ENERGY-BASED MODELS FOR CONTINUAL LEARNING

Shuang Li *
MIT CSAIL
lishuang@mit.edu

Yilun Du
MIT CSAIL
yilundu@mit.edu

Gido M.van de Ven
Baylor College of Medicine
ven@bcm.edu

Igor Mordatch
Google Brain
igor.mordatch@gmail.com

ABSTRACT

We motivate Energy-Based Models (EBMs) as a promising model class for continual learning problems. Instead of tackling continual learning via the use of external memory, growing models, or regularization, EBMs change the underlying training objective to cause less interference with previously learned information. Our proposed version of EBMs for continual learning is simple, efficient, and outperforms baseline methods by a large margin on several benchmarks. Moreover, our proposed contrastive divergence-based training objective can be combined with other continual learning methods, resulting in substantial boosts in their performance. We further show that EBMs are adaptable to a more general continual learning setting where the data distribution changes without the notion of explicitly delineated tasks. These observations point towards EBMs as a useful building block for future continual learning methods. Project page: https://energy-based-model.github.io/Energy-Based-Models-for-Continual-Learning/.

1 INTRODUCTION

Humans are able to rapidly learn new skills and continuously integrate them with prior knowledge. The field of Continual Learning (CL) seeks to build artificial agents with the same capabilities (Parisi et al., 2019; Hadsell et al., 2020; De Lange et al., 2021). In recent years, continual learning has seen increased attention, particularly in the context of classification problems. Continual learning requires models to remember prior skills as well as incrementally learn new skills, without necessarily having a notion of an explicit task identity. Standard neural networks (He et al., 2016; Simonyan & Zisserman, 2014; Szegedy et al., 2015) experience catastrophic forgetting and perform poorly in this setting. Different approaches have been proposed to mitigate catastrophic forgetting, but many rely on the usage of external memory (Lopez-Paz & Ranzato, 2017; Li & Hoiem, 2017; Hayes et al., 2020; Buzzega et al., 2020), additional models (Shin et al., 2017; von Oswald et al., 2019; Liu et al., 2020), or auxiliary objectives and regularization (Kirkpatrick et al., 2017; Schwarz et al., 2018; Maltoni & Lomonaco, 2019; Zenke et al., 2017), which can constrain the wide applicability of these methods.

In this work, we propose a new approach towards continual learning on classification tasks. Most existing CL approaches tackle these tasks by utilizing normalized probability distribution (i.e., softmax output layer) and trained with a cross-entropy objective. In this paper, we argue that by viewing classification from the lens of training an un-normalized probability distribution, we can significantly improve continual learning performance in classification problems. In particular, we interpret classification as learning an Energy-Based Model (EBM) across classes. Training becomes a wake-sleep process, where the energy of an input data at its ground truth label is decreased while the energy of the input at (an)other selected class(es) is increased. An important advantage is that this framework offers freedom to choose what classes to update in the continual learning process. By contrast, the cross entropy objective reduces the likelihood of *all* negative classes when given a new input, creating updates that lead to catastrophic forgetting.

The energy function, which maps an input-label pair to a scalar energy, also provides a way for the model to select and filter portions of the input that are relevant towards the classification on hand. This enables EBMs training updates for new data to interfere less with previous data. In particular, our formulation of the energy function allows us to compute the energy of an input by learning a conditional gain based on the class label, which serves as an attention filter to select the most relevant information. In the event of a new class, a new conditional gain can be learned.

* Correspondence to: Shuang Li <lishuang@mit.edu>

These unique properties benefit EBMs in addressing two important open challenges in continual learning. 1) First, we show that EBMs are promising for class-incremental learning, which is one of the most challenging settings for continual learning (van de Ven & Tolias, 2019; Hayes & Kanan, 2020; Masana et al., 2020; Prabhu et al., 2020). Generally, successful existing methods for class-incremental learning store data, use generative replay, or pretrain their model on another dataset, which has disadvantages in terms of memory and/or computational efficiency. We show that EBMs perform well in class-incremental learning without using replay and without relying on stored data. 2) The second open challenge that EBMs can address is continual learning without task boundaries (Aljundi et al., 2019a; Lee et al., 2020; Jin et al., 2020). Typically, a continual learning problem is set up as a sequence of distinct tasks with clear boundaries that are known to the model (the *boundary-aware* setting). Most existing continual learning methods rely on these known boundaries for performing certain consolidation steps (e.g., calculating parameter importance, updating a stored copy of the model). However, assuming such clear boundaries is not always realistic, and often a more natural scenario is the *boundary-free* setting (Zeno et al., 2018; Rajasegaran et al., 2020), in which data distributions gradually change without a clear notion of task boundaries. While many common CL methods cannot be used without clear task boundaries, we show that EBMs can be naturally applied to this more challenging setting.

There are three main contributions of our work. First, we introduce energy-based models for classification continual learning problems. We show that EBMs can naturally deal with challenging problems in CL, including the boundary-free setting and class-incremental learning without using replay. Secondly, we propose an energy-based training objective that is *simple* and broadly applicable to different types of models, with significant boosts on their performance. This contrastive divergence based training objective can naturally handle the dynamically growing number of classes and significantly reduces catastrophic forgetting. Lastly, we show that the proposed EBMs perform strongly on four standard CL benchmarks: split MNIST, permuted MNIST, CIFAR-10, and CIFAR-100 without using extra generative model or storing data. These observations point towards EBMs as a class of models naturally inclined towards the CL regime and as an important new baseline upon which to build further developments.

2 Related Work

2.1 Continual learning settings

Boundary-aware versus boundary-free during training. In most existing CL studies, models are trained in a *boundary-aware* setting, in which a sequence of distinct tasks with clear task boundaries is given (*e.g.*, (Kirkpatrick et al., 2017; Zenke et al., 2017; Shin et al., 2017)). There are no overlaps between any two tasks. Models are first trained on the first task and then move to the second one. Moreover, models are typically told when there is a transition from one task to the next. However, it could be argued that it is more realistic for tasks to change gradually and for models to not be explicitly informed about the task boundaries. Such a *boundary-free* setting has been explored in (Zeno et al., 2018; Rajasegaran et al., 2020; Aljundi et al., 2019b; Madireddy et al., 2020). In this setting, models learn in a streaming fashion and the data distributions gradually change over time. Importantly, most existing CL methods are not applicable to this setting as they require the task boundaries to decide when to perform certain consolidation steps. In this paper, we show that our method can naturally handle both the boundary-aware and boundary-free settings.

Task-incremental versus class-incremental learning. Another important distinction in CL is between task-incremental learning (*Task-IL*) and class-incremental learning (*Class-IL*) (van de Ven & Tolias, 2019; Prabhu et al., 2020; Belouadah et al., 2020; Maltoni & Lomonaco, 2019; Hayes & Kanan, 2020). In *Task-IL*, also referred to as the multi-head setting (Farquhar & Gal, 2018), models predict the label of an input data by choosing only from the labels in the task where the data come from. In *Class-IL*, also referred to as the single-head setting, models chose between the classes from all tasks so far when asked to predict the label of an input data. *Class-IL* is more challenging than *Task-IL* as it requires models to select the correct labels from the mixture of new and old classes.

In some previous works, class-incremental learning only refers to the boundary-aware setting where the tasks have clear boundaries. In our paper, we use class-incremental learning in a more general way, allowing it to contain both boundary-aware and boundary-free settings, where the tasks do not necessarily have clear boundaries.

2.2 Continual learning approaches

Task-specific methods. One way to reduce interference between tasks is by using different parts of a neural network for different tasks (Fernando et al., 2017; Serra et al., 2018; Masse et al., 2018; Zeng et al., 2019; Hu et al., 2019; Wortsman et al., 2020). Other methods let models grow or recruit new resources when learning new tasks (Rusu et al., 2016; Yoon et al., 2017; Vogelstein et al., 2020). Although these methods are generally successful in reducing catastrophic forgetting, a key disadvantage is that they require knowledge of task identities during training and testing. Unless they are combined with a mechanism for task-inference, they are not suitable for *Class-IL*.

Regularization-based methods. Regularization is used in CL to encourage the stability of those aspects of the network that are important for previous tasks (Li & Hoiem, 2017; Zenke et al., 2017; Nguyen et al., 2017; Titsias et al., 2020; Kolouri et al., 2020; Kao et al., 2021). A popular strategy is to add a regularization loss to penalise changes of important parameters. For example, EWC (Kirkpatrick et al., 2017) and online EWC (Schwarz et al., 2018) evaluate the importance of parameters based on the fisher information matrices. A disadvantage of regularization-based methods is that typically they gradually reduce the model's capacity for learning new tasks. Moreover, while in theory these methods can be used for *Class-IL*, in practice they have been shown to fail on such problems (Farquhar & Gal, 2018; van de Ven & Tolias, 2019).

Replay methods. To preserve knowledge, replay methods periodically rehearse previous information during training (Robins, 1995; Rebuffi et al., 2017; Aljundi et al., 2019a; Chaudhry et al., 2019b). Exact or experience replay based methods store data from previous tasks and revisit them when training on new tasks. Although straightforward, such methods face critical non-trivial questions, such as how to select the data to be stored and how to use them (Lopez-Paz & Ranzato, 2017; Hou et al., 2019; Wu et al., 2019; Chaudhry et al., 2019a; Mundt et al., 2020; Pan et al., 2020). An alternative is to generate the replayed data (Shin et al., 2017). While both types of replay can mitigate forgetting, an important disadvantage is that they are computationally relatively expensive. Additionally, storing data might not always be possible while incrementally training a generative model is a challenging problem in itself (Lesort et al., 2019; van de Ven et al., 2020).

In contrast, we propose EBMs for CL that reduce catastrophic forgetting without requiring knowledge of task-identity, without restricting the model's learning capabilities, and without using stored data.

3 Rethinking Existing Continual Learning

The most common way to do classification with deep neural networks is to use a softmax output layer in combination with a cross-entropy loss. In continual learning, most existing methods for classification are based on the softmax-based classifier (SBC).

3.1 Softmax-based classification

Given an input $\mathbf{x} \in \mathbb{R}^D$ and a discrete set $\mathcal{Y} = \{1, \ldots, N\}$ of N possible class labels, a traditional softmax-based classifier defines the conditional probabilities of those labels as:

$$p_{\boldsymbol{\theta}}(y|\mathbf{x}) = \frac{\exp([f_{\boldsymbol{\theta}}(\mathbf{x})]_y)}{\sum_{i \in \mathcal{Y}} \exp([f_{\boldsymbol{\theta}}(\mathbf{x})]_i)}, \quad \text{for all } y \in \mathcal{Y}, \tag{1}$$

where $f_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^N$ is a feed-forward neural network, parameterized by $\boldsymbol{\theta}$, that maps an input $\mathbf{x}$ to a N-dimensional vector of logits. $[\cdot]_i$ indicates the i^{th} element of a vector.

Training. A softmax-based classifier is typically trained by optimizing the cross-entropy loss function. For a given input $\mathbf{x}$ and corresponding ground truth label y^+, the cross-entropy loss is $\mathcal{L}_{CE}(\boldsymbol{\theta}; \mathbf{x}, y^+) = -\log(p_{\boldsymbol{\theta}}(y^+|\mathbf{x}))$. A schema of SBC is shown in the left part of Figure 1.

Inference. Given an input $\mathbf{x}$, the class label predicted by the softmax-based classifier is the class with the largest conditional probability $\hat{y} = \arg\max_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\mathbf{x})$.

3.2 Why are softmax-based classifiers not the best way to do continual learning?

When used for continual learning, and in particular when used for class-incremental learning, softmax-based classifiers face several challenges. While the cross-entropy loss is designed for i.i.d. datapoints and labels, the data found in continual learning does not follow such a distribution. As a result, when training on a new task, the likelihood of the currently observed classes is increased, but the likelihood of old classes is too heavily suppressed since they are not encountered in the new task. The softmax operation itself introduces competitive, winner-take-all dynamics that make the classifier catastrophically forget past tasks. We show such phenomenon of SBC in Section 5.1.4.

4 Continual learning with Energy-Based Models

In this section, we propose a simple but efficient energy-based training objective that can successfully mitigate the catastrophically forgetting in CL. We first introduce EBMs in section 4.1 and then show how EBMs are used for classification in section 4.2 and continual learning in sections 4.3.

4.1 ENERGY-BASED MODELS

EBMs (LeCun et al., 2006) are a class of maximum likelihood models that define the likelihood of a data point $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^D$ using the Boltzmann distribution with the partition function $Z(\boldsymbol{\theta})$:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{\exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))}{Z(\theta)}, \quad Z(\boldsymbol{\theta}) = \int_{\mathbf{x} \in \mathcal{X}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x})) \tag{2}$$

where $E_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}$, known as the energy function, maps each data point $\mathbf{x}$ to a scalar energy value. In deep learning applications, the energy function $E_{\boldsymbol{\theta}}$ is a neural network parameterized by $\boldsymbol{\theta}$.

EBMs are powerful models that have been applied to different domains, such as structured prediction (Belanger & McCallum, 2016; Rooshenas et al., 2019; Tu & Gimpel, 2019), text generation (Deng et al., 2020), RL (Haarnoja et al., 2017), image generation (Salakhutdinov & Hinton, 2009; Du & Mordatch, 2019; Du et al., 2020; Nijkamp et al., 2019), and classification (Grathwohl et al., 2019). An important difference from prior EBM methods for image classification (Grathwohl et al., 2019) is that we do not utilize negative samples from MCMC in the training objective. Utilizing MCMC to sample negative images makes EBMs difficult to scale to complex datasets and makes training much slower. Our method that uses negative labels in the training objective is faster and more scalable. In this paper, we study continual learning on classification tasks. As far as we are aware, EBMs for continual learning has so far remained unexplored.

4.2 ENERGY-BASED MODELS FOR CLASSIFICATION

To solve the classification tasks, we adapt the above general formulation of an EBM as follows. Given inputs $\mathbf{x} \in \mathbb{R}^D$ and a discrete set $\mathcal{Y}$ of possible class labels, we propose to use the Boltzmann distribution to define the conditional likelihood of label y given $\mathbf{x}$:

$$p_{\boldsymbol{\theta}}(y|\mathbf{x}) = \frac{\exp(-E_{\boldsymbol{\theta}}(\mathbf{x}, y))}{Z(\boldsymbol{\theta}; \mathbf{x})}, Z(\boldsymbol{\theta}; \mathbf{x}) = \sum_{y' \in \mathcal{Y}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}, y')) \tag{3}$$

where $E_{\boldsymbol{\theta}}(\mathbf{x}, y) : (\mathbb{R}^D, \mathbb{N}) \to \mathbb{R}$ is the energy function that maps an input-label pair $(\mathbf{x}, y)$ to a scalar energy value, and $Z(\boldsymbol{\theta}; \mathbf{x})$ is the partition function for normalization.

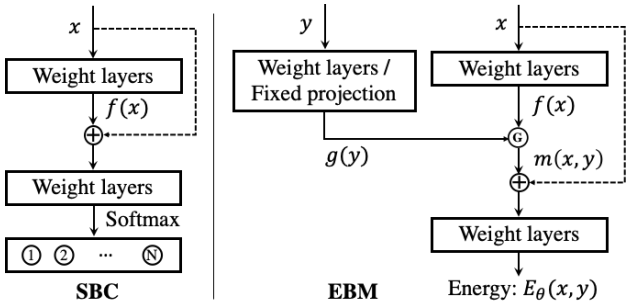

Figure 1: Schematic of the model architectures of the softmax-based classifier (SBC) and energy-based models (EBM). SBC takes an image $\mathbf{x}$ as input and outputs a fixed pre-defined N-dimensional vector. EBM takes a data $\mathbf{x}$ and a class y as input and outputs their energy value. The dash lines are optional skip connections.

Training. We want the distribution defined by $E_{\boldsymbol{\theta}}$ to model the data distribution p_D, which we do by minimizing the negative log likelihood of the data $\mathcal{L}_{\text{ML}}(\boldsymbol{\theta}) = \mathbb{E}_{(x,y)\sim p_D}[-\log p_{\boldsymbol{\theta}}(y|\mathbf{x})]$ with the expanded form:

$$\mathcal{L}_{\text{ML}}(\boldsymbol{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim p_D}\left[E_{\boldsymbol{\theta}}(\mathbf{x}, y) + \log(\sum_{y' \in \mathcal{Y}} e^{-E_{\boldsymbol{\theta}}(\mathbf{x}, y')})\right]. \tag{4}$$

Equation (4) minimizes the energy of $\mathbf{x}$ at the ground truth label y and minimizes the overall partition function by increasing the energy of $\mathbf{x}$ at other labels y'.

Inference. Given an input $\mathbf{x}$, the class label predicted by our EBMs is the class with the smallest energy at $\mathbf{x}$, which can be written as $\hat{y} = \arg\min_{y' \in \mathcal{Y}} E_\theta(\mathbf{x}, y')$.

4.3 ENERGY-BASED MODEL AS A BUILDING BLOCK FOR CONTINUAL LEARNING

EBM training objective. In Equation (4), the energy across all class labels y' given data $\mathbf{x}$ is maximized. Directly maximizing energy across all labels raises the same problem as the softmax-based classifier models that the old classes are suppressed when training a model on new classes and thus cause the catastrophic forgetting. Inspired by (Hinton, 2002), we find that the contrastive divergence approximation of Equation (4) can mitigate this problem and lead to a simpler equation. To do so, we define the following contrastive divergence loss:

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}; \mathbf{x}, y) = \mathbb{E}_{(\mathbf{x},y)\sim p_D}\left[E_{\boldsymbol{\theta}}(\mathbf{x}, y) - E_{\boldsymbol{\theta}}(\mathbf{x}, y^-)\right], \tag{5}$$

where y is the ground truth label of data $\mathbf{x}$ and y^- is a negative class label randomly sampled from the set of class labels in the current training batch $\mathcal{Y}_B$ such that $y^- \neq y$.

Different from the softmax-based classifier, EBMs maximize likelihood not by normalizing over all classes but instead by contrastively increasing the energy difference between the ground truth label and another negative label for a given data point. This operation causes less interference with previous classes and enables EBMs to suffer less from catastrophic forgetting.

Importantly, such a sampling strategy also allows EBMs to be naturally applied, without any modification, to different CL settings (Section 2.1). Since we select the negative sample(s) from the current batch, our EBMs do not require knowledge of the task on hand. This allows application of EBMs in the *boundary-free* setting, in which the underlying tasks or task boundaries are not given. See Appendix A.2 for more details about the *boundary-free* setting.

We find that the proposed EBM training objective is efficient enough to achieve good performance on different CL datasets (Table 1 and Table 5). We note however that it is possible to use other strategies for choosing the negative classes in the partition function in Equation (4). In Table 3, we explore alternative strategies: 1) using all classes in the current training batch $\mathcal{Y}_{\mathcal{B}}$ as negative classes, and 2) using all classes seen so far as negative classes. The usage of negative samples in the EBM training objective provides freedom for models to choose which classes to train on which is important for preventing catastrophic forgetting in CL.

Energy network. Another important difference from softmax-based classifiers is that the choice of model architectures becomes more flexible in EBMs. Traditional classification models only feed in $\mathbf{x}$ as input. In contrast, EBMs have many different ways to combine $\mathbf{x}$ and y in the energy function with the only requirement that $E_\theta(\mathbf{x}, y) : (\mathbb{R}^D, \mathbb{N}) \to \mathbb{R}$. In EBMs, we can treat y as an attention filter or gate to select the most relevant information between $\mathbf{x}$ and y.

To compute the energy of any data $\mathbf{x}$ and class label y pair, we use y to influence a conditional gain on $\mathbf{x}$, which serves as an attention filter (Xu et al., 2015) to select the most relevant information between $\mathbf{x}$ and y. In Figure 1 (right), we first send $\mathbf{x}$ into a network to generate a feature $f(\mathbf{x})$. The label y is mapped into a same dimension feature $g(y)$ using a small learned network or a fixed projection. We use the gating block G to select the most relevant information between $\mathbf{x}$ and y:

$$m(\mathbf{x}, y) = G(f(\mathbf{x}), g(y)). \tag{6}$$

The output is finally sent to weight layers to generate the energy value $E_\theta(\mathbf{x}, y)$. See Appendix C for more details about our model architectures.

Our EBMs allow any number of classes in new batches by simply training or defining a new conditional gain $g(\mathbf{y})$ for the new classes and generating its energy value with data point $\mathbf{x}$.

Inference. During inference, because we evaluate according to the class-incremental learning scenario, the model must predict a class label by choosing from all classes seen so far. Let $\mathbf{x}$ be one data point with an associated discrete label y. There are $\mathcal{Y}$ different class labels in the dataset. The MAP estimate is $\hat{y} = \arg\min_{y'} E_\theta(\mathbf{x}, y')$, where $y' \in \mathcal{Y}$ and E_θ is the energy function with parameters θ after training on all the training data.

Alternative EBM training objective. EBMs are not limited to modeling the conditional distribution between $\mathbf{x}$ and y as shown in Equation (3). Another way to use the Boltzmann distribution is to define the joint likelihood of $\mathbf{x}$ and y. In Appendix F, we show that modeling the joint likelihood can further improve the results. Since the main focus of this paper is to propose a simple but efficient EBM training objective for continual learning, we only show the results of using Equation (5) in this main paper.

5 Experiments

In this section, we want to answer the following questions: How do the proposed EBMs perform on different CL settings? Can we apply the EBM training objective to other methods? And what is the best architecture for label conditioning? To answer these questions, we first report experiments on the *boundary-aware* setting in Section 5.1. We then show that EBMs can also be applied to the *boundary-free* setting in Section 5.2.

5.1 Experiments on Boundary-Aware Setting

Datasets. We evaluate the proposed EBMs on four commonly used CL datasets, including split MNIST (Zenke et al., 2017), permuted MNIST (Kirkpatrick et al., 2017), CIFAR-10 (Krizhevsky et al., 2009), and CIFAR-100 (Krizhevsky et al., 2009). The split MNIST is obtained by splitting the original MNIST (LeCun et al., 1998) into 5 tasks with each task having 2 classes. It has 60,000 training images and 10,000 test images. We follow the implementation of (van de Ven & Tolias, 2019) for permuted MNIST. There are 10 tasks and each task has 10 classes. We separate CIFAR-10

Table 1: Evaluation of class-incremental learning on the *boundary-aware* setting on four datasets. Experiments ran by ourselves are performed 10 times with different random seeds, the results are reported as the mean ± SEM over these runs. Results with (-) means they are not reported in their original paper. Results with ± (-) means a SEM is not reported in their original paper.

Without Replay	**split MNIST**	**permuted MNIST**	**CIFAR-10**	**CIFAR-100**
SBC	19.90 ± 0.02	17.26 ± 0.19	19.06 ± 0.05	8.18 ± 0.10
EBM	**53.12 ± 0.04**	**87.58 ± 0.50**	**38.84 ± 1.08**	**30.28 ± 0.28**
EWC (Kirkpatrick et al., 2017)	20.01 ± 0.06	25.04 ± 0.50	18.99 ± 0.03	8.20 ± 0.09
Online EWC (Schwarz et al., 2018)	19.96 ± 0.07	33.88 ± 0.49	19.07 ± 0.13	8.38 ± 0.15
SI (Zenke et al., 2017)	19.99 ± 0.06	29.31 ± 0.62	19.14 ± 0.12	9.24 ± 0.22
LwF (Li & Hoiem, 2017)	23.85 ± 0.44	22.64 ± 0.23	19.20 ± 0.30	10.71 ± 0.11
MAS (Aljundi et al., 2019b)	19.50 ± 0.30	-	20.25 ± 1.54	8.44 ± 0.27
BGD (Zeno et al., 2018)	19.64 ± 0.03	84.78 ± 1.30	-	-

With Replay	**split MNIST**	**permuted MNIST**	**CIFAR-10**	**CIFAR-100**
SBC+ER	90.65 ± 0.45	93.70 ± 0.09	42.07 ± 0.64	28.57 ± 0.35
EBM+ER	91.13 ± 0.35	94.59 ± 0.09	**44.76 ± 0.73**	**34.07 ± 0.55**
iCaRL (Rebuffi et al., 2017)	**94.57 ± 0.11**	**94.85 ± 0.03**	-	-
DGR (Shin et al., 2017)	91.30 ± 0.60	92.19 ± 0.09	17.21 ± 1.88	9.22 ± 0.24
BI-R (van de Ven et al., 2020)	94.41 ± 0.15	-		21.51 ± 0.25
GSS-Greedy (Aljundi et al., 2019c)	84.80 ± 1.80	77.30 ± 0.50	33.56 ± 1.70	-
CTN (Pham et al., 2021)	-	79.01 ± 0.65	-	-
PGMA (Hu et al., 2019)	81.70 ± (-)	-	40.47 ± (-)	-

into 5 tasks, each task with 2 classes. CIFAR-100 is split into 10 tasks with each task having 10 classes. CIFAR-10 and CIFAR-100 each has 50,000 training images and 10,000 test images.

Evaluation protocol. All experiments are performed according to the class-incremental learning scenario, which is considered as the most natural and also the hardest setting for continual learning (Tao et al., 2020a; He et al., 2018; Tao et al., 2020b). Many CL approaches that perform well for task-incremental learning, fail when asked to do class-incremental learning (van de Ven & Tolias, 2019). More details can be found in Appendix A.1.

5.1.1 Comparisons with existing methods

The first comparison of interest is how the performance of the proposed EBMs compares with the performance of the softmax-based classifier (SBC). But we aim higher, and we additionally compare how the proposed EBMs perform relative to methods specifically designed for continual learning. For this, we compare against the methods EWC (Kirkpatrick et al., 2017), Online EWC (Schwarz et al., 2018), SI (Zenke et al., 2017), LwF (Li & Hoiem, 2017), MAS (Aljundi et al., 2019b), BGD (Zeno et al., 2018), BI-R (van de Ven et al., 2020), DGR (Shin et al., 2017), iCaRL (Rebuffi et al., 2017), GSS-Greedy (Aljundi et al., 2019c), CTN (Pham et al., 2021), and PGMA (Hu et al., 2019).

Comparison with methods without replay. We first compare EBMs with CL methods without using replay. In our experiments, we control the baselines and EBMs to have similar model architectures with similar number of model parameters. For SBC, EWC, Online EWC, Online EWC, LwF on splitMNIST, permuted MNIST, and CIFAR-100, we use the results reported in (van de Ven & Tolias, 2019; van de Ven et al., 2020). For BGD, the results are from (Zeno et al., 2018). For MAS, we use the result from (Prabhu et al., 2020; Zhang et al., 2020). We performed the other experiments ourselves using the public available code. The detailed model architectures are listed in Appendix C.

The *Class-IL* results on four datasets are shown in Table 1. Experiments ran by ourselves were performed 10 times with different random seeds, with results reported as the mean ± SEM. Similar training regimes were used for the EBMs and baselines. On split MNIST, permuted MNIST, and CIFAR-10, we trained for 2000 iterations per task. On CIFAR-100, we trained for 5000 iterations per task. All experiments used the Adam optimizer with learning rate $1e^{-4}$.

On these *Class-IL* benchmarks, we observe that EBMs have a significant improvement over SBC, as well as over several softmax-based CL methods that do not rely on an external quota of memory or use generative replay.

Several recent methods (Wortsman et al., 2020; Henning et al., 2021; Yan et al., 2021; Douillard et al., 2022) also achieve good performance on *Class-IL* without using replay. However, our main goal is to use EBM, similar to the softmax-based classifier, as a building block for continual learning. We believe that subsequently optimizing the model architecture, using a larger model, adding additional techniques, or saving previous data, can further improve the performance, but this is not the focus of the paper.

Table 2: Class-IL results of EBM and baselines using our contrastive divergence training objective and their original one.

Dataset	Method	CLS	EWC	Online EWC	SI	EBM
split MNIST	Original Objective	19.90 ± 0.02	20.01 ± 0.06	19.96 ± 0.07	19.99 ± 0.06	-
	CD Objective	44.98 ± 0.05	50.68 ± 0.04	50.99 ± 0.03	49.44 ± 0.03	$\mathbf{53.12 \pm 0.04}$
CIFAR-10	Original Objective	19.06 ± 0.01	18.99 ± 0.01	19.07 ± 0.01	19.14 ± 0.02	-
	CD Objective	19.22 ± 0.02	36.51 ± 0.03	36.16 ± 0.02	35.12 ± 0.02	$\mathbf{38.84 \pm 0.01}$

Table 3: Performance of EBM on CIFAR-100 with different strategies for selecting the negative samples.

Dataset	CIFAR-100
All Neg Seen	8.07 ± 0.10
All Neg Batch	29.03 ± 0.53
1 Neg Batch	$\mathbf{30.28 \pm 0.28}$

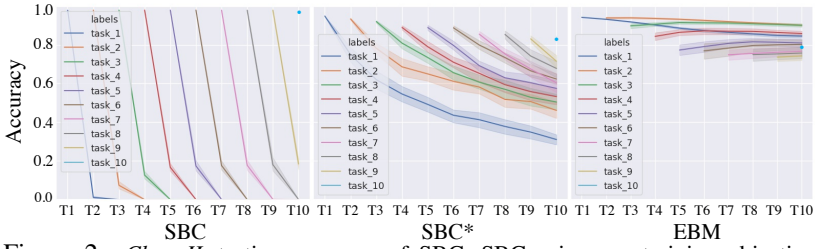

Figure 2: *Class-IL* testing accuracy of SBC, SBC using our training objective (SBC*), and EBMs on each task on the permuted MNIST dataset.

Comparison with methods with replay. We notice that many of the successful existing methods for *Class-IL* rely on an external quota of memory (Rebuffi et al., 2017; Hayes et al., 2020) or use generative models (Shin et al., 2017; van de Ven et al., 2021; 2020; Liu et al., 2020; Cong et al., 2020). A disadvantage of these methods is that they are relatively expensive in terms of memory and/or computation. Thus it is not fair to compare the basic version of EBMs to methods using replay.

Importantly, however, EBMs are flexible enough to be combined with existing continual learning approaches to further improve the performance. In Table 1, we show the result of combining EBMs with exact replay, *i.e.* **EBM+ER**. As EBM+ER uses previous data during training, for fair comparisons, we design another baseline that combines SBC with exact replay, *i.e.* SBC+ER. See Appendix D for the implementation details of EBM+ER and SBC+ER. Combined with replay, both SBC and EBM have improvements. EBMs still outperform SBC, especially on more challenging datasets, *e.g.* CIFAR-10 and CIFAR-100. Interestingly, on CIFAR-100, we find that EBMs without using replay (EBM: 30.28%) perform better than SBC with using replay (SBC+ER: 28.57%). Moreover, the performance of EBM+ER is comparable to the performance of established replay-based methods such as iCaRL, DGR, and BI-R.

Our results indicate that the proposed EBM formulation provides a promising building block for tackling CL problems. Other approaches, such as replay-based approaches, can build on top of EBMs.

5.1.2 Effect of Contrastive Divergence Training Objective and Label Conditioning

Contrastive divergence training objective on existing approaches. The proposed contrastive divergence training objective is simple and can also be directly applied to existing CL approaches. We test this by modifying the training objective of baseline models to that of our proposed contrastive divergence objective, which computes the softmax normalization only over the ground truth class y and a negative class i sampled from the current training batch: $\mathcal{L}_{\text{SBC\_CD}}(\mathbf{x}, y) = -[f_{\boldsymbol{\theta}}(\mathbf{x})]_y + [f_{\boldsymbol{\theta}}(\mathbf{x})]_i$. We only modify their training objective without changing their model architectures. In Table 2, we find that our proposed training objective (CD Objective) significantly improves the performance of different CL methods. This is because the contrastive divergence objective does not suppress the probability of old classes when improving the probability of new classes. Our training objective is simple to implement on existing CL approaches.

Effect of training objectives. We conduct an experiment on the CIFAR-100 dataset to investigate how different EBM training objectives influence the CL results. We compare three different strategies for selecting the negative samples as described in Section 4.3. The first strategy uses all seen classes so far as negative labels (**All Neg Seen**), which is most similar to the way that the traditional classifier is optimized. The second one takes all the classes in the current batch as negative labels (**All Neg Batch**). The last one — the contrastive divergence objective we proposed in Equation (5) — randomly selects one class from the current batch as the negative (**1 Neg Batch**). In Table 3, we find that using only one negative sample generates the best result, and using negatives sampled from classes in the current batch is better than from all seen classes. Since our EBM training objective aims at improving the energy of negative samples while decreasing the energy of positive ones, sampling negatives from the current batch has less interference with previous classes than sampling from all seen classes. The proposed contrastive divergence training objective uses a single negative and causes the minimum suppression on negative samples.

Table 4: Performance of EBM with different label conditioning architectures on the CIFAR-10 dataset.

Model architectures		**Normalization types**	
Beginning (V1)	13.69 ± 1.12	End Fix (V4)	34.30 ± 1.03
Middle (V2)	20.16 ± 1.05	End Fix Norm2 (V4)	33.91 ± 1.13
Middle (V3)	18.36 ± 0.97	End Fix Softmax (V4)	35.97 ± 1.09
End (V4)	38.13 ± 0.59	End Norm2 (V4)	37.23 ± 1.20
		End Softmax (V4)	$\mathbf{38.84 \pm 1.08}$

Effect of label conditioning. Next, we test whether the label conditioning in our EBMs is important for their performance. As mentioned in Table 2, we modify baselines using our training objective. The only difference is that EBMs have the label conditioning architecture while baselines do not. EBMs outperform the modified baselines, *e.g.* Online EWC — the best baselines with the contrastive divergence training objective — is 50.99% while EBM is 53.12% on split MNIST. This implies that the label conditioning architecture mitigates catastrophic forgetting in EBMs.

We further show the testing accuracy of each task as the training progresses in Figure 2. We compare the standard classifier (SBC), classifier using the contrastive divergence training objective (SBC*), and our EBMs on the permuted MNIST dataset. We find that the accuracy of old tasks in SBC drop sharply when learning new tasks, while the contrastive divergence training objective used in SBC* is better. The curve on EBMs drops even slower than SBC*. As both SBC* and EBMs use the contrastive divergence training objective, the good performance of EBMs implies the effectiveness of the label conditioning architecture in EBMs.

To summarize, we show that the strong performance of our EBMs is due to both the contrastive divergence training objective and the label conditioning architecture. Moreover, these results indicate that surprisingly, and counterintuitively, directly optimizing the softmax-based architecture with the cross-entropy loss used by existing approaches may not be the best way to approach continual learning.

5.1.3 Comparisons of different model architectures

EBMs allow flexibility in integrating data information and label information in the energy function. To investigate where and how to combine the information from data $\mathbf{x}$ and label y, we conduct a series of experiments on CIFAR-10. Table 4 shows four model architectures (V1-V4) that combine $\mathbf{x}$ and y in the early, middle, and late stages, respectively (see Appendix C for more details). We find that combining $\mathbf{x}$ and y in the late stage (V4) performs the best. We note that instead of learning a feature embedding of label y, we can use a fixed projection matrix which is randomly sampled from the uniform distribution $\mathcal{U}(0, 1)$. Even using this fixed random projection can already generates better results than most baselines without replay in Table 1. Note further that the number of trainable parameters in the "Fix" setting is much lower than that of the baselines. We notice that the standard classifiers have to change their model architecture by adding new class heads to the softmax output layer when dealing with new classes. In contrast, the EBMs "Fix" setting do not need to modify the model architecture or resize the network when adding new classes. Using a learned feature embedding of y can further improve the result. We apply different normalization methods over the feature channel of y. We find that Softmax (End Softmax (V4)) is better than the L2 normalization (End Norm2 (V4)) and no normalization (End (V4)).

5.1.4 Qualitative analysis

To better understand why EBMs suffer less from catastrophic forgetting, we qualitatively compare our EBMs and SBC in this part. We provide additional analyses of CL performance of SBC and EBMs in Appendix B and Appendix E.

Energy landscape. In Figure 3, we show the energy landscapes after training on task 9 and task 10 of the permuted MNIST dataset. For SBC, the energy is given by the negative of the predicted probability. Each datapoint has 100 energy values (EBM) or probabilities (SBC) corresponding to the 100 labels in the dataset. For each datapoint, these values are normalized over all 100 classes. Dark elements on the diagonal indicate correct predictions. After training on task T_9, SBC assigns high probabilities to classes from T_9 (80-90) for almost all the data from T_1 to T_9. After learning task T_{10}, the highest probabilities shift to classes from task T_{10} (90-100). SBC tends to assign high probabilities to new classes for both old and new data, indicating forgetting. In contrast, EBM has low energies across the diagonal, which means that after training on new tasks, EBM still assigns low energies to the true labels of data from previous tasks. This shows that EBM is better than SBC at learning new tasks without catastrophically forgetting the old tasks.

Predicted class distribution. In Figure 4, for the split MNIST dataset, we plot the proportional distribution of predicted classes. Only data from the tasks seen so far was used for this figure. Taking the second panel in the

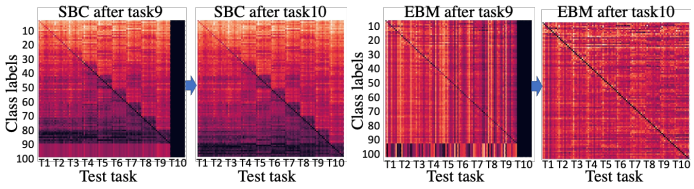

Figure 3: Energy landmaps of SBC and EBMs after training on task T_9 and T_{10} on permuted MNIST. The darker the diagonal is, the better the model is in preventing forgetting previous tasks.

Figure 4: Predicted label distribution after learning each task on split MNIST. SBC only predicts classes from the current task, while EBM predicts classes from all seen classes.

Table 5: Evaluation of class-incremental learning performance on the *boundary-free* setting. Each experiment is performed 5 times with different random seeds, average test accuracy is reported as the mean ± SEM over these runs.

Method	**split MNIST**	**permuted MNIST**	**CIFAR-10**	**CIFAR-100**
SBC	24.03 ± 0.59	21.42 ± 0.11	23.30 ± 0.81	9.85 ± 0.02
EBM	**81.78 ± 1.22**	**92.35 ± 0.11**	**49.47 ± 1.25**	**34.39 ± 0.24**
Online EWC (Schwarz et al., 2018)	39.62 ± 0.14	41.37 ± 0.04	22.53 ± 0.41	9.57 ± 0.02
SI (Zenke et al., 2017)	28.79 ± 0.24	35.71 ± 0.11	26.26 ± 0.72	10.42 ± 0.01
BGD (Zeno et al., 2018)	21.65 ± 1.15	26.15 ± 0.22	17.03 ± 0.82	8.50 ± 0.02

first row as an example, it shows the distribution of predicted labels on test data from the first two tasks after finishing training on the second task. This means that so far the model has seen four classes: {1,2,3,4}. Since the number of test images from each class are similar, the ground truth proportional distribution should be uniform over those four classes. After training on the first task, the predictions of SBC are indeed roughly uniformly distributed over the first two classes (first panel). However, after learning new tasks, SBC only predicts classes from the most recent task, indicating that SBC fails to correctly memorize classes from previous tasks. In contrast, the predictions of EBM are substantially more uniformly distributed over all classes seen so far.

5.2 Experiments on Boundary-Free Setting

When applying continual learning in real life, boundaries are not usually well defined between different tasks. However, most existing CL methods rely on the presence of sharp, known boundaries between tasks to determine when to consolidate the knowledge. We show that EBMs are able to flexibly perform CL across different setups, and perform well on the *boundary-free* setting as well.

5.2.1 Datasets and Evalution Protocols

For the *boundary-free* setting, we use the same datasets as the *boundary-aware* setting in Section 5.1. We use the code of "continuous task-agnostic learning" proposed by (Zeno et al., 2018) to generate a continually changing data stream during training. In this setting, the frequency of each subsequent class increases and decreases linearly. See Appendix A.2 for more details of the *boundary-free* setting. All experiments are again performed in a class-incremental setup.

5.2.2 Comparison with Existing Methods

In this setting, we restrict our comparison to methods that do not use replay. Most of the non-replay methods that were compared in Table 1 cannot be applied to the *boundary-free* setting without modification, because these methods rely on known task boundaries to perform certain consolidation operations (e.g., to update the parameter regularization term). A relatively straight-forward adaptation of these methods is to perform their consolidation operation after every mini-batch step instead of after each task (see also (Zeno et al., 2018)), although this adaptation is impractical for some algorithms (e.g., EWC) because of large computational complexity. We managed to run the Online EWC and SI baselines in this way, while the BGD baseline is designed to be suitable for the *boundary-free* setting.

We further note that there are continual learning methods that might be applicable to settings in which task boundaries are unknown (i.e., a *boundary-agnostic* setting), but these methods are not necessarily also applicable to the *boundary-free* setting, because often these methods still require that there are underlying task boundaries. For example, the method proposed by (Wortsman et al., 2020) is able to infer task boundaries if they are not given (i.e., *boundary-agnostic*), but this method cannot deal with the setting in which there are no underlying boundaries at all (i.e., *boundary-free*).

The results on the *boundary-free* setting are shown in Table 5. All compared methods used similar model architectures as in the *boundary-aware* setting. Each experiment was performed 5 times with different random seeds, the results are reported as the mean $\pm$ SEM over these runs. We observe that EBMs have a significant improvement on all the datasets. The experiments show that EBMs have good generalization ability for different continual learning problems as EBMs can naturally handle data streams with and without task boundaries.

6 Discussion

In this paper, we show that energy-based models are a promising class of models in a variety of different continual learning settings. We demonstrate that EBMs exhibit many desirable characteristics to prevent catastrophic forgetting in continual learning, and we experimentally show that EBMs obtain strong performance on the challenging class-incremental learning scenario on multiple benchmarks, both on the boundary-aware and boundary-free settings. One drawback of the current EBM formulation is that we need to compute the energy between a data point and each class during inference. However, in the actual implementation, we found that the difference between the softmax-based classifier (SBC) and our EBM is minor. Another potential limitation of this work is that we only focus on evaluating the average accuracy, and mostly only after training has finished. It might be interesting to also focus on other evaluation metrics, such as backward and forward transfer that evaluates models' ability to transfer knowledge across tasks (Lopez-Paz & Ranzato, 2017; Vogelstein et al., 2020), or to perform continual evaluation (De Lange et al., 2022). We do provide several additional results and analyses in the appendix.

Acknowledgements

This work has been partly supported by the Lifelong Learning Machines (L2M) program of the Defence Advanced Research Projects Agency (DARPA) via contract number HR0011-18-2-0025.

References

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. *arXiv preprint arXiv:1908.04742*, 2019a.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11254–11263, 2019b.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019c.

David Belanger and Andrew McCallum. Structured prediction energy networks. In *International Conference on Machine Learning*, pp. 983–992, 2016.

Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. Initial classifier weights replay for memoryless class incremental learning. *arXiv preprint arXiv:2008.13710*, 2020.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2019a.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019b.

Yulai Cong, Miaoyun Zhao, Jianqiao Li, Sijia Wang, and Lawrence Carin. Gan memory with no forgetting. *Advances in Neural Information Processing Systems*, 33, 2020.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Matthias De Lange, Gido van de Ven, and Tinne Tuytelaars. Continual evaluation for lifelong learning: Identifying the stability gap. *arXiv preprint arxiv:2205.13452*, 2022.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. Residual energy-based models for text generation. *arXiv preprint arXiv:2004.11714*, 2020.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295, 2022.

Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation and inference with energy based models. *arXiv preprint arXiv:2004.06030*, 2020.

Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

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.

Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*, 2020.

Tyler L Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 220–221, 2020.

Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pp. 466–483. Springer, 2020.

Chen He, Ruiping Wang, Shiguang Shan, and Xilin Chen. Exemplar-supported generative reproduction for class incremental learning. In *BMVC*, pp. 98, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Christian Henning, Maria Cervera, Francesco D'Angelo, Johannes Von Oswald, Regina Traber, Benjamin Ehret, Seijin Kobayashi, Benjamin F Grewe, and João Sacramento. Posterior meta-replay for continual learning. *Advances in Neural Information Processing Systems*, 34:14135–14149, 2021.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8): 1771–1800, 2002.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019.

Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao, Jinwen Ma, Dongyan Zhao, and Rui Yan. Overcoming catastrophic forgetting for continual learning via model adaptation. In *International Conference on Learning Representations*, 2019.

Xisen Jin, Arka Sadhu, Junyi Du, and Xiang Ren. Gradient based memory editing for task-free continual learning. *arXiv preprint arXiv:2006.15294*, 2020.

Ta-Chu Kao, Kristopher Jensen, Gido van de Ven, Alberto Bernacchia, and Guillaume Hennequin. Natural continual learning: success is a journey, not (just) a destination. *Advances in Neural Information Processing Systems*, 34: 28067–28079, 2021.

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*, 114(13):3521–3526, 2017.

Soheil Kolouri, Nicholas A Ketz, Andrea Soltoggio, and Praveen K Pilly. Sliced cramer synaptic consolidation for preserving deeply learned representations. In *International Conference on Learning Representations*, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits, 1998. *URL http://yann. lecun. com/exdb/mnist*, 10:34, 1998.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In *International Conference on Learning Representations*, 2020.

Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Andrei Stoian, and David Filliat. Generative models from the perspective of continual learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

Timothée Lesort, Thomas George, and Irina Rish. Continual learning in deep networks: an analysis of the last layer. *arXiv preprint arXiv:2106.01834*, 2021.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D Bagdanov, Shangling Jui, and Joost van de Weijer. Generative feature replay for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 226–227, 2020.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

Sandeep Madireddy, Angel Yanguas-Gil, and Prasanna Balaprakash. Neuromodulated neural architectures with local error signals for memory-constrained online continual learning. *arXiv preprint arXiv:2007.08159*, 2020.

Davide Maltoni and Vincenzo Lomonaco. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116:56–73, 2019.

Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020.

Nicolas Y Masse, Gregory D Grant, and David J Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44):E10467–E10475, 2018.

Martin Mundt, Yong Won Hong, Iuliia Pliushch, and Visvanathan Ramesh. A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. *arXiv preprint arXiv:2009.01797*, 2020.

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

Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. *arXiv preprint arXiv:1903.12370*, 2019.

Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard E Turner, and Mohammad Emtiyaz Khan. Continual deep learning by functional regularisation of memorable past. In *Advances in Neural Information Processing Systems*, 2020.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

Quang Pham, Chenghao Liu, Doyen Sahoo, and HOI Steven. Contextual transformation networks for online continual learning. In *International Conference on Learning Representations*, 2021.

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. 2020.

Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13588–13597, 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

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

Amirmohammad Rooshenas, Dongxu Zhang, Gopal Sharma, and Andrew McCallum. Search-guided, lightly-supervised training of structured prediction energy networks. In *Advances in Neural Information Processing Systems*, pp. 13522–13532, 2019.

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.

Ruslan Salakhutdinov and Geoffrey Hinton. Deep boltzmann machines. In David van Dyk and Max Welling (eds.), *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pp. 448–455, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR. URL http://proceedings.mlr.press/v5/salakhutdinov09a.html.

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 Serra, Didac Suris, 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.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. 2020a.

Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12183–12192, 2020b.

Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. In *International Conference on Learning Representations*, 2020.

Lifu Tu and Kevin Gimpel. Benchmarking approximate inference methods for neural structured prediction. *arXiv preprint arXiv:1904.01138*, 2019.

Gido M van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Gido M van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications*, 11:4069, 2020.

Gido M van de Ven, Zhe Li, and Andreas S Tolias. Class-incremental learning with generative classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 3611–3620, 2021.

Joshua T Vogelstein, Jayanta Dey, Hayden S Helm, Will LeVine, Ronak D Mehta, Ali Geisa, Haoyin Xu, Gido M van de Ven, Chenyu Gao, Weiwei Yang, Bryan Tower, Jonathan Larson, Christopher M White, and Carey E Priebe. Representation ensembling for synergistic lifelong learning with quasilinear complexity. *arXiv preprint arXiv:2004.12908*, 2020.

Johannes von Oswald, Christian Henning, João Sacramento, and Benjamin F Grewe. Continual learning with hypernetworks. In *International Conference on Learning Representations*, 2019.

Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33, 2020.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 374–382, 2019.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057, 2015.

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023, 2021.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3987–3995. JMLR. org, 2017.

Chen Zeno, Itay Golan, Elad Hoffer, and Daniel Soudry. Task agnostic continual learning using online variational bayes. *arXiv preprint arXiv:1803.10123*, 2018.

Song Zhang, Gehui Shen, and Zhi-Hong Deng. Self-supervised learning aided class-incremental lifelong learning. *arXiv preprint arXiv:2006.05882*, 2020.

Appendices

In this appendix, we provide more details of the experiment setup in Section A. We show more experimental results of our EBMs and baselines in Section B. Section C lists the model architecture details of the proposed EBMs and baselines on difference datasets. Section D has more experimental details of EBMs using replay. We provide additional analysis of why EBM can mitigate the catastrophic forgetting problem in continual learning in Section E. Section F describes the alternative EBM training objective mentioned in the main paper Section 4.3.

A More Experiment Setup Details

A.1 More details about the class-incremental setting

In the main paper Section 2.1, we have talked about the task-incremental and class-incremental settings. In this part, we provide more details of the class-incremental setting.

Class-incremental learning is a standard continual learning setting and has been used in many existing continual learning papers. In the class-incremental learning, the model is sequentially trained on each task. During testing, the model needs to predict the correct class from all classes.

Training: Taking the split MNIST dataset under the boundary-aware setting as an example, task1 has images from two classes [0, 1]; task2 has images from two classes [2, 3]; ...; and task5 has images from classes [8, 9]. The model is first trained on task1, and then task2, until the final task. Note that the model is trained sequentially, so when training the final task, the model only sees images and classes in the final task, and thus it tends to forget previous tasks which is called catastrophic forgetting.

Testing: In the class-incremental learning, given an image from a task, the model needs to predict its class label from all seen classes (e.g. 10 classes in split MNIST). Note that this is challenging as 1) during training, the model only needs to predict the class label of an image from classes in the current task (e.g. 2 classes in split MNIST); 2) after the model is trained on the final task, it is more likely to predict classes from the final task (e.g. classes [8, 9] in split MNIST), even the input image is from previous tasks. Please see Figure 3 and Figure 4 in the main paper for a better understanding.

A.2 More details about the boundary-free and boundary-aware settings

In the main paper Section 2.1, we have talked about the boundary-free and boundary-aware settings. In this part, we provide more details of the less frequently used boundary-free setting.

Boundary-free and boundary-aware are two different settings of how the data is provided during training. We follow the general implementation of (Zeno et al., 2018) for the boundary-free setting. In this setting, models learn in a streaming fashion and the data distributions gradually change over time. Taking the split MNIST dataset as an example, it has 10 classes. During training, the percentage of '1s' in each training batch gradually decreases while the percentage of '2s' increases, and then the percentage of '2s' in each training batch gradually decreases while the percentage of '3s' increases, and so on. The model can observe data from different classes during training and there is no a clear task boundary. It might also be useful to have a look at Figure 2 of the paper (Zeno et al., 2018) for a better intuitive understanding of the boundary-free setting.

In our contrastive divergence training objective, we randomly select 1 negative sample from the current batch. If there are multiple negative samples, we just randomly select 1 from them. In Table 3 of the main paper, our contrastive divergence training objective (1 Neg Batch=30.28) is better than using all the negative classes in the current batch (All Neg Batch=29.03) and is much better than using all the negative classes seen so far (All Neg Seen=8.07).

In the contrastive divergence training objective, the positive and negative classes are sampled from the current batch regardless of how the batch data is provided. The batch data can be sampled based on either the boundary-free setting or the boundary-aware setting. For example, In the boundary-aware setting, the first batch may only contain classes (1,2) from the first task, the second batch may only contain classes (3,4) from the second task, the third batch may only have classes (5,6) from the third task, and so on. There are clear task boundaries. In the boundary-free setting, the first batch may contain classes (1), later batches may contain classes (1,2), and then classes (2,3), and so on. There are no task boundaries.

The contrastive divergence loss allows our method to be trained without requiring knowledge of the task boundaries, and thus it can naturally handle both the boundary-aware and boundary-free settings. In contrast, many existing

Table 6: Comparison of our EBM with baselines on different variants of the split CIFAR-100 protocol. Results of the *Class-IL* performance on the *boundary-aware* setting are reported.

Method	**CIFAR-100 split up into:**			
	5 tasks	**10 tasks**	**20 tasks**	**50 tasks**
SBC	14.74 ± 0.20	8.18 ± 0.10	4.46 ± 0.03	1.91 ± 0.02
EBM	$\mathbf{34.88 \pm 0.14}$	$\mathbf{30.28 \pm 0.28}$	$\mathbf{25.04 \pm 0.33}$	$\mathbf{13.60 \pm 0.50}$
EWC (Kirkpatrick et al., 2017)	14.78 ± 0.21	8.20 ± 0.09	4.46 ± 0.03	1.91 ± 0.02
SI (Zenke et al., 2017)	14.07 ± 0.24	9.24 ± 0.22	4.37 ± 0.04	1.88 ± 0.03
LwF (Li & Hoiem, 2017)	25.75 ± 0.14	10.71 ± 0.11	12.18 ± 0.16	7.68 ± 0.16

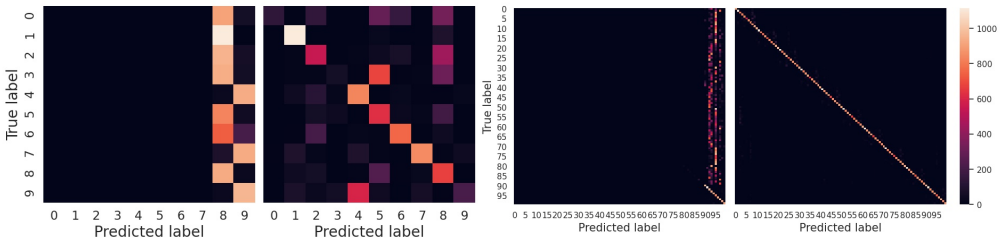

Figure 5: Confusion matrices between ground truth labels and predicted labels at the end of learning on split MNIST (left) and permuted MNIST (right). The lighter the diagonal is, the more accurate the predictions are.

CL approaches, such as EWC, LwF, are not directly applicable to the boundary-free setting as they require the task boundaries to decide when to perform certain consolidation steps.

B Additional Results

Extending the results presented in the main paper, here we further compare EBMs with the baseline models by providing additional experiments of their performance. We first investigate the effect of different numbers of tasks in Section B.1. We then evaluate the computational complexity for inference in Section B.2 and show the confusion matrices between the ground truth labels and model predictions in Section B.3.

B.1 Effect of different numbers of tasks

To test the generality of our proposed EBMs, we repeat the boundary-aware experiments on CIFAR-100 for different number of classes per task in Table 6. In the main paper Table 1, the CIFAR-100 dataset was split into 10 tasks, resulting in 10 classes per task. Here we additionally split CIFAR-100 into 5 tasks (i.e., 20 classes per task), 20 tasks (i.e., 5 classes per task), and 50 tasks (i.e., 2 classes per task). Our EBM substantially outperforms the baselines on all settings.

B.2 Inference computational complexity

One drawback of the current EBM formulation is that we need to compute the energy between a data point and each class during inference. However, in the actual implementation, we found that the difference between the softmax-based classifier (SBC) and our EBM is minor.

We test the inference time needed for a single forward sweep of the softmax-based classifier (SBC) and our EBM on the split MNIST and CIFAR-10 datasets. We use TITAN Xp 12G GPUs for testing. We evaluate every single model on 1 single GPU. Each experiment is run 5 times with different random seeds.

On the split MNIST dataset, the average inference time of SBC is 0.00022s while EBM is 0.00027s. On CIFAR-10, the average inference time of SBC is 0.00059s while EBM is 0.00063s. The difference between SBC and EBM is less than 0.0001s for a single forward pass.

B.3 Class confusion matrix at the end of learning

We show confusion matrices for EBMs and SBC. A confusion matrix illustrates the relationship between the ground truth labels and the predicted labels. Figure 5 shows the confusion matrices after training on all the tasks on the split

Table 7: The model architectures used on split MNIST. $h = 400$.

(a) The architecture of EBMs.

x = FC(784, h) (x)
y = Embedding (y)
x = x * Norm2 (y) + x
x = ReLU (x)
out = FC(h, 1) (x)

(b) The architecture of baseline models.

x = FC(784, h) (x)
x = ReLU (x)
x = FC(h, h) (x)
x = ReLU (x)
out = FC(h, 10) (x)

Table 8: The model architectures used on permuted MNIST. $h = 1000$.

(a) The architecture of EBMs.

x = FC(1024, h) (x)
y = Embedding (y)
x = x * Norm2 (y) + x
x = ReLU (x)
out = FC(h, 1) (x)

(b) The architecture of baseline models.

x = FC(1024, h) (x)
x = ReLU (x)
x = FC(h, h) (x)
x = ReLU (x)
out = FC(h, 100) (x)

MNIST dataset and permuted MNIST dataset. The standard classifier tends to only predict the classes from the last task (class 8, 9 on the split MNIST dataset and classes 90-100 on the permuted MNIST dataset). The EBMs on the other hand have high values along the diagonal, which indicates that the predicted results match the ground truth labels for all the sequentially learned tasks.

The poor performance of SBC might be caused by an imbalance between bias and norms of vectors biased toward the last classes in a fully-connected layer. (Wu et al., 2019) and (Hou et al., 2019) correct the classification result by reserving the samples for old classes and/or training extra correction layers. (Wu et al., 2019) train a bias correction layer using the data from both the old classes and the new classes. (Hou et al., 2019) also uses old data to rebalance the training process. In contrast, EBM can predict the classes correctly without storing previous data and without training extra layers to correct the bias. (Lesort et al., 2021) investigates different sources of performance decrease for the output layer and proposes three different methods to address the catastrophic forgetting in the output layer: a simplified weight normalization layer, two masking strategies, and an alternative to Nearest Mean Classifier using median vectors. Some techniques may also be applied to EBM, such as weight normalization.

C Model Architectures

In this section, we provide details of the model architectures used on different datasets.

Images from the split MNIST and permuted MNIST datasets are grey-scale images. The baseline models for these datasets, similar as in (van de Ven & Tolias, 2019), consist of several fully-connected layers. We use model architectures with similar number of parameters for EBMs. The model architectures of EBMs on the split MNIST dataset and permuted MNIST dataset are the same, but have different input and output dimensions and hidden sizes. The model architectures of EBMs and baseline models on the split MNIST dataset are shown in Table 7. The model architectures of EBMs and baseline models on the permuted MNIST dataset are shown in Table 8.

Images from the CIFAR-10 and CIFAR-100 datasets are RGB images. For CIFAR-10, we use a small convolutional network for both the baseline models and EBMs. We investigate different architectures to search for the effective label conditioning on EBMs training as described in the main paper Section 5.1.3. The model architectures of EBMs and baseline models on the CIFAR-10 dataset are shown in Table 9. The model architectures used on the CIFAR-100 dataset are shown in Table 10.

D More Details of EBMs using Replay

In the main paper Section 5.1.1, we show that the proposed EBM formulation can be combined with existing continual learning methods, such as exact replay. In this section, we provide more experimental details of EBMs using replay.

When training a new task, we mix the new data with data sampled from a memory buffer that stores examples of previously learned tasks to train the models. The examples stored in the buffer are randomly selected from the classes

Table 9: The model architectures used on the CIFAR-10 dataset.

(a) EBM: **Beginning (V1)**
Input: x, y
y = Embedding(N, 3) (y)
y = Softmax(dim=-1) (y)
y = y * y.shape[-1]
x = x * y
x = Conv2d(3×3, 3, 32) (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 32) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = Conv2d(3×3, 32, 64) (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 64) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = FC(2304, 1024) (x)
x = ReLU(x)
out = FC(1024, 1) (x)

(b) EBM: **Middle (V2)**
Input: x, y
x = Conv2d(3×3, 3, 32) (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 32) (x)
x = ReLU (x)
y = Embedding(N, 32) (y)
y = Softmax(dim=-1) (y)
y = y * y.shape[-1]
x = x * y
x = Maxpool(2, 2) (x)
x = Conv2d(3×3, 32, 64) (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 64) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = FC(2304, 1024) (x)
x = ReLU(x)
out = FC(1024, 1) (x)

(c) EBM: **Middle (V3)**
Input: x, y
x = Conv2d(3×3, 3, 32) (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 32) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = Conv2d(3×3, 32, 64) (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 64) (x)
x = ReLU (x)
y = Embedding(N, 64) (y)
y = Softmax(dim=-1) (y)
y = y * y.shape[-1]
x = x * y
x = Maxpool(2, 2) (x)
x = FC(2304, 1024) (x)
x = ReLU(x)
out = FC(1024, 1) (x)

(d) EBM: **End Fix Softmax (V4)**
Input: x, y
x = Conv2d(3×3, 3, 32) (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 32) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = Conv2d(3×3, 32, 64) (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 64) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = FC(2304, 1024) (x)
y = Random Projection (y)
y = Softmax(dim=-1) (y)
y = y * y.shape[-1]
x = x * y
out = FC(1024, 1) (x)

(e) EBM: **End Softmax (V4)**
Input: x, y
x = Conv2d(3×3, 3, 32) (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 32) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = Conv2d(3×3, 32, 64) (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 64) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = FC(2304, 1024) (x)
y = Embedding(N, 1024) (y)
y = Softmax(dim=-1) (y)
y = y * y.shape[-1]
x = x * y
out = FC(1024, 1) (x)

(f) Baseline models
Input: x
x = Conv2d(3×3, 3, 32) (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 32) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = Conv2d(3×3, 32, 64) (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 64) (x)
x = ReLU (x)
x = Maxpool(2, 2) (x)
x = FC(2304, 1024) (x)
x = ReLU (x)
x = FC(1024, 1024) (x)
x = ReLU (x)
out = FC(1024, 10) (x)

encountered so far, and the available memory budget k is equally divided over all classes encountered so far. On the split MNIST, permuted MNIST, and CIFAR-10 datasets, we use a memory budget of $k = 1000$. On the CIFAR-100 datasets, we use a memory budget of $k = 2000$.

Table 10: The model architectures used on the CIFAR-100 dataset. Following (van de Ven et al., 2020), the convolutational layers were pre-trained on CIFAR-10 for all models. The 'BinaryMask'-operation fully gates a randomly selected subset of $X\%$ of the nodes, with a different subset for each y. Hyperparameter X was set using a gridsearch.

(a) The architecture of EBMs.

Input: x, y
x = Conv2d(3×3, 3, 16) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 16, 32) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 64) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 128) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 128, 256) (x)
x = FC(1024, 2000) (x)
x = ReLU (x)
x = BinaryMask (x, y)
x = FC(2000, 2000) (x)
x = ReLU (x)
x = BinaryMask (x, y)
out = FC(2000, 1) (x)

(b) The architecture of baseline models.

Input: x
x = Conv2d(3×3, 3, 16) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 16, 32) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 32, 64) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 64, 128) (x)
x = BatchNorm (x)
x = ReLU (x)
x = Conv2d(3×3, 128, 256) (x)
x = FC(1024, 2000) (x)
x = ReLU (x)
x = FC(2000, 2000) (x)
x = ReLU (x)
out = FC(2000, 100) (x)

Taking the split MNIST dataset as an example, after training the first task, we randomly select 1000 data-label pairs from the first task and save them in the replay buffer. Then we train the model on the second task. In each training batch, we randomly sample a set of data-label pairs from the second task, and randomly sample a set of data-label pairs from the replay buffer. We compute the final loss by adding the loss of data from the current task $\mathcal{L}_{\text{CD current}}(\boldsymbol{\theta}; \mathbf{x}, y)$ and the loss of data from the replay buffer $\mathcal{L}_{\text{CD replay}}(\boldsymbol{\theta}; \mathbf{x}, y)$ using the following equation:

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}; \mathbf{x}, y) = \mathcal{L}_{\text{CD current}}(\boldsymbol{\theta}; \mathbf{x}, y) + \mathcal{L}_{\text{CD replay}}(\boldsymbol{\theta}; \mathbf{x}, y). \tag{7}$$

After we finishing training the second task, we randomly select 500 data-label pairs from the replay buffer and randomly sample 500 data-label pairs from the second task and update the replay buffer using the new sampled data-label pairs. Note we always keep the number of data-label pairs in the replay buffer at 1000. Then we train the model on the third task. In each training batch, we randomly sample a set of data-label pairs from the third task, and randomly sample a set of data-label pairs from the replay buffer. We compute the final loss using Equation (7). We use such a strategy to train the models until the final task.

The baseline model, *i.e.* "SBC+ER", in the main paper Table 1 uses replay in the same way. The only difference is that "SBC+ER" uses the cross-entropy loss as described in the main paper Section 3.1 but "EBM+ER" uses the contrastive divergence training objective.

We use the similar training regimes for EBMs and SBC. On split MNIST, permuted MNIST, and CIFAR-10, we trained for 2000 iterations per task. On CIFAR-100, we trained for 5000 iterations per task. In each training batch, we sampled 128 data-label pairs from the current task and 128 data-label pairs from the replay buffer (start from the second task) to train the model. All experiments used the Adam optimizer with learning rate $1e^{-4}$.

Table 11: The model architectures used for the model capacity analysis. h are 512, 1024, and 4096 for the small, medium and large network, respectively.

(a) The architecture of EBMs.

x = FC($32 \times 32 \times 3$, h) (x)
x = ReLU (x)
y = Embedding (y)
x = x * y
x = ReLU (x)
out = FC(h, 1) (x)

(b) The architecture of the standard classifier.

x = FC($32 \times 32 \times 3$, h) (x)
x = ReLU (x)
x = FC(h, h) (x)
x = ReLU (x)
out = FC(h, 10) (x)

E Additional Analyses

We show the model capacity comparisons in Section E.1.

E.1 Model capacity

Another hypothesized reason for why EBMs suffer less from catastrophic forgetting than standard classifiers is potentially their larger effective capacity. We thus test the model capacity of the standard classifier and EBMs on both the generated images and natural images.

Model capacity on generated images. We generate a large randomized dataset of 32×32 images with each pixel value uniformly sampled from -1 to 1. Each image is then assigned a random class label between 0 and 10. We measure the model capacity by evaluating to what extent the model can fit a such dataset. For both the standard classifier and the EBM, we evaluate three different sizes of models (small, medium, and large). For a fair comparison, we control the EBM and classifier have similar number of parameters. The Small EBM and SBC have $2,348,545$ and $2,349,032$ parameters respectively. The medium models have $5,221,377$ (EBM) and $5,221,352$ (SBC) parameters while the large models have $33,468,417$ (EBM) and $33,465,320$ (SBC) parameters. The model architectures of EBMs and classifiers are shown in Table 11.

The resulting training accuracies are shown in Figure 6 with the number of data ranges from one to five millions. Given any number of datapoints, EBM obtains higher accuracy than the classifier, demonstrating that EBM has larger capacity to memorize data given a similar number of parameters. The gap between EBM and SBC increases when the models become larger. The larger capacity of EBM potentially enables it to memorize more data and mitigate the forgetting problem.

Model capacity on natural images. We also compare classifiers and EBMs on natural images from CIFAR-10. Each image is assigned a random class label between 0 and 10. We use the same network architecture as in Table 11, but with a hidden unit size of $h = 256$. Since there are only $50,000$ images on CIFAR-10, we use a small classifier and EBM and train them on the full dataset. After training 100000 iterations, the EBM obtains a top-1 prediction accuracy of 82.81%, while the classifier is 42.19%. We obtain the same conclusion on natural images that EBM has a larger capacity to memorize data given a similar number of parameters.

F Alternative EBM Training Objective

F.1 Energy-based models for classification

In the main paper Section 4.3, we introduce an alternative EBM training objective. Here we provide more details about this training objective. We propose to use the Boltzmann distribution to define the joint likelihood of image $\mathbf{x}$ and label y:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, y) = \frac{\exp(-E_{\boldsymbol{\theta}}(\mathbf{x}, y))}{Z(\boldsymbol{\theta})}, \qquad Z(\boldsymbol{\theta}) = \sum_{\mathbf{x}' \in \mathcal{X}, y' \in \mathcal{Y}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}', y')) \tag{8}$$

where $E_{\boldsymbol{\theta}}(\mathbf{x}, y) : (\mathbb{R}^D, \mathbb{N}) \to \mathbb{R}$ is the energy function that maps an input-label pair $(\mathbf{x}, y)$ to a scalar energy value, and $Z(\boldsymbol{\theta})$ is the partition function for normalization.

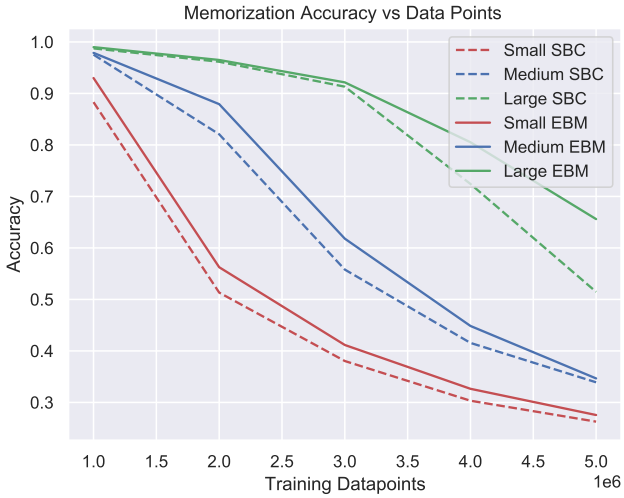

Figure 6: Model capacity of the standard classifier (SBC) and EBM using different model sizes.

Training. We want the distribution defined by $E_{\boldsymbol{\theta}}$ to model the joint data distribution p_D, which we do by minimizing the negative log likelihood of the data

$$\mathcal{L}_{\text{ML}}(\boldsymbol{\theta}) = \mathbb{E}_{(x,y)\sim p_D}[-\log p_{\boldsymbol{\theta}}(\mathbf{x}, y)]. \tag{9}$$

with the expanded form:

$$\mathcal{L}_{\text{ML}}(\boldsymbol{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim p_D}\left[E_{\boldsymbol{\theta}}(\mathbf{x}, y) + \log(\sum_{\mathbf{x}'\in\mathcal{X}, y'\in\mathcal{Y}} e^{-E_{\boldsymbol{\theta}}(\mathbf{x}', y')})\right]. \tag{10}$$

Equation (10) minimizes the energy of $\mathbf{x}$ at the ground truth label y and minimizes the overall partition function by increasing the energy of any other randomly paired $\mathbf{x}'$ and y'.

Inference. Given an input $\mathbf{x}$, the class label predicted by our EBMs is the class with the smallest energy at $\mathbf{x}$:

$$\hat{y} = \arg\min_{y'\in\mathcal{Y}} E_\theta(\mathbf{x}, y'), \tag{11}$$

F.2 Energy-based models for continual learning

As described in the main paper Section 4.3, directly maximizing energy across all labels of a data point $\mathbf{x}$ raises the same problem as the softmax-based classifier models that the old classes are suppressed when training a model on new classes and thus cause catastrophic forgetting. Inspired by (Hinton, 2002), we find that the contrastive divergence approximation of Equation (10) can mitigate this problem and lead to a simpler equation. We approximate Equation (10) by sampling a random pair of image $\mathbf{x}'$ and label y' from the current training batch to approximate the partition function. Our training objective is given by:

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}; \mathbf{x}, y) = \mathbb{E}_{(\mathbf{x},y)\sim p_D}\left[E_{\boldsymbol{\theta}}(\mathbf{x}, y) - E_{\boldsymbol{\theta}}(\mathbf{x}', y')\right], \tag{12}$$

where y is the ground truth label of data $\mathbf{x}$.

This training objective is reminiscent of the contrastive divergence training objective used to train EBMs in the main paper Equation (5). The major difference is that we utilize both images and labels from the current batch as our contrastive samples instead of just labels used in the main paper. We show in the experiments that using the proposed contrastive training objective in Equation (12) can further improve the continual learning performance.

F.3 Inference

We use the same inference methods as described in the main paper to perform the *Class-IL* evaluation on the continual learning datasets. The model predicts the class label $\hat{y}$ of a data point $\mathbf{x}$ from all class labels $\mathcal{Y}$, where $\mathbf{x}$ is one data point with an associated discrete label $y \in \mathcal{Y}$. The MAP estimate is

$$\hat{y} = \arg\min_{y'} E_\theta(\mathbf{x}, y'), \quad y' \in \mathcal{Y}, \tag{13}$$

where E_θ is the energy function with parameters θ after training on all the training data.

Table 12: Evaluation of class-incremental learning on the *boundary-aware* setting on the split MNIST and permuted datasets. Each experiment is performed at least 10 times with different random seeds, the results are reported as the mean ± SEM over these runs. Note our comparison is restricted to methods that do not replay stored or generated data.

Method	**splitMNIST**	**permMNIST**
SBC	19.90 ± 0.02	17.26 ± 0.19
EBM	**53.12 ± 0.04**	**87.58 ± 0.50**
EBM Alt CD	**60.14 ± 1.66**	**89.15 ± 0.89**
EWC (Kirkpatrick et al., 2017)	20.01 ± 0.06	25.04 ± 0.50
Online EWC (Schwarz et al., 2018)	19.96 ± 0.07	33.88 ± 0.49
SI (Zenke et al., 2017)	19.99 ± 0.06	29.31 ± 0.62
LwF (Li & Hoiem, 2017)	23.85 ± 0.44	22.64 ± 0.23
MAS (Aljundi et al., 2019b)	19.50 ± 0.30	-
BGD (Zeno et al., 2018)	19.64 ± 0.03	84.78 ± 1.30

F.4 Comparisons with existing methods

We follow the experiments performed in the main paper and evaluate the *Class-IL* on the split MNIST (Zenke et al., 2017) and permuted MNIST (Kirkpatrick et al., 2017) datasets on the *Boundary-Aware* setting.

We compare EBMs using different training objectives and the baseline approaches in Table 12. All the baselines and EBMs use similar model architectures with similar number of model parameters for fair comparisons. "EBM" means the results of the training objective used in the main paper Equation (5). "EBM Alt CD" represents the alternative training objective described in Equation (12). EBMs have a significant improvement over the baseline methods on all the datasets, showing that EBMs forget less when updating models for new tasks. "EBM Alt CD" can further improve the continual learning performance.