Exploring Bayesian Neural Networks for Continual Learning on MNIST Dataset

Ram J. Zaveri
rz0012@mix.wvu.edu

West Virginia University

1 Abstract

The artificial systems have significantly improved over the last decade, much progress has been made with tremendous accuracy gain. However, there are still major issues present at the core: the challenge of lifelong learning and reliability. Many approaches tried tackling the problem of continual learning and became victim to the phenomenon called Catastrophic forgetting. Present artificial systems can learn domains they have seen before fairly well, but when they come across something new, their performance either significantly declines or they forget what they have learned in the past when trying to teach them the new data distribution. Additionally, they are also prone to being overly confident when performing inference on seen as well as unseen data, causing significant reliability issues when lives are at stake. Therefore, there is a need to formulate an approach that will be continually adaptable as well as reliable. The free energy hypothesis in brain science suggests that the biological systems give us a solid understanding of the reasoning under uncertainty which takes us to the domain of Bayesian reasoning. In this work, we investigate Bayesian Neural Network for a continual learning task. We tested the approach on a vision dataset, MNIST, and show relative performance improvement under the conditions when the model is forced to predict and when the model is not.

2 Introduction

Artificial Neural Networks (ANNs) have been the subject of numerous issues in recent years. For instance, despite being generally good approximators, ANNs suffer greatly from significant performance degradation when they encounter information that does not fit into their prior distributions. Biological systems are capable of managing that. According to population modeling of the neural dynamics, every neuron has a threshold for processing information flow both spatially and temporally before moving on to the next neuron. In essence, this aids in deciding "if" the cell should be stimulated by the previous input at this precise moment. A build-up of ions in the presynaptic cleft increases the likelihood that the subsequent neuron will fire to process the input [1]. The free energy hypothesis proposes to measure surprise in terms of variational free energy, as explained in [2]. It was once known as the Bayesian brain hypothesis. When we consider it in the context of the biological brain, it continuously modifies an internal model of the environment and attempts to minimize the information-theoretic surprise. Essentially, synaptic plasticity can be described as a Bayesian learning mechanism that monitors the distribution of synaptic weights across time rather than the weights themselves [3, 4]. Thus, a more symmetrical and broad perspective on the issue of continuous adaptability would result through examining the likelihood that an event will occur as opposed to declaring that it has. The main objective is to forecast a continuous variable in order to ascertain the likelihood that an event will occur and provide a response that is more consistent with biology.

Weight and bias are two factors that ANNs use to assist them fine-tune the distribution clusters in the hyperspace. The computations are carried out by building up the weights of earlier neurons that have been processed by an activation function (ReLU, Sigmoid, etc.) to add nonlinearity to the previous layer before processing it to the subsequent layer. Naturally, activation layers are crucial because they give the neural network non-linearity and prevent the preceding layer's weights from blowing up. It is presumed that the learnt parameters, weights, and biases are optimal for the unknown data once the model has been trained.

In general, this isn't the case since ANNs overfit to the observed data, which causes them to either perform poorly or catastrophically forget when trained on fresh data. Essentially, we must incorporate a method of assurance into the process of learning. An alternative way to put it is to give each individual neuron a degree of uncertainty. By defining the weights' means and variances, scientists are attempting to address this issue through what are known as Bayesian Neural Networks (BNNs). In this work, we leverage this information and suggest that Bayesian inference significantly mitigates the problem of overfitting while providing significantly good priors for the continual learning paradigm.

3 Related Works

Continual Learning

Continual learning refers to a setting where an algorithm is learning in a continuous fashion. Thus said, the algorithm does need to preserve the past information it learned; however, most approaches struggle in this particular setting and go through catastrophic forgetting [5]. Furthermore, a plethora of outstanding research works have discussed various mathematical and empirical strategies to prevent catastrophic forgetting. Replay [6] based mechanisms make use of a memory buffer that consists of examples from previously learned tasks and are presented with examples during the continual learning phase. Elastic Weight Consolidation [7, 8] is a regularization base method that constrains the model's weights based on their importance for previously learned tasks. Another way to approach continual learning is to keep expanding the model architecture to the new classes/tasks [9, 10]. As described in [11], there are various categories and paradigms in continual learning as well; for example, instance-incremental learning describes learning instances in the incremental fashion, task-incremental learning refers to learning tasks with disjoint data label spaces incrementally, class-incremental learning refers to learning tasks with disjoint label spaces with class generalizable capability, and so on. In this work, we are utilizes class-incremental learning paradigm where we introduce new classes as number of tasks increases.

Bayesian Neural Networks (BNNs)

Bayesian Neural Networks are increasingly gaining popularity in deep learning community, where more recently, [12] combines the most recent research to prepare a tutorial specifically for deep learning practitioners. BNNs are stochastic models and learn continuous distributions (one can also discretize this [2]), probabilistic approximations are more stable compared to their ANNs counterparts.The advantage of BNNs is that they are not prone to overfitting as can be observed in traditional learning schemes. They are being used in many fields, i.e., computer vision [13], network traffic monitoring [14], medicine [15], active learning [16, 17], online learning [18], and so on. Additionally, as observed in [5], since the calculated posteriors can be reused as priors for the data the model has not seen before, it can inherently avoid the major problem of catastrophic forgetting. Other key advantages are they naturally learn to quantify uncertainty, they can even distinguish between uncertainty associated with the data it has seen and the data it has never seen, making it robust against anomaly [19]. Additionally, they are also different from, so called, black box algorithms with traditional deep learning models since, the prior that are used in BNNs are explicit [12]. [20, 21] further incorporates Bayesian Learning rule with natural gradient descent in a fashion it works, if not exactly, similarly, without destroying the core fundamentals of machine learning principles, with machine learning algorithms. Therefore, we are investigating the plausibility of BNNs in a continual learning situation in this paper.

4 Methods

4.1 Overview

We investigate Bayesian Neural Networks for Continual Learning on MNIST Dataset in this work. We first describe the Bayesian Inference scheme used in this work. Further, we discuss continual learning paradigm that was used to test the capability of Bayesian Neural Network in a class-incremental learning setting.

4.2 Bayesian Inference

The theory of variational free energy describes the probabilistic model to predict observations from the postulated causes [12, 22, 23]. Unlike the frequentist approach, Bayesian Inference considers the probability as a measure of belief in the occurrences of events rather than the limit on the frequency. As a result, prior beliefs affect posterior beliefs in Bayesian inference. See equation 1.

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)} = \frac{P(D,H)}{\int_H P(D,H')dH'} \tag{1}$$

Where $P(D|H)P(H)$ is the likelihood, $P(H)$ is the prior, $P(D) = \int_H P(D,H')dH'$ is the evidence from the data D, $P(H|D)$ is the posterior, and $P(D,H)$ is the joint probability.

As described in [12], BNNs are referred to as stochastic artificial neural networks using Bayesian Inference. Here, stochastic models utilize probability distributions rather than a point estimate of the value unlike the traditional neural networks. This gives a better understanding of precision, or uncertainty, of the weight vector that is associated with every neuron. This is typically done by incorporating a stochastic activation or weight as observed in BNNs. As a result, they are capable of simulating multiple potential outcomes, thereby forming an ensemble of networks by nature. For example, assume $\theta = (\mathbf{W}, \mathbf{b})$, where θ is the model parameters, $\mathbf{W}$ and $\mathbf{b}$ are weights and biases, respectively. The traditional networks will learn through a point estimate equation, $s(\mathbf{Wl} + \mathbf{b})$, where s is the activation function. In stochastic models, either s or $\theta = (\mathbf{W}, \mathbf{b})$ are associated with probability distributions. And since the computation is inherently dependent on the probability distributions of either s or θ, stochastic models are forming ensembles of distributions, thus, can be used as a special case of ensemble learning [24].

To train the BNNs, we will first have to establish the Bayesian Posterior:

$$p(\theta|D) = \frac{p(D_y|D_x,\theta)p(\theta)}{\int_\theta p(D_y|D_x,\theta')p(\theta')d\theta'} \propto p(D_y|D_x,\theta)p(\theta) \tag{2}$$

During prediction for a given $p(\theta|D)$, we can compute the prediction, $p(y|x,D)$ (sampled from $y = \Phi_\theta(x) + \epsilon$, where Φ is an approximation and ϵ is the noise) as follows:

$$p(y|x,D) = \int_\theta p(y|x,\theta')p(\theta'|D)d\theta' \tag{3}$$

Here, computing $\int_\theta p(D_y|D_x,\theta')p(\theta')d\theta'$, is extremely difficult and time-consuming, since we are trying to perform integral over all the possible parameters, extremely in-efficient. Therefore, one of the two major approaches is used in this work, called Variational Inference. The other one is Markov Chain Monte Carlo; which falls short with increasing scalability as opposed to variational inference [25].

Variational Inference [25] is not exact; however, provides very good approximation with increasing scale. Here, instead of sampling from an exact posterior mentioned in Equation 3, we compute a distribution $q_\phi(H)$, namely variational distribution, parameterized by parameters ϕ. Overall, we want $q_\phi(H)$ to be as close to the exact posterior $P(H|D)$ as possible. We use Kullback-Leiber (KL) divergence [26] function to measure that based on Shannon's information theory [27]. The overall KL-Divergence loss is as follows:

$$D_{KL}(q_\phi||P) = \int_H q_\phi(H') log\Big(\frac{q_\phi(H')}{P(H'|D)}\Big)dH' \tag{4}$$

However, KL-Divergence loss still takes $P(H|D)$ into account, and will still need to calculate it; rather, if we simplify the formula, we get the following:

$$log(P(D)) - D_{KL}(q_\phi||P) = \int_H q_\phi(H') log\Big(\frac{q_\phi(H')}{P(H')}\Big)dH' \tag{5}$$

Here, during gradient calculation, $log(P(H'|D))$ becomes $log(P(H',D)) - log(P(D))$, where $log(P(D))$ is a constant and becomes 0. Therefore, we can safely eliminate this term and reduce it to the equation 5, which is also referred to as evidence lower bound ELBO loss and the way to optimize this loss is called stochastic variational inference [28].

Now, to perform backpropagation over a neural network, we need to calculate gradients over equation 5 according to the following equation:

$$\frac{\partial}{\partial \phi} \int_{\phi} q_{\phi}(\theta') f(\theta', \phi) d\theta' = \int_{\epsilon} q(\epsilon) \Big(\frac{\partial f(\theta, \phi)}{\partial \theta} \frac{\partial \theta}{\partial \phi} + \frac{\partial f(\theta, \phi)}{\partial \phi} \Big) \tag{6}$$

where, $q_{\phi}(\theta)\partial\theta = q(\epsilon)\partial\epsilon$.

Further, to estimate the priors during inference, we first need to establish that equation 5 is KL-divergence of $q(\phi)$ substracted from the log-likelihood of the data, $log(P(D)) - D_{KL}(q_{\phi}||P)$, which means equation 5 is a function of both the variational parameters, ϕ, to estimate the posterior, and (addition of) a parameterized prior distribution, $p_{\xi}(H)$, given the parameters, ξ. Therefore, the loss function now becomes,

$$L = log(q_{\phi}(\theta)) - log(p_{\xi}(D_y|D_x, \theta)p_{\xi}(\theta)) \tag{7}$$

Here, we perform gradient decent with respect to both ϕ and ξ using the equation 6 (replace ϕ with ξ).

4.3 Reliability Measure

Section 4.2 describes the overall method used for performing Bayesian Inference; however, to test the ability of BNNs to be reliable, we use a simple sanity test. We train a simple fully connected network on all the digits of MNIST [29] dataset and test in under two conditions: when the model is forced to predict, and when the model is not. Meaning, in the first instance, we disregard the uncertainty prediction and use argmax of the prediction as the final inference. In short, regardless of the confidence being low or unstable, the model is forced to predict. In the other situation, we discard the predictions where the certainty was below 0.8 (sampled from a population of 100 predictions), and calculated the accuracy when the model made predictions.

4.4 Continual Learning

To test the BNNs in a continual learning situation, we choose MNIST [29] dataset that consists of 10 digits, and split them in 5 groups ([0,1], [2,3], [4,5], [6,7], [8,9]) sequentially. We train the model with each group once and in sequence and calculate the performance in terms of accuracy for this particular task. Similar to 4.3, the model is a fully connected network with 10 classes as predictions; however, only two classes are being trained at a time. Additionally, we use replay as our continual learning paradigm, and keep a memory buffer of 960 samples. This number was decided arbitrarily. The memory buffer consists of samples from previously learning tasks, where random samples are stored evenly as the number of tasks increase. The results are shown in section 5.

5 Results

5.1 Implementation Details

All the code was written in Python 3.10 and the models are implemented in PyTorch 2.0 [30]. All the models in this work are using 784 as input dimension (MNIST [29] dimensions are 784 $\in R^1$), 1024 as hidden dimensions, and 10 as output dimensions given the number of digits in MNIST [29] are 10. All the nodes are fully connected nodes. For direct test on all 10 digits, we train the model for 5 epochs, and during continual learning, we only train it for one epoch every task in an incremental fashion, i.e., [0,1] category, then [2,3] category, so on and so forth. Adam [31] optimizer strategy with the learning rate of 0.001 and momentum of 0.9 was used for both BNNs and generic ANNs. Cross-entropy loss was used as a categorical loss during training. All the training and testing was done on a single TITAN Xp GPU. Additionally, we sample inferred results 100 times and take the average in BNNs.

5.2 Reliability Testing

As described in 4.3, we test Bayesian Inference scheme on the whole MNIST [29] dataset under two conditions, when the model is forced to predict vs when the model is not. Please see table 1. We can

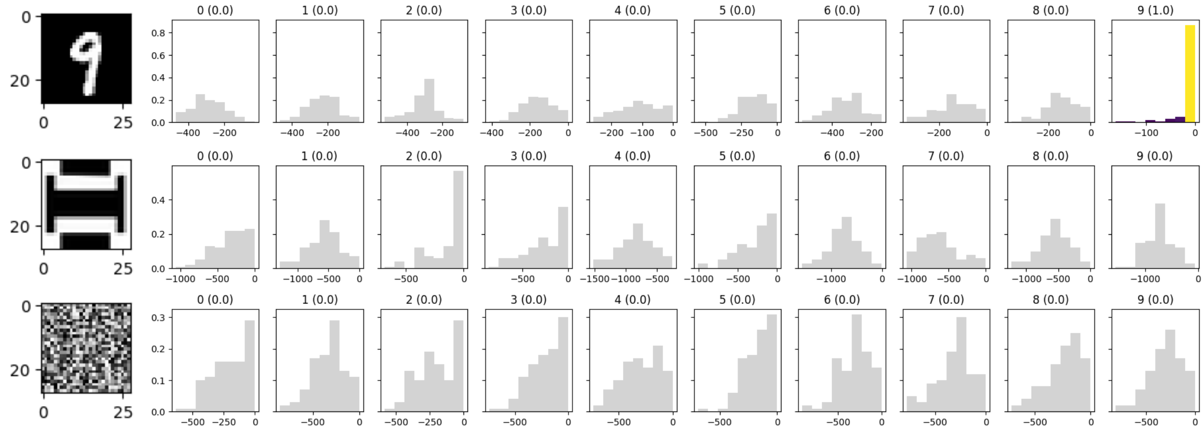

Figure 1: **Qualitative results on BNNs.** First row shows results from the learned classes, the second row shows results on alphabets, (meaningful but) not learned domain, and the last row shows results on random noise.

Table 1: Preliminary Testing when the model is forced to predict VS when the model is not.

	Forced Pred.		Not Forced Pred.	
	Skipped	Accuracy	Skipped	Accuracy
BNNs	0/10000	0.87	1415/10000	0.95

Table 2: Comparative study on Generic ANNs and BNNs (with and without skipping allowed).

	Tasks														
Training	Task 1	Task 2		Task 3			Task 4				Task 5				
	Task 1	Task 1	Task 2	Task 1	Task 2	Task 3	Task 1	Task 2	Task 3	Task 4	Task 1	Task 2	Task 3	Task 4	Task 5
Generic ANNs	0.997	0.570	0.950	0.989	0.201	0.751	0.989	0.761	0.121	0.794	0.989	0.810	0.510	0.497	0.469
BNNs	0.998	0.950	0.973	0.987	0.845	0.813	0.967	0.880	0.790	0.790	0.949	0.850	0.899	0.704	0.596
BNNs (skipping allowed)	0.999	0.978	0.968	0.997	0.901	0.902	0.965	0.883	0.840	0.930	0.976	0.896	0.895	0.842	0.610

observe that when the model was forced to predict, the accuracy was 0.87 and was not quite satisfactory. Here, the model does not consider its uncertainty predictions. On the other hand, when we consider the uncertainty predictions, the model skips 1415 samples, but the accuracy is 0.95, which is significantly higher than previous results. This essentially proves the reliability of the model in which when the model is certain the accuracy is indeed relatively high. For qualitative analysis, please consider Figure 1.

5.3 Class-Incremental Learning

Table 2 described empirical study on incrementally changing tasks, specifically on generic ANNs, and BNNs in conditions where model is forced to predict and when the model is not (skipping predictions are allowed based on the probability). Here, this continually changing tasks, as described in section 4.4, are five sequential tasks split evenly from the MNIST [29] dataset. We can observe that generic ANNs either overfit to the first task or to their current task. Their performance on the intermediate tasks is unstable, and sometimes, experiences catastrophic forgetting. For example, when learning Task 4, its accuracy decreased significantly on Task 3. On the other hand, BNNs are relatively stable during incrementally learning sequential tasks. Additionally, if we allow skipping, the accuracy improves even further, specifically in the final task. This suggests the model's ability to be reliable under incrementally changing conditions. We should note that, we do make use of replay in this case, and randomly sample examples from previous tasks as described in 4.4. Further, it also suggests, that BNNs can indeed learn for sparse data or maintain their priors throughout the continual learning process.

6 Discussion and Conclusion

We study Bayesian Neural Networks for Continual Learning on MNIST [29] Dataset in this paper. We show that BNNs are indeed reliable networks and can learn even in a continual learning scenario where replay buffer is present. Further, we discuss, how we can make use of their confidence/uncertainty predictions to leverage their capacity to make accurate predictions which would be extremely useful in a real-world situation. A few limitations of this approach are the scalability and complexity. Since the task was relatively simple and the data scale was relatively small, it requires more investigation to understand the true potential of BNNs.

References

[1] L. Barrett, *Seven and a Half Lessons about the Brain*. Houghton Mifflin Harcourt Publishing Company, 2020.

[2] N. Skatchkovsky, H. Jang, and O. Simeone, “Bayesian continual learning via spiking neural networks,” *Frontiers in Computational Neuroscience*, vol. 16, nov 2022.

[3] L. Aitchison, J. Jegminat, J. A. Menendez, J.-P. Pfister, A. Pouget, and P. E. Latham, “Synaptic plasticity as bayesian inference,” *Nature Neuroscience*, vol. 24, no. 4, pp. 565–571, mar 2021.

[4] K. Friston, “The history of the future of the bayesian brain,” *NeuroImage*, vol. 62, no. 2, pp. 1230–1233, aug 2012.

[5] H. Ritter, A. Botev, and D. Barber, “Online structured laplace approximations for overcoming catastrophic forgetting,” *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[6] L. Alssum, J. L. Alcázar, M. Ramazanova, C. Zhao, and B. Ghanem, “Just a glimpse: Rethinking temporal information for video continual learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2473–2482.

[7] H. Ahn, S. Cha, D. Lee, and T. Moon, “Uncertainty-based continual learning with adaptive regularization,” *Advances in neural information processing systems*, vol. 32, 2019.

[8] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, “Overcoming catastrophic forgetting in neural networks,” *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[9] S. Srivastava, M. Yaqub, and K. Nandakumar, “Lifelong learning of task-parameter relationships for knowledge transfer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2524–2533.

[10] S. Sanyal, R. V. Babu *et al.*, “Continual domain adaptation through pruning-aided domain-specific weight modulation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2456–2462.

[11] L. Wang, X. Zhang, H. Su, and J. Zhu, “A comprehensive survey of continual learning: Theory, method and application,” *arXiv preprint arXiv:2302.00487*, 2023.

[12] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun, “Hands-on bayesian neural networks—a tutorial for deep learning users,” *IEEE Computational Intelligence Magazine*, vol. 17, no. 2, pp. 29–48, 2022.

[13] A. Kendall and Y. Gal, “What uncertainties do we need in bayesian deep learning for computer vision?” *Advances in neural information processing systems*, vol. 30, 2017.

[14] T. Auld, A. W. Moore, and S. F. Gull, “Bayesian neural networks for internet traffic classification,” *IEEE Transactions on neural networks*, vol. 18, no. 1, pp. 223–239, 2007.

[15] W. Beker, A. Wołos, S. Szymkuć, and B. A. Grzybowski, “Minimal-uncertainty prediction of general drug-likeness based on bayesian neural networks,” *Nature Machine Intelligence*, vol. 2, no. 8, pp. 457–465, 2020.

[16] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International conference on machine learning*. PMLR, 2017, pp. 1183–1192.

[17] T. Tran, T.-T. Do, I. Reid, and G. Carneiro, "Bayesian generative active deep learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6295–6304.

[18] M. Opper and O. Winther, "A bayesian approach to on-line learning," 1999.

[19] S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1184–1193.

[20] M. Khan and W. Lin, "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 878–887.

[21] M. E. Khan and H. Rue, "The bayesian learning rule," *arXiv preprint arXiv:2107.04562*, 2021.

[22] J. Bruineberg, J. Kiverstein, and E. Rietveld, "The anticipating brain is not a scientist: the free-energy principle from an ecological-enactive perspective," *Synthese*, vol. 195, no. 6, pp. 2417–2444, 2018.

[23] A. Etz, Q. F. Gronau, F. Dablander, P. A. Edelsbrunner, and B. Baribault, "How to become a bayesian in eight easy steps: An annotated reading list," *Psychonomic bulletin & review*, vol. 25, no. 1, pp. 219–234, 2018.

[24] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. CRC press, 2012.

[25] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[26] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[27] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[28] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *Journal of Machine Learning Research*, 2013.

[29] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.