Task Agnostic Continual Learning via Meta Learning

Abstract

Most continual learning approaches implicitly assume that there exists a multi-task solution for the sequence of tasks. In this work, we motivate and discuss realistic scenarios when this assumption does not hold. We argue that the traditional metric of zero-shot remembering is not appropriate in such settings, and, inspired by the meta-learning literature, we focus on the speed of remembering previous tasks. A natural approach to deal with this case is to separate the concerns into *what* task is currently being solved and *how* the task should be solved. At each step, the *what* algorithm performs task inference, which allows our framework to work in absence of task boundaries. The *how* algorithm is conditioned on the inferred task, allowing for task-specific behaviour, hence relaxing the assumption of a multi-task solution. From the perspective of meta-learning, our framework is able to deal with a sequential presentation of tasks, rather than having access to the distribution of all tasks. We empirically validate the effectiveness of our approach and discuss variations of the proposed algorithm.

1. Introduction

Connectionist networks are known to suffer from catastrophic forgetting (CF) (McCloskey & Cohen, 1989), when the underlying data stream is not independently and identically distributed (i.i.d). Continual Learning (CL) explores this problem, where non-stationarity of data is given as a sequence of distinct tasks. One typical goal of many CL algorithms is to ensure that, after training on a sequence of tasks, the performance of the network is close to a network trained on all tasks at the same time. Hence, all these methods implicitly assume that there is always a multi-task solution that fits all previous tasks. Another common assumption is that either the identities of different tasks or the boundaries between them are available to the learner, and many CL methods often depend on this information to know when to consolidate the knowledge learned so far.

However, there are many realistic scenarios where both of the above-mentioned assumptions do not hold. Consider a multi-agent game in reinforcement learning (RL), where all agents are learning and adapting their policies. For any of these agents, the objective it tries to optimize (in other words, its task) depends not only on itself and the environment, but also on the policies and configurations of others, which are usually not directly observable. Moreover, the other agents might change their policies at any moment as they are also learning. As a result, the task for this agent is changing all the time and there are no clearly defined boundaries available to the agent. It has been observed that such non-stationarity in multi-agent systems usually causes catastrophic forgetting of the agent (Hernandez-Leal et al., 2019). For example, Vinyals et al. (2019) trained agents to play the video game StarCraft II by self-play (Tesauro, 1995), and they noticed that one salient drawback of this approach is in fact forgetting: the agent may forget how to defeat a previous version of itself as training progresses, and this may lead to a "tail-chasing" cycle where the agents always relearn a previously learned strategy and training never converges.

Furthermore, since the tasks now depend on the configurations of other agents, in general, there is no guarantee that a multi-task solution would exist in these settings. A potential example is Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), where a generator G and a discriminator D are trained together by playing a minimax game. The objective of D is to classify the data as real or fake, whereas the goal of G is to fool D as much as possible by generating fake data. It was shown in (Goodfellow et al., 2014) that the optimal discriminator $D^*(x)$ is a function of the generator probability density function $p_G(x)$,

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$

Therefore, if we take two snapshots G_1, G_2 of the generator at different moments of the training process such that $p_{G_1}(x) \neq p_{G_2}(x)$ for some x where $p_{data}(x) \neq 0$, then their corresponding optimal discriminators have to be different. In other words, there is no multi-task solution for the discriminator to be optimal for both generators. As a result, optimizing the discriminator with respect to a new version of the generator will inevitably lead to degradation of its performance with respect to a past generator, which is considered *forgetting* by the traditional metric of continual learning. Indeed, it has been shown empirically by Thanh-Tung & Tran (2019); Liang et al. (2019) that GANs suffer from

catastrophic forgetting, and they adapted CL methods such as Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) and Synaptic Intelligence (SI) (Zenke et al., 2017) to alleviate forgetting in GANs . However, as we pointed out before, these methods were initially designed with implicit assumptions that a multitask solution always exists and precise task boundaries are available, which makes them unsuitable for the setting of multi-agent games like GANs.

In this work, we propose a CL framework that does not make these assumptions and is applicable in a task agnostic scenario where the tasks can potentially be conflicting with each other. Furthermore, to evaluate our framework, we shift our focus from less forgetting to faster remembering: to rapidly recover the performance on a previously learned task, given the right context as a cue.

2. Formal Statement

We consider an online learning scenario similar to Hochreiter et al. (2001); Nagabandi et al. (2019), where at each time step t, a model $\hat{f}$ parametrised by θ_t receives an observation x_t and makes a prediction $\hat{y}_t := \hat{f}(x_t; \theta_t)$. It then gets the ground truth y_t on that task, which can be used to optimize its parameters for better performance in the future. If the data distribution is non-stationary (for example, (x, y) are sampled from task A for a while, then the task switches to task B at some moment t'), then training on the new data might lead to catastrophic forgetting – the new parameters θ' can solve task B but not task A anymore.

Many continual learning methods were proposed to alleviate the problem of catastrophic forgetting. Most of them require either the task identities (A and B in the example) or at least the moment when the task switches (t' in this case). This information, however, is not available when the ground truth y_t depends not only on the observation x_t but also on some hidden task (or context) variable T_t: $y_t = f(x_t, T_t)$, a common situation in partially observable environments (Monahan, 1982; Cassandra et al., 1994). Only recently, the CL community started to look at the task agnostic setting (Zeno et al., 2018; Aljundi et al., 2019). However, all these methods have the underlying assumption that no matter what tasks the learner has been learning, at any time t, it is always possible to find parameters θ_t that fit all previous tasks: $\exists \theta_t \; s.t. \; \forall t' \leq t, \hat{f}(x_{t'}, \theta_t) \approx y_{t'}$. As discussed in the previous section, this assumption does not hold in many realistic scenarios where different tasks conflict with each other: $f(x_t, T_t) \neq f(x_{t'}, T_{t'})$ even when $x_t = x_{t'}$. It follows that, in those settings, catastrophic forgetting cannot be avoided if the model $\hat{f}(\cdot; \theta_t)$ does not depend on the hidden task variable T_t.

3. What & How Framework

Here we propose a framework for task agnostic continual learning that explicitly infers the current task from some context data $\mathcal{D}_t^{\text{ctx}}$ and makes predictions based on both the inputs x_t and the inferred task representations c_t. The framework consists of two modules: a task inference encoder algorithm $\mathcal{F}^{\text{what}} : \mathcal{D}_t^{\text{ctx}} \to c_t$ that predicts the current task representation c_t based on the context data $\mathcal{D}_t^{\text{ctx}}$, and a decoder algorithm $\mathcal{F}^{\text{How}} : c_t \to \hat{f}_t$ that maps the task representation c_t to a task specific model $\hat{f}_t : x \to \hat{y}$.

Under this framework, even when the inputs x_t and $x_{t'}$ are the same, the predictions $\hat{y}_t$ and $\hat{y}_{t'}$ can be different from each other depending on the context. In this work, we choose the recent k observations $\{(x_{t-k}, y_{t-k}), \cdots (x_{t-1}, y_{t-1})\}$ as the context dataset $\mathcal{D}_t^{\text{ctx}}$. This choice is reasonable in an environment where the task variable T_t is piece-wise stationary or changes smoothly. An overview of this framework is illustrated in Figure 1.

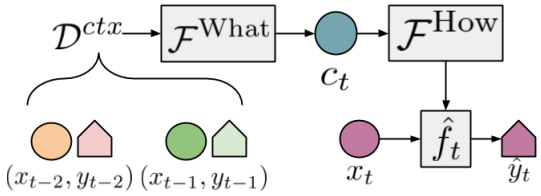

Figure 1. What & How framework

3.1. Meta Learning as Task Inference

In fact, many recently proposed meta-learning methods can be seen as decomposing the problem into What and How modules. For example, Conditional Neural Processes (CNP) (Garnelo et al., 2018) embed the observation and target pairs in context data $(x_i, y_i) \in \mathcal{D}_t^{\text{ctx}}$ by an encoder network $r_i = h(x_i, y_i; \theta_h)$. The embeddings are then aggregated by a commutative operation $\oplus$ (such as the mean operation) to obtain a single embedding of the context: $r_t = \mathcal{F}^{\text{What}}(\mathcal{D}_t^{\text{ctx}}; \theta_h) = \bigoplus_{x_i, y_i \in \mathcal{D}_t^{\text{ctx}}} h(x_i, y_i; \theta_h)$. At inference time, the context embedding is passed as an additional input to a decoder g to produce the conditional outputs: $\mathcal{F}^{\text{How}}(r_t) = g(\cdot, r_t; \theta_g)$.

Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) infers the current task by applying one or a few steps of gradient descent on the context data $\mathcal{D}_t^{\text{ctx}}$. In this case, the gradient descent algorithm is the What encoder and the resulting task-specific parameters can be considered a high-dimensional representation of the current task: $\theta_t^k = \mathcal{F}^{\text{What}}(\mathcal{D}_t^{\text{ctx}}; \theta^{\text{init}}) = U^k(\theta^{\text{init}}, \mathcal{D}_t^{\text{ctx}}, \lambda^{\text{in}}) := \theta_t^{k-1} - \lambda^{\text{in}} \nabla_\theta \mathcal{L}^{\text{in}}(\hat{f}(\cdot; \theta_t^{k-1}), \mathcal{D}_t^{\text{ctx}})$, where the meta parameters θ_t^{init} are the initial values of the model parameters, U^k is the operator that updates θ^{init} by k steps of gradient descent on the

context data $\mathcal{D}_t^{\text{ctx}}$ with an inner loop learning rate λ^{in} and an inner loop loss function $\mathcal{L}^{\text{in}}$. The How decoder of MAML returns the task-specific model by simply re-parametrizing the model $\hat{f}$ with θ_t: $\mathcal{F}^{\text{How}}(\theta_t) := \hat{f}(\cdot; \theta_t)$.

In Fast Context Adaptation via Meta-Learning (CAVIA) (Zintgraf et al., 2019), a neural network model $\hat{f}$ takes a context vector c_t as an additional input: $\hat{y} = \hat{f}(x, c_t; \theta)$. The context vector is inferred from context data by a few steps of gradient descent: $c_t = \mathcal{F}^{\text{What}}(\mathcal{D}_t^{\text{ctx}}; \theta) := c^{\text{init}} - \lambda^{\text{in}} \nabla_c \mathcal{L}^{\text{in}}(\hat{f}(\cdot, c; \theta), \mathcal{D}_t^{\text{ctx}})$. Then a context-dependent model is returned by the How decoder: $\mathcal{F}^{\text{How}}(c_t) := \hat{f}(\cdot, c_t; \theta)$.

In this work, to showcase our framework, we choose a simple meta learning method called Reptile (Nichol et al., 2018), mainly for its simplicity and for being computationally inexpenssive as it does not require second order gradients. Similar to MAML, Reptile tries to learn an initialization of model parameters θ_t^{init} such that optimization on a test task is fast, so its What encoder and How decoder are exactly the same as those of MAML. To update the meta parameters θ_t^{init}, Reptile simply uses the difference between the task-specific parameters and the initialization as the gradient direction: $g_{\theta^{\text{init}}} := \theta_t^{\text{init}} - \theta_t^k = \theta_t^{\text{init}} - U^k(\theta_t^{\text{init}}, \mathcal{D}_t^{\text{ctx}}, \lambda^{\text{in}})$.

3.2. Continual Meta Learning

In order to train a meta-learning model, one normally needs access to a task distribution so that i.i.d task samples are available at the same time during training. This is not possible in the online learning setting where tasks are presented sequentially one after the other. Finn et al. (2019) proposed an online meta-learning algorithm called *follow the meta leader* (FTML) based on the framework of regret-minimization. However, the computational cost of FTML grows linearly over time as new losses are accumulated, and it requires to store all datapoints from previous tasks, which is usually considered infeasible in continual learning due to limited resources or privacy reasons. In this work, we choose an alternative framework of online learning called online variational Bayes (Minka et al., 2009; Opper, 1998), since it does not require unbounded computational and memory budget. Furthermore, when additional memory budget are available for storing datapoints, online variational Bayes can also be extended by combining it with memory-based online learning methods (Minka et al., 2009; Nguyen et al., 2017; Kurle et al., 2020). In this work, we focus on an algorithm that does not have a growing memory cost over time.

Formally, let ϕ be the collection of meta parameters in $\mathcal{F}^{\text{What}}$ and $\mathcal{F}^{\text{How}}$ (for instance, θ^{init} in MAML and Reptile) such that $\hat{f}_t = \mathcal{F}^{\text{How}} \circ \mathcal{F}^{\text{What}}(\mathcal{D}_t^{\text{ctx}}; \phi)$. Using the Bayes rule, the posterior $p(\phi|\mathcal{D}_{0:t})$ can be recursively updated by

$$p(\phi|\mathcal{D}_{0:t}) = \frac{p(\mathcal{D}_t|\phi, \mathcal{D}_{0:t-1}) p(\phi|\mathcal{D}_{0:t-1})}{p(\mathcal{D}_t|\mathcal{D}_{0:t-1})} \tag{1}$$

where $\mathcal{D}_t = \{(x_t, y_t)\}$ and $\mathcal{D}_{0:t}$ is the union of all datapoints up to t. By our assumption that a moving window of context data is informative enough about the task variable, we have

$$\begin{aligned} p(\mathcal{D}_t|\phi, \mathcal{D}_{0:t-1}) &\approx p(\mathcal{D}_t|\phi, \mathcal{D}_t^{\text{ctx}}) = p(y_t|\hat{f}_t(x_t)) \\ &= p(y_t|\mathcal{F}^{\text{How}} \circ \mathcal{F}^{\text{What}}(\mathcal{D}_t^{\text{ctx}}; \phi)(x_t)) \end{aligned} \tag{2}$$

In online variational Bayes, the true posterior is approximated by a parametric distribution $q_t(\phi)$ by minimizing the Kullback-Leibler divergence

$$q_t(\phi) = \arg\min_{q(\phi)} \text{KL}(q(\phi)||p(\phi|\mathcal{D}_{t_0:t_n})) \tag{3}$$

and if we use a parametric distribution at every time step, the optimization problem above can be simplified as maximizing the evidence lower bound (ELBO) $\mathcal{E}(q(\phi), \mathcal{D}_{0:t}, q_{t-1}(\phi)) = \mathbb{E}_{q(\phi)}[\log p(\mathcal{D}_t|\phi, \mathcal{D}_{0:t-1})] - \text{KL}(q(\phi)||q_{t-1}(\phi))$.

In this work, we choose the parametric distribution to be a factorized Gaussian $q_t(\phi) = \prod_i \mathcal{N}(\phi_i|\mu_i(t), \sigma_i(t))$, where ϕ_i is the i-th component of ϕ. Using the re-parametrization trick: $\phi_i = \mu_i + \sigma_i \epsilon_i,\ \epsilon_i \sim \mathcal{N}(0, 1)$ and let $q_{t-1}(\phi) = \prod_i \mathcal{N}(\phi_i|\mu_i(t-1), \sigma_i(t-1))$, we can find the maximum of the ELBO by solving the following equations

$$\frac{\partial}{\partial \mu_i(t)} \mathcal{E}(q(\phi), \mathcal{D}_{0:t}, q_{t-1}(\phi)) = 0 \tag{4}$$

$$\frac{\partial}{\partial \sigma_i(t)} \mathcal{E}(q(\phi), \mathcal{D}_{0:t}, q_{t-1}(\phi)) = 0 \tag{5}$$

and the results are update rules for μ_i and σ_i that are similar to Bayesian Online Learning (Opper, 1998) and Bayesian Gradient Descent (BGD) (Zeno et al., 2018), but on the meta-level:

$$\mu_i(t) = \mu_i(t-1) - \sigma_i^2(t-1)\mathbb{E}_\epsilon\Big[\frac{\partial \mathcal{L}_t(\phi)}{\partial \phi_i}\Big], \tag{6}$$

$$\begin{aligned} \sigma_i(t) =& \sigma_i(t-1)\sqrt{1 + \Big(\frac{1}{2}\sigma_i(t-1)\mathbb{E}_\epsilon\Big[\frac{\partial \mathcal{L}_t(\phi)}{\partial \phi_i}\epsilon_i\Big]\Big)^2} \\ &- \frac{1}{2}\sigma_i^2(t-1)\mathbb{E}_\epsilon\Big[\frac{\partial \mathcal{L}_t(\phi)}{\partial \phi_i}\epsilon_i\Big], \end{aligned} \tag{7}$$

where $\mathcal{L}_t(\phi) = -\log p(\mathcal{D}_t|\phi, \mathcal{D}_{0:t-1})$. An intuitive interpretation of these learning rules is that weights μ_i with smaller uncertainty σ_i are more important for the knowledge accumulated so far, thus they should change slower in the future in order to preserve the learned knowledge.

Algorithm 1 What & How (using Reptile)

Input: $\mu, \sigma, \lambda^{\text{in}}, k, J, \mathcal{D}_0^{\text{ctx}}, \eta_\mu, \eta_\sigma,$
for $t = 0, 1, \ldots$ **do**
  $c_t \leftarrow \theta_t = U^k(\mu; \mathcal{D}_t^{\text{ctx}}, \lambda^{\text{in}}) = \mathcal{F}^{\text{What}}(\mathcal{D}_t^{\text{ctx}}; \mu),$
  $\hat{y}_t \leftarrow \hat{f}(x_t; \theta_t) = \mathcal{F}^{\text{How}}(c_t)(x_t)$
  $\Delta_\mu \leftarrow \eta_\mu \frac{\partial \mathcal{L}_t(\phi)}{\partial \phi}\Big|_{\phi=\mu} \approx \eta_\mu(\mu - U^k(\mu, \mathcal{D}_t^{\text{ctx}}, \lambda^{\text{in}}))$
  for $j = 1$ **to** J **do**
    $\epsilon^j \sim \mathcal{N}(0, 1)$
    $\phi^j = \epsilon^j \sigma + \mu$
    $\Delta_\sigma^j \leftarrow \epsilon^j \frac{\partial \mathcal{L}_t(\phi)}{\partial \phi}\Big|_{\phi=\phi^j} \approx \epsilon^j(\phi^j - U^k(\phi^j, \mathcal{D}_t^{\text{ctx}}, \lambda^{\text{in}}))$
  end for
  $\Delta_\sigma \leftarrow \eta_\sigma \frac{1}{J} \sum_{j=1} \Delta_\sigma^j$
  $\mu_i \leftarrow \mu_i - \sigma_i^2 \Delta_{\mu_i}$
  $\sigma_i \leftarrow \sigma_i \sqrt{1 + \left(\frac{1}{2}\sigma_i \Delta_{\sigma_i}\right)^2} - \frac{1}{2}\sigma_i^2 \Delta_{\sigma_i},$
  Update $\mathcal{D}_t^{\text{ctx}}$ with $\{(x_t, y_t)\}$ to get $\mathcal{D}_{t+1}^{\text{ctx}}$
end for

In practice, we introduce learning rates for both 6 and 7. Maximum a posterior (MAP) estimate of ϕ is used for prediction. We approximate the expectation in the μ update rule 6 by the gradient at the mean, and for the expectation in the σ update rule 7, we estimate it by Monte Carlo sampling method. The final algorithm called $W\&H$ is described in Algorithm 1.

Complexity The number of parameters required by the $W\&H$ algorithm is 3 times that of the base model, since it needs to store the mean and the standard deviation of the initialization and a copy of the current task-specific parameters. In terms of time complexity, the computation of the mean update Δ_μ and the Monte Carlo (MC) sample of the standard deviation update Δ_σ^j can be parallelized. In that case, $W\&H$ has only constant computational overhead compared to Reptile due to sampling and gradient averaging. If the MC sampling process is implemented sequentially, the total time complexity is $\mathcal{O}(J + 1)$ times that of the Reptile algorithm. Similar to the findings in Zeno et al. (2018), we find that in practice the number of MC samples has negligible effect on the performance of the algorithm.

4. Related Work

Continual learning has seen a surge in popularity in the last few years, with multiple approaches being proposed to address the problem of catastrophic forgetting. These approaches can be largely categorized into the following types (Parisi et al., 2019): ***Rehearsal based methods*** focus on techniques to either efficiently store data points from previous tasks (Robins, 1995; Lopez-Paz et al., 2017) or to create pseudo datasets that are representative of past tasks. Then the stored or generated data can be used to approximate the losses of previous tasks. For example, Learning without Forgetting (LwF) (Li & Hoiem, 2017) first labels the inputs of the current task with the previous model, then use the resulting input-output pairs for rehearsal. Deep Generative Replay (DGR) (Shin et al., 2017) trains a generative model together with a classifier, and when the task switches, the previous generative model can be used to produce pseudo-examples for rehearsing the old tasks. ***Structural based methods*** exploit modularity to reduce interference, localizing the updates to a subset of weights. Rusu et al. (2016) proposed to learn a new module for each task with lateral connection to previous modules. This prevents catastrophic forgetting by construction and allows forward transfer, at the cost of quadratic growth in model size. He & Jaeger (2018) proposed to use Conceptors to identify the linear subspaces in a network that are not used by previous tasks for learning future tasks. This method does not increase the size of the network as long as the linear subspaces are not exhausted, but the network capacity will eventually saturate. In (Golkar et al., 2019), pruning techniques were applied to minimize the growth of the model after each task. Finally, ***Regularization based methods*** draw inspiration from Bayesian learning, and can be seen as utilizing the posterior after learning a sequence of tasks as *a prior* to regularize learning of the new task. These methods differ from each other in how the prior and implicitly the posterior are parametrized and approximated. For instance, Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) relies on a Gaussian approximation with a diagonal covariance, estimated using a Laplace approximation. Variational Continual Learning (VCL) (Nguyen et al., 2017) learns directly the parameters of the Gaussian relying on the re-parametrization trick. (Ritter et al., 2018) achieved better approximation with a block-diagonal covariance. Synaptic Intelligence (SI) by Zenke et al. (2017) proposed to estimate the importance of parameters by the path length of the updates on the previous task, then discourage future changes on important parameters by a quadratic penalty.

While effective at preventing forgetting, the above-mentioned methods either rely on knowledge of task boundaries or require task labels to select a sub-module for adaptation and prediction, hence cannot be directly applied in the task agnostic scenario considered here. To circumvent this issue, (Kirkpatrick et al., 2017) used Forget-Me-Not (FMN) (Milan et al., 2016) to detect task boundaries and combined it with EWC to consolidate memory when task switches. However, FMN requires a generative model that computes exact data likelihood, which limits it from scaling to complex tasks. Bayesian Gradient Descent (BGD) (Zeno et al., 2018), as we discussed before, adopts the framework of online variational Bayes, and approximates the posterior with a diagonal Gaussian distribution. More recently,

(Aljundi et al., 2019) proposed a rehearsal-based method to select a finite number of data that are representative of all data seen so far. All of these methods assume that it is possible to learn one model that fits all previous data, neglecting the scenario where different tasks may conflict each other, hence does not allow task-specific adaptations.

Meta-learning, or learning to learn (Schmidhuber, 1987), trains a model on a distribution of tasks and focuses on its ability to quickly learn a new task at meta-testing time. As with continual learning, different families of approaches exist for meta-learning. ***Memory based methods*** Santoro et al. (2016) rely on a recurrent model (optimizer) such as LSTM to learn a history-dependent update function for the lower-level learner (optimizee). Andrychowicz et al. (2016) trained an LSTM to replace the stochastic gradient descent algorithm by minimizing the sum of the losses of the optimizees on multiple prior tasks. Ravi & Larochelle (2016) use an LSTM-based meta-learner to transform the gradient and loss of the base-learners on every new example to the final updates of the model parameters. ***Metric based methods*** learn an embedding space in which new tasks can be solved efficiently. Koch (2015) trained siamese networks to tell if two images are similar by converting the distance between their feature embeddings to the probability of whether they are from the same class. Vinyals et al. (2016) proposed the matching network to improve the embeddings of a test image and the support images by taking the entire support set as context input. The approaches discussed in Section 3.1 instead belong to the family of ***optimization based meta-learning*** methods. Beside the online meta learning work (Finn et al., 2019) discussed before, the most relevant work in this domain is from (Nagabandi et al., 2019), where they studied fast adaptation in a non-stationary environment by learning an ensemble of networks, one for each task. Unlike our work, they used MAML for initialization of new networks in the ensemble instead of task inference. A drawback of this approach is that the size of the ensemble grows over time and is unbounded, hence can become memory-consuming when there are many tasks.

5. Experiments

We design a series of experiments to thoroughly evaluate the effectiveness of the What & How framework, and compare it to BGD and other CL methods (EWC, online EWC, SI, LwF, DGR, DGR+Distill) implemented by van de Ven & Tolias (2019). We also include the following baselines: **None**: the model is trained sequentially using Adam (Kingma & Ba, 2014) in the standard way without applying any continual learning method; **Joint**: all tasks seen so far are trained at the same time, this scheme usually sets the upper bound for continual learning methods.

Table 1. Zero-shot and few-shot recall ($k = 5$ steps) accuracies for different continual learning methods on the label-permuted MNIST tasks. The results are average accuracies over all tasks. The mean and the standard error of mean (SEM) are computed over 5 runs with different random seeds for each method.

METHODS	ZERO-SHOT	FEW-SHOT
NONE	28.07± 1.19	27.68± 0.98
JOINT	**35.74± 0.99**	33.44± 1.79
BGD	28.18± 1.17	27.67± 1.04
EWC	27.86± 1.19	29.85± 0.95
ONLINE EWC	27.85± 1.20	30.42± 0.69
SI	28.06± 1.19	29.50± 1.07
LWF	33.70± 1.02	32.91± 1.29
DGR	27.88± 1.12	23.06± 0.89
DGR+DISTILL	28.03± 1.14	28.88± 1.26
W&H (OURS)	28.16± 1.19	**88.88± 1.00**

5.1. Label-Permuted MNIST

In this experiment, we first create a different permutation of 10 classes for every task, with which we shuffle the classes in the labels. For instance, digit 0 might be the first class in one task but the second class in another task. The reason for this design is to ensure that a multi-task solution does not exist since the network has to map the same image to different labels for different tasks. In this way, we can test whether our framework is able to quickly adapt its behavior according to the current context. Five tasks are created with this method and are presented sequentially for 1000 iterations each to an MLP with 2 hidden layers of 1000 neurons. In each iteration, a mini-batch of 128 images is presented to the network. Note that except BGD, None and our method, all the other baselines simply cannot be directly applied in the task agnostic scenario, so we provide the necessary task information for these methods in order to perform the comparison.

Zero-shot vs. Few-shot Recall at the end of the entire learning process, we test the learner's classification accuracy of each task in two ways. The first way is to directly apply the final model on the testing data without any adaptations, this corresponds to the *zero-shot recall* accuracy traditionally used in continual learning. For the $W\&H$ algorithm, this means we apply the model simply with the learned initialization. In the second way, we provide the final model with a mini-batch of 128 images from the training set of each task as context data, then let the model take $k = 5$ steps of gradient descent on the context data before it is tested on the testing data of that task. We call this metric *few-shot recall* accuracy.

Table 1 summarizes the performance of our method and other CL methods. Note that in this experiment, it is impossible to achieve good performance with zero-shot recall

Table 2. Average test accuracy (over all 10 tasks) on the permuted MNIST experiment. The accuracies for BGD and $W\&H$ reported here are the mean ($\pm$ SEM) over 5 runs. Other results are taken directly from van de Ven & Tolias (2019)

APPROACHES	METHODS	DOMAIN-IL
BASELINES	NONE	78.51$\pm$ 0.24
	JOINT	97.59$\pm$ 0.01
REGULERIZATION	EWC	94.31$\pm$ 0.11
	ONLINE EWC	94.42$\pm$ 0.13
	SI	95.33$\pm$ 0.11
REPLAY	LWF	72.64$\pm$ 0.52
	DGR	95.09$\pm$ 0.04
	DGR+DISTILL	97.35$\pm$ 0.02
ONLINE BAYESIAN	BGD	93.02$\pm$ 0.33
	W&H (OURS)	93.37$\pm$ 0.33

Table 3. Average test accuracy (over all 5 tasks) on the split MNIST experiment. The accuracies for BGD and $W\&H$ reported here are the mean ($\pm$ SEM) over 5 runs. Other results are taken directly from van de Ven & Tolias (2019)

APPROACHES	METHODS	DOMAIN-IL
BASELINES	NONE	59.21$\pm$ 2.04
	JOINT	98.42$\pm$ 0.06
REGULERIZATION	EWC	63.95$\pm$ 1.90
	ONLINE EWC	64.32$\pm$ 1.90
	SI	65.36$\pm$ 1.57
REPLAY	LWF	71.50$\pm$ 1.63
	DGR	95.72$\pm$ 0.25
	DGR+DISTILL	96.83$\pm$ 0.20
ONLINE BAYESIAN	BGD	66.07$\pm$ 2.13
	W&H (OURS)	67.33$\pm$ 2.03

since a multi-task solution does not exist. Even the joint training scheme which is considered the upper bound for CL achieves very poor accuracies. In the few-shot setting, our framework significantly outperforms the other baselines, even without access to any task information during training. Interestingly, some baselines perform worse in the few-shot setting due to over-fitting on the context data.

5.2. Permuted and Split MNIST

We also test the What & How framework on the standard CL benchmarks called Permuted MNIST (Goodfellow et al., 2013; Kirkpatrick et al., 2017) and Split MNIST (Zenke et al., 2017). In these experiments, tasks are not conflicting with each other, so multi-task solutions do exist. van de Ven & Tolias (2019) described three scenarios for these experiments based on what task information are available at test time: **task-incremental learning**: models are always informed about which task is presented; **domain-incremental learning**: task ID is not provided at test time; **class-incremental learning**: task ID is not provided and should be inferred at test time. However, they assumed that *during training* there are clear and well-defined task boundaries available for the learner. Since our focus is task agnostic CL during training, we only consider the *domain-incremental scenario* at test time, because in the other two scenarios, the task boundaries are anyways available during training (in the class-incremental case, a class corresponds to a task, which can be simply detected from the labels), it is not necessary to apply a task-agnostic method.

In the permuted MNIST protocol, a new task is created by shuffling the pixels of all images in MNIST by a fixed permutation. We present 10 such tasks sequentially for 5000 iterations each. After all tasks are learned, the network has to predict the digit from an image without knowing the permutation. For the split MNIST protocol, the original MNIST dataset is divided into 5 subsets with 2 digits each. We present one of these subsets at a time for 2000 iterations. At the end of the training, the network has to predict if an image was the first class or the second class in its subset, without knowing which subset the image is from. To be comparable with the results from van de Ven & Tolias (2019), we use exactly the same network architecture, experiment setup and hyper-parameters for these two experiments.

The results of these two experiments are displayed in Table 2 and Table 3. Again, BGD and *our method do not have access to any task information*, while the other methods cannot be directly applied without task information. For our method, we use the previous mini-batch as context data during training, and at testing time, the learned initialization was directly used without any context-dependent adaption, it can be seen from the tables that our method performs the same as BGD, which also has similar performance to the regularization-based task-aware methods. This means our framework is also applicable when multi-task solutions exist.

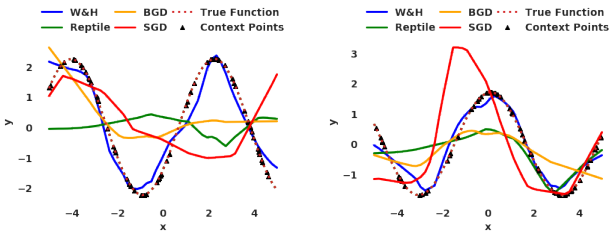

Figure 2. Few-shot adaptation on sine curves after trained on 200 such Sine curves. Left: predictions on the first presented Sine curve. Right: predictions of different learners on a Sine curve that did not occur during training.

5.3. Sine Regression

One desideratum of continual learning is the ability of *forward transfer*. Lopez-Paz et al. (2017) defined a metric for forward transfer based on a model's "zero-shot" performance on a future task. We generalize the concept of forward transfer to the "few-shot" setting: positive forward transfer should allow a model to learn faster on a future task similar to the ones it has learned. This definition coincides with generalization in meta-learning, which refers to how fast a meta-learner can learn a new task at meta-test time. We show that our method is capable of forward transfer by comparing it to Reptile, BGD and SGD on the sine regression tasks commonly used in meta learning literature. We randomly generate 200 different sine curves and present them sequentially to the learners for 100 iterations each. At the end of the learning process, we evaluate the few-shot performance (mean squared error) of all learners on these 200 sine curves as well as 200 new sine curves *it has not seen before*. In particular, the learners are evaluated after they takes $k = 5$ steps of SGD on a mini-batch of 64 context data. Table 4 summarizes the results of this experiment. The other baselines perform poorly due to catastrophic forgetting and no multi-task solution. Figure 2 visualizes the predictions of different learners on the first seen and a unseen sine curve at the end of the entire training procedure.

5.4. Continual GAN

Finally, we apply our framework in the setting of GAN training, where the ultimate goal is to find the optimal parameters θ_G^* for a generator $G(z;\theta_G)$ by optimizing a minimax objective (e.g. Goodfellow et al., 2014; Metz et al., 2016):

$$\begin{aligned}\theta_G^* &= \operatorname*{argmin}_{\theta_G} \max_{\theta_D} l(\theta_G, \theta_D) \\ &= \operatorname*{argmin}_{\theta_G} l(\theta_G, \theta_D^*(\theta_G)) \end{aligned} \tag{8}$$

where $\theta_D^*(\theta_G) = \operatorname{argmax}_{\theta_D} l(\theta_G, \theta_D)$ and $l(\theta_G, \theta_D) = \mathbb{E}_{x\sim p_{data}(x)}[\log(D(x;\theta_D))] + \mathbb{E}_{z\sim\mathcal{N}(0,1)}[\log(1 - D(G(z;\theta_G);\theta_D))]$.

Following our discussion in the Section 1, continually learning $D(\cdot;\theta_D)$ is impossible, because at different moments t and t' during training, the generator distributions may be different ($p_G \neq p_{G'}$), hence the corresponding optimal discriminator parameters cannot be the same: $\theta_D^*(\theta_G) \neq \theta_D^*(\theta_{G'})$. In other words, the tasks for the discriminator at time t and t' are conflicting with each other. Adopting the What & How framework, we focus on continually learning a model for $\theta_D^*(\cdot)$ in Eq.8 instead of the discriminator $D(\cdot)$. This model corresponds to our What encoder: given a set of data points $\mathcal{D}_G := \{G(z_n;\theta_G)\}_n$ sampled from the current generator as context data, the What encoder is trained to approximate the optimal discriminator parameters $\theta_D^*(\theta_G) \approx \mathcal{F}^{\text{What}}(\mathcal{D}_G;\phi)$ by k steps of inner loop updates on the context data, where the meta parameter ϕ corresponding to an initialization of the discriminator $\phi := \theta_D^{\text{init}}$ is learned by Alg.1. The generator-specific discriminator returned by the How decoder $\mathcal{F}^{\text{How}} \circ \mathcal{F}^{\text{What}}(\mathcal{D}_G;\phi) := D(\cdot;\mathcal{F}^{\text{What}}(\mathcal{D}_G;\phi))$ is used to compute the loss of θ_G and to update the current generator[1]. This way, the conflict at the level of θ_D is resolved at the level of ϕ: it is possible, in theory, to find a single ϕ that maximizes both $l(\theta_G, \mathcal{F}^{\text{What}}(\mathcal{D}_G;\phi))$ and $l(\theta_{G'}, \mathcal{F}^{\text{What}}(\mathcal{D}_{G'};\phi))$, even when $\theta_G \neq \theta_{G'}$.

Table 4. Average MSE of different methods on the sine curve regression tasks at the end of training. The "Seen" column contains the MSE over 200 sine curves presented during training. The "Unseen" column contains the MSE over 200 new sine curves the learners have not seen before. The results reported here are the mean (± SEM) over 5 trials.

METHODS	SEEN	UNSEEN
SGD	3.71± 0.39	3.82± 0.40
BGD	3.34± 0.24	3.27± 0.22
REPTILE	3.23 ± 0.68	3.08± 0.65
W&H (OURS)	**1.04 ± 0.09**	**1.08 ± 0.12**

Thanh-Tung & Tran (2019) and Liang et al. (2019) showed that a notorious problem for GAN training called mode collapse (Che et al., 2016) is interrelated with catastrophic forgetting and can cause the training process to never converge, since the generator is always optimized to revisit a mode that the discriminator has forgotten. Therefore, overcoming catastrophic forgetting problem in the discriminator should be able to break this mode revisiting cycles and reduce mode collapse, as shown in the experiments below.

2D Mixture of Gaussian To directly visualize the effect of our method, we first apply it to train a simple GAN from synthetic data generated from a mixture of 8 Gaussian distributions on 2D space. The network architecture and experiment setup are exactly the same as in (Metz et al., 2016). We first train a vanilla GAN with standard techniques on this dataset, the first row (Vanilla GAN) of Figure 3 shows that it enters a non-convergent cycle of revisiting the modes. We then apply the What & How method (with $k = 3$ steps in the inner loop) to the discriminator while keeping the rest of the experiment setup and hyper-parameters the same. As can be seen in the second row (WHGAN) of Figure 3, although mode collapse also occurs at the beginning of the training process, our networks can avoid the repeating cycle, and eventually converge to a distribution that covers all modes.

[1]Unlike the Unrolled GAN (Metz et al., 2016), we do not backprop through the inner loop optimization of the discriminator when we compute the gradients of the generator, even though this can provide more accurate gradients for the generator and further improve the performance of our GANs. The reason is that we want to isolate the effect of our method from that of the Unrolled GAN.

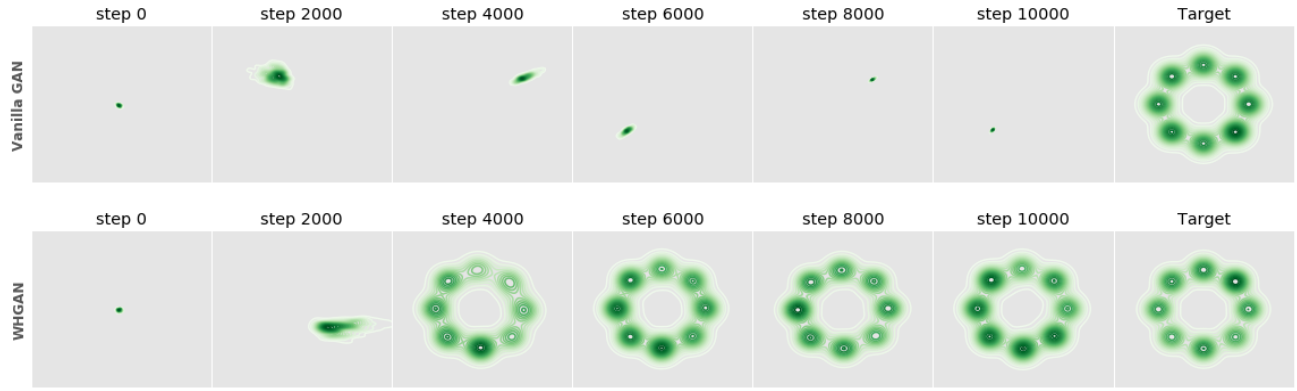

Figure 3. The What and How framework prevents mode collapse on the 2D mixture of Gaussians dataset. The first six columns show KDE plots of 512 samples from the generator at different training steps. The last column are created from 512 samples of the real distribution. The first row shows standard training of a vanilla GAN. The second row shows the same GAN with the discriminator trained by the What and How framework.

Table 5. Quantitative Evaluation of DCGAN and WHGAN on CIFAR10. The results reported here are the mean ($\pm$ SEM) over 5 random trials. $\downarrow$ (resp. $\uparrow$) indicates lower (resp. higher) is better.

Metrics	**DCGAN**	**WHGAN**(Ours)
NDB$\downarrow$	48.20$\pm$ 4.68	**26.00** $\pm$ **0.89**
JSD$\downarrow$	0.013$\pm$ 0.0016	**0.0069** $\pm$ **0.00019**
FID$\downarrow$	47.52$\pm$ 0.49	46.78 $\pm$ 0.63
IS $\uparrow$	4.40 $\pm$ 0.028	4.47 $\pm$ 0.027

DCGAN on CIFAR10 In this experiment, we compare the differences between a DCGAN (Radford et al., 2015) trained with and without our method on the CIFAR10 dataset (Krizhevsky et al., 2009). Since it is hard to visualize mode collapse for high dimensional image data, we use two bin-based metrics called NDB and JSD (Richardson & Weiss, 2018) to evaluate the resulting GANs. To compute NDB, the real samples are first clustered by K-means into $K = 200$ bins, which can be considered as modes of the data distribution. Then $N = 50000$ images sampled from the generator are assigned to their nearest bins. For each bin, a two-sample test is performed to decide if the synthesized samples are statistically different from the real samples. NDB is then simply the *number of statistically different bins* and JSD is the Jensen-Shannon divergence between the real distribution and the generator distribution over these bins. Lower NDB and JSD scores imply more similarity between two distributions, and hence less mode collapse. In addition, we also evaluate the resulting GANs with the Inception Score (IS) (Salimans et al., 2016) and the Fréchet Inception Distance (FID) (Heusel et al., 2017), which are metrics based on the image features extracted by the Inception Network (Szegedy et al., 2015). Higher IS and lower FID indicate better quality of the generated images.

Table 5 compares the performance of the original DCGAN and one trained with our framework (WHGAN). Again, we keep the network architecture and all hyper-parameters the same, except that the discriminator in WHGAN is trained with the What & How method. In both cases, we train the networks for 50000 iterations with a mini-batch size 64. For our method, $k = 3$ steps are used in the inner loop of the What encoder. The results show that WHGAN achieved significantly lower NDB and JSD while maintaining the same image quality.

6. Conclusions

In this work, we showed that when a multi-task solution does not exist, catastrophic forgetting is inevitable. A framework that can infer task information explicitly from context data was proposed to resolve this problem. The framework separates the inference process into two components: one for representing *What* task is presented, and the other for describing *How* to solve the given task. In addition, our framework unifies many meta learning methods and establishes a connection between continual learning and meta learning, leveraging the advantages of both.

From the meta learning perspective, our framework addresses the continual meta learning problem by applying CL techniques on the meta variables, therefore allowing meta knowledge to accumulate over an extended period; from the continual learning perspective, our framework addresses the task agnostic continual learning problem by explicitly inferring the task when the task information is not available and a multi-task solution does not exist. This allows us to shift the focus of continual learning from less forgetting to faster remembering, given the right context.

References

Aljundi, R., Lin, M., Goujaud, B., and Bengio, Y. Online continual learning with no task boundaries. *arXiv preprint arXiv:1903.04476*, 2019.

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., and de Freitas, N. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pp. 3981–3989, 2016.

Cassandra, A. R., Kaelbling, L. P., and Littman, M. L. Acting optimally in partially observable stochastic domains. 1994.

Che, T., Li, Y., Jacob, A. P., Bengio, Y., and Li, W. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135, 2017.

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. *arXiv preprint arXiv:1902.08438*, 2019.

Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., and Eslami, S. A. Conditional neural processes. In *International Conference on Machine Learning*, pp. 1690–1699, 2018.

Golkar, S., Kagan, M., and Cho, K. Continual learning via neural pruning. *arXiv preprint arXiv:1903.04476*, 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A., and Bengio, Y. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

He, X. and Jaeger, H. Overcoming catastrophic interference using conceptor-aided backpropagation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1al7jg0b.

Hernandez-Leal, P., Kartal, B., and Taylor, M. E. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pp. 6626–6637, 2017.

Hochreiter, S., Younger, A. S., and Conwell, P. R. Learning to learn using gradient descent. In *Proceedings of the International Conference on Artificial Neural Networks*, ICANN '01, pp. 87–94, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42486-5. URL http://dl.acm.org/citation.cfm?id=646258.684281.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Koch, G. Siamese neural networks for one-shot image recognition. 2015.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. 2009.

Kurle, R., Cseke, B., Klushyn, A., van der Smagt, P., and Günnemann, S. Continual learning with bayesian neural networks for non-stationary data. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJlsFpVtDB.

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

Liang, K. J., Li, C., Wang, G., and Carin, L. Generative adversarial network training is a continual learning problem, 2019. URL https://openreview.net/forum?id=SJzuHiA9tQ.

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

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

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

Milan, K., Veness, J., Kirkpatrick, J., Bowling, M., Koop, A., and Hassabis, D. The forget-me-not process. In

Advances in Neural Information Processing Systems, pp. 3702–3710, 2016.

Minka, T. P., Xiang, R., and Qi, Y. Virtual vector machine for bayesian online classification. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 411–418, 2009.

Monahan, G. E. State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

Nagabandi, A., Finn, C., and Levine, S. Deep online learning via meta-learning: Continual adaptation for model-based RL. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyxAfnA5tm.

Nguyen, C. V., Li, Y., Bui, T. D., and Turner, R. E. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Opper, M. A bayesian approach to on-line learning. *On-line learning in neural networks*, pp. 363–378, 1998.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. 2016.

Richardson, E. and Weiss, Y. On gans and gmms. In *Advances in Neural Information Processing Systems*, pp. 5847–5858, 2018.

Ritter, H., Botev, A., and Barber, D. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pp. 3738–3748, 2018.

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

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *Advances in neural information processing systems*, pp. 2234–2242, 2016.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.

Schmidhuber, J. Evolutionary principles in self-referential learning. *Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1987.

Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pp. 2990–2999, 2017.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Tesauro, G. Temporal difference learning and td-gammon. 1995.

Thanh-Tung, H. and Tran, T. On catastrophic forgetting and mode collapse in gans. 2019.

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

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

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

Zeno, C., Golan, I., Hoffer, E., and Soudry, D. Bayesian gradient descent: Online variational bayes learning with increased robustness to catastrophic forgetting and weight pruning. *arXiv preprint arXiv:1803.10123*, 2018.

Zintgraf, L. M., Shiarlis, K., Kurin, V., Hofmann, K., and Whiteson, S. CAVIA: Fast context adaptation via meta-learning, 2019.