UNCERTAINTY-GUIDED LIFELONG LEARNING IN BAYESIAN NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

The ability to learn in a setting where tasks arrive in a sequence without having access to previous data is vital for continual learning algorithms when restricted in capacity. In this lifelong learning setting a single model is challenged to learn a new task, while at the same time using what it deems important to avoid forgetting about previous tasks as well as freeing up capacity for future tasks. We argue that the ability to identify the most contributing network parameters in a learned task, plays a critical role to decide which ones to remember. In this work we propose to rely on Bayesian Networks, which inherently model the distribution of network parameters to formulate a continual learning framework, using the parameter uncertainty for two lifelong learning directions; i.e. 1) eliminating catastrophic forgetting using weight pruning, where a hard selection is made on the most certain weights saved as a task-based binary masks, 2) reducing catastrophic forgetting by adaptively correlating the learning rate of the parameters with their uncertainty allowing for a task-agnostic model which does not require to know when a new task arrives. We show the benefit of our approach using diverse object classification datasets in both cases compared to various baselines.

1 INTRODUCTION

Humans can easily accumulate and maintain the knowledge gained from previously observed tasks and continuously learn to solve new problems or tasks. Inspired by this capability of humans, Lifelong Learning (LLL) intends to learn sequential tasks without forgetting leaned concepts as new tasks are being introduced. Unlike the powerful capabilities observed in mammalian brains in acquiring and preserving the knowledge, artificial agents still struggle in continual learning.

One of the main challenges for a lifelong learning agent is known as *catastrophic forgetting* (McCloskey & Cohen, 1989; McClelland et al., 1995) which refers to the significant drop in the performance of a learner when switching from a trained task to a new one. This phenomenon occurs because trained parameters on initial task require to change in favor of learning new objectives. This is the reason that naive finetuning intuitively suffers the most from *catastrophic forgetting*. Given limited capacity of a network, one way to see this problem as to deciding which part of network is best to keep and which part is less important and can thus be used by future tasks without significantly effecting current task performance. In this work we propose to turn to Bayesian Networks as introduced by Blundell et al. (2015). These networks represent each parameter with a distribution defined by a mean and variance which more naturally allows to determine which parameters are most important for a task. We integrate this idea with two different LLL paradigms: Hard pruning of part of the network and soft regularization based importance weighting of network parameters.

To avoid adding capacity when learning new tasks, one can first prune or compress the network (Liu et al., 2015; Aghasi et al., 2017) and then learn new tasks using free parameters as proposed by (Mallya & Lazebnik, 2017; 2018). Here a part of the network is frozen, while the remaining network is free to adapt for future tasks. While this eliminates forgetting over time, it reduces flexibility when learning large number tasks as the network fills up over time as the percentage which is frozen by previous tasks increases over time.

The second paradigm we study is regularization based LLL where forgetting is reduced by introducing a loss which penalizes when *important* weight parameters are changed while learning new tasks. In contrast to the hard pruning this is "softer" way to avoid forgetting. The main challenge

for the approaches in this domain is how to define and measure importance. Different techniques have explored different quantities to measure importance with Kirkpatrick et al. (2017); Zenke et al. (2017); Aljundi et al. (2017). While we go over more details about these methods in section 4.2 we argue that the externally defined importance definition used in these woks might yield sub-optimal regularization performance. Moreover, importance computation in these methods requires an extra computational step (whether online or offline). Despite of the difficulties in defining and measuring importance, these approaches have the benefit of good scalability as well as high efficiency.

Most of the work in lifelong learning have been proposed within the realm of non-Bayesian models where model parameters tend to overconfidently fit to training data and there is no internal definition of weight importance. In contrast, Bayesian models, enhanced by their statistically-grounded theoretical framework, provide clear definitions of uncertainty in model weights and/or predictions. Although exact Bayesian inference remains intractable in neural networks, variational approximations have provided a powerful alternative to this problem. In this work we exploit an efficient back-propagation-compatible Bayesian formulation in neural networks, called Bayes-by-Backpop (BBB) Blundell et al. (2015), to obtain weights' uncertainty as a guidance to perform continual learning. We show how lifelong learning using pruning and regularization can be enhanced when accurate weights importance are present to identify task-specific parameters. The main intuition behind is that Bayesian neural network provide precise uncertainty measurements in weights which is the bottleneck for regularization methods.

2 Related Work

Here we list some of the previously introduced approaches taken in lifelong learning that mitigate catastrophic forgetting to different extents. Conceptually, these approaches can be divided into the following categories. We list pros and cons for each and explain why (not) our approach is inspired or build upon them.

Dynamic architectural methods: In this setting the architecture will grow such that it carries over the past knowledge as well as storing new gained knowledge in different forms such as additional layers, nodes, or dual-architecture. In this approach the objective function remains fixed whereas the model capacity grows often exponentially with the number of tasks. Progressive networks Rusu et al. (2016); Schwarz et al. (2018) was one of the earliest work in this path which was successfully applied on reinforcement learning problems where the base architecture is duplicated and lateral connections are added in response to new tasks. Roy et al. (2018) also uses hierarchical network, where CNNs at multiple levels, could grow in tree-shape manner to accommodate new tasks. Dynamically Expandable Network (DEN) Yoon et al. (2018) also expands the network by selecting the *drifting* units and retraining them separately based on the current tasks in hand. Despite the ability of these methods in preventing forgetting, they have large overheads upon arrival of new tasks. We have completely avoided using architectural growth in our method due to this reason.

Memory-based methods: In this regime, previous information is partially stored in different fashions to be used later as a form of *rehearsal* Robins (1995). GeppNet Gepperth & Karaoguz (2016) and its variant GeppNet-STM were introduced in this category for class incremental learning settings. They use a self-organizing map (SOM), a linear regression layer, to re-structure the input data topographically. Once the SOM is trained, it will be only updated when a an input belong to a new class arrive. This example is stored by a short-term memory buffer to prevent forgetting it in the future. iCaRL Rebuffi et al. uses this idea to perform class incremental learning through nearest-mean-of-exemplars using current data and stored examples of previously seen classes. It was shown to be capable of incrementally learning large scale datasets such as ImageNet. However, this method can not be used in learning multiple sequential tasks from different datasets. Gradient episodic memory (GEM) (Lopez-Paz et al., 2017) was another work for class incremental learning where data is stored at the end of each episode to be used later to restrict gradient updates from deviating away from their previous values. GEM is not only able to mitigate forgetting, but also allows for positive backward knowledge transfer which is referred to improvement on previously learned tasks. Another value of this work was that it was the first method capable of learning with seeing each training example only once. Since storing examples can be costly and sometimes unpractical, a generative model was employed to encode/decode pseudo information Draelos et al. (2017); Shin

et al. (2017); Nguyen et al. (2017). However, in practice generative models are difficult to train specially in sequential learning scenarios.

Regularization methods: These approaches do not require storing data while retaining the whole network. Catastrophic forgetting is alleviated via preventing significant changes to representation learned for previous tasks. This can be performed through regularizing the objective function, or directly applied on weight parameters such that important weights to one task are not allowed to change in the future. Various methods have been explored in both weight *importance* measurements as well as regularization techniques. Elastic weight consolidation (EWC) method Kirkpatrick et al. (2017) imposes a quadratic penalty on changes in weights that are found to be important to prior tasks. However, the importance is measured through computing the diagonal of the Fisher information metric over parameters of the old task resulting in inefficiency on high-dimensional output spaces. Synaptic Intelligence (SI) suggested an online way of computing importance which had comparable results to EWC while being more efficient. Similar to SI, Memory-aware Synapses (MAS) Aljundi et al. (2017) proposed an online way of computing importance adaptive to the test set using the change in the model outputs w.r.t the inputs. PackNet Serrà et al. (2018) used iterative pruning to fully restrict gradient updates on important weights via saving binary masks. While applying this binary mask at inference results in zero forgetting of the initial performance, it requires to know which task we are performing to use the appropriate mask. PackNet also ranks the weight importance by their magnitude which is not guaranteed to be a proper importance indicative. A follow-up work to PackNet Mallya & Lazebnik (2018) suggested learning these binary masks end-to-end using a meta-learner.

3 Background: Variational Bayes-by-Backprop

In this section we review the Bayesian framework we use to be able to learn sequential tasks by remembering task-specific parameters. The Bayesian formulation used here was first introduced by Blundell et al. (2015) that learns the probability distribution over network parameters. In the original paper, it was shown that this novel back-propagation-compatible algorithm acts as a regularizer and yields comparable performance to dropout on MNIST dataset. In the next section (Section 4) we show how to use the derived weight uncertainty to measure the importance in performing the task in hand and use the rest of the network to learn later tasks with.

3.1 Variational inference in neural networks

In Bayesian models, latent variables are drawn from a prior density $p(\mathbf{w})$ which are related to the observations through the likelihood $p(x|\mathbf{w})$. During the inference, the posterior distribution $p(\mathbf{w}|\mathbf{x})$ is computed conditioned on the given input data. However, in practice this probability distribution is intractable and is often estimated through approximate inference. Markov chain Monte Carlo (MCMC) sampling (Hastings, 1970) has been widely used and explored for this purpose (see Robert & Casella (2013) for different methods under this category). However, MCMC algorithms, despite providing guarantees in finding asymptotically exact samples from the target distribution, they are not suitable for large datasets and/or large models as they are bounded by speed and scalability issues. Alternatively, Variational inference provides a faster solution to the same problem in which the posterior is approximated using optimization rather than being sampled from a chain Peterson (1987); Hinton & Van Camp (1993); Jaakkola & Jordan (1996; 1997). Similar to any other approximation algorithms, there will be trade-offs between the precision of the results and the efficiency of the optimization. However, variational inference methods always take advantage of fast optimization techniques such as citestochastic methods, distributed methods, which allows them to explore data models faster to find the best approximation. See Blei et al. (2017) for a complete review of the theory.

3.2 Bayes by Backprop

Let $\mathbf{x} \in \mathbb{R}^n$ be a set of observed variables and $\mathbf{w}$ be a set of latent variables parametrized by θ sharing a joint density of $p(\mathbf{w}, \mathbf{x})$. Given the observations $\mathbf{x}$ and by using the set of weight parameters $\mathbf{w}$, a neural network, as a probabilistic model, $p(\mathbf{y}|\mathbf{x}, \mathbf{w})$ can output a probability associated with a possible output $\mathbf{y} \in \mathcal{Y}$. Variational inference aims to calculate this conditional probability distribu-

tion over the latent variables by finding the closest proxy to the exact posterior through solving an optimization problem.

We first propose a family $\mathcal{D}$ of probability densities over the latent variables ($q(\mathbf{w}|\theta) \in \mathcal{D}$). We then find the closest member of this family to the true conditional probability of interest $p(\mathbf{w}|\mathbf{x})$ by minimizing the Kullback-Leibler (KL) divergence between q and p:

$$q^*(\mathbf{w}|\theta) = \underset{q(\mathbf{w}|x)\in\mathcal{D}}{\arg\min}\ \mathrm{KL}\big(q(\mathbf{w}|\theta)\|p(\mathbf{w}|\mathbf{x})\big) \tag{1}$$

Once solved, $q(\cdot)$ would be the closest approximation to variational posterior parametrized by θ^* to construct the parameters of the network. We can unpack Eq. 1 and call it a loss function commonly known as variational free energy or expected lower bound :

$$\mathcal{L}_{BBB}(\mathbf{w},\theta) = \mathrm{KL}\big[q(\mathbf{w}|\theta)\|p(\mathbf{w})\big] - \mathbb{E}_{q(\mathbf{w}|\theta)}\big[\log(p(\mathbf{x}|\mathbf{w})\big] \tag{2}$$

Blundell et al. (2015) showed that Eq. 2 can be approximated using i Monte Carlo samples from the variational posterior:

$$\mathcal{L}_{BBB}(\mathbf{w},\theta) \approx \sum_{i=1}^{n} \log q(\mathbf{w^{(i)}}|\theta) - \log \mathbf{p}(\mathbf{w^{(i)}}) - \log p(\mathbf{x}|\mathbf{w}^{(i)}) \tag{3}$$

We assume $q(\mathbf{w}|\theta)$ to have a Gaussian pdf with diagonal covariance. θ is parametrized point-wise with a mean μ and standard deviation of $\sigma = \log(1+\exp(\rho))$. Variational posterior parameters will have a form of $\mu + \log(1+\exp(\rho)) \circ \epsilon$ where ϵ is a sample drawn from posited q. For the prior, as it was suggested in Blundell et al. (2015), a scale mixture of two Gaussian pdfs are chosen which are zero-centered while having difference variances of σ_1^2 and σ_2^2. The uncertainty obtained for every weight parameter, as opposed to hidden units Rezende et al. (2014); Kingma & Welling (2013) has been successfully used in model compression Blundell et al. (2015), uncertainty-based exploration in reinforcement learning application Blundell et al. (2015); Lipton et al. (2016). In this work we want to use this framework to learn sequential tasks without forgetting using per-weight derived uncertainties.

4 Lifelong learning using Bayes by Backprop (BLLL)

Unlike regular deep neural networks, in BBB model, weight parameters are represented by probability distributions parametrized by their mean and standard deviation. In order to take into account both mean and variance we use the quantity of signal-to-noise ratio (SNR) for each parameter defined as

$$\mathrm{SNR} = \frac{|\mu|}{\sigma} \tag{4}$$

SNR is a commonly used measure in signal processing to distinguish between "useful" information from unwanted noise contained in a signal. In the context of neural models, SNR can be thought as an indicative of parameter importance; the higher the SNR, the more effective or important the parameter is to the model predictions for a given task.

In our method called, Bayesian Lifelong Learning (BLLL), we have considered two common scenarios of lifelong learning: 1) sequential learning of multiple datasets; 2) class-incrementally learning of a single or two similar datasets together. We perform continual learning using weight pruning (BLLL-PR), and adaptive scaled gradient update method based upon the Bayesian uncertainty defined per weight parameters.

4.1 BLLL without forgetting using weight pruning (BLLL-PR)

Weight pruning is a long standing solution to reduce inference computation or network compression (Liu et al., 2017; Molchanov et al., 2016). Moreover, it has been recently used to perform lifelong learning (Mallya & Lazebnik, 2017), where the goal is to sequentially learning multiple tasks using a single network's capacity. In (Mallya & Lazebnik, 2017) this has been accomplished by freeing up parameters deemed to be unimportant to the current task according to their magnitude. Forgetting is prevented in pruning by saving a task-specific binary mask of important vs. unimportant parameters.

Algorithm 1 Bayesian Lifelong Learning with Zero Forgetting (`BLLL-PR`) or Reduced Forgetting (`BLLL-LR`)

Require: hyper parameters for BBB-only: σ, σ_1, σ_2, ρ, μ, π, n
Require: hyper parameters for training: learning rate α

$\epsilon \sim \mathcal{N}(0, I)$
$\sigma = \log(1 + \exp(\rho))$ ▷ Ensures σ is always positive
$\mathbf{w} = t(\theta, \epsilon) = \mu + \sigma \circ \epsilon$ ▷ $\mathbf{w}$:= a posterior sample of weights
while not done **do**
  $l_1 \leftarrow \sum_{i=1}^{n} \log \mathcal{N}(\mathbf{w}_i | \mu, \sigma^2)$ ▷ log-posterior
  $l_2 \leftarrow \sum_{i=1}^{n} \log \left(\pi \mathcal{N}(\mathbf{w}_i \mid 0, \sigma_1^2) + (1-\pi)\mathcal{N}(\mathbf{w}_i \mid 0, \sigma_2^2)\right)$ ▷ log prior
  $l_3 \leftarrow \sum_{i=1}^{n} \log(p(x|\mathbf{w}))$ ▷ log-likelihood of the data
  $\mathcal{L}_{BBB} \leftarrow \frac{1}{M}(l_1 - l_2) - l_3$ ▷ M := number of minibatches
  if BLLL-PR **then**
    $\mathrm{SNR}_i \leftarrow \frac{|\mu_i|}{\sigma_i}$ ▷ signal-to-noise ratio for every weight
    for all layers **do**
      for all parameters in layer **do**
        if $\mathrm{SNR}_i \in$ Top p portion of SNR values in layer **then**
          $\alpha = 0$
    $\mu' \leftarrow \mu - \alpha \nabla \mathcal{L}_{BBB_\mu}$ ▷ Update parameter using gradient descent
    $\rho' \leftarrow \rho - \alpha \nabla \mathcal{L}_{BBB_\rho}$ ▷ Update parameter using gradient descent
  if BLLL-LR **then**
    $\beta \leftarrow \sigma \times \alpha$
    $\mu' \leftarrow \mu - \beta \nabla \mathcal{L}_{BBB_\mu}$
    $\rho' \leftarrow \rho - \alpha \nabla \mathcal{L}_{BBB_\rho}$ ▷ Update the remaining uniformly using gradient descent

Our work is significantly different from (Mallya & Lazebnik, 2017) in the different criterion we propose to use for importance measurement; the statistically-grounded uncertainty defined in Bayesian neural networks.

BLLL-pruning is performed as follows: for every layer, convolutional or fully-connected, the parameters are ordered by their SNR value and those with the lowest importance are pruned (set to zero). The pruned parameters, are "saved" using a binary mask so that they can be used later in learning new tasks whereas the important parameters remain fixed throughout training on future tasks. Once a task is learned, an associated binary mask is saved which will be used at inference to recover key parameters to the desired task. The overhead memory caused by saving the binary mask is negligible given the fact it completely eliminates the forgetting by recovering the exact performance achieved on a previously learned task.

4.2 BLLL WITH REDUCED FORGETTING

Another strategy to perform continual learning is to reduce forgetting by controlling further changes in a representation learned for a task. This can be enforced in a form of regularization in the objective function or through a varying learning rate for weight parameters.

Adaptive learning rate (`BLLL-LR`): In order to preven drastic changes to the predicted posterior for a learned task, we propose to condition the changes in the mean of each parameter's distribution on its uncertainty i.e., once a task is learned, the more certain a mean of a distribution is, the less it should be updated when a new task arrives. We impose this through means of the parameters distributions, μ_i, to be adaptive to their uncertainty represented by σ_i. The key benefit of the learning rate-based Bayesian continual learning is that it neither requires additional memory, as opposed to pruning technique, nor tracking the change in parameters with respect to the previous learned task, as needed in the weight regularization methods. More importantly, this method does not need to be aware of task switching as it only needs to adjust the learning rates of the means in the posterior distribution based on their current standard deviations. Complete algorithm for BLLL with pruning (`BLLL-PR`), regularization (`BLLL-Reg`) and, adaptive learning rate (`BLLL-LR`). is shown in Algorithm 1.

5 EXPERIMENTAL SETUP

Datasets: Toy benchmarks such as MNIST(LeCun et al., 1998) and CIFAR10/100 (Krizhevsky & Hinton, 2009) have been widely used to measure forgetting and accuracy of various lifelong learning approaches. As a result, we include them as part of our evaluation. We use MNIST split and permuted MNIST (Srivastava et al., 2013) for class incremental learning with similar experimental settings used in Serrà et al. (2018); Lopez-Paz et al. (2017) in terms of data splits. Furthermore, to have a better understanding of our method, we also evaluate our approach on sequentially learning of various datasets belonging to different distributions similar to the sequence of 8 tasks used in (Serrà et al., 2018) including FaceScrub (Ng & Winkler, 2014), MNIST, CIFAR100, Not NotMNIST (not), SVHN (Netzer et al., 2011), CIFAR10, TrafficSigns (Stallkamp et al., 2011), and FashionMNIST (Xiao et al., 2017). Details of each are summarized in Table 1. In all the experiments we resized images to $32 \times 32 \times 3$ if necessary. For datasets with monochromatic images, we replicate the image across all RGB channels. No data augmentation of any kind has been used in our analysis.

Table 1: Utilized datasets summary

Names	#Classes	Train	Test
FaceScrub (Ng & Winkler, 2014)	100	20,600	2,289
MNIST (LeCun et al., 1998)	10	60,000	10,000
CIFAR100 (Krizhevsky & Hinton, 2009)	100	50,000	10,000
NotMNIST (not)	10	16,853	1,873
SVHN (Netzer et al., 2011)	100	73,257	26,032
CIFAR10 (Krizhevsky & Hinton, 2009)	10	39,209	12,630
TrafficSigns (Stallkamp et al., 2011)	43	39,209	12,630
FashionMNIST (Xiao et al., 2017)	10	60,000	10,000

Baselines: Within Bayesian framework, we have three reference baselines of fine-tuning, feature extraction, and joint training. In fine-tuning (`BLLL-FT`), training with regular SGD optimization continues upon arrival of new tasks without any forgetting avoidance strategy. Feature extraction, denoted as (`BLLL-FE`) in our experiments, refers to freezing all layers in the network except for the last layer when a new task arrives. In joint training (`BLLL-JT`) we learn all the tasks in a multi-task learning fashion which serves as the upper bound for average accuracy on all tasks. We have compared these three Bayesian references with their counterparts using an ordinary (non-Bayesian) network referred to as `ORD-FT`, `ORD-FE`, and `ORD-JT`. We have compared them on Split MNIST class incremental learning experiments. Along with the BLLL variants of reference baselines, we compare with the state-of-the-art approaches including EWC Kirkpatrick et al. (2017), IMM (Lee et al., 2017), learning without forgetting (LWF) (Li & Hoiem, 2017), less-forgetting learning (LFL) (Jung et al., 2016), PathNet (Fernando et al., 2017), PNNs (Rusu et al., 2016), and Hard attenstion mask (HAT) (Serrà et al., 2018) using implementations provided in (Serrà et al., 2018). On Permuted MNIST dataset average accuracies are directly reported from GEM (Lopez-Paz et al., 2017), SI (Zenke et al., 2017), and VCL (Nguyen et al., 2017) without re-implementation.

Network architecture: For Split MNIST and Permuted MNIST experiments we have used a multilayer perceptron with three hidden layers with 660 units whereas other baselines use a two-layer MLP with 800 units in each. Because there exists double number of parameters in our bayesian network compared to its equivalent regular neural net, we ensured fair comparison by matching the total number of parameters between the two to be 1.9M unless otherwise is stated. For multiple datasets learning scenario as well as alternating incremental CIFAR10/100 datasets we have used a ResNet18 Bayesian network with 7.1-11.3M parameters depending on the experiment. Note that the majority of the baselines provided in this work were originally developed using some variants of AlexNet structure. We tried change that to ResNet type of network, however, it resulted in degrade in their reported and experimented performance. On the other hand, best results for our Bayesian network were achieved using ResNet18. Therefore, we kept the architectures as is and only matched their number of parameters to ensure having equal capacity across different approaches.

Hyperparameter tuning: We have used the approach introduced in (Anonymous, 2019) specific to continual learning setting for hyperparameter tunning. Unlike commonly used tuning techniques which use a validation set composed of all the classes in the dataset, this method realistically assumes to have access to a few classes only on which we can tune the hyperparameters on a held-out

validation set. In all our experiments we consider 0.15 split for validation set on the first two tasks in all the learning sequences.

Training details: It is important to note that in all our experiments, no pre-trained model is used. We used stochastic gradient descent with learning rate of 0.01 decaying by a factor of 0.3 upon being plateau for 5 consecutive epochs. Batch size of 64 was consistently used throughout all experiments. Dataset splits and batch shuffle has been performed identically across BLLL experiments and all the baselines.

Performance measurement: Let n be the total number of tasks. Once all are learned, we evaluate our model on all n tasks. ACC is the average test classification accuracy across all tasks. To measure forgetting we report backward transfer, BWT, which indicates how much learning new tasks has influenced the performance on previous tasks. While $\text{BWT} < 0$ directly reports *catastrophic forgetting*, $\text{BWT} > 0$ indicates that learning new tasks has helped with the preceding tasks. In mathematical notation BWT and ACC are as follows:

$$\text{BWT} = \frac{1}{n-1}\sum_{i=1}^{n-1} R_{i,n} - R_{i,i}, \quad \text{ACC} = \frac{1}{n}\sum_{i=1}^{n} R_{i,n} \tag{5}$$

where $R_{i,n}$ is the test classification accuracy on task i after sequentially finishing learning the nth task.

6 Results & Discussion

We present our results for class incremental learning of single datasets (Split MNIST and Permuted MNIST) as well as sequential learning of 8 distinct tasks using variant of BLLL approach compared against various recent lifelong learning approaches.

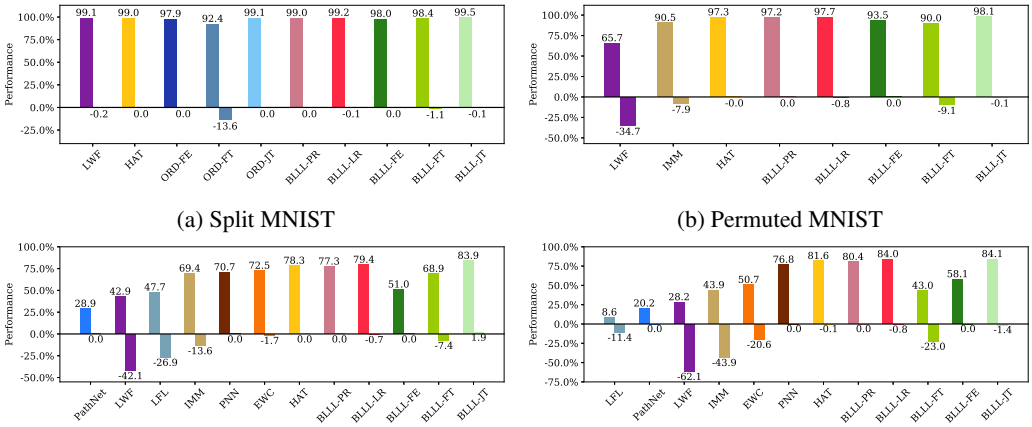
6.1 Split MNIST

We first present our results for class incremental learning of MNIST (Split MNIST) in which we learn $0-9$ digits in two tasks with 5 randomly shuffled classes at a time. All presented methods in this setup have 1.9M parameters. Figure 1a shows the results for reference baselines in Bayesian and non-Bayesian networks including fine-tuning (`BLLL-FT`,`ORD-FT`), feature extraction (`BLLL-FE`,`ORD-FE`) and, joint training (`BLLL-JT`,`ORD-JT`) averaged over 5 runs. Although MNIST dataset is too easy to draw broad conclusions on it, we observe throughout all experiments that Bayesian fine-tuning and joint training, perform significantly better than their counterparts `ORD-FT` and `ORD-JT` that use a regular network where forgetting in `BLLL-FT` is comparable and even better than some of the baselines presented in literature. We also compare against HAT (Serrà et al., 2018) and LWF (Li & Hoiem, 2017) where HAT has similar performance to our `BLLL-PR` with zero forgetting while `BLLL-LR` is able to increase the ACC by 2% at the cost of 0.1% drop in forgetting.

Figure 2a shows how the accuracy carries from the first half of the dataset to the second half. Ordinary fine-tuning is the only reference method that experience a performance boos upon change in the task whereas both `BLLL-FE` and `ORD-FE` degrade after new task arrives which is fully expected. All other baselines as well as our BLLL invariants steadily perform above $\approx 99\%$ on both tasks.

6.2 Permuted MNIST

Permuted MNIST is a popular variant of MNIST dataset to evaluate continual learning approaches in which each task is considered as a random permutation of the original MNIST pixels. Following the literature, we learn a sequence of 10 random permutations and report average accuracy at the end. Figure 1b shows ACC and BWT performance of `BLLL-LR` and `BLLL-PR` against that of the state of the art models. We can again observe that `BLLL-FT`, despite having no intention in preventing forgetting, exhibits reasonable negative BWT values which makes it a successor over IMM (Lee et al., 2017) and LWF (Li & Hoiem, 2017) baselines. While HAT (Serrà et al., 2018) and `BLLL-PR` have a similar ACC performance with no forgetting, `BLLL-LR` shows a slight increase in the tendency to forget (-0.8%) in exchange for a negligible boost in ACC (0.4%). Accuracy evolution over 10 tasks is shown in 2b.



(a) Split MNIST

(b) Permuted MNIST

(c) Alternating Incremental CIFAR10/100

(d) Multiple datasets (8 tasks)

Figure 1: ACC and BWT on (a) Split MNIST (2 tasks), (b) Permuted MNIST (10 tasks), (c) Alternating incremental CIFAR10/100 (10 tasks), (d) 8 consecutive tasks including FaceScrub (Ng & Winkler, 2014), MNIST (LeCun et al., 1998), CIFAR100 (Krizhevsky & Hinton, 2009), Not NotMNIST (not), SVHN (Netzer et al., 2011), CIFAR10, TrafficSigns (Stallkamp et al., 2011), and FashionMNIST (Xiao et al., 2017)

Results in Fig. 1b are computed on a $1.9\mathrm{M}$ network. However ACC for this experiment is also reported in HAT (Serrà et al., 2018), VCL (Nguyen et al., 2017), EWC (Kirkpatrick et al., 2017), SI (Zenke et al., 2017), and GEM(Lopez-Paz et al., 2017) using a smaller MLP with $0.1\mathrm{M}$ parameters. ACC for these approaches are given as 91.0%, 90%, 88.2%, 86.0%, 82.6%, respectively. However, `BLLL-LR` achieves 92.2% with BWT of -0.4% under the same conditions.

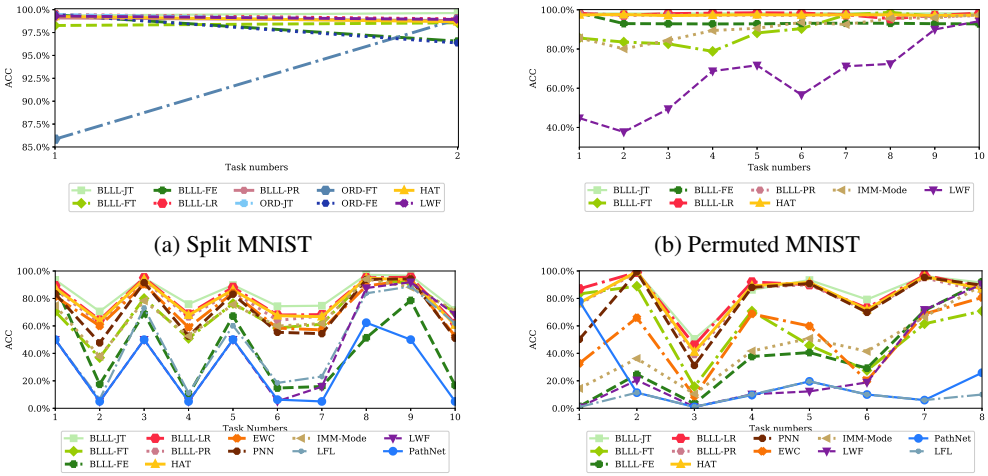
6.3 Alternating CIFAR10 and CIFAR100

In this experiment we alternate between class incrementally learning of CIFAR10 and CIFAR100. Both datasets are divided into 5 tasks each with 2 and 20 classes per task, respectively. Figure 1c presents ACC and BWT obtained with `BLLL-PR`, `BLLL-LR`, and three `BLLL` reference methods compared against various continual learning baselines. It is interesting to note that in this setup, some baselines (LWF and PathNet) do not perform better than the naive accuracy achieved by feature extraction . LFL is able to marginally outperform feature extraction in ACC at the cost of twice of the forgetting we can have in fine-tuning while IMM performs only as well as fine-tuning in both ACC and BWT. PNN, EWC, and HAT are the only baselines that perform between the minimum expected ACC (`BLLL-FE`) and maximum acceptable negative BWT (`BLLL-FT`) bounded by the high ACC and near zero forgetting achieved in joint training (`BLLL-JT`) for this setup.

Among the three acceptable baselines, PNN is the only zero-forgetting-guaranteed approach and hence can be properly compared with our `BLLL-PR` which also never forgets by definition. Note that PathNet is also a zero-forgetting approach which suffers from bad pre-assignment of network's capacity per task which causes poor performance on the beginning task from which it never recovers. `BLLL-PR` consistently outperforms PNN by overall 10% better ACC where the gap between them is more pronounced on CIFAR100 tasks which are more difficult compared to binary classification tasks of CIFAR10. EWC and HAT are both allowed to forget by construction, however HAT shows zero forgetting behavior. While EWC is outperformed by both of our BLLL variants, HAT exhibits 1% better ACC over `BLLL-PR`. `BLLL-LR` is able to compensate for that at a cost of 0.7% drop in forgetting. A trade off between ACC and BWT exists when choosing between `BLLL-PR` and `BLLL-LR` depending on what is needed in the problem.

6.4 Multiple datasets learning

Finally we present our results for sequentially learning of 8 tasks using `BLLL-PR` and `BLLL-LR`. Similar to the previous experiments we look at both ACC and BWT obtained for `BLLL-PR`,



(a) Split MNIST

(b) Permuted MNIST

(c) Alternating incremental CIFAR10/CIFAR100

(d) Multiple datasets

Figure 2: Evolution of the test classification accuracy obtained across tasks when all tasks are learning

BLLL−LR, BLLL references (FT,FE,JT) as well as various baselines. Considering the ACC achieved by BLLL−FE or BLLL−FT (58.1%) as a lower bound we observe again that some baselines are not able to do better than BLLL−FT including LFL, PathNet, LWF, IMM, and EWC. PNN and HAT remain the only considerable baselines for our BLLL−PR and BLLL−LR approaches. BLLL−PR continues to outperform PNN by 5% in ACC while the two ensure zero forgetting. HAT exhibits -0.1% BWT while being overcome by BLLL−LR in ACC by 3%.

We believe the main insight of our Bayesian approach is that instead of computing extra measurements of importance, which are often task, input or output dependent, we are able to use the predicted weights uncertainty as a bi-product of the framework to deem the most contributing parameters in the overall posterior distribution and choose to freeze them using a binary mask, as in BLLL−PR, or to enforce minimum changes in them conditioned on their current uncertainty, as in BLLL−LR, allowing for more capacity to learn with and most importantly developing a task-agnostic model that does not require to be warned of task arrival. This makes our approach an excellent candidate for applications in which such information is inevitably unavailable.

7 Conclusion

In this work we propose a lifelong learning formulation for Bayesian networks, called B−LLL, that uses the Bayesian-embedded uncertainty predictions as a guidance to perform lifelong learning; i.e. to identify important parameters that can be either fully preserved through a saved binary mask (BLLL−PR) or allowed to change conditioned on their uncertainty for learning new tasks (BLLL−LR). We demonstrated how the probabilistic uncertainty distributions per weights are helpful in sequentially learning short and long sequences of benchmark datasets compared against many other approaches. We showed that both BLLL−PR and BLLL−LR perform superior or marginally better than the state-of-the-art models across all the experiments. In order to choose between the two one should know that BLLL−PR results in zero forgetting while requiring a small binary mask to be saved per task whereas BLLL−LR is memory-free and allows for more capacity in the network by allowing small forgetting to occur. The key benefit from BLLL−PR is that the model no longer need to know when tasks are being switched leading to a task-agnostic model that can be deployed where it is not possible to distinguish tasks in a continuous stream of the data.

REFERENCES

Notmnist dataset. technical report.

Alireza Aghasi, Afshin Abdi, Nam Nguyen, and Justin Romberg. Net-trim: Convex pruning of deep neural networks with performance guarantee. In *Advances in Neural Information Processing Systems*, pp. 3180–3189, 2017.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. *arXiv preprint arXiv:1711.09601*, 2017.

Anonymous. Efficient lifelong learning with a-gem. In *Submitted to International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Hkf2_sC5FX`. under review.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Timothy J Draelos, Nadine E Miner, Christopher C Lamb, Jonathan A Cox, Craig M Vineyard, Kristofor D Carlson, William M Severa, Conrad D James, and James B Aimone. Neurogenesis deep learning: Extending deep networks to accommodate new classes. In *Neural Networks (IJCNN), 2017 International Joint Conference on*, pp. 526–533. IEEE, 2017.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

Alexander Gepperth and Cem Karaoguz. A bio-inspired incremental learning architecture for applied perceptual problems. *Cognitive Computation*, 8(5):924–934, 2016.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pp. 5–13. ACM, 1993.

Tommi Jaakkola and Michael Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, volume 82, pp. 4, 1997.

Tommi S Jaakkola and Michael I Jordan. Computing upper and lower bounds on likelihoods in intractable networks. In *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*, pp. 340–348. Morgan Kaufmann Publishers Inc., 1996.

Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, pp. 201611835, 2017.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in Neural Information Processing Systems*, pp. 4652–4662, 2017.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Zachary C Lipton, Jianfeng Gao, Lihong Li, Xiujun Li, Faisal Ahmed, and Li Deng. Efficient exploration for dialogue policy learning with bbq networks & replay buffer spiking. *arXiv preprint arXiv:1608.05081*, 2016.

Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 806–814, 2015.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 2755–2763. IEEE, 2017.

David Lopez-Paz et al. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. *arXiv preprint arXiv:1711.05769*, 1(2):3, 2017.

Arun Mallya and Svetlana Lazebnik. Piggyback: Adding multiple tasks to a single, fixed network by learning to mask. *arXiv preprint arXiv:1801.06519*, 2018.

James L McClelland, Bruce L McNaughton, and Randall C O'reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient transfer learning. *CoRR, abs/1611.06440*, 2016.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.

Hong-Wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pp. 343–347. IEEE, 2014.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

Carsten Peterson. A mean field theory learning algorithm for neural networks. *Complex systems*, 1:995–1019, 1987.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Christian Robert and George Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

Deboleena Roy, Priyadarshini Panda, and Kaushik Roy. Tree-cnn: A deep convolutional neural network for lifelong learning. *arXiv preprint arXiv:1802.05800*, 2018.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018.

Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *arXiv preprint arXiv:1801.01423*, 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pp. 2990–2999, 2017.

Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In *Advances in neural information processing systems*, pp. 2310–2318, 2013.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pp. 1453–1460. IEEE, 2011.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks, 2018.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *arXiv preprint arXiv:1703.04200*, 2017.