CAML: Catastrophically-Aware Meta-Learning

Tristan Gosakti, Jonathan Gomes-Selman & Woodrow Z. Wang

Extended Abstract

Introduction: The ability to build reliable AI agents for real-world applications is contingent on the development of algorithms that allow agents to incrementally learn from new experiences without forgetting knowledge from previous experiences. Catastrophic forgetting is a significant problem that hinders a model's ability to perform well in a continual multi-task learning setting.

We build upon the work of Riemer et al. (2018) to incorporate the use of prioritized replay buffers for settings where a model must learn from a sequence of tasks. Previous works consider uniform sampling from the meta-experience replay buffers for simplicity. In this paper, *our key insight is that examples in the replay buffer are not equally important in terms of optimizing for the meta-objective function.*

Our main contributions in this paper are three-fold:

- We develop **Catastrophically-Aware Meta-Learning (CAML)**, inspired by the Meta-Experience Replay algorithm (Riemer et al., 2018), but modified with various proposed prioritized replay methods.
- We empirically demonstrate the improved sample efficiency and robustness to catastrophic forgetting of our proposed CAML model compared to various continual learning baselines on the MNIST Rotations dataset.
- We qualitatively perform inter-task and intra-task analysis of our learned models. Moreover, through saliency map analysis, we show that CAML learns to attend to semantically meaningful input features so as to maximize transfer and minimize interference.

Methods: We add various novel prioritized meta-experience replay methods to the MER algorithm (Riemer et al., 2018). We propose the **Catastrophically-Aware Meta-Learning (CAML)** algorithm, a meta-learning approach that learns how to learn without forgetting with prioritized replay. We explore four prioritization schemes: 1) Loss priority (prioritize examples with higher loss), 2) Newest priority (prioritize most recent examples), 3) Oldest priority (prioritize oldest examples), and 4) Dynamic priority (a combination of loss and age).

Experiments: We perform experiments on the MNIST Rotations dataset, which is a common continual learning benchmark (Lopez-Paz & Ranzato, 2017). The dataset contains 20 tasks, each defined by rotating MNIST digits (0-9) by a fixed angle between 0 and 180 degrees.

In order to evaluate different characteristics of model success and the impact of forgetting, we use the following metrics: 1) Retained accuracy (RA), the final accuracy across all tasks after sequential training is complete, 2) Learned accuracy (LA), the average accuracy for each task directly after it is learned, and 3) Backward transfer and interference (BTI), the average change in accuracy from when a task is learned to the end of training, where a low BTI reflects catastrophic forgetting.

Empirically, we show that each of the prioritization schemes have their own benefits, summarized in Table 1(a). CAML (Newest) performs best in terms of learned accuracy, or the accuracy immediately after training on a task. This is due to focusing on learning new tasks at the cost of forgetting how to perform well on old tasks. On the contrary, CAML (Oldest) performs well on BTI, as prioritizing old examples helps mitigate catastrophic forgetting. CAML (Loss) performs best on retained accuracy, or final accuracy after training on all tasks sequentially. Finally, CAML (Dynamic) leverages the best of both worlds by combining both loss and age in order to obtain the best BTI by far and a highly competitive retained accuracy.

Qualitatively, we carefully examine the performance of CAML across various tasks and across various classes within a single task. Through careful introspection of CAML's performance compared to various baselines, we are able to show CAML's robustness to catastrophic forgetting as well as the susceptibility of more naive baselines to catastrophic forgetting, as seen in Figure 1(b). Moreover, we further inspect CAML's ability to learn to learn without forgetting through the lens of saliency maps, a popular qualitative visualization in computer vision. From the saliency maps in Figure 1(c), CAML is learning to attend to semantically meaningful parts of the inputs so as to maximize gradient transfer and minimize gradient interference, whereas less successful baselines attend to seemingly random pixels in the image.

Future Directions: Our results motivate future exploration into prioritized replay buffers for continual learning. While we used fixed heuristics for suitable prioritization schemes, future work could involve directly learning the priorities of examples instead. This would be similar to tackling the exploration vs. exploitation tradeoff, where learning how to prioritize examples could allow a model to choose when to sample more uniformly vs. greedily.

Model	*LA*	*RA*	*BTI*
Online	0.2529	0.1879	−0.0650
Independent	0.1016	0.1016	-
Task Input	0.1313	0.1100	−0.0213
MER	0.3473	0.5105	0.1632
CAML (Newest)	**0.3911**	0.4462	0.0551
CAML (Oldest)	0.3482	0.5248	0.1766
CAML (Loss)	0.3731	**0.5438**	0.1707
CAML (Dynamic)	0.3227	0.5369	**0.2142**

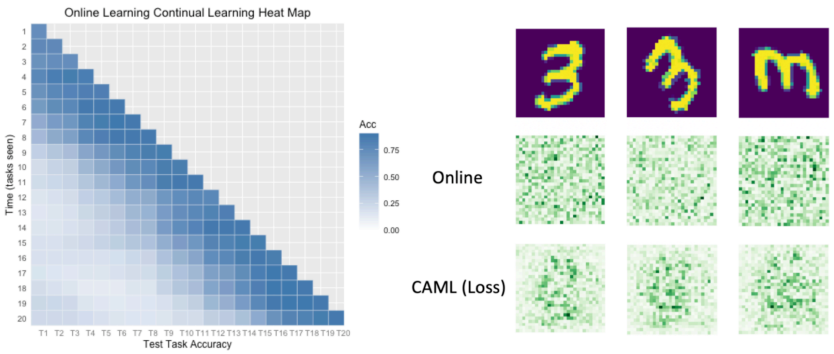

Figure 1: (a) Left: Model performances on MNIST Rotations. **(b) Middle:** Heat map of Online baseline showing catastrophic forgetting. **(c) Right:** Saliency maps for model introspection, where CAML attends to semantically meaningful parts of the input.

CAML: CATASTROPHICALLY-AWARE META-LEARNING

Tristan Gosakti, Jonathan Gomes-Selman & Woodrow Z. Wang
Department of Computer Science
Stanford University
{tgosakti,jgs8,wwang153}@stanford.edu

ABSTRACT

The ability to build reliable AI agents for real-world applications is contingent on the development of algorithms that allow agents to incrementally learn from new experiences without forgetting knowledge from previous experiences. Catastrophic forgetting is a significant problem that hinders a model's ability to perform well in a continual multi-task learning setting. We propose a novel algorithm, Catastrophically-Aware Meta-Learning (CAML), that combines the usage of a meta-experience replay buffer with prioritized replay in order to learn how to learn without forgetting in a sample efficient manner. CAML learns parameters such that gradient transfer is maximized and gradient interference is minimized. We perform experiments on the MNIST Rotations dataset and show that CAML outperforms common continual learning baselines. Moreover, we qualitatively inspect our model's performance through inter-task, intra-task, and saliency map analysis.

1 INTRODUCTION

Catastrophic forgetting is a common issue that plagues multi-task learning problems (McCloskey & Cohen, 1989). When training on tasks sequentially, such as in the continual learning setting, the learning from new tasks may cause the model to unlearn – or catastrophically forget – how to perform well on previous tasks. Lifelong or continual learning is important for the deployment of AI in the real world; since tasks and data may evolve over time, models must stay updated to avoid this covariate shift.

We specifically examine a setting of continual learning, where the model is given a sequence of tasks from a single data distribution D_g. From this sequence of tasks, we would like to learn how to quickly learn from new tasks also drawn from D_g: a meta-learning problem. Critically, we would like to do this without forgetting or losing information on how to perform tasks seen previously during training. Our hypothesis is that by addressing catastrophic forgetting issues, we can increase training stability and improve the meta-learned models performance on new tasks.

The gradient transfer-interference tradeoff is a common way to examine the issue of catastrophic forgetting in continual multi-task learning settings (McCloskey & Cohen, 1989). Successful models must retain knowledge from previous tasks by minimizing gradient interference, whilst also being flexible enough to learn new knowledge by maximizing gradient transfer.

There have been promising attempts at addressing catastrophic forgetting by explicitly influencing gradient updates, either by gradient surgery (Yu et al., 2020) or limiting weight sharing (Kirkpatrick et al., 2016). While these methods have shown promising results, we are interested in further examining approaches that can learn how to learn without forgetting. Namely, we focus on investigating meta-learning algorithms to address catastrophic forgetting.

We build upon the work of Riemer et al. (2018) to incorporate the use of prioritized replay buffers for settings where a model must learn from a sequence of tasks. Previous works consider uniform sampling from the meta-experience replay buffers for simplicity. In this paper, *our key insight is that examples in the replay buffer are not equally important in terms of optimizing for the meta-objective function.*

Therefore, we propose the novel **Catastrophically-Aware Meta-Learning (CAML)** algorithm, a meta-learning approach that learns how to learn without forgetting with prioritized replay. We explore four prioritization schemes: 1) Loss priority (prioritize examples with higher loss), 2) Newest priority (prioritize most recent examples), 3) Oldest priority (prioritize oldest examples), and 4) Dynamic priority (a combination of loss and age). In order to evaluate different characteristics of model success and the impact of forgetting, we use the following metrics: 1) Retained accuracy (RA), the final accuracy across all tasks after sequential training is complete, 2) Learned accuracy (LA), the average accuracy for each task directly after it is learned, and 3) Backward transfer and interference (BTI), the average change in accuracy from when a task is learned to the end of training, where a low BTI reflects catastrophic forgetting.

We perform experiments on the MNIST Rotations dataset, which is a common continual learning benchmark (Lopez-Paz & Ranzato, 2017). The dataset contains multiple tasks, where each task is defined by rotating MNIST digits (0-9) by a fixed angle between 0 and 180 degrees. We choose this dataset due to its prevalence in related works and the reduced need for high compute.

Empirically, we show that each of the prioritization schemes have their own benefits. CAML (Newest) performs best in terms of learned accuracy, or the accuracy immediately after training on a task. This is due to focusing on learning new tasks at the cost of forgetting how to perform well on old tasks. On the contrary, CAML (Oldest) performs well on BTI, as prioritizing old examples helps mitigate catastrophic forgetting. CAML (Loss) performs best on retained accuracy, or final accuracy after training on all tasks sequentially. Finally, CAML (Dynamic) leverages the best of both worlds by combining both loss and age in order to obtain the best BTI and a highly competitive retained accuracy.

Qualitatively, we carefully examine the performance of CAML across various tasks and across various classes within a single task. Through careful introspection of CAML's performance compared to various baselines, we are able to show CAML's robustness to catastrophic forgetting as well as the susceptibility of more naive baselines to catastrophic forgetting. Moreover, we further inspect CAML's ability to learn to learn without forgetting through the lens of saliency maps, a popular qualitative visualization in computer vision. With the saliency maps, we are able to show that CAML is learning to attend to semantically meaningful parts of the inputs so as to maximize gradient transfer and minimize gradient interference, whereas less successful baselines attend to seemingly random pixels in the image.

Our main contributions in this paper are three-fold:

- We develop **Catastrophically-Aware Meta-Learning (CAML)**, inspired by the Meta-Experience Replay algorithm (Riemer et al., 2018), but modified with various proposed prioritized replay methods.
- We empirically demonstrate the improved sample efficiency and robustness to catastrophic forgetting of our proposed CAML model compared to various continual learning baselines on the MNIST Rotations dataset.
- We qualitatively perform inter-task and intra-task analysis of our learned models. Moreover, through saliency map analysis, we show that CAML learns to attend to semantically meaningful input features so as to maximize gradient transfer and minimize gradient interference.

2 Related Work

Continual Learning: We examine the specific setting of continual learning, where the sequence of tasks is drawn from the same underlying data distribution (Riemer et al., 2018). This setting has been particularly interesting, as researchers look to design models that can incrementally learn from new incoming data autonomously over a long period of time. In the lifelong learning setting, models must adapt to changing environments while retaining knowledge from memories of past experiences, which presents a difficult non-stationarity issue (Thrun, 1994; 1995).

Transfer-Interference Tradeoff: Catastrophic forgetting is particularly potent when training on a sequence of tasks, where a model must balance a tradeoff between learning from new tasks while not forgetting previous learnings on old tasks. Previous works have shown that forgetting can occur due to catastrophic interference in the gradient updates from different tasks (McCloskey & Cohen, 1989). The transfer-interference tradeoff closely parallels the stability-plasticity dilemma, where stability refers to a model's ability to preserve knowledge and plasticity refers to a model's ability to rapidly adapt and learn from a new experience.

Gradient Surgery: Methods have been proposed to directly modify the gradients in order to promote efficient incremental learning (Yu et al., 2020; Hu et al., 2019). These methods focus on addressing the issue of gradient alignment by performing projections on the gradients so as to maximize transfer and minimize interference. Yu et al. (2020) identify the tragic triad of conflicting gradients, high positive curvature, and large gradient differences as a poignant issue in multi-task learning. In our work, we investigate methods that do not directly manipulate the gradients, but rather employ meta-learning techniques to minimize gradient interference and maximize gradient transfer as part of the objective function.

Weight Sharing: A commonly proposed solution to mitigating gradient interference is through limiting weight sharing. By limiting weight sharing, researchers have shown that past knowledge can be better preserved (Zenke et al., 2017; Kirkpatrick et al., 2016). Intuitively, with no weight sharing, gradient interference is completely avoided, but weight sharing can improve a model's ability to have positive gradient transfer, creating a difficult tradeoff to optimize over. This issue of optimal weight sharing is an interesting issue that can be examined from the lens of meta-learning.

Meta-Learning: Meta-learning has shown great promise in being able to address the issue of learning how to optimally share weights for models learning to solve multiple tasks at once (Finn et al., 2017). Meta-learning methods can

be quite computationally expensive due to the higher order gradient updates, but researchers have developed first-order approximations accordingly (Nichol et al., 2018). For our continual learning setting, meta-learning is a promising approach to learn how to update a model's parameters in order to optimally balance the gradient transfer-interference tradeoff.

Replay Buffers: Other works have noted that leveraging past experience can potentially help mitigate catastrophic forgetting (Lin, 1992; Atkinson et al., 2018; Riemer et al., 2018). These works suggest using an experience replay buffer to combine past experience with online data, which has shown success in reinforcement learning (Mnih et al., 2013). However, the replay buffers used by these works are either unbounded in size or sampling is done uniformly randomly (Finn et al., 2019). The use of replay buffers appears critical to successful continual learning models, as without a memory of past experiences, catastrophic forgetting becomes much more difficult to avoid. We would like to investigate the effect of various properties of replay buffers on the mitigation of catastrophic forgetting.

Prioritized Replay: Our hypothesis is that the order of which tasks are seen can greatly affect the extent to which catastrophic forgetting occurs. Prioritized experience replay has shown success in reinforcement learning settings by replaying important transitions more frequently, instead of simply uniformly randomly sampling (Schaul et al., 2015; Moore & Atkeson, 1993). Prioritizing the samples from an experience replay buffer can allow for more sample efficient and effective training. In supervised learning, prioritized replay draws analogies to resampling, under-sampling, and over-sampling techniques used to address class imbalance Galar et al. (2012); Hinton (2007). Intuitively, replay buffers serve as *memory* for the models in the continual learning domain, so it is important to understand how to maximize the benefits of the model's memory.

3 Transfer-Inteference Tradeoff and Meta-Learning Objective

3.1 Transfer-Inteference Tradeoff

One perspective for viewing transfer and interference is through that of gradient alignment. Given two distinct training data examples (x_i, y_i) and (x_j, y_j) with shared model parameters θ and loss L, gradient *transfer* occurs when:

$$\frac{\partial L(x_i, y_i)}{\partial \theta} \cdot \frac{\partial L(x_j, y_j)}{\partial \theta} > 0 \tag{1}$$

Namely, *transfer* occurs when the angle between gradient updates is < 90, indicating that the gradient steps are aligned within the objective landscape. Although not guaranteed, the hope is that when gradients are in alignment, learning on data point i also helps performance on j and vice versa. On the other hand, gradient *interference* occurs in the opposite situation:

$$\frac{\partial L(x_i, y_i)}{\partial \theta} \cdot \frac{\partial L(x_j, y_j)}{\partial \theta} < 0 \tag{2}$$

In this situation, the angle between gradients is > 90, meaning that the gradients are misaligned and conflicting. Therefore, learning on data point i conflicts with learning on data point j, as the gradient steps counteract each other. We refer to this situation throughout the paper as gradient interference, often resulting in unlearning or catastrophic forgetting of previously learned knowledge.

3.2 Learning to Maximize Transfer Without Forgetting: a Meta-Learning Perspective

In continual learning, maximizing transfer and minimizing interference are crucial objectives for developing robust, successful models. Using the perspective discussed previously, we can extend the traditional optimization objective over a stationary data distribution D to additionally optimize for gradient alignment:

$$\theta = arg \min_{\theta} \mathbb{E}_{[(x_i,y_i),(x_j,y_j)]\sim D} \big[L(x_i, y_i) + L(x_j, y_j) - \alpha \frac{\partial L(x_i, y_j)}{\partial \theta} \cdot \frac{\partial L(x_j, y_j)}{\partial \theta} \big] \tag{3}$$

Specifically, given two randomly sampled data examples i and j, in addition to minimizing their respective losses, we look to maximize the alignment of their gradients to maximize gradient transfer (Equation 1). Intuitively, we seek model parameter θ that not only optimally minimizes individual data losses but also encourages shared learning between different data examples. While Equation 3 directly addresses the *Transfer-Interference Tradeoff*, several problems must be addressed when applying it to continual learning.

Experience Replay: Firstly, in continual learning, data is presented as a streaming, non-stationary distribution. To address this, Riemer et al. (2018) introduce a fixed size replay buffer M, used to approximate the true stationary distribution D. This replay buffer M is updated throughout the continual learning process using reservoir sampling, a

technique for maintaining a random sample from a data stream with unknown length. In this way, M most approximately captures a stationary distribution over past examples at each time step. By replacing D with a distribution over the replay buffer M, we transform the objective for continual learning. One caveat to note is that in addition to sampling from the replay buffer during optimization, we always interleave the current data example to ensure that we train on every revealed example.

Meta-Learning Objective: A second problem that must be addressed is the second order optimization within Equation 3 that is very expensive to compute. Rather than model this objective exactly, Riemer et al. (2018) draw inspiration from first order meta-learning algorithms that learn how to learn. Specifically, Riemer et al. (2018) build upon the lightweight, first order meta-learning algorithm Reptile (Nichol et al., 2018). This model has several attractive properties. Firstly, in comparison to other optimization-based meta-learning models, such as MAML (Finn et al., 2017), Reptile does not require the expensive computation of second order gradients. Secondly, unlike MAML and other first order variants such as FOMAML, Reptile does not require training over (support, query) set pairs for each task. This crucially allows us to apply the Reptile objective in a purely multi-task, supervised learning setting. Lastly, Reptile approximately optimizes for the same objective as MAML, which further shows that it approximately optimizes for expected loss as well as *within task gradient alignment*. In a meta-learning setting, Reptile optimizes on each task individually by optimizing across s batches of data. After s SGD optimization steps, the meta update in Reptile updates parameters θ as $\theta_0 = \theta_0 + \beta * (\theta_s - \theta_0)$. This process then repeats over many iterations and tasks. Using a first order Taylor approximation, Nichol et al. (2018) show that Reptile approximately optimizes for the following objective over the set of k batches for a particular meta task:

$$\theta = arg \min_{\theta} \mathbb{E}_{B_1,\cdot,B_s \sim D} \big[2 \sum_{i=1}^{s} [L(B_i) - \sum_{j=1}^{i-1} \alpha \frac{\partial L(B_i)}{\partial \theta} \cdot \frac{\partial L(B_j)}{\partial \theta}]\big] \tag{4}$$

Namely, the optimization minimizes the expected loss over a task while also ensuring fast learning over the given task by maximally aligning gradients across training batches. Compared to Equation 3, we can immediately see the similarity, and, thus look to adapt Reptile for a continual learning setting.

The MER algorithm does exactly this (Riemer et al., 2018). Firstly, just like the transformation of Equation 3 to handle streaming data, we can replace the stationary distribution D in Equation 4 with the approximate replay buffer M.

$$\theta = arg \min_{\theta} \mathbb{E}_{B_1,\cdot,B_s \sim M} \big[2 \sum_{i=1}^{s} [L(B_i) - \sum_{j=1}^{i-1} \alpha \frac{\partial L(B_i)}{\partial \theta} \cdot \frac{\partial L(B_j)}{\partial \theta}]\big] \tag{5}$$

$B_1, \cdot, B_k$ are batches sampled from the replay buffer and are not necessarily drawn from one individual task. This first version of a continual learning based Reptile objective optimizes for gradient alignment between batches samples from the replay buffer that may contain data from different tasks. Although a perfectly valid approach, through experimental analysis, Riemer et al. (2018) discover that breaking each individual batch into constituent data samples and then performing both an inner (within batch) meta Reptile update and an outer (between batch) meta Reptile update performs even better. Specifically, we first write the inner (within batch) optimization as follows:

$$\theta = arg \min_{\theta} \mathbb{E}_{[(x_{i1},y_{i1}),\ldots,(x_{ik},y_{ik})] \sim B_i} \big[2 \sum_{j=1}^{k} [L(x_{ij}, y_{ij}) - \sum_{r=1}^{j-1} \alpha \frac{\partial L(x_{ij}, y_{ij})}{\partial \theta} \cdot \frac{\partial L(x_{ir}, y_{ir})}{\partial \theta}]\big] \tag{6}$$

where we perform one meta Reptile update over batch B_i, with each of the k examples within the batch treated as their own Reptile batch of size 1 from Equation 5. Now, adding a second meta optimization step over the s batches sampled from M, we extend Equation 6 to define our general continual learning objective:

$$\theta = arg \min_{\theta} \mathbb{E}_{[(x_{11},y_{11}),\ldots,(x_{sk},y_{sk})] \sim M} \big[2 \sum_{i=1}^{s} \sum_{j=1}^{k} [L(x_{ij}, y_{ij}) - \sum_{q=1}^{i-1} \sum_{r=1}^{j-1} \alpha \frac{\partial L(x_{ij}, y_{ij})}{\partial \theta} \cdot \frac{\partial L(x_{qr}, y_{qr})}{\partial \theta}]\big] \tag{7}$$

Breaking this equation down, the key difference as compared with Equation 5 for Reptile is that we now optimize not for gradient alignment across batches $1, \ldots, s$, but across the k elements within each batch. Overall, Equation 7 defines the general objective that we use to learn to learn in a way that promotes gradient alignment at each point in time.

4 CAML: Catastrophically-Aware Meta-Learning

With CAML, we investigate various prioritization schemes with the intuition that not all samples are equally important to learn from. Namely, we build upon the MER algorithm proposed by Riemer et al. (2018) and add various novel

prioritized meta-experience replay methods. We focus on describing the prioritization techniques used for CAML in this section.

4.1 Prioritized Meta-Experience Replay:

The meta-experience replay buffer M contains seen examples that are interleaved with the training of current examples with the goal of maximizing gradient transfer and minimizing gradient interference to avoid forgetting.

The buffer is updated via reservoir sampling, which ensures that every time step, the probability that any of the N samples seen has probability $\frac{M_{size}}{N}$ of being in the buffer. Each example i in the replay buffer M has a corresponding priority p_i. The priority assigned to each of these samples are determined by the different prioritization schemes introduced below. Concretely, we define the probability of sampling example i in the replay buffer as

$$P(i) = \frac{e^{(p_i/T)}}{\sum_{j=1}^{M_{size}} e^{(p_j/T)}} \tag{8}$$

We note here that the priority p_i can take on any real value. T denotes the temperature of the softmax distribution, and as T increases, the probabilities become more uniform. Moreover, our transformation is monotonic in the priorities, and we ensure that there is always a non-zero probability of sampling a given example. Tuning the hyperparameter T interpolates between greedy sampling and uniform random sampling, which can be useful in varying contexts. Further research in dynamically adjusting T throughout training is extremely promising, but we choose to fix T in our experiments.

Whenever an example is sampled from the replay buffer, its priority is recomputed using the respective prioritization scheme. This is as opposed to recomputing the priorities of examples every time step, which would be much more computationally expensive, but could potentially provide more insightful and accurate priorities.

4.2 Prioritization Schemes:

We investigate four prioritization schemes for CAML: 1) prioritizing on oldest samples, 2) newest samples, 3) samples with highest loss, and finally 4) a dynamic priority (a weighted average of age and highest loss).

A variable age is incremented by 1 every time a new sample is seen during training. We use this measure of absolute age in order to encode when an example is added to the replay buffer.

Oldest Priority: Prioritizing oldest samples leverages the intuition that training on samples seen a long time ago will help a model retain its ability to perform well on previous tasks and mitigate catastrophic forgetting.

We define the Oldest priority as follows:

$$p_i^{oldest} = -age$$

Since the age is absolute and increasing throughout training, we negate the age in order to prioritize older examples. We refer to this model as CAML (Oldest).

Newest Priority: On the contrary to Oldest priority, we also inspect Newest priority. We define the Newest priority as follows:

$$p_i^{newest} = age$$

We refer to this model as CAML (Newest).

Loss Priority: We also investigate prioritizing on samples with the highest loss. This uses the intuition that the model would perform better given access to samples that it performed worst on previously. Intuitively, by training on high loss examples, learning should be accelerated and we should observe increased sample efficiency.

We define the Loss priority as follows:

$$p_i^{loss} = Loss(x_i, y_i)$$

We refer to this model as CAML (Loss).

Dynamic Priority: Finally, with the intuition that different prioritization schemes may provide different benefits, we designed the Dynamic priority that combines relative age with highest loss. Intuitively, the Loss priority should encourage maximizing transfer, whereas the Oldest priority should encourage minimizing interference. Combining the two could lead to more optimal improvements of both objectives.

Concretely, the Dynamic priority is defined as follows:

$$p_i^{dynamic} = Loss(x_i, y_i) + \alpha_d \cdot age_{rel}$$

Here, age_{rel} represents the relative age of an example in the replay buffer. In contrast to the absolute age variable, age_{rel} represents the amount of time since an example was last sampled from the replay buffer. Note that absolute age continually and monotonically increases throughout training, so it would eventually dominate the loss component if used in this prioritization scheme. We use relative age instead of absolute age in order to prevent a divergence in the distributions of the individual components in the Dynamic priority. α_d represents the weight each time step should be worth relative to the loss component and can be tuned accordingly as a hyperparameter. (We use 0.05 in our experiments). We refer to this model as CAML (Dynamic).

5 EXPERIMENTS

5.1 DATASET

For experimental analysis, we focus on the MNIST Rotations dataset, a common continual learning benchmark (Lopez-Paz & Ranzato, 2017). This dataset augments the well known MNIST dataset, where different tasks presented throughout the continual learning process contain MNIST digits (0-9) rotated by different fixed angles between 0 and 180 degrees. In total, the dataset consists of 20 different tasks to be presented sequentially through continual learning. In order to gauge sample efficiency and highlight the importance of prioritized replay in the few-shot setting, we use a buffer size of 128 and learn from 10 samples from each of the 20 tasks. This differs significantly from the larger data regime used by Riemer et al. (2018), which accounts for the difference in reported results.

5.2 BASELINE MODELS

We compare our prioritization variants of the CAML algorithm with standard continual learning baselines.

1. **Online:** The model uses a vanilla supervised learning approach by training on one example at a time on the incoming data by simply applying SGD.
2. **Independent:** A separate, independent model is trained for each task, so no parameters are shared.
3. **Task Input:** The model contains an independent input layer per task and a shared output layer with the same architecture as the Online model.
4. **MER:** The model combines Reptile with a replay buffer to learn how to learn without forgetting.

5.3 EVALUATION

We consider three quantitative metrics to evaluate model success over the continual learning task and to measure catastrophic forgetting:

1. Retained Accuracy (RA): Final accuracy across all tasks after sequential training complete.
2. Learned Accuracy (LA): Average accuracy for each task immediately after it is learned.
3. Backward Transfer and Interference (BTI): Average change in accuracy from when a task is learned to end of training. A low BTI represents catastrophic forgetting.

5.4 GENERAL MODEL COMPARISON

Table 1 summarizes the results of CAML compared to baselines. CAML (Loss) achieves the highest retained accuracy. CAML (Oldest) does best with respect to BTI, so it is most robust to catastrophic forgetting, which makes sense since it prioritizes training on old data so as not to forget how to perform well on old tasks. We see CAML (Newest) does the best in learned accuracy. This makes sense since it prioritizes learning on more recent data during training, which

Model	*LA*	*RA*	*BTI*
Online	0.2529	0.1879	-0.0650
Independent	0.1016	0.1016	-
Task Input	0.1313	0.1100	-0.0213
MER	0.3473	0.5105	0.1632
CAML (Newest)	**0.3911**	0.4462	0.0551
CAML (Oldest)	0.3482	0.5248	0.1766
CAML (Loss)	0.3731	**0.5438**	0.1707
CAML (Dynamic)	0.3227	0.5369	**0.2142**

Table 1: Model performances on MNIST Rotations. CAML (Newest) performs best in terms of learned accuracy, or the accuracy immediately after training on a task, which makes sense as it prioritizes learning on new data from the new task at the expense of catastrophic forgetting on old tasks. CAML (Loss) performs best in terms of retained accuracy – or average final accuracy on all tasks. CAML (Dynamic) combines the benefits of both Oldest and Loss prioritization methods to achieve the best BTI score by far and nearly the highest retained accuracy, which shows promise in developing a prioritization scheme that combines multiple objectives.

is likely to be from the same task as the current example. CAML (Newest) does worse in RA than MER, and has a low BTI, which suggests catastrophic forgetting, which makes sense as well because when prioritizing newer data, the model is likely to forget how to perform well on older tasks. CAML (Dynamic) is able to leverage the benefits from both Oldest and Loss prioritizations, showing the most robustness to catastrophic forgetting as well as nearly the highest retained accuracy.

5.5 Varying Buffer Size

Model	*Buffer Size*	*LA*	*RA*	*BTI*
CAML (Loss)	16	0.3128	0.3788	0.0660
	64	0.3536	0.4787	0.1251
	128	**0.3731**	**0.5438**	**0.1707**
	200	**0.3756**	**0.5384**	0.1628
MER	16	**0.3203**	**0.4090**	**0.0887**
	64	**0.3565**	**0.5009**	**0.1445**
	128	0.3473	0.5105	0.1632
	200	0.3469	0.5157	**0.1688**

Table 2: Varying buffer sizes experiment. CAML outperforms MER as the buffer size increases, suggesting that prioritizing replay becomes more important in a setting where there are more samples to choose from.

We compare MER with CAML (Loss) across different buffer sizes (up to a max buffer size of 200, which holds all 200 samples in our limited data regime setting) in Table 2. MER appears to outperform CAML in the lower buffer size regime. However, as buffer size increases to 128 and 200, we observe that CAML begins to outperform MER. We hypothesize that prioritizing replay with highest loss is more beneficial when the replay buffer has more samples to choose from. Specifically, when the buffer size is small, there is less of a need to proactively select examples to learn from, and uniform sampling is likely sufficient.

6 Qualitative Analysis: Highlighting Catastrophic Forgetting

6.1 Continual Learning Heat maps

To visualize model performance throughout the continual learning process, we generate task performance heat maps. Descending the y-axis of the heat map represents the sequential revealing of tasks through time, while the x-axis labels the individual tasks. For a given time step t (i.e. revealing of task Tt), a row in the heat map represents the accuracy of the *current model* over the individual tasks learned so far ($T1, \ldots, Tt$). Therefore, the rows of the heat map track retained performance over previously learned tasks, while the models learns on new tasks in the continual learning

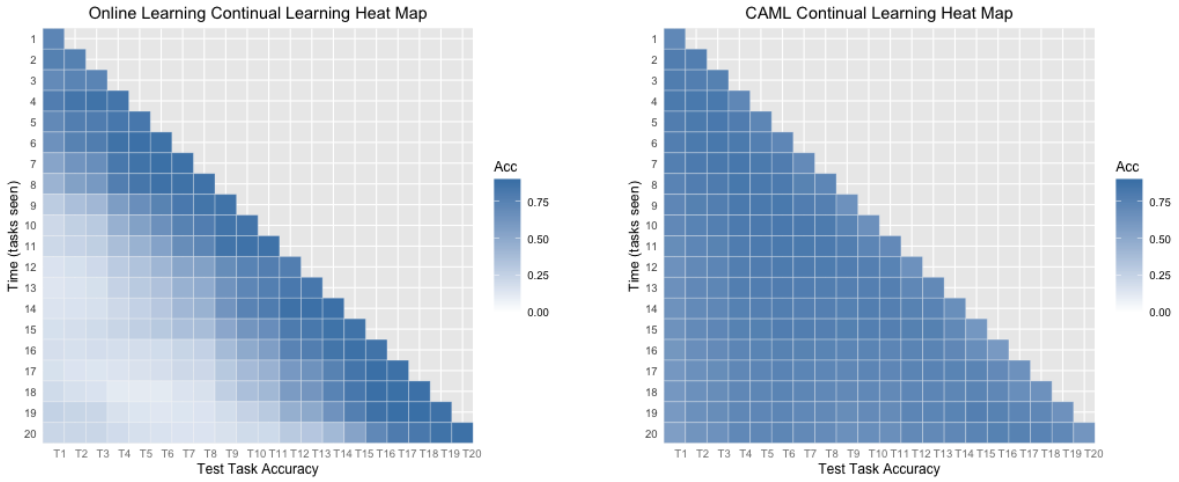

Figure 1: Heat map representations of model performance throughout the continual learning process. The y-axis represents the progression of time and the x-axis labels the individual tasks. **(a) Left:** The Online baseline suffers from catastrophic forgetting, as shown by the diminished accuracy on old tasks as new tasks are learned. **(b) Right:** CAML (Loss) displays robustness to forgetting by learning new tasks while retaining performance on old tasks.

process. For guided interpretation, the diagonal represents accuracies immediately after learning a task and reduced accuracy going down a single column reveals catastrophic forgetting.

To generate the heat maps presented in Figure 1, we compare the Online baseline with CAML (Loss). Specifically, we compare training the Online baseline with 1000 examples per task against CAML trained with 100 examples per task with a replay buffer of size 128. We explicitly expose the Online model to a greater number of examples per class to emphasize the forgetting gap between the Online model and CAML, even when the Online model has access to more training examples for each task.

Figure 1 demonstrates the improved overall performance of CAML and highlights CAML's superior ability to combat catastrophic forgetting. Analyzing the heat map of the Online model, we observe that the model performs very strongly on the current learning task (i.e. along the diagonal); however, learning over new tasks shows significant interference with performance over previous tasks, which we visualize as diminishing accuracies as one moves away from the diagonal or down a given task's column in time. In contrast, the heat map generated for CAML clearly shows the strong performance of CAML throughout the continual learning process across all learned tasks. In fact, we even see continued improvements away from the diagonal, as the replay buffer and gradient alignment optimization from Reptile encourage past tasks to aid in the learning of newer tasks, while maintaining or even boosting performance over the sampled previous tasks.

6.2 Forgetting Through Time: A Single Task Case Study

As a follow up to the heat map analysis of the previous section, we zoom in and analyze the performance on a single task during continual learning. Following the same experimental setting as in the heat map study, we track progress of the learned model just over the 4th rotation task ($34°$). We specifically select the 4th task, as opposed to e.g. the 1st, to avoid the *cold start problem* and allow for several iterations of initial training. Mixing qualitative and quantitative analysis, we visualize a grid of histogram plots after sequential learning of each task $t = 4, \ldots, 20$. Each histogram then displays the error counts for each MNIST digit (i.e. how many time the ground truth digit class was misclassified) based on the current model at time t, for just the task 4 test data set.

Figure 2 shows this fine grained comparison between the Online baseline model and CAML. In both settings we see that in the tasks learned immediately after the reference task 4, both models have comparatively low MNIST digit error rates. This makes sense as the Online model focuses its energy purely on newly revealed tasks and for a short while accuracy remains high over past tasks; however, we quickly see that there is a limit to this, and after 7 or 8 tasks (by rotation 85 and 95), we see error rates have significantly increased. The growth in error rates over time for the old task 4 further confirms previous observations of explicit forgetting in the Online model; in contrast, the relatively stable error bars through time in the CAML grid additionally confirms the robustness of CAML to task interference.

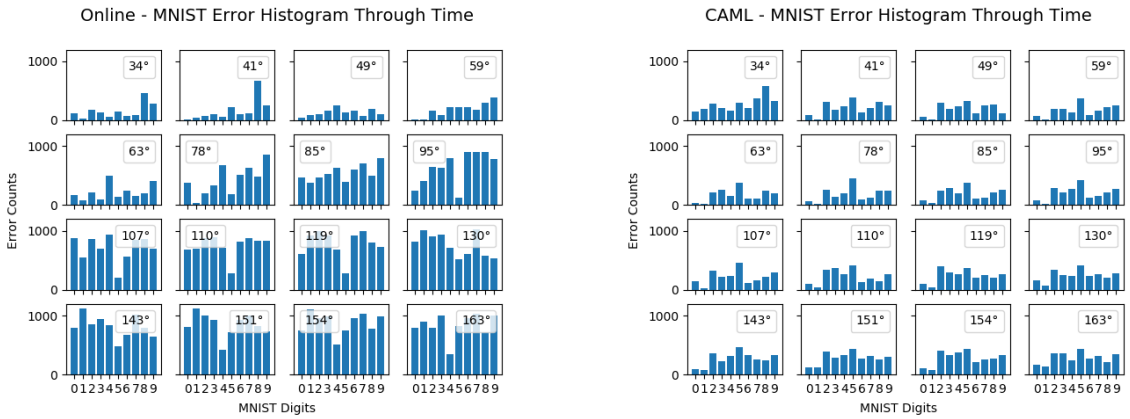

Figure 2: Histogram comparison of MNIST digit class error rates through time on rotation task 4 (34°). Each histogram displays error counts for each MNIST digit based on the current model at time t, for just task 4. **(a) Left:** The Online baseline shows severe forgetting over class 4 shown by a progressive increase in error rates through time. **(b) Right:** In contrast, CAML shows stable performance over task 4 performance with only slight fluctuations in error rates through time.

One interesting follow up that we leave for future work is an analysis of the fluctuations in errors for different time steps, which may shed light on different levels of interference that exists between tasks.

6.3 Saliency Maps: A Window Into What Models See

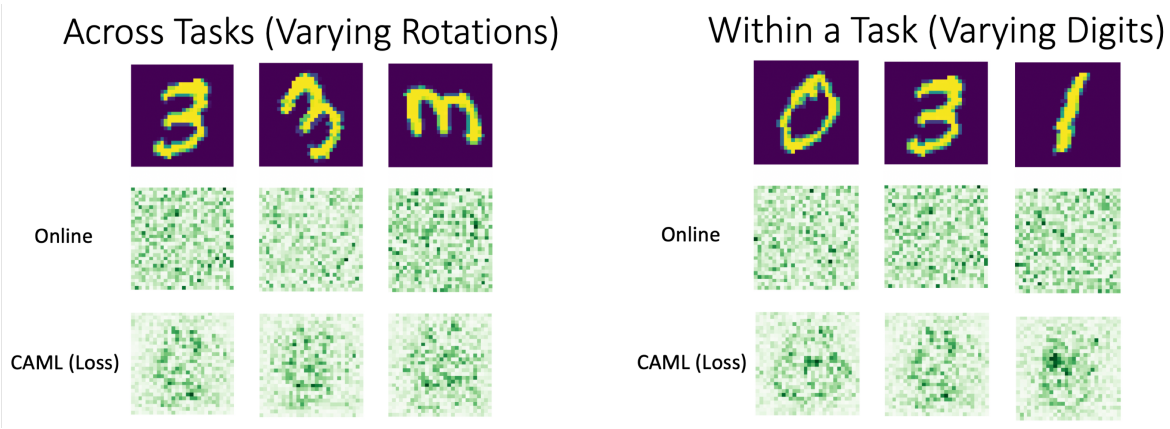

Figure 3: Saliency maps for insight into model interpretability. In both sets of saliency maps, the Online baseline appears to be attending to random parts of the image, leading to greater catastrophic forgetting as new tasks are seen. CAML attends to semantically meaningful parts of the image so as to maximize transfer and minimize interference. **(a) Left:** Inter-task saliency maps. **(b) Right:** Intra-task saliency maps.

Saliency maps represent a window into the eyes of a neural network. Namely, these visual representations attempt to explain or highlight the parts of an image that strongly influence its classification. One common approach for generating these saliency maps is through visualizing the gradient of the true class score with respect to the input image. The SmoothGrad method (Smilkov et al., 2017) extends this idea to visually sharpen gradient-based saliency maps by averaging across a collection of saliency maps generated from randomly perturbed versions of the input image:

$$\hat{M}_c(x) = \frac{1}{n} \sum_{i=1}^{n} \left| M_c(x + \mathcal{N}(0, \sigma^2)) \right| \tag{9}$$

where M_c represents a saliency map for the given input x ($M_c = \frac{\partial}{\partial x} Loss(x, y)$) and n and σ are hyperparameters controlling the sample size and amount of noise added. We use default parameters suggested by the paper $n = 50$ and $\sigma = 0.1 * (x_{max} - x_{min})$.

Using SmoothGrad, we generate saliency maps to further analyze the issue of catastrophic forgetting and task interference from a qualitative lens. We compare saliency maps between an Online model and CAML (Loss). Saliency maps are generated by selecting random data samples from held out task test sets, after training is completed. Different from previous settings, we adjust training hyperparameters by increasing the number of samples seen per task to be 1000 and increasing the CAML buffer size to be 5120; by evaluating models in a more data rich setting, as opposed to the few-shot and data-sparse settings used before, we focus primarily on visualizing the effects of forgetting and task interference through time.

Figure 3 shows two different settings for saliency map comparison. On the left, we visualize saliency maps generated across tasks, which in this case equates to different rotations of the same MNIST digit. On the right, we visualize saliency maps generated for varying digits within a single task. In both settings, we can see that Online attends to random aspects of the image, whereas CAML attends more clearly to semantically meaningful parts of the image near the center and near the digits. We hypothesize that this observation helps explain CAML's overall better performance as well as resistance to forgetting; by attending to semantically meaningful regions within/around the digits, CAML demonstrates that it has learned a more flexible representation of rotated digits. In contrast, the random noise revealed in the Online model's saliency maps qualitatively suggests the presence of strong task interference. High gradient interference between tasks (i.e. different MNIST rotations having different meaningful image patches) likely leads to task vs. task conflict within the learning process, causing the model to unlearn past tasks while only partially learning new tasks. This results in a confused model that likely learns based on semantically less meaningful features, as important features conflict across task to task rotations.

7 Conclusion

Summary: In this work, we propose a novel meta learning algorithm that learns to learn without forgetting in a sample efficient manner by prioritizing the most important samples to learn from in the meta-experience replay buffer. We find that CAML has several attractive properties: different prioritization schemes outperform baselines in specific metrics. CAML (Newest) performs best on learned accuracy, CAML (Loss) performs best on retained accuracy, and CAML (Oldest) performs well in backward transfer and interference. CAML (Dynamic) is able to achieve a good tradeoff between maximizing transfer and minimizing interference by achieving nearly the highest average final accuracy and by far the most robustness to catastrophic forgetting. We perform significant qualitative analysis both inter-task and intra-task as a form of model introspection.

Limitations: In the small data, few-shot regime, CAML displays significant gains in performance over baseline algorithms. However, preliminary analysis into the larger data regimes suggest that the priorities of the examples in the meta-experience replay buffer are relatively uniform, which would make CAML extremely similar to uniform sampling. Our experiments are also only performed on MNIST Rotations due to computational and time constraints, but further analysis on other non-stationary, more complex settings would be useful.

Future Work: Our results motivate future exploration into prioritized replay buffers for continual learning. While we used fixed heuristics for suitable prioritization schemes, future work could involve directly learning the priorities of examples instead. This would be similar to tackling the exploration vs. exploitation tradeoff, where learning how to prioritize examples could allow a model to choose when to sample more uniformly vs. greedily.

Group Member Contributions:

Woody: Woody helped develop the CAML prioritization schemes as well as implement the CAML architecture. He ran experiments to obtain the paper's main empirical results and helped write the manuscript.

Tristan: Tristan analyzed the effects the buffer size had for different models, conducted experiments to understand prioritization's efficacy by examining the contents of the buffer during every step of training, and built graphs to visualize those effects. He is taking this class for credit/no credit, and believes a bigger portion of this project should be credited to his teammates Woody and Jonathan, if possible. He also helped write the manuscript.

Jon: Jon helped develop the CAML prioritization schemes and formalized the meta-learning objective. He performed significant qualitative analysis to for introspection into model interpretability. He also helped write the manuscript.

Code available here: `https://github.com/jonathangomesselman/CS330-CAML`

REFERENCES

Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. Pseudo-rehearsal: Achieving deep reinforcement learning without catastrophic forgetting. *CoRR*, abs/1812.02464, 2018. URL http://arxiv.org/abs/1812.02464.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400, 2017. URL http://arxiv.org/abs/1703.03400.

Chelsea Finn, Aravind Rajeswaran, Sham M. Kakade, and Sergey Levine. Online meta-learning. *CoRR*, abs/1902.08438, 2019. URL http://arxiv.org/abs/1902.08438.

M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2012. doi: 10.1109/TSMCC.2011.2161285.

Geoffrey E. Hinton. To recognize shapes, first learn to generate images. *Progress in brain research*, 165:535–47, 2007.

Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao Tao, Dongyan Zhao, Jinwen Ma, and Rui Yan. Overcoming catastrophic forgetting for continual learning via model adaptation. *International Conference on Learning Representations*, May 2019. URL http://par.nsf.gov/biblio/10120468.

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. URL http://arxiv.org/abs/1612.00796.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach. Learn.*, 8 (3–4):293–321, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992699. URL https://doi.org/10.1007/BF00992699.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continuum learning. *CoRR*, abs/1706.08840, 2017. URL http://arxiv.org/abs/1706.08840.

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, January 1989. ISSN 0079-7421. doi: 10.1016/S0079-7421(08)60536-8.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS 2013 Deep Learning Workshop*, 2013.

Andrew W. Moore and Christopher G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Mach. Learn.*, 13(1):103–130, October 1993. ISSN 0885-6125. doi: 10.1023/A:1022635613229. URL https://doi.org/10.1023/A:1022635613229.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999, 2018. URL http://arxiv.org/abs/1803.02999.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *CoRR*, abs/1810.11910, 2018. URL http://arxiv.org/abs/1810.11910.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

S. Thrun. A lifelong learning perspective for mobile robot control. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, volume 1, pp. 23–30 vol.1, 1994. doi: 10.1109/IROS.1994.407413.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? In *Proceedings of the 8th International Conference on Neural Information Processing Systems*, NIPS'95, pp. 640–646, Cambridge, MA, USA, 1995. MIT Press.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning, 2020.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Improved multitask learning through synaptic intelligence. *CoRR*, abs/1703.04200, 2017. URL `http://arxiv.org/abs/1703.04200`.