# Editing Bayesian Neural Network

Jiarao Liu (jl14412) [1] Yuan Zhang (yz5892) [1] Jiangjie Bian (jb6942) [1]

## Abstract

This paper studies the problem of approximately updating a Bayesian model from a small subset of the training data to be replaced by new data. We frame this problem as one of minimizing the Kullback-Leibler divergence between the approximate posterior belief of model parameters after directly updating, and compare it to the exact posterior belief from retraining with remaining data. Using the variational inference (VI) framework, we show that it is equivalent to minimizing an evidence upper bound. In model training with VI, we apply to use optimization method of Black-Box Variational Inference, which could quickly be applied to many models with little additional deviation.

## 1. Introduction

Our interactions with machine learning (ML) applications have surged in recent years such that large quantities of users' data are now deeply ingrained into the ML models being trained for these applications. Most of data experience a high pace of updating, meaning that the data used to train the previous model would quickly become obsolete, and the accuracy of the model trained on these outdated data would not enjoy high accuracy as before. In this way, it would desirable to ll the model to be upgraded with new data.

A naive alternative to machine unlearning is to simply retrain an ML model from scratch with the remaining useful and the new coming data. In practice, this is prohibitively expensive in terms of time and space costs since the remaining data is often large such as in the above scenarios. How then can a trained ML model directly and efficiently upgrade from a small subset of data to be replaced to become approximately close to that from retraining with the large remaining data and new data?

Our work here addresses the above question by focusing on the family of Bayesian models, whose uncertainty would make the model upgrading more likely to occur. Our proposed loss function measures the Kullback-Leibler (KL) divergence between the approximate posterior belief of model parameters by directly upgrading from new data vs. the exact posterior belief from retraining with remaining data. Using the variational inference (VI) framework, we show that minimizing this KL divergence is equivalent to minimizing (instead of maximizing) a counterpart of the evidence lower bound called the evidence upper bound (EUBO)

## 2. Related Works

Previously, scholars from MIT and NUS and proposed the method that allow part of training data of bayesian neural network to be erased, whose result is quite promising. Our work extended their work by introducing a new group of data which would become the substitute of the dat to be erased. Such scenario could be more consistent with many real-would situation. Additionally, our proposed method could also be regarded as a Continual Learning Approach under the Bayesian context, where new data continuously appear while old data need to be forget. Our approach could also combine these two phases all together with Variational Inference to directly obtain a posterior of upgraded model weight distribution.

Black-Box Variational Inference is a algorithm specially design to approximate the posterior in Variational Inference. Such method has use Monte Carlo Sampling and stochastic optimization to approximate the posterior of variational distribution. Despite a little deriation, such method could surely adapt to various of VI schemes and obtain the result efficiently.

Yarin Gal et al. applied variational inference to train BCNN, as described in "Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference" (2016). In this paper, Gal et al. train networks that approximate Bernoulli variational inference in Bayesian Neural Networks. While the variational approach is expensive to compute when applied to posterior approximation for BNNs as the number of parameters becomes large, the research

[*]Equal contribution [1]New York University, New York, United States. Correspondence to: Leo Liu <lj14412@nyu.edu>.

*Proceedings of the 39th International Conference on Machine Learning*, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

group chose Bernoulli to approximate variational distributions, which requires no extra parameters. Moreover, since overfitting is caused by using dropout in CNNS after inner-product layers only, the kernels of the CNN are integrated to fix overfitting. Bayesian CNN is thus applied with dropout for all layers.

## 3. Methods

Here we have a posterior distribution of a model parameter $P(\theta|D)$ which is trained on $D$. The $D$ itself consists of two parts: $D_r$ and $D_e$ such that: The $D_r$ is the correct dataset that we would like to preserve on the model, and $D_e$ is the error data for us to replace. $D_n$ is the new data set here to replace $D_e$. Now our job is to find the posterior $P(\theta|D, D_r)$ with only access to $D_e$, $D_n$ and $P(\theta|D)$. This is because that in most re-world cases, $D_e$ is often a small set of errors for us to correct while allocating $D_e$ from the database could be much more costly and inefficient.

$D$ : Original Training Dataset
$D_r$ : Correct Dataset
$D_e$ : Error Dataset
$D_n$ : New Data to replace $D_e$

A Bayesian Neural Network $P(\theta|D)$ that has been trained previously based on dataset $D$ is available. The set of training features are compared with corresponding labels and added to the correct dataset $D_r$ or the error dataset $D_e$, depending on whether the data are correctly labeled. Therefore, $D_r \cup D_e = D$, $D_r$ intersects $D_e = \emptyset$.

The new dataset $D_n$ is combined with the correct dataset $D_r$ that has been filtered out to improve the given network. Notice that the error dataset $D_e$ is left out. In this new training process, instead of retraining the whole network system, only editing based on the new dataset $D_n$ is needed. This reduces the load and cost of training.

We have the Bayesian Neural Network $P(\theta|D)$, the error dataset $D_e$, and the new dataset $D_n$.
Now we would like to find $P(\theta|D_r, D_n)$
(Note: $D_r \cup D_e = D$; $D_r \cap D_e = \emptyset$)

- Theorem 1: $P(\theta|D_r, D_n) = \frac{P(D_n|\theta)\cdot P(\theta|D_r)}{P(D_n|D_r)}$
- Theorem 2: We assume that $D_r$ and $D_e$ are conditionally independent on $\theta$, then $P(D|\theta) = P(D_r|\theta)\cdot P(D_e|\theta)$

  $\Rightarrow$ Therefore, $P(\theta|D_r) = \frac{P(\theta|D)\cdot P(D_e|D_r)}{P(D_e|\theta)}$
- Theorem 3: Based on the previous two theorems, we now want to use Variational Inference to approximate the posterior $P(\theta|D_r, D_n)$. Define $q_\phi(\theta)$ parameterized by $\phi$.

  $\Rightarrow$ $KL[q_\phi(\theta)\|p(\theta|D_r, D_n)] = H[q_\phi(\theta)] - E_{\theta,q}[\log P(D_n|\theta)] - E_{\theta,q}[\log P(\theta|D)] + E_{\theta,q}[\log P(D_e|\theta)] + \log P(D_n|D_r) - \log P(D_e|D_r)$

Thus, $\phi = \min[\ H[q_\phi(\theta)] - E_{\theta,q}[\log P(D_n|\theta)]$
$-E_{\theta,q}[\log P(\theta|D)] + E_{\theta,q}[\log P(D_e|\theta)]\ ]$

The distribution of model parameter theta based on the correct dataset $D_r$ and the new dataset $D_n$, $p(\theta|D_r, D_n)$ , is expanded by definition into $\frac{p(D_n, D_r, \theta)}{p(D_n, D_r)}$, which yields the distribution of theta when all the data needed for retraining are fed to the network. The probabilistic distribution is examined under the context of correctly classified previous data. Divide both the numerator and the denominator by Dr to get $\frac{\frac{P(D_n, D_r, \theta)}{P(D_r)}}{\frac{P(D_n, D_r)}{P(D_r)}}$, which is equivalent to $\frac{P(D_n, \theta|D_r)}{P(D_n|D_r)}$. The theta trained from D and thus Dr is used and modified to fit new data $D_n$. This leads to the last step and gives the output $\frac{P(D_n|\theta)\cdot P(\theta|D_r)}{P(D_n|D_r)}$.

**Theorem 3.1.** $P(\theta|D_r, D_n) = \frac{P(D_n|\theta)\cdot P(\theta|D_r)}{P(D_n|D_r)}$

*Proof.* $\frac{P(D_n|\theta)\cdot P(\theta|D_r)}{P(D_n|D_r)} = \frac{P(D_n, \theta|D_r)}{P(D_n|D_r)} = \frac{\frac{P(D_n, D_r, \theta)}{P(D_r)}}{\frac{P(D_n, D_r)}{P(D_r)}} = \frac{P(D_n, D_r, \theta)}{P(D_n, D_r)} = P(\theta|D_r, D_n)$ □

**Theorem 3.2.** *Assume dataset are conditionally independent on $\theta$, then* $P(\theta|D_r) = \frac{P(\theta|D)\cdot P(D_e|D_r)}{P(D_e|\theta)}$

*Proof.* Given that $P(D|\theta) = P(D_r|\theta)\cdot P(D_e|\theta)$, then

$$P(\theta|D_r) = \frac{P(\theta|D)\cdot P(D_e|D_r)}{P(D_e|\theta)}$$
$$= P(\theta|D)\cdot \frac{P(D_r|\theta)}{P(D|\theta)}\cdot P(D_e|D_r)$$
$$= \frac{P(\theta|D)}{P(D|\theta)}\cdot P(D_r|\theta)\cdot P(D_e|D_r)$$
$$= \frac{\frac{P(\theta,D)}{P(D)}}{\frac{P(\theta,D)}{P(\theta)}}\cdot P(D_r|\theta)\cdot P(D_e|D_r)$$
$$= \frac{P(\theta)}{P(D)}\cdot P(D_r|\theta)\cdot P(D_e|D_r)$$
$$= \frac{P(D_r,\theta)}{P(D)}\cdot \frac{P(D)}{P(D_r)}$$
$$= \frac{P(D_r,\theta)}{P(D_r)}$$
$$= P(\theta|D_r)$$ □

Recall that we use $p(\theta|D_r, D_n)$ to represent the parameters retrained using previous correct data $D_r$ and newly added data $D_n$. We attempt to construct a probability model $q_\phi(\theta)$ that best fits $p(\theta|D_r, D_n)$. To achieve the best approximation, the KL divergence between $q_\phi(\theta)$ and $p(\theta|D_r, D_n)$ needs to be minimized. The variational inference is used here. Since the KL divergence between $q_\phi(\theta)$ and $p(\theta|D_r, D_n)$ can be deduced as $H[q_\phi(\theta)] - E[\log P(D_n|\theta)] - E[\log P(\theta|D)] + E[\log P(D_e|\theta)] + \log P(D_n|D_r)$. We achieve to find the upper bound for such a KL divergence.

**Theorem 3.3.** $KL\left[q_\phi(\theta) \| p(\theta|D_r, D_n)\right] = H\left[q_\phi(\theta)\right] - E_{\theta,q}\left[\log P(D_n|\theta)\right] - E_{\theta,q}\left[\log P(\theta|D)\right] + E_{\theta,q}\left[\log P(D_e|\theta)\right] + \log P(D_n|D_r) - \log P(D_e|D_r)$

*Proof.* $KL\left[q_\phi(\theta) \| p(\theta|D_r, D_n)\right] = \int q_\phi(\theta) \log \frac{q_\phi(\theta)}{P(\theta|D_r,D_n)} d\theta$
$= E\left[\log q(\theta)\right] - E\left[\log P(\theta|D_r D_n)\right]$
$= H\left[q_\phi(\theta)\right] - E\left[\log \frac{P(D_n|\theta)\cdot P(\theta|D_r)}{P(D_n|D_r)}\right]$
$= H\left[q_\phi(\theta)\right] - E\left[\log P(D_n|\theta)\cdot P(\theta|D_r)\right] + E\left[\log P(D_n|D_r)\right]$
$= H\left[q_\phi(\theta)\right] - E\left[\log P(D_n|\theta) + \log P(\theta|D_r)\right] + \log P(D_n|D_r)$
$= H\left[q_\phi(\theta)\right] - E\left[\log P(D_n|\theta)\right] - E\left[\log P(\theta|D_r)\right] + \log P(D_n|D_r)$
Given that $P(\theta|D_r) = \frac{P(\theta|D)\cdot P(D_e|D_r)}{P(D_e|\theta)}$, then $E\left[\log P(\theta|D_r)\right] = E\left[\log \frac{P(\theta|D)\cdot P(D_e|D_r)}{P(D_e|\theta)}\right]$
$= E\left[\log P(\theta|D)\cdot P(D_e|D_r)\right] - E\left[\log P(D_e|\theta)\right]$
$= E\left[\log P(\theta|D)\cdot P(D_e|D_r)\right] - E\left[\log P(D_e|\theta)\right]$
$= E\left[\log P(\theta|D)\right] + E\left[P(D_e|D_r)\right] - E\left[\log P(D_e|\theta)\right]$
$= E\left[\log P(\theta|D)\right] - E\left[\log P(D_e|\theta)\right] + \log P(D_e|D_r)$
Therefore, $KL\left[q_\phi(\theta) \| p(\theta|D_r, D_n)\right]$
$= H\left[q_\phi(\theta)\right] - E\left[\log P(D_n|\theta)\right] - \left[E\left[\log P(\theta|D)\right] - E\left[\log P(D_e|\theta)\right] + \log P(D_e|D_r)\right] + \log P(D_n|D_r)$
$= H\left[q_\phi(\theta)\right] - E\left[\log P(D_n|\theta)\right] - E\left[\log P(\theta|D)\right] + E\left[\log P(D_e|\theta)\right] - \log P(D_e|D_r) + \log P(D_n|D_r)$
$\Rightarrow KL\left[q_\phi(\theta) \| p(\theta|D_r, D_n)\right] \leq H\left[q_\phi(\theta)\right] - E\left[\log P(D_n|\theta)\right] - E\left[\log P(\theta|D)\right] + E\left[\log P(D_e|\theta)\right] + \log P(D_n|D_r)$ □

## 4. Experiment

Based on our assumption, the dataset $D$ should contain error data $D_e$. In order to achieve this, we use the MNIST dataset and preprocess the data. Specifically, we change the label for pictures and regard this as a mislabeled class. Here are five mislabeled classes we used in the experiment:

- picture ”7” labeled as 1
- picture ”9” labeled as 6
- picture ”2” labeled as 7
- picture ”3” labeled as 2
- picture ”5” labeled as 3

A sample of $D$ with 3 mislabeled classes is shown in figure 1

Then, we input the $D$ into our Bayesian Convolutional Neural Network. The Structure of our network contains two Bayesian convolutional network layers with prior mean and variance set to 0 and 0.1, and a Bayesian linear network layer with prior mean and variance set to 0 and 0.1 which output the label.

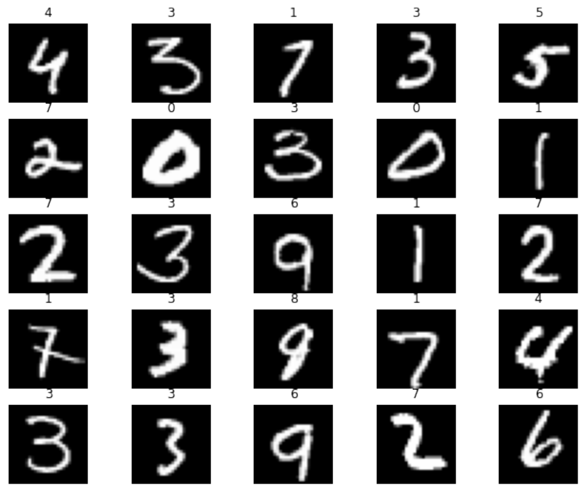

*Figure 1.* Modified MNIST Dataset with 3 Mislabeled Classes

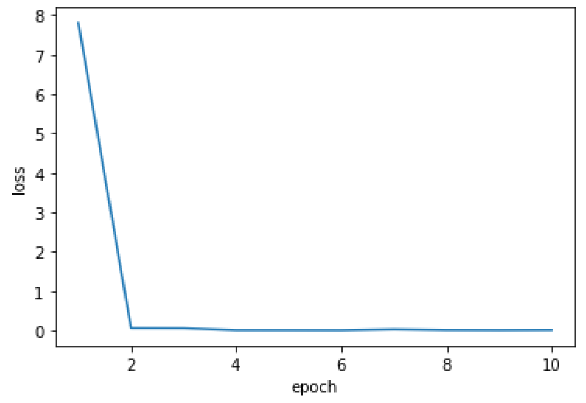

*Figure 2.* Loss for Training Edited Bayesian Convolutional Neural Network

Next, we edit the Bayesian Convolutional Neural Network with our approach. During training the edited network shown in figure 2, the loss decreases quickly as the epoch increases. In order to compare the performance of the edited Bayesian convolutional neural network with the retrained Bayesian convolutional neural network, we test the edited network under datasets containing different numbers of mislabeled classes and record the accuracy and training time. Also, we retrain a new Bayesian convolutional neural network on the correct dataset. The result is shown in table 1.

As the data shows, when the original Bayesian convolutional neural network is tested on the correct dataset, the accuracy is 69.3%, which makes sense since three out of ten classes are mislabeled. The accuracy of the retrained Bayesian

| Model | Mislabeled Class # | Accuracy(%) | Training Time(s) |
|---|---|---|---|
| Old BCNN | 3 | 69.3 | 109.32 |
| Edited BCNN | 1 | 98.7 | 52.47 |
| | 2 | 98.6 | 52.91 |
| | 3 | 97.9 | 53.18 |
| | 4 | 97.8 | 53.54 |
| | 5 | 97.8 | 53.78 |
| Retrained BCNN | | 99.4 | 110.11 |

*Table 1.* Experiment result

convolutional neural network and that of edited Bayesian convolutional neural networks based on different datasets are all good in classification. On the other hand, the training time for the edited network is only about half of the training time for retraining a new network. Also, the number of mislabeled classes contained in the original dataset only influences the training time by 0.5 seconds for one more mislabeled class.

## 5. Discussion

The original Bayesian Neural Network achieves 69.3% accuracy for the modified datasets, where we replace part of the correct labels with wrong ones. We carry out the retraining and the edited training in parallel. While both achieved performance close to perfection, edited training had accuracy slightly lower than retraining but saved about half the training time. The training time would need further improvement since the computation time is not greatly reduced with an observable accuracy loss. This suggests future work of further reducing the edited training time while having the model's accuracy closer to retraining.

## References

Gal, Y. and Ghahramani, Z. Bayesian convolutional neural networks with bernoulli approximate variational inference, 2015. URL `https://arxiv.org/abs/1506.02158`.

Kumar, A., Raghunathan, A., Jones, R., Ma, T., and Liang, P. Fine-tuning can distort pretrained features and underperform out-of-distribution, 2022. URL `https://arxiv.org/abs/2202.10054`.

Mitchell, E., Lin, C., Bosselut, A., Manning, C. D., and Finn, C. Memory-based model editing at scale, 2022. URL `https://arxiv.org/abs/2206.06520`.

Murphy, K. P. *Probabilistic machine learning : an introduction*. CThe MIT Press, Cambridge, Massachusetts, 2022.

Nguyen, Q. P., Low, B. K. H., and Jaillet, P. Variational bayesian unlearning, 2020. URL `https://arxiv.org/abs/2010.12883`.

Ranganath, R., Gerrish, S., and Blei, D. M. Black box variational inference, 2014. URL `https://arxiv.org/abs/1401.0118`.

Santurkar, S., Tsipras, D., Elango, M., Bau, D., Torralba, A., and Madry, A. Editing a classifier by rewriting its prediction rules. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Sinitsin, A., Plokhotnyuk, V., Pyrkin, D., Popov, S., and Babenko, A. Editable neural networks, 2020. URL `https://arxiv.org/abs/2004.00345`.