Combating Catastrophic Forgetting without the Model

Ian Huang
Columbia University

Lauren Arnett
Columbia University

Carl Vondrick
Columbia University

Abstract

When the distribution of tasks shifts during the course of learning, deep learning models often catastrophically forget the previous tasks. In this paper, we introduce a model-free approach to mitigate these effects through the manipulation of the training data. Inspired by how adversarial examples hide information to change the classifier performance, we instead perturb information in the training data such that it shifts the optimization trajectory away from forgetting the old task while retaining the ability to learn the new task. Consequently, our approach is general and can be applied to most models and loss functions. We formalize this approach through a gradient alignment objective, and our experiments on two image classification datasets show substantial improvements over baselines on task retention under distribution shift.

1 Introduction

Deep learning has enabled significant progress in computer vision, language understanding, and robotics (1; 4; 7; 16). However, when the distribution of tasks shifts during the course of learning, models often catastrophically forget the previous tasks (3; 11; 14). This limits the application of deep learning models in environments where continuous lifelong learning is needed to allow for introduction of novel classes and information the model has not previously seen.

Due to the importance of this problem, there have been several efforts that address catastrophic forgetting. Previous work has focused on regularizations on the model through transfer learning, introduction of new model architectures, and replay mechanisms (12). However, the key limitation of these methods is that they are either a) specific to a particular model or loss, thus restricting the type of architecture that is capable of minimizing catastrophic forgetting, or b) they require the storage of training data which increases linearly with time, thus trading scalability of the solution for the higher accuracy of learning in batches. Our goal is to create an approach without these limitations.

We introduce a model-free approach to mitigate catastrophic forgetting. Our approach works by perturbing information in the training examples such that the network will less likely forget the old tasks. While adversarial examples are often viewed as a weakness of machine learning methods (17), we instead leverage them to help mitigate catastrophic forgetting. We identify per-pixel perturbations on an input image that minimize catastrophic forgetting in an adversarial manner while still maintaining the original semantic content. To accomplish this, we introduce *gradient alignment*, a metric we show

Preliminary work. Under review by AISTATS 2020. Do not distribute.

to be negatively correlated with catastrophic forgetting.

By optimizing the input images by back-propagation according to gradient alignment, we seek to show this approach improves the model's ability to retain its previous tasks without changing its architecture. Forming a model-agnostic method of combating catastrophic forgetting allows for more existing state-of-the-art architectures that are not necessarily optimized for lifelong learning to be trained in an online and lifelong fashion. Experiments suggest that this is a promising method for anticipating and preventing catastrophic forgetting without regard to a particular model or loss.

The main contributions of this work are twofold. First, we present a metric by which to minimize the expected amount of catastrophic forgetting without assuming or restricting anything about the nature of the optimization method, loss function or model architecture. Second, we show through experimentation that using an iterative method of optimizing input images with respect to this metric in a way that preserves semantics is capable of diminishing catastrophic forgetting while preserving the ability to learn from new incoming classes.

2 Related Works

Transfer learning involves directly transferring hidden layers from previously learned models into new models, as a form of knowledge transfer (12). For example, Lopez-Paz and Ranzato proposed Gradient Episodic Memory to alleviate catastrophic forgetting while allowing for transfer of learning to previously learned tasks (10). However, this approach is not scalable to continuously learning systems, as it requires previously learned model parameters to be preserved.

Replay mechanisms typically involve a system that "mixes" previously learned data into the batch containing new information (13). The intuition is that by relearning samples in conjunction with new data, the model can better reconcile the newly observed samples with previously learned. Draelos *et al.* preserves the weights of the model necessary for retaining older information with a technique they call "intrinsic replay" (2). Shin *et al.* propose Deep Generative Replay with a deep generative model that samples training data from previously learned tasks to be introduced alongside data from new tasks for the discriminator architecture (15). The replay technique typically either requires the network to have a supply of old training data, or requires the model to re-learn each new joint task from scratch.

Parameter regularization approaches regulate how far "important" parameters of the model can deviate, where "importance" is evaluated by a metric of that parameter's relevance to older tasks. Kirkpatrick *et al.* propose Elastic Weight Consolidation (EWC), where the relevance of a parameter is given by its posterior distribution over the dataset (5). However, EWC makes some assumptions about the distribution of the posterior distribution of the parameters over the older dataset, which has been proven to lead to sub-optimal learning of the new task.

Distributional regularization attempts to match the output distribution of the networks for older tasks (13). This approach however requires the preservation of previously observed data, thus hindering its scalability.

3 Method

3.1 Gradient Collision and Catastrophic Forgetting

We define catastrophic forgetting as losing accuracy over the prior classes while learning new classes. Formally, let new sample x_t in class y_t be presented to the model at training iteration t. Let $f(x; \theta_t)$ be the neural network with parameters θ_t on the t-th training iteration. At every timestep, let z denote the single class that the new sample x_t represents, let $\mathcal{L}(\theta_t)$ be the classification loss over the set of previously seen training data. We define θ_t as the parameters of the model *before* the gradient step upon the input of (x_t, y_t). Then, we define *disruption* as:

$$D(x_t; \theta_t) = \mathcal{L}(\theta_{t+1}) - \mathcal{L}(\theta_t)$$

where θ_{t+1} is the parameters after updating the parameters to the tth example. Naturally, if an image is highly *disruptive* to old classes with respect to the model with parameters θ_t, then we expect $D(x_t; \theta_t)$ to be large.

At every iteration, we would like to directly minimize disruption with respect to the input image x_t: $\tilde{x_t} = \arg\min_{x_t} \mathcal{L}(\theta_{t+1}; X) - \mathcal{L}(\theta_t; X)$. However, not only would this entail expressing θ_{t+1} as a fixed differentiable function of x_t and θ_t (we would have to make assumptions about the update rule such as $\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta; x_t)|_{\theta=\theta_t}$), but optimizing this quantity above iteratively is computationally expensive since one would have to compute the gradient of $\mathcal{L}$ at two separate parameter configurations θ_t and θ_{t+1} in order to evaluate the gradient step for this quantity.

Since gradient-based methods of optimization dominate in the the training of deep neural networks, we look to the *directions* in which the gradients of the old task and new task point to provide valuable clues for the extent of disruption. To this end, we define *gradient alignment* as:

$$G(x_t; \theta_t) = \frac{\nabla_\theta \mathcal{L}(\theta; X)|_{\theta=\theta_t}^T \nabla_\theta \mathcal{L}(\theta; x_t)|_{\theta=\theta_t}}{||\nabla_\theta \mathcal{L}(\theta; X)|_{\theta=\theta_t}||_2 ||\nabla_\theta \mathcal{L}(\theta; x_t)|_{\theta=\theta_t}||_2}$$

where X is the dataset learned before time t.

Under the assumption that the step size is small enough to prevent overshoot during stochastic gradient descent, we show that maximizing $G(x_t; \theta_t)$ is a necessary and sufficient condition for the minimization of $D(x_t; \theta_t)$. More elaboration will be given in Section 3.3. Empirically, we see that this is the case. For the LeNet MNIST classification architecture, we run over many experiments where the model was pretrained on a 20-image dataset of two classes of MNIST for 50 epochs before a single image is introduced at time step 21 (8; 9). For every possible triple (N_1, N_2, y_t), we draw a line of best fit between $D(x_{21}; \theta_{21})$ and $G(x_{21}; \theta_{21})$. Figure 1 shows the distribution of these scores, from which we can see that the correlation between D and G is overwhelmingly negative. These preliminary results thus suggest the idea of maximizing $G(x_t; \theta_t)$ with respect to input data x_t at every timestep is a sensible goal to minimize anticipated forgetting after every gradient step at time t.

3.2 The Need to Regularize

Directly optimizing the above objective can produce large changes to the input image, which is undesirable for two reasons. Firstly, large changes can modify the image so much that it eliminates the signal for the new task that we are trying to learn. Secondly, the change may cause the training examples to shift off the natural image manifold, which can cause generalization failures during inference. Ideally, we would like to regularize the model such that produced image has a high likelihood under the natural image distribution. However, for realistic datasets such as images, this is unknown and challenging to estimate.

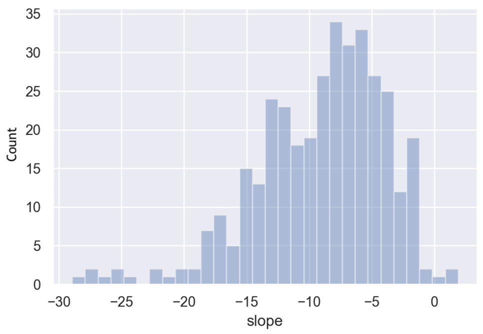


Figure 1: Histogram of the slopes for lines of best fit drawn through disruption and gradient alignment across all experiments.

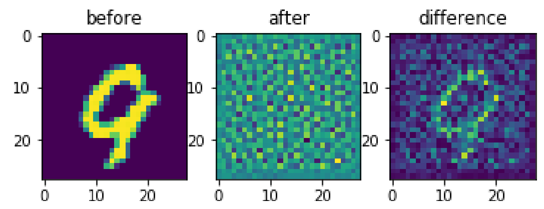


Figure 2: Initial image x_t and the perturbed image $x_t + \arg\max_\delta G(x_t + \delta; \theta_t)$ (without the regularization term) for a model that was pre-trained on 20 images of old classes 8 and 6. This optimization was unsuccessful at preserving the input's semantic content.

Instead, we will regularize the example alteration so that the change is minimal by subtracting a regularization term:

$$\tilde{x_t} = x_t + \arg\max_\delta (G(x_t + \delta; \theta_t) - \lambda ||\delta||_2)$$

The Frobenius norm of the δ-matrix acts as a proxy to achieve a similar effect as regularizing with the natural image distribution. Assuming that the marginal probability density is continuous over the image space, for all $\epsilon > 0$ there always exist an open ball $\mathcal{N}_\xi(x_t)$ for which $\forall x' \in N_\xi(x_t), |P(y_t|x_t) - P(y_t|x')| < \epsilon$.

Maximizing $G(x_t + \delta; \theta_t)$ subject to the regularization encourages images to retain their semantic information. The benefit of this is such that at prediction time, the network $f(x; \theta_t)$ will be able to take in x without needing to first transform it to $\tilde{x}$ (which can only be calculated with label information), and perform well on the task.

3.3 Connection between Gradient Alignment and Disruption

Why do we choose gradient alignment as the objective in the first place? Gradient alignment has an intuitive geometric understanding, and allows us to make some guarantees about the outcome of catastrophic forgetting under the assumption that the alignment is large enough (i.e. positive).

In a two dimensional analog of the geometry that justifies the optimization of $G(x_t + \delta, \theta_t)$ over δ, we find that at timestep t, $\nabla_\theta \mathcal{L}(\theta; X)|_{\theta=\theta_t}$ and $\nabla_\theta \mathcal{L}(\theta; x_t)|_{\theta=\theta_t}$ are pointing in somewhat opposite directions. The gradient step by (x_t, y_t) would cause a component-wise *ascent* in $\mathcal{L}(\theta; X)$. Ideally, we would like to shift the loss contours in the θ-space by changing x_t such that $\nabla_\theta \mathcal{L}(\theta; \tilde{x_t})|_{\theta=\theta_t}$ is now in the same direction as $\nabla_\theta \mathcal{L}(\theta; X)|_{\theta=\theta_t}$.

Alternatively, to see why we expect this to be sensible for SGD, assume that:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta; x_t)|_{\theta=\theta_t}$$

Recall that disruption is defined as $\mathcal{L}(\theta_{t+1}; X) - \mathcal{L}(\theta_t; X)$. Plugging our expression for θ_{t+1} into this and finding the x_t that minimizes this quantity, we arrive at:

$$\tilde{x_t} = \arg\min_{x_t} \mathcal{L}(\theta_t - \eta \nabla_\theta \mathcal{L}(\theta; x_t)|_{\theta=\theta_t})$$

We see that the original second term in the definition of disruption has no contribution from x_t. By definition of the gradient $\nabla_\theta \mathcal{L}(\theta; x_t)$, we know that $\mathcal{L}(\theta_{t+1})$ is minimum for fixed η if and only if $\nabla_\theta(\mathcal{L}(\theta; X))|_{\theta=\theta_t} = \nabla_\theta(\mathcal{L}(\theta; x_t))|_{\theta=\theta_t}$. We see that if this is true then gradient alignment is maximized:

$$\frac{\nabla_\theta(\mathcal{L}(\theta; X))|_{\theta=\theta_t}^T \nabla_\theta(\mathcal{L}(\theta; x_t))|_{\theta=\theta_t}}{||\nabla_\theta(\mathcal{L}(\theta; X))|_{\theta=\theta_t}||_2 ||\nabla_\theta(\mathcal{L}(\theta; x_t))|_{\theta=\theta_t}||_2} = 1$$

Thus, the minimization of disruption of x_t implies the maximization of gradient alignment. The contrapositive statement implies that if the gradient alignment is *not* maximal, then the disruption of x_t is *not* minimal. This thus incentivizes our maximization of gradient alignment for x_t.

3.4 Implementation

To solve the problem of finding δ in

$$\arg\max_\delta (G(x_t + \delta; \theta_t) - \lambda ||\delta||_2)$$

we employ gradient ascent. Expressed in terms of the original losses of function f, each update for δ takes the following form:

$$\delta_{k+1} = \delta_k + \eta \nabla_\delta (G(x_t + \delta; \theta_t) - \lambda ||\delta||_2)$$

where δ_0 is randomly initialized using a multivariate standard normal distribution in the image space.

The optimizer efficacy in discovering perturbations on x_t is crucial. However, because every new image x_t at each timestep results in a different loss function $\mathcal{L}(\theta; x_t)$ over the θ-space, so too does the each new image change the curvature of the objective function, $G(x_t + \delta; \theta_t) - \lambda ||\tilde{x_t} - x_t||$ over the image space. As such, it is difficult to find a single set of hyperparameters that works for all input images. For our purposes, we pick:

$$\delta_t = \underset{\delta \in \{\delta_0, \delta_1, \delta_2 \ldots, \delta_K\}}{\arg\max} (G(x_t + \delta; \theta_t) - \lambda ||\delta||_2)$$

where K is the total number of gradient steps to run.

4 Experimentation

4.1 Experimental Setup

To evaluate the effects of our preprocessing method on online learning models, we implement a simple model with the LeNet architecture (8) for the MNIST dataset (9) and a similar but deeper version of the same architecture for the CIFAR-100 dataset (6). Two instances of each model are then compared in their retention of prior classes (through disruption) and their learning of the new class. The "baseline model" receives the unaltered image x_t at every timestep t, and the model back-propagates on x_t at the end of every timestep. Meanwhile, the "preprocessed model" receives $\tilde{x_t}$.

At every time step, the disruption is calculated for both the baseline and preprocessed models over a set X that contains only data points both models were trained on before time step 0. A held-out set of class y_t is used to evaluate the loss over the class y_t (i.e. the ability to learn y_t). While the preprocessed model is trained on perturbed images, the input at test time will be unaltered.

For both MNIST and CIFAR-100, we simulate a live continuous stream of information to both models by first randomly selecting 2 classes from the dataset on which we pretrain both models. Then, randomly selecting a new class from the remaining ones, we simulate 10 timesteps; during each, the optimization described in Section 3.4 is done on an image sample of that class, and the sample (x_t, y_t) is given to the baseline model while $(\tilde{x_t}, y_t)$ is given to the preprocessed model, to perform the tth gradient step. We repeat this experiment 1000 times.

We furthermore conduct experiments studying the trajectory of accuracies over the whole dataset for both the preprocessed and baseline model on CIFAR-100. We pretrain the model on 400 images of 5 different classes, then incrementally introduce the 6th and 7th classes through 15 timesteps. The same rules apply for the learning and preprocessing of (x_t, y_t) as before, except changes are now applied based on two different classes. At each timestep, we measure its accuracy over a held-out set of old classes, over a held-out set of the new class, and over the union of both. We repeat this experiment 50 times.

4.2 Quantifying Retention and Learning of New Class

Our experiments on the MNIST and on the CIFAR-100 datasets give rise to the distribution displayed in Figure 3 and Figure 4. In our analysis for both, we consider only the samples where optimization over x_t was successful at every timestep.

In Figures 3a and 4a, we compare the distributions of

$$\mathcal{L}(\theta_{t+1}; X)_{baseline} - \mathcal{L}(\theta_{t+1}; X)_{preprocessed}$$

at every timestep t. Positive values of this quantity means that our preprocessing allows the preprocessed model to outperform its baseline counterpart in retention of old classes. Indeed, for the MNIST dataset, the mean of the distributions of this quantity at each timestep except the last is positive. In expectation, our preprocessing approach allows for better retention of older tasks at each timestep. For the CIFAR-100 dataset, the mean of this quantity is consistently positive and increasing at every single timestep, therefore also showing the efficacy of our method in expectation.

Figure 3b displays the distributions of $\mathcal{L}(\theta_t; x_{test})_{baseline} - \mathcal{L}(\theta_t; x_{test})_{preprocessed}$ for the held-out dataset of the new class at every timestep. Figure 3b shows that while the learning of class y_t is initially compromised by a large margin after preprocessing x_t, the mean loss tends to 0 as the number of timesteps increase, and the distribution of this quantity gets narrower. [1] However, we see that the opposite happens for the CIFAR-100 dataset. While the distributions are still closely centered at 0, the distributions get wider as the number of timesteps increase. This suggests a greater uncertainty in the effects on the learning of class y_t as the model goes forward in time.

4.3 Visualizing Alterations

Figure 5 displays a sample from our experiments. These examples show that regularization aids in preserving semantic information, so the gradient ascent method was able to discover a δ that creates $\tilde{x_t}$ within the neighborhood of x_t for each t. We find consistently in the MNIST dataset that the pixels corresponding to the numerical characters are being altered the most (see Figures 5b and 5d), while our experimentation on the CIFAR-100 dataset consistently show a tendency to apply a 3x3 pattern of perturbations onto the image (see Figures 5c and 5e), regardless the semantics or variation in the position of the entity in the image. This suggests that while optimization on the gradient alignment tends to *erase* information of new classes to be found in the MNIST image, it is able to find in the CIFAR-100 image space adversarial perturbations that does not try to diminish the pixels specifically corresponding to the new class. We suspect that because the dimensionality of the CIFAR-100 dataset is larger, such adversarial subspaces are easier to find. As such from our qualitative experiments, we show that our method works better (in preserving semantics of the input image) in higher-dimensional problems.

4.4 Trajectory of Accuracies

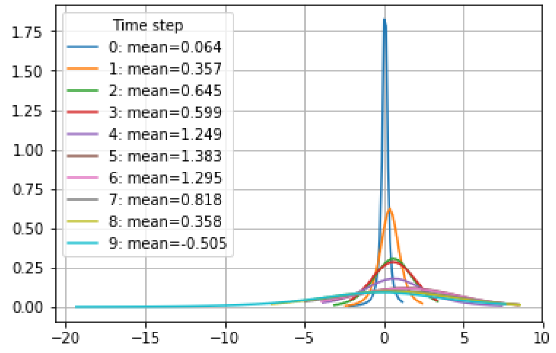
Figure 6 shows a graph of the area under the accuracy curve (evaluated over the union of old and new classes) across the 15 timesteps. We see that the distribution for the preprocessed model is further to the right than that of the baseline model, suggesting higher accuracies on average. Looking at a specific example in Figure 7b, we see the optimization algorithm has found a perturbation that increases gradient alignment without altering the semantics, albeit at the cost of some blurring of the original image. However, this altered image not only increases the classification accuracy over the new class (in Figure 7a we see an increase in accuracy of the preprocessed model over the new class from timestep 14 to 15), but moreover preserves the older classes better than the baseline model (the accuracy over the old classes is higher at the last timestep for the preprocessed model than the baseline model).

5 Discussion

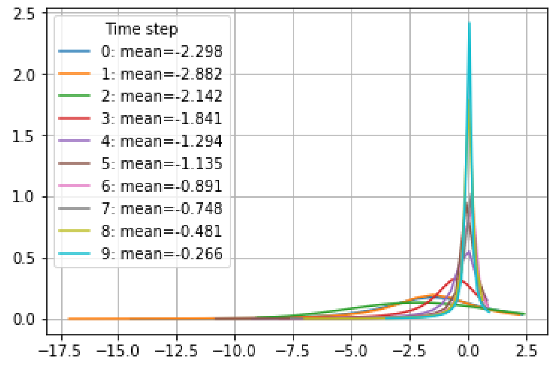
Although adversarial examples are typically viewed as a weakness of a neural networks, this paper shows we can leverage them as a model-agnostic approach to improve them. Our

[1] Intuitively, this makes sense since small perturbations around x_t in the image space assigns the probability $p(z|x_t)$ to sample $\tilde{x_t}$. Under the assumption of the continuity of the marginal probability density function, we know that $|p(z|x_t) - p(z|\tilde{x_t})|$ can be made arbitrarily small.

results suggest perturbing the input training data via gradient alignment can help deep networks retrain previous tasks and mitigate catastrophic forgetting. Moreover, although we have focused on catastrophic forgetting in this paper, our approach is general and can be used to modify and improve neural network classifiers across other properties. We expect that viewing adversarial examples as a beneficial tool can spur new techniques to improve the representations and learning dynamics of neural networks.

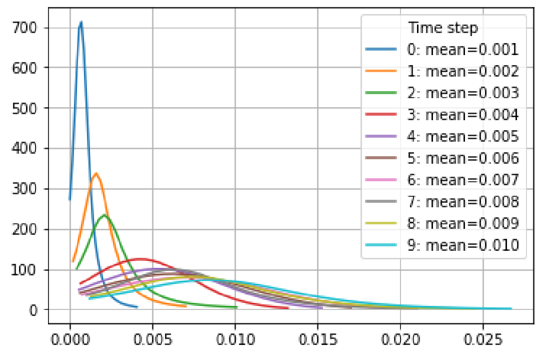


(a) $\mathcal{L}(\theta_{t+1};X)_{baseline} - \mathcal{L}(\theta_{t+1};X)_{preprocessed}$ over timesteps. The mean of the distribution is largely positive, showing that our approach is expected to lower disruption.

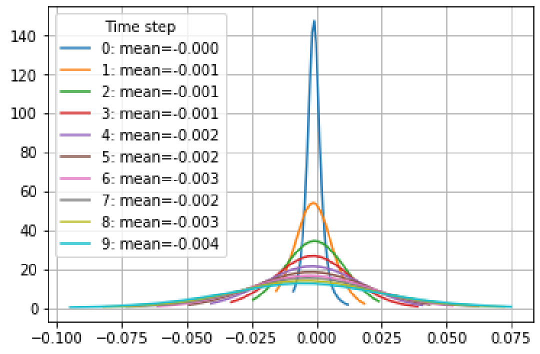


(b) $\mathcal{L}_(\theta_t;x_{test})_{baseline} - \mathcal{L}(\theta_t;x_{test})_{preprocessed}$ through time. On average, learning over class y_t is affected at first, but this impact decreases as the number of z samples increase.

Figure 3: On MNIST: Distribution of differences in disruption and z learning scores (fit to a t-distribution) for experiments with successful optimization over x_t.

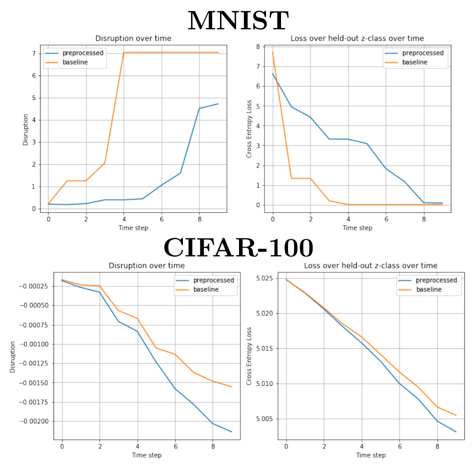


(a) $\mathcal{L}_N(\theta_{t+1})_{baseline} - \mathcal{L}_N(\theta_{t+1})_{preprocessed}$ over timesteps. Every distribution is shifting to the right, suggesting a aggregating effect of disruption minimization through timesteps.

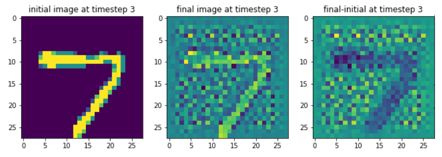


(b) $\mathcal{L}_(\theta_t;x_{test})_{baseline} - \mathcal{L}(\theta_t;x_{test})_{preprocessed}$ through time. On average, learning over class y_t is affected at first, but this impact decreases as the number of z samples increase.

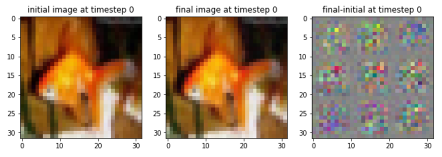
Figure 4: On CIFAR-100: Distributions of differences in disruption and z learning scores (fit to a t-distribution) for experiments with successful optimization over x_t.



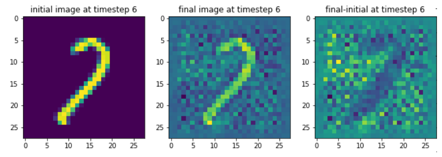
(a) The trajectory of the disruption over X at every timestep, as well as the trajectory of loss evaluated over a held-out dataset of class y_t. We see that the model with the preprocessed input does better in both retaining prior classes while the learning of class y_t across the timesteps.



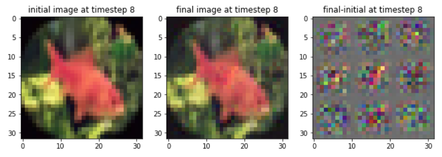
(b) The input modifications at the 4th timestep. The output retains the semantics of the input.



(c) The input modifications at the 1st timestep. The output retains the semantics of the input.



(d) The input modifications at the 7th timestep. We see that the alignment scores over 500 optimization steps have an overall increasing trend.



(e) The input modifications at the 9th timestep. We see that the alignment scores over 500 optimization steps have an overall increasing trend.

Figure 5: Samples of our experiments on the MNIST and CIFAR-100 datasets.

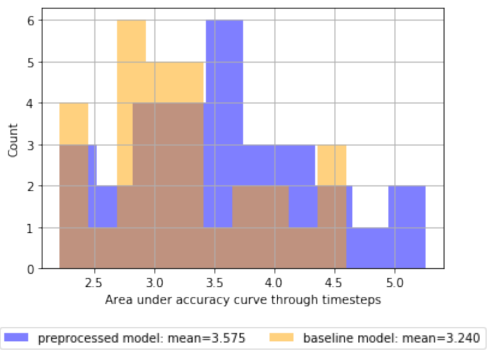
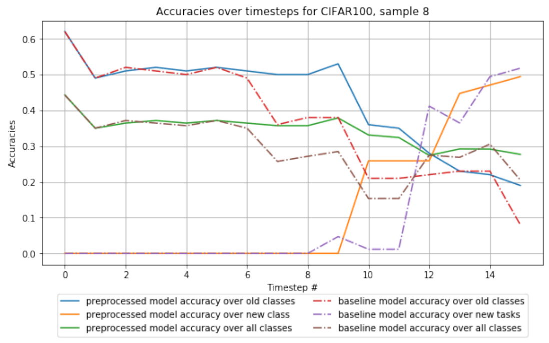
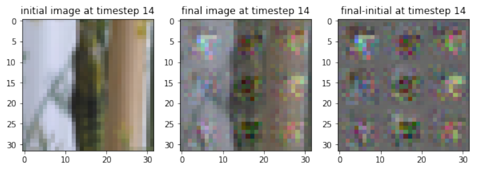


Figure 6: Histogram of area under the accuracy curve for the preprocessed and the baseline models over CIFAR-100 over 50 trials of online learning with 15 timesteps. On average, our method beats the baseline in accuracy over the union of old and new classes at every timestep.



(a) Accuracies over timesteps on CIFAR-100.



(b) Perturbations at the last timestep.

Figure 7: Distribution of differences in disruption and z learning scores (fit to a t-distribution) for experiments with successful optimization over x_t on the MNIST dataset.

References

[1] Yoshua Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[2] Timothy J. Draelos, Nadine E. Miner, Christopher C. Lamb, Craig M. Vineyard, Kristofor D. Carlson, Conrad D. James, and James B. Aimone. Neurogenesis deep learning. *CoRR*, abs/1612.03770, 2016.

[3] Robert French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3:128–135, 05 1999.

[4] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[5] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016.

[6] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[8] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[9] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[10] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continuum learning. *CoRR*, abs/1706.08840, 2017.

[11] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C):109–165, 1989.

[12] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *CoRR*, abs/1802.07569, 2018.

[13] Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *CoRR*, abs/1705.09847, 2017.

[14] A. Robins. Catastrophic forgetting in neural networks: the role of rehearsal mechanisms. In *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 65–68, Nov 1993.

[15] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *CoRR*, abs/1705.08690, 2017.

[16] Niko Sünderhauf, Oliver Brock, Walter J. Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter Corke. The limits and potentials of deep learning for robotics. *CoRR*, abs/1804.06557, 2018.

[17] Christian Szegedy, Google Inc, Wojciech Zaremba, Ilya Sutskever, Google Inc, Joan Bruna, Dumitru Erhan, Google Inc, Ian Goodfellow, and Rob Fergus. Intriguing

properties of neural networks. In *ICLR*, 2014.