a test accuracy of 0.6129, which is lower compared to the other models.

Its training accuracy was 0.4688, indicating a potential issue of overfitting.

**InceptionV3** had a higher training accuracy of 0.9429, suggesting better generalization compared to Vgg16.

**Vgg19** had a training accuracy of 0.7667, indicating a moderate level of overfitting.



	Models	Train	Time (s)	Test	Val
CNN Models	Simple CNN	0.620	59.27	0.468	0.428
	Vgg19	0.865	85.37	0.767	0.684
	Inceptionv3	0.857	66.80	0.942	0.711

Figure 3.3: Before applying noise on data(CNN Model)

it is noteworthy that the GPU used here, so the time of training was short and fairly acceptable.


## After Noise

**Simple CNN** experienced a significant decrease in test accuracy to 0.5161, indicating that the introduced noise adversely affected its performance

**Vgg19** also showed a decrease in test accuracy,it was impacted by the noise as well.

dispite his strenght,it was affected too well by noise and became anxious about certainty in his predictions.

**InceptionV3** it was also noticeably affected and did not fulfill the function well.



<b>Models</b>	<b>Train</b>	<b>Time (s)</b>	<b>Test</b>	<b>Val</b>
<b>Simple CNN</b>	0.5161	47.38	0.480	0.482
<b>Vgg19</b>	0.596	58.03	54.50	0.711
<b>Inceptionv3</b>	0.742	72.01	0.726	0.544

**Figure 3.4: After applying noise on data(CNN Model)**

so the test accuracy began to indicate that the model was not coping with this type of noisy data.

Now we will see the effectiveness of Bayesian methods and immediately after noise.

<b>Models</b>	<b>Train</b>	<b>Time (s)</b>	<b>Test</b>	<b>Val</b>
<b>Bayes by prop</b>	0.52	57.38	0.474	0.58
<b>MCD</b>	0.46	55.19	0.56	0.44

**Figure 3.5: After applying noise on data(Bayesian Methods)**

I want to say that using the CPU in these Bayesian methods inhibits them and does not make them learn efficiently due to the many and complex calculations.

We noticed that the Bayesian methods were not effective after using noise, and it was intentional to use the CPU here and add 5 epochs for training.

Bayesian methods suffered greatly from overfit, and finding values capable of producing results and understanding the data was difficult and took a lot of time, and we know that this type of calculation requires a GPU. so this low efficiency just tell us that these bayesian methods did not deal well with the uncertainty estimation.



### **Comparing Bayesian Approximation with Other Methods**

Before noise, InceptionV3 and Vgg19 outperformed Simple CNN in terms of both test and training accuracies. However, after introducing noise, InceptionV3 exhibited better resilience compared to SimpleCNN and Vgg19. This suggests that InceptionV3 have a more robust feature representation that is less sensitive to noise.

uncertainty estimation through Dropout can help mitigate the impact of noise and improve model robustness.

After the introduction of noise, the performance of the non-Bayesian methods, such as Vgg16 and Vgg19, decreased significantly. These methods lack the ability to capture and quantify uncertainty in their predictions, which can make them more susceptible to the adverse effects of noise.

the Bayesian approximation method (Dropout) and the Monte Carlo Dropout (MCD) technique did not perform well after noise as it used to be.

It is worth noting that the variational inference and many Bayesian methods were not very effective, and it was like trying all the methods, and this in itself may not be possible for me to be certain of the effectiveness of combining these methods inevitably with CNN's architecture.

The Bayesian methods provide an advantage in their ability to quantify uncertainty in predictions only in the theory and allowing them to make more informed decisions, especially in the presence of noise.

This uncertainty estimation helps mitigate the negative impact of noise and improves the overall robustness and reliability of the models.

## 3.4 Discussion of Findings and Insights

### 3.4.1 Interpretation of Experimental Results and Implications

After evaluating the uncertainty estimation methods and assessing model robustness, we obtained valuable insights from the experimental results.

Let's discuss the interpretation of these findings and their implications for real-world applications. **Uncertainty Estimation Methods** Based on the evaluation metrics such as calibration, sharpness, and coverage, we found that Monte Carlo Dropout not demonstrated effective uncertainty estimation capabilities within Bayesian Deep Learning models.

i want to mention that MCD was tried, and although it was effective in the third data, it completely failed in both first and second data.

**Monte Carlo Dropout** leverages dropout during training and testing to approximate the posterior distribution and obtain a distribution of predictions.

Although bayesian methods such as MCD (Monte Carlo Dropout) and Bayesian Dropout seem to be promising in theory, they have several challenges in practice.

One of the primary challenges behind these methods is that they can actually be computationally efficient and complex, which makes them less scalable in the case of larger data sets and deeper architectures. Further, these methods need to be tuned hyperparameters appropriately; otherwise, the suboptimal performance and instability during the training will be the case.

It has also been observed that the quality, and the distribution of the data could have a significant bearing on the performance of Bayesian methods. Uncertain, non-representative, and low quality data is evidence that provides reasons why these methods cannot adequately capture the necessary uncertainty or make correct generalizations to unseen examples.

Implementing Bayesian methods becomes another issue. ForEach verification of the uncertainties and fixing the bugs in the Bayesian methods can also become challenging sometimes. This may require deep analysis and usage of diagnostic tools.

Last but not least, they can be resource-intensive meaning that they could require additional computational resources and longer training times, which may limit their practicality in real-time or resource-constrained applications.

### **3.5 Conclusion**

As we saw in this chapter, we numerically compared traditional convolutional neural networks (cnns) methods and the most famous models with Bayesian methods and concluded that some of them were useful and others made training and understanding the data more ambiguous.

What I can say in this section specifically is that choosing the appropriate Bayesian methods requires a deep understanding, a lot of experimentation, and understanding and analyzing the data in advance so that we can know what is wrong with it.

We will talk in the next chapter about general Discussion overview based on Bayesian methods.