ENERGY-BASED MODELS FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

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 have a natural way to support a dynamically-growing number of tasks and classes and less interference with old tasks. We show that EBMs are adaptable to a more general continual learning setting where the data distribution changes without the notion of explicitly delineated tasks. We also find that EBMs outperform the baseline methods by a large margin on several continual learning benchmarks. These observations point towards EBMs as a class of models naturally inclined towards the continual learning regime.

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). In recent years, CL has seen increased attention, particularly in the context of classification problems. A crucial characteristic of continual learning is the ability to learn new data without forgetting prior data. Models must also be able to incrementally learn new tasks, without necessarily having a notion of an explicit task identity. However, standard neural networks (He et al., 2016; Simonyan & Zisserman, 2014; Szegedy et al., 2015) experience the catastrophic forgetting problem 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), larger models (Shin et al., 2017), or additional objectives and regularization (Kirkpatrick et al., 2017; Schwarz et al., 2018; Zenke et al., 2017; Maltoni & Lomonaco, 2019) which can constrain the wide applicability of these methods.

In this work, we primarily focus on classification tasks. These tasks are usually tackled by utilizing normalized probability distribution 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 seperate classes (Grathwohl et al., 2019). Training becomes a wake-sleep process, where the energy of an input data and its ground truth label is decreased while the energy of the input and another selected class is increased. This offers freedom to choose what classes to update in the CL process. By contrast, the cross entropy objective reduces the likelihood of *all* negative classes when given a new input, creating updates that lead to forgetting.

The energy function, which maps a data and class pair to a scalar energy, also allows us to learn a dynamically changing number of classes by varying the class part of the input. This is useful in the continual learning setting, where number of classes and tasks can grow and the total number of classes may not be known ahead of time. By contrast, traditional classification methods assume a fixed number of classes and thus cannot incorporate new classes as learning progresses.

This functional form of an energy function can further let the model select and filter portions of the input that are relevant towards the classification on hand. We show that 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 a data by learning a conditional gain based on the input 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.

These unique benefits are applicable across a range of continual learning tasks. Most existing works (Kirkpatrick et al., 2017; Zhao et al., 2020) toward continual learning typically learn a sequence of distinct tasks with clear task boundaries (*Boundary-Aware*). Many of these methods depend on knowing the task boundaries that can provide proper moments to consolidate knowledge. However, this scenario is not very common in the real world, and a more natural scenario is the *Boundary-Agnostic* setting (Zeno et al., 2018; Rajasegaran et al., 2020), in which data gradually changes without a clear notion of task boundaries. This setting has also been used as a standard evaluation in the continual RL (Al-Shedivat et al., 2017; Nagabandi et al., 2018). Many common CL methods are not applicable to the *Boundary-Agnostic* scenario as the task boundaries are unknown or undefined. In contrast, EBMs are readily applied to this setting without any modification and are able to support both *Boundary-Aware* and *Boundary-Agnostic* settings.

There are four primary contributions of our work. First, we introduce energy-based models for classification CL problems in both boundary-aware and boundary-agnostic regimes. Secondly, we use the standard contrastive divergence training procedure and show that it supports dynamically growing number of classes and significantly reduces catastrophic forgetting. Thirdly, we propose to learn new conditional gains during the training process which makes EBMs parameter updates cause less interference with old data. Lastly, we show that in practice EBMs bring a significant improvement on four standard CL benchmarks, split MNIST, permuted MNIST, CIFAR-10, and CIFAR-100. These observations towards EBMs as a class of models naturally inclined towards the CL regime.

2 Related work

In recent years, numerous methods have been proposed for CL. Here we broadly partition these methods into three categories: task-specific, regularization, and replay-based approaches.

Task-specific methods. One way to reduce interference between tasks is by using different parts of a neural network for different problems. For a fixed-size network, such specialization could be achieved by learning a separate mask for each task (Fernando et al., 2017; Serra et al., 2018) or by *a priori* defining a different, random mask for every task to be learned (Masse et al., 2018). Other methods let a neural network grow or recruit new resources when it encounters new tasks, examples of which are progressive neural networks (Rusu et al., 2016) and dynamically expandable networks (Yoon et al., 2017). Although they are generally successful in reducing catastrophic forgetting, an important disadvantage is that they require knowledge of the task identity at both training and test time. This means that these methods are not suitable for class-incremental learning.

Regularization-based methods. Regularization is used in continual learning to encourage stability of those aspects of the network that are important for previously learned tasks. A popular strategy is to add a regularization loss to penalise changes to model parameters that are important for previous tasks. EWC (Kirkpatrick et al., 2017)) and online EWC (Schwarz et al., 2018) evaluate the importance of each parameter using the diagonal elements in the fisher information matrices, while SI (Zenke et al., 2017) tracks the past and current parameters and estimates their importance online. An alternative strategy is to regularize the network at the functional level. Learning without Forgetting (Li & Hoiem, 2017) uses knowledge distillation to encourage stability of the network's learned input-output mapping. However, these regularization-based approaches gradually reduce the model's capacity for learning new tasks and they have been shown to consistently fail in the class-incremental learning (Farquhar & Gal, 2018; van de Ven & Tolias, 2019).

Replay methods. To preserve knowledge, replay methods periodically rehearse previously acquired information during training (Robins, 1995). Exact or experience replay store data from previous tasks and revisit them when training on a new task. Although this might seem straightforward, critical non-trivial questions are how to select the data to be stored as well as exactly how to use them (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017; Mundt et al., 2020). An alternative to storing data is to generate the data to be replayed. A generative model (Shin et al., 2017) is sequentially trained to generate input samples representative of those from previously seen tasks. While both types of replay can prevent catastrophic forgetting in both task- and class-incremental learning settings, an important disadvantage is that 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, our EBMs reduce catastrophic forgetting without requiring knowledge of task-identity, without gradually restricting the model's learning capabilities and without using stored data.

3 BACKGROUND

In this section, we will introduce traditional continual learning methods, which are typically modified based on the feed-forward classifier. We will show the limitation of using such structures and how EBMs can be applied to these problems in the next section.

3.1 CONTINUAL LEARNING WITH FEED-FORWARD CLASSIFIER

Define T as a classification task with inputs $\mathbf{x}$ and associated discrete classification labels $\mathbf{y} \in Y$, where $Y = \{1, \ldots, N\}$ contains N classes. Traditional classifier approaches to continual learning typically predict the probabilities of all the N classes using a feed-forward network

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{\exp([f(\mathbf{x})]_{\mathbf{y}})}{\sum \exp([f(\mathbf{x})]_{\mathbf{y}_i})}, \quad \mathbf{y}_i \in Y \tag{1}$$

The input is a data $\mathbf{x} \in \mathbb{R}^D$ and the output is probabilities of all classes $p_\theta \in \mathbb{R}^N$. $f(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^N$ is the feed-forward function that maps data x into a N-dimensional vector. The final layer in such classifier networks is predefined and is fixed to predict up to N different classes.

Existing continual learning methods (Kirkpatrick et al., 2017; Masse et al., 2018) tackle the continual learning problem typically using a feed-forward classifier. However, such classifier architectures have the following challenges 1) They must assume a fixed number of classes to define the final network layer and thus cannot incorporate new classes as learning progresses. But in many continual learning problems, each subsequent task adds a new set of classes and may not be possible to predefine ahead of time. 2) In the *Class-IL* setting, where models predict the label of a given data from all classes (Tao et al., 2020a; He et al., 2018), the classifier architectures compute the Softmax and cross-entropy loss over all seen classes. When training on a new task, they improve the likelihood of ground truth classes but suppress the likelihood of old classes since they are all negative classes for data in the current task. This operation introduces competitive, winner-take-all dynamics that impact the strength of old classes and thus make the classifier forget past tasks.

3.2 ENERGY-BASED MODELS

Energy based models (LeCun et al., 2006) are a class of maximum likelihood models that define the likelihood of a data point $\mathbf{x} \in \mathbb{R}^D$ using the Boltzmann distribution

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

where $E_\theta(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}$, known as the energy function, maps each input data to a scalar, and $Z(\mathbf{x})$ is the partition function over all data points.

4 CONTINUAL LEARNING WITH ENERGY-BASED MODELS

In this section, we introduce EBMs and show how they can overcome the above mentioned problems. Note our analysis is based on the *Class-IL* setting which is one of the most natural and also the hardest settings used in CL (Tao et al., 2020a; He et al., 2018; Tao et al., 2020b).

4.1 ENERGY-BASED MODELS FOR CLASSIFICATION

In the classification problem, the likelihood of $\mathbf{y}$ conditioned on $\mathbf{x}$ is defined using our EBM as

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{\exp(-E_\theta(\mathbf{x}, \mathbf{y}))}{Z(\mathbf{y}|\mathbf{x})}, \quad Z(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{y}_i} \exp(-E_\theta(\mathbf{x}, \mathbf{y}_i)) \tag{3}$$

where $E_\theta(\mathbf{x}, \mathbf{y}) : (\mathbb{R}^D, \mathbb{N}) \rightarrow \mathbb{R}$ is the energy function that maps $\mathbf{x}$ and $\mathbf{y}$ to a scalar energy value, and $Z(\mathbf{y}|\mathbf{x})$ is the normalization constant given $\mathbf{x}$. $\mathbf{y}_i$ is sampled from classes in the current batch Y_B and the probability of $p_\theta(\mathbf{y}|\mathbf{x})$ must sum to 1 over all $\mathbf{y}$ given $\mathbf{x}$. To minimize the negative log likelihood of the ground truth label $\mathbf{y}^+$, we minimize the energy of $\mathbf{x}$ and the ground truth label $\mathbf{y}^+$ while increasing the energy of $\mathbf{x}$ at other labels. We use the following loss and its corresponding gradient

$$\mathcal{L}_{\text{ML}}(\theta) = \mathbb{E}_{\mathbf{x} \sim T}[E_\theta(\mathbf{x}, \mathbf{y}^+) + \log \sum_{\mathbf{y}_i \sim Y_B} \exp(-E_\theta(\mathbf{x}, \mathbf{y}_i))]. \tag{4}$$

$$\frac{\partial \mathcal{L}_{\text{ML}}}{\partial \theta} = \mathbb{E}_{\mathbf{x} \sim T}\left(\frac{\partial E_\theta(\mathbf{x}, \mathbf{y}^+)}{\partial \theta} - \mathbb{E}_{\mathbf{y}_i \sim Y_B}\left[\frac{\partial E_\theta(\mathbf{x}, \mathbf{y}_i)}{\partial \theta}\right]\right). \tag{5}$$

In the continual learning setting, we assume classes in Y_B are uniformly distributed in every new batch (this assumption is adaptable for both the *Boundary-Aware* and *Boundary-Agnostic* settings). Inspired by the contrastive divergence proposed by (Hinton, 2002), we approximate the above objective by sampling one negative class $\mathbf{y}_i \neq \mathbf{y}^+$ from the current batch.

As opposed to the classifier architectures which output a fixed N-dimensional probability vector of all classes, this formulation gives us substantial freedom in the choice of class labels during training as the classes labels $\mathbf{y}_i$ for each data $\mathbf{x}$ are sampled from Y_B. EBMs allow any number of classes in new batches and avoid needing to predefine the number of classes in advance, as is necessary with traditional classification models.

Even assuming a known fixed number of classes and tasks, classifier models suppress the strength of old classes and thus make the classifier forget past tasks as mentioned in Section 3.1. In contrast, EBMs define the probability without normalizing over all classes but instead sampling a single negative class from the current training batch. This operation cause less interference with previous classes without using any extra information which contributes to why EBMs suffer less from catastrophic forgetting.

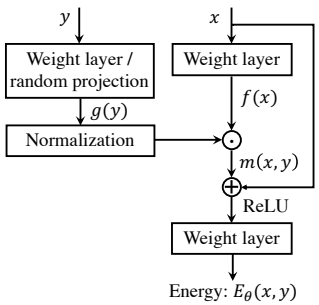

Figure 1: Model architecture used in EBM. EBM takes a data $\mathbf{x}$ and a class $\mathbf{y}$ as input and outputs their energy value.

Additionally, the choice of model architectures also becomes more flexible. The traditional classification models can only feed in $\mathbf{x}$ as input and the classification results only depend on the neural network trained on the current data $\mathbf{x}$. Thus the models usually fail on the old data after updating network parameters trained on the new data. In contrast, EBMs have many different ways to combine $\mathbf{x}$ and $\mathbf{y}$ in the energy function $E_\theta(\mathbf{x}, \mathbf{y})$ with the only requirement that $E_\theta(\mathbf{x}, \mathbf{y}) : (\mathbb{R}^D, \mathbb{N}) \rightarrow \mathbb{R}$. In EBMs, we can treat $\mathbf{y}$ as an attention filter or gate to select the most relevant information between $\mathbf{x}$ and $\mathbf{y}$ as described in Section 4.2.

Since the negative sample can be sampled from the current batch, EBMs do not require knowledge of the task on hand, allowing applications across different continual learning scenarios, including those when the underlying task is unclear. More broadly speaking, EBMs can go out of the scope of classification problems, and similar ideas can be applied to regression (Gustafsson et al., 2020), generation (Du & Mordatch, 2019; Du et al., 2020), and reinforcement learning tasks (Parshakova et al., 2019), although we do not explore them in this work.

4.2 Energy network.

To compute the energy of any data $\mathbf{x}$ and class label $\mathbf{y}$ pair, we use $\mathbf{y}$ 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 $\mathbf{y}$. In Figure 1, we first send $\mathbf{x}$ into a small network to generate the feature $f(\mathbf{x})$. The label $\mathbf{y}$ is mapped into a same dimension feature space $g(\mathbf{y})$ using a FC layer or a random projection. We use the Softmax function over each channel of $g(\mathbf{y})$ and combine it with $\mathbf{x}$:

$$m_c(\mathbf{x}, \mathbf{y}) = f_c(\mathbf{x}) \cdot \frac{\exp(g_c(\mathbf{y}))}{\sum_j \exp(g_j(\mathbf{y}))}, \tag{6}$$

where f_c and g_c are the cth channel of $f(\mathbf{x})$ and $g(\mathbf{y})$ respectively. The output is finally sent to a fully connected layer to generate the energy value $E_\theta(\mathbf{x}, \mathbf{y})$. For more details on our model architectures, see Appendix A.6.

The classification results not only rely on the network trained on data point $\mathbf{x}$ as in the feed-forward classifiers, but also $g(\mathbf{y})$. We show that such structure has less interference with previous data in Section 5.1.5. EBM allows any number of classes in new batches by simply training a new conditional gain $g(\mathbf{y})$ for the new classes and generating its energy value with data point $\mathbf{x}$. This formulation gives us freedom to learn new classes without pre-defining their number in advance and mitigates the catastrophic forgetting problem.

Class inference. During inference, the model must predict a class label from all classes seen so far. let $\mathbf{x}_k$ be one data point from a batch B_k with an associated discrete label $\mathbf{y} \in Y_k$, where Y_k contains classes in B_k. There are $Y = \bigcup_{k=1}^{K} Y_k$ different classes in total after seeing all the batches. The MAP estimate is

$$\hat{\mathbf{y}} = \arg\min_{\mathbf{y}} E_{\theta_K}(\mathbf{x}_k, \mathbf{y}), \quad \mathbf{y} \in \bigcup Y_k, \tag{7}$$

Table 1: Evaluation of the *Class-IL* performance on the *Boundary-Aware* setting compared to different baselines. The test accuracy on four datasets are reported. Each experiment is performed 20 times with different random seeds, the results are reported as the mean $\pm$ SEM over these runs. Note both EBMs and baselines do not use extra memory to store data or models for fair comparison.

Method	splitMNIST	permMNIST	Cifar10	Cifar100
CLS	19.91 $\pm$ 0.01	16.95 $\pm$ 0.31	19.06 $\pm$ 0.01	8.18 $\pm$ 0.10
EWC	20.03 $\pm$ 0.09	25.14 $\pm$ 0.71	18.99 $\pm$ 0.01	8.20 $\pm$ 0.09
Online	19.89 $\pm$ 0.02	33.20 $\pm$ 0.74	19.07 $\pm$ 0.01	8.38 $\pm$ 0.15
SI	19.98 $\pm$ 0.03	29.35 $\pm$ 0.90	19.14 $\pm$ 0.02	9.24 $\pm$ 0.22
LwF	20.53 $\pm$ 0.11	43.88 $\pm$ 0.88	19.20 $\pm$ 0.30	10.71 $\pm$ 0.11
MAS	19.50 $\pm$ 0.30	-	20.25 $\pm$ 1.54	8.44 $\pm$ 0.27
BGD	19.64 $\pm$ 0.03	84.78 $\pm$ 1.30	-	-
EBM	**53.12** $\pm$ **0.04**	**87.78** $\pm$ **0.01**	**40.19** $\pm$ **0.01**	**30.28** $\pm$ **0.28**

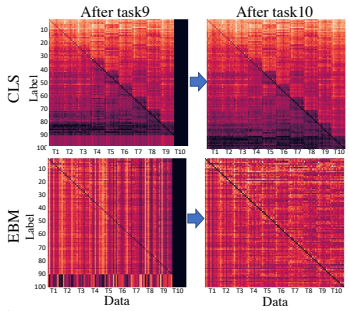

Figure 2: Energy landmaps of CLS and EBMs after training on task T_9 and T_{10} on permuted MNIST. The darker the diagonal is, the better the model is in preventing forgetting previous tasks.

where $E_{\theta_K}(\mathbf{x}_k, \mathbf{y})$ is the energy function with parameters θ_K resulting from training on the batches $\{B_1, \cdots, B_K\}$. The energy function can compute an energy for any discrete class input, including unseen classes. This avoids needing to predefine the number of classes in advance, as is necessary with traditional CL models.

5 Experiments

In this section, we want to investigate several questions, including how does the proposed EBMs perform on different CL settings? Can we qualitatively understand the differences between EBMs and baselines? Is the proposed EBM training objective better than other objectives? Can we apply the EBM training objective to other baseline methods? And finally what is the the best architecture for label conditioning that causes the minimum interference with old data? To answer these questions, we first introduce experiments on the *Boundary-Aware* setting which typically learn a sequence of distinct tasks with clear task boundaries in Section 5.1. We then show that EBMs can be naturally applied to the *Boundary-Agnostic* setting where the data gradually changing in a steaming fashion without the notion of task boundaries in Section 5.2.

5.1 Experiments on Boundary-Aware Setting

5.1.1 Datasets and evalution protocols

Datasets. We evaluate the proposed EBMs on the 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) datasets. We follow existing approaches and separate the datasets into several splits. The split MNIST dataset is obtained by splitting the original MNIST dataset (LeCun et al., 1998) into 5 tasks with each task has 2 classes. It has 60,000 training images and 10,000 test images. The permuted MNIST has 10 tasks, each task with 10 classes. For each task, the original images pixels are randomly permuted to generate 32×32 images. We separate CIFAR-10 into 5 tasks, each task with 2 classes. There are 50,000 training images and 10,000 test images. Similarly, CIFAR-100 is split into ten tasks with each task has 10 classes.

Evaluation protocols. Task-incremental Learning (*Task-IL*), Domain-incremental Learning (*Domain-IL*), and Class-incremental Learning (*Class-IL*) are three common evaluation metrics used by existing works (van de Ven & Tolias, 2019; Prabhu et al., 2020). Most approaches perform fairly good on the first two simpler settings, but fail on the *Class-IL* setting, 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). *Class-IL* predicts the label of a given data from all seen classes. In this paper, we consider *Class-IL* as our evaluation metric.

5.1.2 Comparisons with existing methods

Due to the difficulty on the class incremental setting, most approaches rely an external quota of memory. Such replay based methods (Shin et al., 2017; Rebuffi et al., 2017) use extra memory, larger or extra models, and thus may not be applicable to large datasets due to the prohibitive memory footprint. In this paper, we focus on continual learning *without* a memory footprint.

We compare the proposed method with almost all available baseline models that do not use the replay, including a standard classification model (CLS), EWC (Kirkpatrick et al., 2017), Online EWC (Schwarz et al., 2018), SI (Zenke et al., 2017), LwF (Li & Hoiem, 2017), MAS (Aljundi et al., 2019), and BGD (Zeno et al., 2018). The continual learning results on four datasets are shown in Table 1. All the baselines and EBMs are based on the same model architecture. For split MNIST and permuted MNIST, we use several fully-connected layers as in (van de Ven & Tolias, 2019). For CIFAR-10 and CIFAR-100, we use the convolutional network (see Appendix A.6). For all the baselines and EBMs on CIFAR-100, we pre-train the convolutional layers on CIFAR-10 and only fine-tune the fully-connected layers. Each experiment in Table 1 is run 20 times with different random seeds, with results reported as the mean $\pm$ SEM. EBMs have a significant improvement on all the datasets, showing that EBMs forget less when updating models for new tasks.

5.1.3 Qualitative analysis

Energy landscape. To better understand why EBMs suffer less from catastrophically forgetting, we qualitatively compare the change in energy landscapes of EBMs and baseline methods as the learning progresses. We show the energy landscapes after training on task 9 and task 10 on the permuted MNIST dataset in Figure 2. For the classifier, we show its negative probabilities on all classes. Each datapoint has 100 energy values (EBM) or probabilities (CLS) corresponding to the 100 labels in the dataset and the values are normalized over 100 classes for each datapoint. The dark diagonal elements indicate the model predictions are correct for both old and new data. After training on task T_9, CLS assigns high probabilities to classes from task T_9 (80-90) for almost all the data from task T_1 to T_9. However, the highest probabilities shift to classes from task T_{10} (90-100) after learning task T_{10}. This means classifier tends to assign high probabilities to new classes for both old and new data, indicating forgetting. EBM on the other hand, tends to have low energies across the diagonal, which means that after training on new tasks, EBM still assigns low energies to true labels of previous data. This indicates that EBM is better at learning new tasks without catastrophically forgetting of old tasks.

Predicted class distribution. In Figure 3, we plot the proportional distribution of all seen classes so far on the splitMNIST dataset. Taking the second figure in the first row as an example, it shows the distribution of predicted labels after training on the second task which means there are four seen classes $\{1, 2, 3, 4\}$. For each image from the first and second task, we compute its top-1 class prediction and accumulate the prediction on each class to obtain the proportional distribution graph over all seen classes. Since the number of images in each class are almost the same, the ground truth proportional distribution should be uniform over all the classes. CLS can predict the first two classes uniformly after training on task 1 (first column). However, after learning new tasks, classes from the new task have larger proportions than old classes which means the model tends to predict new classes and forget the old classes. The last figure in row one gives very high proportions for classes from the last task but very low proportions for previously classes. CLS almost forgets all previous tasks and only predict classes from the last task even the input images are from old tasks. EBM covers all seen classes, indicating memorizing the old task and preventing forgetting.

Figure 3: Predicted label distribution after learning each task on the split MNIST dataset. CLS only predicts classes from the current task while EBMs can predict classes for all seen classes.

We provide more analysis about EBMs and baselines from different perspectives in the Appendix, including the confusion matrix in Appendix A.2, the model capacity in Appendix A.3, and parameter importance in Appendix A.4.

5.1.4 Effects of Energy Training Objective on Model Performance

Energy objective effects on EBMs training. We conduct an experiment on the CIFAR-100 dataset to investigate how different training objectives influence the CL results. In Equation 4, the negative labels $\mathbf{y}_i$ ares sampled from Y_B. We test three different sampling strategies in Table 2. The first one uses all seen classes so far as negative labels which is similar to the traditional feed-forward classifiers (**All Neg Seen (V4)**). The second one takes all the classes in the current batch as negative labels (**All Neg Batch (V4)**). The last one randomly selects one class from the current bath as the negative as we reported in Section 4.1 (**1 Neg Batch (V4)**). Note that the negative labels do not include the ground truth class in all these three strategies. We found 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. This conclusion is consistent with our analysis in Section 4.1. Since the loss in Equation 4 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. Using a single negative causes the minimum suppression on negative samples and thus has the best result. Overall, our results indicate that surprisingly, and counterintuitively, directly optimizing the cross-entropy loss used by existing approaches may not be the best way for continual learning.

EBM training objective on other approaches. Next we show that using the EBM training objective can be beneficial to other model types. We modify the training objective of a standard classifier to that of our proposed energy objective. The comparisons between the original approaches and our modified version are listed in Table 3. The new training objective does not suppress the probability of old classes when improving the probability of new classes and thus achieves better results. EBMs still outperform the baselines, implying both the proposed energy objective and label conditioning architecture contribute to why EBMs suffer less from catastrophic forgetting. We also show the testing accuracy curve in Appendix A.1.

5.1.5 Effect of Label Conditioning on Performance

EBMs allow flexibility in integrating data information and label information in the energy function. To investigate where and how to combine the information from $\mathbf{x}$ and $\mathbf{y}$, we conduct a series of experiments on the CIFAR-10 dataset. Table 2 shows four model architectures (V1-V4) that combine $\mathbf{x}$ and $\mathbf{y}$ in the early, middle, and late stages, respectively (see Appendix A.6 for the architecture details). We find combining $\mathbf{x}$ and $\mathbf{y}$ in the late stage (V4) performs the best. We note that instead of learning a feature embedding of label $\mathbf{y}$, 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 in Table 1. Note further that the number of trainable parameters in the "Fix" setting is much lower than that of the baselines. Using a learned feature embedding of $\mathbf{y}$ can further improve the performance. We may also apply different normalization methods over the feature channel of $\mathbf{y}$. We find that Softmax (**End Softmax (V4)**) is better than the L2 normalization (**End Norm2 (V4)**) and no normalization (**End (V4)**).

Table 2: Different EBM training objectives and label conditioning architectures.

Dataset	**Cifar100**
All Neg Seen (V4)	8.07 ± 0.10
All Neg Batch (V4)	29.03 ± 0.53
1 Neg Batch (V4)	$\mathbf{30.28 \pm 0.28}$
Dataset	**Cifar10**
Beginning (V1)	13.69 ± 0.02
Middle(V2)	20.16 ± 0.02
Middle(V3)	18.36 ± 0.02
End Fix (V4)	34.30 ± 0.03
End Fix Norm2 (V4)	33.91 ± 0.02
End Fix Softmax (V4)	35.97 ± 0.01
End (V4)	38.13 ± 0.01
End Norm2 (V4)	37.23 ± 0.02
End Softmax (V4)	$\mathbf{40.19 \pm 0.01}$

Interference with past data. We evaluate the ability of label conditioning (**End Softmax (V4)**) in preventing interference with past data when learning classification on new classes. Formally, let $\mathbf{x}_i$ and $\mathbf{x}_j$ be two data points and θ_i be the model parameters after training on $\mathbf{x}_i$. The model parameters change $\triangle\theta(\mathbf{x}_j)$ after learning new data $\mathbf{x}_j$. We test the difference between the losses $\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i)$ and $\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i + \triangle\theta(\mathbf{x}_j))$. Ideally, we expect $\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i + \triangle\theta(\mathbf{x}_j)) \leq \mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i)$ for $\mathbf{x}_i \neq \mathbf{x}_j$, which means the update on new data has no influence or positive influence on the old data. The first-order expansion gives $\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i + \triangle\theta(\mathbf{x}_j)) \approx \mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i) + \nabla_{\theta_i}\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i)^T \triangle\theta(\mathbf{x}_j)$. To make our desired equality hold, the gradient at $\mathbf{x}_i$ should be $\nabla_{\theta_i}\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i)^T \triangle\theta(\mathbf{x}_j) \leq 0$. $\phi = \nabla_{\theta_i}\mathcal{L}_{\text{ML}}(\mathbf{x}_i \mid \theta_i)^T \triangle\theta(\mathbf{x}_j)$ can be used to measure the influence of updating models on new data on the performance of old data. The smaller the value is, the less negative influence on old data, and thus the better the model is in preventing forgetting.

We compare EBMs with the standard classifier (CLS) and CLS using our EBM training objective (CLS*) (see Appendix A.5 for the implementation details). For results on split MNIST, the average value of ϕ over all tasks of CLS, CLS*, and EBMs are $\phi_{\text{CLS}} = 0.168$, $\phi_{\text{CLS}^*} = 0.004$, and $\phi_{\text{EBM}} = -3.743e^{-5}$, respectively. For CIFAR-10, the results are $\phi_{\text{CLS}} = 1.104$, $\phi_{\text{CLS}^*} = 0.043$, and $\phi_{\text{EBM}} = 4.414e^{-5}$, respectively. We found EBMs have substantially smaller ϕ than CLS and CLS*, which is consistent with our analysis that the conditional gain $g(\mathbf{y})$ serves as an attention filter that could select the relevant information between $\mathbf{x}$ and $\mathbf{y}$ and make parameter updates cause less interference with old data, which contributes to why EBMs suffer less from catastrophic forgetting.

Table 3: Results of baselines using our training objective and their original one.

split MNIST	Ori	Ours
CLS	19.90 ± 0.02	44.98 ± 0.05
EWC	20.01 ± 0.06	50.68 ± 0.04
Online	19.96 ± 0.07	50.99 ± 0.03
SI	19.99 ± 0.06	49.44 ± 0.03
EBM	-	$\mathbf{53.12 \pm 0.04}$
CIFAR-10	Ori	Ours
CLS	19.06 ± 0.01	19.22 ± 0.02
EWC	18.99 ± 0.01	36.51 ± 0.03
Online	19.07 ± 0.01	36.16 ± 0.02
SI	19.14 ± 0.02	35.12 ± 0.02
EBM	-	$\mathbf{40.19 \pm 0.01}$

Table 4: Evaluation of the *Class-IL* performance on the *Boundary-Agnostic* setting compared to different baselines. The test accuracy on four datasets are reported. Each experiment is performed 5 times with different random seeds, the results are reported as the mean $\pm$ SEM over these runs. Note both EBMs and baselines do not use extra memory to store data or models for fair comparison.

Method	**splitMNIST**	**permMNIST**	**Cifar10**	**Cifar100**
CLS	24.03 ± 0.01	21.42 ± 0.01	23.30 ± 0.01	9.85 ± 0.02
Online	39.62 ± 0.04	41.37 ± 0.04	22.53 ± 0.01	9.57 ± 0.02
SI	28.79 ± 0.01	35.71 ± 0.01	26.26 ± 0.01	10.42 ± 0.01
BGD	21.65 ± 0.02	26.15 ± 0.02	17.03 ± 0.01	8.50 ± 0.02
EBM	$\mathbf{81.78 \pm 0.06}$	$\mathbf{92.35 \pm 0.01}$	$\mathbf{49.47 \pm 0.03}$	$\mathbf{34.39 \pm 0.24}$

5.2 Experiments on Boundary-Agnostic Setting

When applying continual learning in real life, boundaries are not usually well defined between different tasks. Most existing continual learning methods are often tailored to the problem setup where a fixed boundary between tasks is defined, and must be informed when the tasks are switched during training. We show that EBMs are able to flexibly perform continual learning across different problem setups, and perform well on the *Boundary-Agnostic* setting as well.

5.2.1 Datasets and evalution protocols

We evaluate our proposed continual learning method on the *Boundary-Agnostic* setting on four datasets, including split MNIST, permuted MNIST, CIFAR-10, and CIFAR-100. We use the code of "continuous task-agnostic learning" proposed by (Zeno et al., 2018) to generate the training/testing data. The transition between tasks goes slowly over time using a sampling function based on the probability of each task. There is a mixture of data from two different tasks during the transition and the proportion of new data gradually increases. Similar to the *Boundary-Aware* setting, We use *Class-IL* as our evaluation metric for the *Boundary-Agnostic* continual learning.

5.2.2 Comparison with existing methods

We focus on the investigation on the performance of different model architectures with similar model size and memory footprint. We do not directly compare with replay-based methods as they use extra memory, larger or extra models. Since there is no knowledge on the number of tasks, previous methods of continual learning that rely on task boundaries are generally inapplicable. One trivial adaptation is to take the core action after every batch step instead of every task. However, doing such adaptation is impractical for most algorithms, such as EWC, because of the large computational complexity. We successfully get the result of CLS, Online EWC, SI, and BGD baselines. Overall classification results on four datasets are shown in Table 4.

All the baselines and EBM are based on the same model architecture 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. EBMs can be naturally applied to both the *Boundary-Aware* and *Boundary-Agnostic* settings without any modification, as EBMs have a very flexible training objective. The experiments demonstrate EBMs have good generalization ability for different continual learning problems. EBMs can naturally handle different number of new classes, new tasks, data streams with and without clear task boundaries.

6 Conclusion

In this paper, we show that energy-based models are a promising class of models towards a variety of different continual learning settings. In contrast to the traditional continual learning approaches that can only learn a fixed number of tasks and classes, EBMs have substantial freedom to learn new tasks using the contrasting divergence training and conditional gains learned from new classes. We show that EBMs exhibit many desirable characteristics to prevent catastrophic forgetting in CL. We experimentally show EBMs obtain the state-of-the-art performance on the *Class-IL* setting on multiple benchmarks.

REFERENCES

Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. *arXiv preprint arXiv:1710.03641*, 2017.

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, 2019.

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.

Fredrik K Gustafsson, Martin Danelljan, Goutam Bhat, and Thomas B Schön. Energy-based models for deep probabilistic regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 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.

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

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.

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. 2006.

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.

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

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

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

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.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

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

Tetiana Parshakova, Jean-Marc Andreoli, and Marc Dymetman. Distributional reinforcement learning for energy-based sequential models. *arXiv preprint arXiv:1912.08517*, 2019.

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.

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 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.

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.

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.

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

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.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13208–13217, 2020.

A APPENDIX

In Section 5.1, we show the comparisons between EBMs and baselines and provide qualitative analysis of their performance. This includes the testing accuracy curve in Section A.1, confusion matrix between the ground truth labels and model predictions in Section A.2, model capacity comparisons in Section A.3, and parameter importance measurement in Section A.4. In Section A.5, we provide more details of the label conditioning analysis mentioned in Section 5.1.5. Finally, Section A.6 shows the model architectures we used on different datasets.

A.1 TESTING ACCURACY CURVE.

In Section 5.1.4, we show that the proposed EBM training objective is also applicable to baseline approaches and improves their performance significantly on the *Class-IL* setting. Figure 4 shows the testing accuracy of each task as the training progresses. We compare the standard classifier (CLS), classifier using our training objective (CLS*), and our EBMs. The left figures show results on the split MNIST dataset while the right figures show results on the permuted MNIST dataset. We observe that the accuracy of old tasks in CLS drop sharply when learning new tasks, while the EBM training objective can mitigate the forgetting problem. The curve on EBMs drops even slower than CLS and CLS*, implying the proposed energy objective can mitigate the catastrophic forgetting problem.

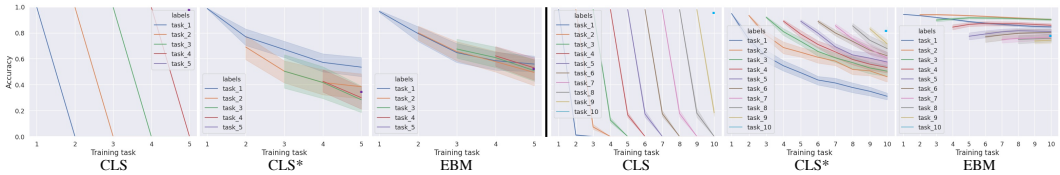

Figure 4: *Class-IL* testing accuracy of the standard classifier (CLS), classifier using our training objective (CLS*), and EBMs on each task on the split MNIST dataset (left) and permuted MNIST dataset (right).

A.2 CLASS CONFUSION MATRIX AT THE END OF LEARNING.

In Section 5.1.3, we show the qualitative analysis of EBMs and baselines. Except the energy landscape and predicted class distribution, we also generate the confusion matrices of EBMs and baseline approaches. Confusion matrix shows the relationship between ground truth labels and predicted labels. In Figure 5, we draw the confusion matrices after training on all the tasks on the split MNIST dataset and permuted MNIST dataset. The classifier tends to predict the classes from the last task (class 8, 9 for split MNIST and 90-100 for permuted MNIST). EBMs have high values along the diagonal that means the predicted results matching the ground truth labels for all the sequentially learned tasks. This demonstrates that EBMs are better at learning new tasks without catastrophically forgetting of old tasks.

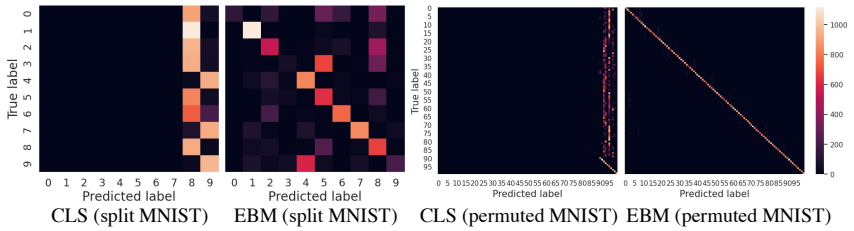

Figure 5: Confusion matrices between ground truth labels and predicted labels at the end of learning on split MNIST (left) and permuted MNIST (right). The lighter the diagonal is, the more accurate the predictions are.

A.3 MODEL CAPACITY

Another hypothesized reason for why EBMs suffer less from catastrophic forgetting is potentially their larger effective capacity. To analyze effective capacity of our models, 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 classifier and 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 CLS have $2,348,545$ and $2,349,032$ parameters respectively. The medium models have $5,221,377$ (EBM) and $5,221,352$ (CLS) parameters while the large models have $33,468,417$ (EBM) and $33,465,320$ (CLS) parameters. We use the model architectures in Table 5 and Table 6 for EBMs and classifiers.

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 indeed EBM has larger capacity to memorize data given a similar number of parameters. The gap between EBM and CLS increases when the models become larger. The larger capacity of EBM potentially enables it to memorize more data and mitigate the forgetting problem.

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 5 and Table 6, but with a hidden unit size of $h = 256$. Since there are only $50,000$ images on CIFAR-10, we use a small classifier ($210,138$ parameters) and EBM ($219,681$ parameters) 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 that EBM has larger capacity to memorize data given a similar number of parameters.

Table 5: The architecture of EBM used for the model capacity analysis. h are 512, 1024, and 4096 for the small, medium and large network respectively.

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

Table 6: The architecture of classifier used for the model capacity analysis. h are 512, 1024, and 4096 for the small, medium and large network respectively.

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

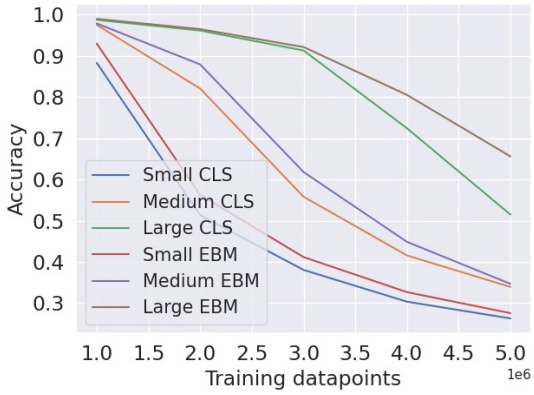

Figure 6: Model capacity of classifier and EBM using different model sizes.

A.4 PARAMETER IMPORTANCE

To further understand why EBMs suffer less from catastrophic forgetting, we design an experiment to test the importance of model parameters on past data. Inspired by the elastic weight consolidation (EWC) (Kirkpatrick et al., 2017), we estimate the importance of parameters for each tasks using the diagonal elements of the Fisher information matrix (FIM) F. Let θ_i be the model parameters after training on task T_i. Given one of previous tasks $T_j, j < i$, we evaluate how important each parameter is for tasks T_j. The kth diagonal of F is defined as the gradient on the EBM loss

$$F_{i,k} = \mathbb{E}_{\mathbf{x}\sim T_j} \nabla_{\theta_{i,k}} \left(E_{\theta_i}(\mathbf{x}, \mathbf{y}^+) + \log \sum_{\mathbf{y}'} \exp(-E_{\theta_i}(\mathbf{x}, \mathbf{y}')) \right)^2, \tag{8}$$

where $\mathbf{x}$ is sampled from tasks T_j and $E(\mathbf{x}, \mathbf{y}^+)$ is the energy value of the input data $\mathbf{x}$ and ground truth label $\mathbf{y}^+$. $\mathbf{y}' \in Y_j$ are classes randomly selected from the current batch. Here we use a single negative class. The above equation assigns high values to parameters crucial to task T_j as their gradients with respect to the loss are larger. Since the diagonal elements of the fisher information matrix measure the importance of each parameter to a given task, the density of diagonal elements represents the proportion of important parameters over all parameters. More density means more parameters are important for the given task and less parameters can be recruited for new tasks. Ideally, we expect these values to be sparse.

In Figure 7, we show the diagonal elements of the standard classifier (CLS), classifier using our training objective (CLS*), and our EBMs on the split MNIST dataset. For CLS and CLS*, we follow (Kirkpatrick et al., 2017) to compute their fisher information matrices. For comparisons across multiple models, we normalize the FIM diagonal elements of each method to be between 0 and 1 and report the normalized results in Figure 7. For example, "Fisher 5 on data 1" shows the diagonal elements of the Fisher information matrix obtained by Equation 8 using the model parameters θ_5 (after training on task T_5) and data $\mathbf{x}, \mathbf{y}^+, \mathbf{y}'$ from task T_1. The distribution of EBMs is sparser than CLS and CLS* indicating that EBMs have fewer important parameters for previous data. Updating parameters for the new task will have less negative impact on old tasks. In addition, more parameters can be used for learning new tasks if the distribution is sparse. This may provide another explanation for why EBMs can mitigate catastrophic forgetting.

A.5 MORE DETAILS OF THE INTERFERENCE WITH PAST DATA ANALYSIS

In Section 5.1.5, we test the interference of models trained on new data with past data. We compare the proposed EBMs with a standard classifier (CLS) and CLS using our EBMs training objective (CLS*). For CLS and CLS*, we use the cross-entropy loss to get their ϕ, *i.e.,* the cross-entropy losses on the old data before and after training on a new data.

We obtain θ_i by training on 200 randomly selected data from task1 and compute its gradient after training. Then we perform a single step optimization on 200 data that are randomly selected from task2 and average their parameters to get θ_j. We compute ϕ using the inner product of $\triangle\theta(\mathbf{x}_j) = \theta_j - \theta_i$ and the gradient $\nabla_{\theta_i}\mathcal{L}_{\text{ML}}(\,\mathbf{x}_i \mid \theta_i\,)$. For comparisons among multiple approaches, we normalize the ϕ of each method by dividing its maximum element value of θ_i and scale ϕ based on the length of θ, as CLS and EBMs have different number of parameters.

A.6 MODEL ARCHITECTURES

Images from the split MNIST and permuted MNIST datasets are gray images. (van de Ven & Tolias, 2019) uses several fully-connected layers for the baseline models. We follow their setting, using similar number of parameters, for EBM. 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 and Table 8 respectively. The model architectures of EBMs and baseline models on the permuted MNIST dataset are shown in Table 9 and Table 10.

Images from the CIFAR-10 and CIFAR-100 are RGB images. We use a small convolutional network for both the baseline models and EBMs. The model architectures of EBMs on the CIFAR-10

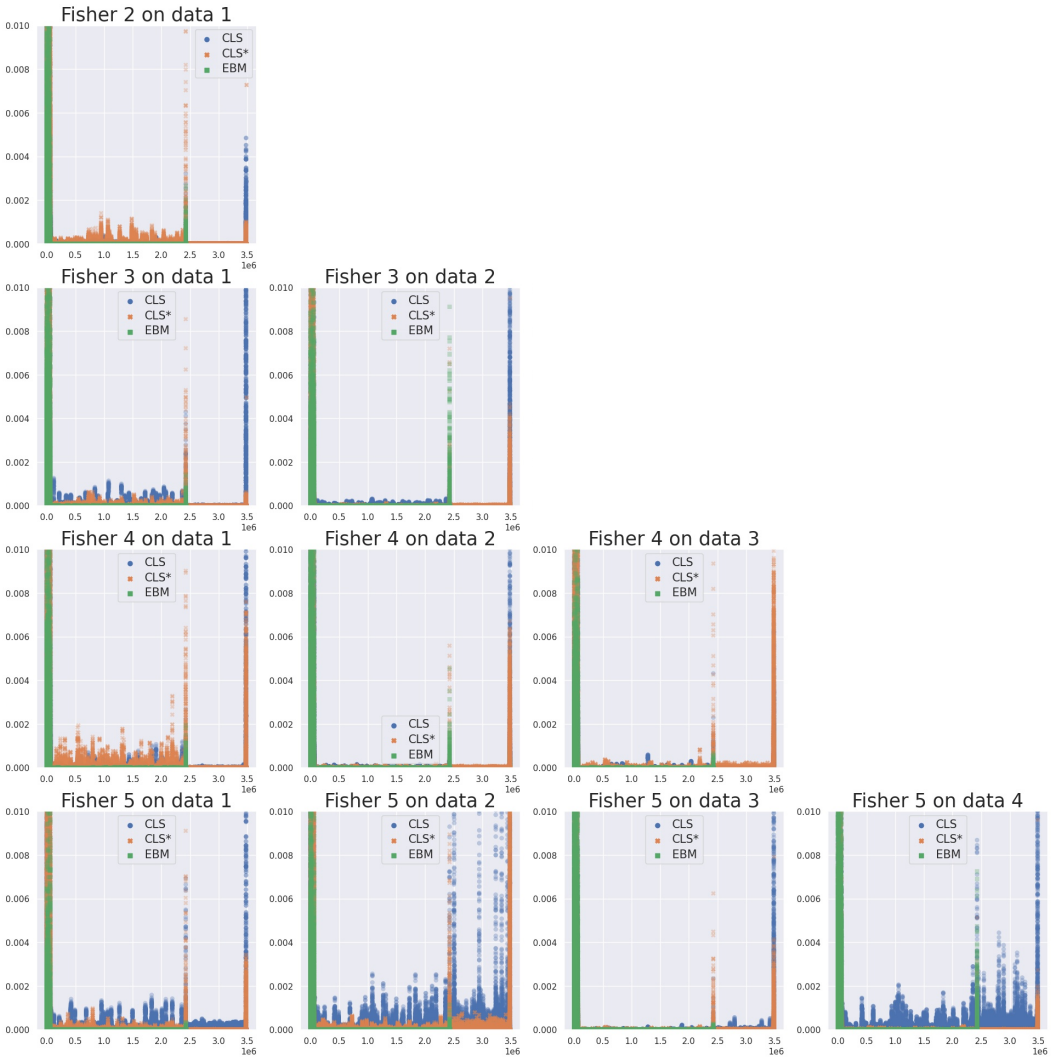

Figure 7: Parameter importance on different tasks. The x-axis represents each different parameter and y-axis is the FIM value in Equation 8. The sparser the parameters are, the fewer important parameters for previous data, which may contribute to why EBMs has less influence on previous tasks after training on new data. "Fisher 5 on data 1" means the diagonal elements of the fisher information matrix obtained by Equation 8 using the model parameters θ_5 (after training on task T_5) and data from task T_1.

and CIFAR-100 datasets are shown in Table 11, Table 12, Table 13, Table 14, and Table 15. We investigate different architectures to search for the effective label conditioning on EBMs training as described in Section 5.1.5. The baseline models on CIFAR-10 and CIFAR-100 is Table 16.

Table 7: The architecture of EBMs on the split MNIST dataset. $h = 400$.

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

Table 8: The architecture of baseline models on the split MNIST dataset. $h = 400$.

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

Table 9: The architecture of EBMs on the permuted MNIST dataset. $h = 1000$.

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

Table 10: The architecture of baseline models on the permuted MNIST dataset. $h = 1000$.

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

Table 11: The architecture of EBMs on the CIFAR-10 and CIFAR-100 datasets. **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)

Table 12: The architecture of EBMs on the CIFAR-10 and CIFAR-100 datasets. **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)

Table 13: The architecture of EBMs on the CIFAR-10 and CIFAR-100 datasets. **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)

Table 14: The architecture of EBMs on the CIFAR-10 and CIFAR-100 datasets. **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)

Table 15: The architecture of EBMs on the CIFAR-10 and CIFAR-100 datasets. **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)

Table 16: The architecture of baseline models on the CIFAR-10 and CIFAR-100 datasets. $N = 10$ on CIFAR-10 and $N = 100$ on CIFAR-100.

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, N) (x)