GENERATION AND CONSOLIDATION OF RECOLLECTIONS FOR EFFICIENT DEEP LIFELONG LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep lifelong learning systems need to efficiently manage resources to scale to large numbers of experiences and non-stationary goals. In this paper, we explore the relationship between lossy compression and the resource constrained lifelong learning problem of function transferability. We demonstrate that lossy episodic experience storage can enable efficient function transferability between different architectures and algorithms at a fraction of the storage cost of lossless storage. This is achieved by introducing a generative knowledge distillation strategy that does not store any full training examples. As an important extension of this idea, we show that lossy recollections stabilize deep networks much better than lossless sampling in resource constrained settings of lifelong learning while avoiding catastrophic forgetting. For this setting, we propose a novel dual purpose recollection buffer used to both stabilize the recollection generator itself and an accompanying reasoning model.

1 INTRODUCTION

In this work we focus on developing a biologically motivated deep neural network architecture to accomplish the task of general purpose function transferrability. We believe this is a critical component of an ideal lifelong learning system. Optimal neural network architectures for learning have often been found to be problem specific and by enabling function transferrability we can more readily apply background knowledge using an architecture focused on the current goal. We achieve this by extending knowledge distillation techniques (Bucilu et al., 2006; Hinton et al., 2015) focused on transferring knowledge from a *teacher neural network* to a *student neural network*. Our proposed architecture based on recent discrete latent variable variational autonencoder models (Jang et al., 2017; Maddison et al., 2017) generates recollections of comparable effectiveness in training student models to real data, but with multiple orders of magnitude less memory resources consumed. Additionally, we demonstrate that we can speed up knowledge distillation by biasing our generation towards difficult examples for the current student model. We will then show how our proposed recollection generator can be used to stabilize incremental lifelong learning without catastrophic forgetting. In incremental lifelong learning settings across many tasks, we highlight that we can be considerably more effective at a given scarce incremental resource budget when using a large diversity of lossy recollections than we could when using a small diversity of lossless recollections.

1.1 MODELING THE HIPPOCAMPUS WITH DEEP NEURAL NETWORKS

A primary motivation of ours is to develop a modern artificial neural network model of the *hippocampal memory index theory* in light of recent advancements in Deep Learning. *Hippocampal memory index theory* was first proposed by Teyler & DiScenna (1986) as a theory for brain function in the hippocampus and later slightly revised in (Teyler & Rudy, 2007) after twenty years of related research. The theory primarily involves the hippocampus and neocortex, which are believed to be involved in the processes of recalling previous experiences and reasoning respectively. The crux of the theory is the idea that the hippocampus does not literally store previous experiences, but rather efficiently stores light weight *indexes* corresponding to information that can then be retrieved from a complimentary *association cortex*. The *association cortex* is considered to be part of the human neocortex, however, its location varies by species. As a result, in our work we model it as a separate component of our system.

Our artificial neural network implementation of this model consists of three primary modules. For clarity, we provide an illustration of our proposed three module architecture in Figure 1. We model a reasoning module with a standard supervised deep neural network that has an architecture suited to the current goal of the system. We also have an association module, which we model as a variational autoencoder Kingma & Welling (2014) with discrete latent variables as in (Jang et al., 2017; Maddison et al., 2017). By leveraging discrete latent variables, we can store codes with significantly smaller storage footprint at the same average distortion achieved by more traditional autoencoder models. The final component of our system is a recollection buffer that stores latent codes affiliated with prior experiences. In conjunction with the decoder of the association module, we can sample from the buffer to create approximate recollections that, for example, stabilize training for the reasoning and association modules. More generally, we will demonstrate using this capability to transfer old knowledge to a new machine learning model without fully storing any data.

1.2 Catastrophic Forgetting in Deep Neural Networks

Catastrophic forgetting refers to the elimination of previously learned information known by a neural network while learning new information. The principles of catastrophic forgetting are very influential on model performance for typical single task learning, multiple task learning, and reinforcement learning problems. It was nicely explained in (Goodrich & Arel, 2014) that for both standard single task learning with variants of stochastic gradient descent and standard parallel multi-task learning (Caruana, 1997), it is implicitly assumed that samples are drawn in an i.i.d. manner from a stationary distribution. Catastrophic forgetting emerges any time these assumptions are violated.

In standard single task learning this is a leading reason why it is advantageous to train on many passes over the data permuted in different orders. The condition is also violated in the context of lifelong learning (Thrun, 1996) of multiple tasks in sequential order. In standard fine-tuning based knowledge transfer, the representation learned on the previous task is used to initialize the model for the next task before training it on only the next task's data. In this setting neural networks tend to forget information from old tasks as they are not given an explicit incentive to remember it during training. In actuality we would ideally like the network to perform well on all tasks, so our acceptance criteria matches the sum of the loss functions for all tasks. On the other hand, we focus the model on only a part of that data for often a large number of network updates.

As we just illustrated for fine-tuning, catastrophic forgetting is intimately related to the well known stability-plasticity dilemma. Proposing solutions to it has been so vexing in part because there are legitimate tradeoffs governing the dynamics of the issue. One tradeoff is the degree to which the representation is plastic to incoming data. Another is between using small representations that tend to generalize and large representations that are less prone to forgetting because of smaller dot products between activations from different input observations. High dot products in the represenation space

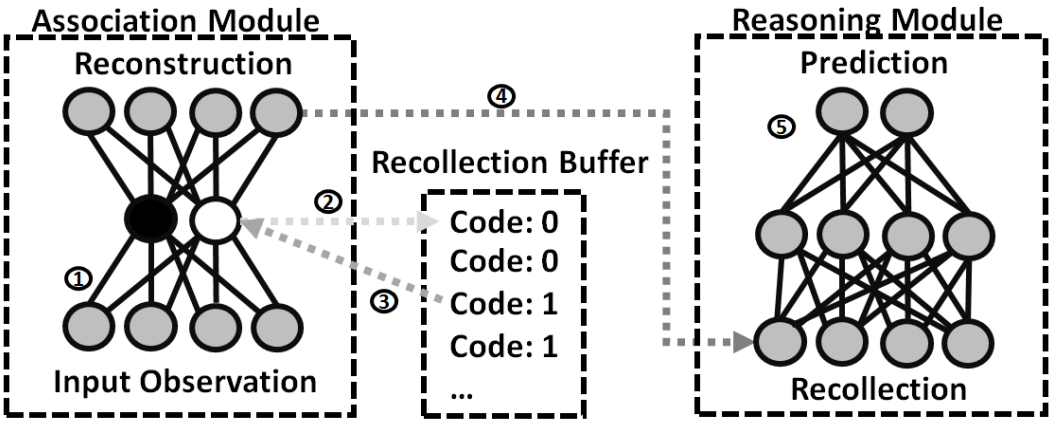

Figure 1: An example illustration of our proposed three module architecture. First, the input observation is encoded to a latent code that is stored in a recollection buffer. Later, a code is selected from the buffer and sent through the decoder to provide a recollection for which the reasoning module makes a prediction.

were theoretically shown to lead to maximal forgetting for linear networks in (Frean & Robins, 1999) and multilayer perceptrons in(Goodrich & Arel, 2014). In practice, the optimal balance depends on the current goal as well as the previous learning history of the system.

To date, the most effective general purpose strategy for reducing forgetting is rehearsals (Murre, 1992), (Robins, 1995) and its more popular reinforcement learning analogue experience replay (Lin, 1992). With these techniques, online lifelong learning is achieved by simply storing a buffer of training examples and randomly interleaving them with current examples in the same mini-batch network update. This is very effective in practice because it brings training conditions closer per incoming batch to the i.i.d. stationary ideal case. However, it has the major disadvantage of requiring the explicit storage of full training examples, which becomes an unrealistic requirement in the lifelong many task learning scenario.

1.3 Biologically Motivated Solutions to Avoid Forgetting

In our work, we focus on designing a way to scale experience rehearsal solutions by reducing the footprint of storing an experience for recollection later in a non-stationary continual learning environment. This idea actually has biologically inspired motivation relating back to the pioneering work of McClelland et al. (1995), describing the potential complementary dynamics of the hippocampus and neocortex. In this theory, the hippocampus is responsible for fast learning providing a very plastic representation for storing short term memories. Because the neocortex responsible for reasoning would otherwise suffer from catastrophic forgetting, the hippocampus also plays a key role in generating approximate recollections to interleave with incoming experiences to stabilize the learning of the neocortex.

Pseudorehearsals (Robins, 1995) is a related approach for preventing catastrophic forgetting that does not require explicit storage of patterns. Instead it relies on learning a generative model alongside the main model. The generative model produces pseudo-experiences that are combined in batches with real experiences during training. As the true labels for pseudo-experiences are obviously not available, the current main model's representation is used to create a target not to forget. For simple learning problems, very crude approximation of the real data such as randomly generated data from an appropriate distribution can be sufficient. However, for complex problems like those found in NLP and Computer Vision with highly structured high dimensional inputs, more refined approximations are needed to stimulate the network with relevant old representations. The view taken in (Li & Hoiem, 2016) for Computer Vision and (Riemer et al., 2017) for NLP is that input generation can be a very challenging problem in its own right that can be side-stepped by using the data of the current task as inputs to prevent forgetting. As demonstrate in (Aljundi et al., 2016), this strategy works best when the inputs of the old task and new task are drawn from a similar subspace. Unfortunately, using the current data creates a large bias in the distribution used to prevent forgetting that cannot be suitable for truly non-stationary problems. In our work, we address these various concerns by proposing a novel pseudo-experience generator module that leverages episodic storage to efficiently model the statistics of more complex distributions of high dimensional observations.

2 Related Work

Recent work attempting to tackle the catastrophic forgetting problem in lifelong learning typically leverages many sweeps over the training data, and saving full models specific to each prior task (Li & Hoiem, 2016), (Rusu et al., 2016), (Riemer et al., 2017), (Kirkpatrick et al., 2017). Our proposed strategy addresses a setting more compatible with continual lifelong learning as the number of tasks becomes large because all of our tasks share a common multi-task representation (Caruana, 1997) and we only perform one sequential sweep over the data. Ultimately, the notion of what constitutes a task for a machine learning algorithm is somewhat arbitrary and defined by humans developing an algorithm. For example, ImageNet is often considered a single task in the eyes of machine learning models, but a human can also easily think of the dataset as a collection of many binary present or not present classification tasks. As such, an ideal strategy for addressing catastrophic forgetting should not be highly dependent on the notion of task boundaries to avoid this human engineering step.

Our work is also related to popular generative models such as variational autoencoders (Kingma & Welling, 2014) and generative adversarial networks (Goodfellow et al., 2014). However, our

work is unique in its use of an approximate replay buffer to stabilize learning and in allowing for faster generative knowledge transfer. Recent work also looks at the problem of generative lifelong learning (Ramapuram et al., 2017) with a variational autoencoder, introducing a modified objective that would potentially be complementary to our contribution.

Even the first work on the topic of knowledge distillation (Bucilu et al., 2006) introduced a strategy for producing synthetic data to amplify real data. Additionally, unlabelled data has been widely used (Riemer et al., 2017; Laine & Aila, 2017; Ao et al., 2017; Kulkarni et al., 2017) for knowledge distillation. Generative models have also been used for distillation before in the context of language models (Shin et al., 2017), but not in the more general case where there is a separate input and output to generate for each example. Graves et al. (2017) recently looked at the problem of automated curriculum learning in a multi-task setting. In this work, we model a setting where the curriculum designer is responsible for the constructing inputs and outputs as opposed to simply choosing among a set of them. In (Matiisen et al., 2017) they learn a task generator that is appropriate for curriculum learning, but do not learn a function that constructs synthetic inputs. By achieving high quality purely generative distillation, our goal is to obtain a form of general purpose knowledge transfer. As a result, our work is related in motivation to techniques that look to preserve knowledge after transforming the network architecture (Chen et al., 2015; Wei et al., 2016).

Kaiser et al. (2017) recently also looked at the problem of memory management with respect to lifelong learning. That work was primarily concerned with mechanisms for memory consolidation and context dependent look-up for reasoning. In contrast, we simply use reservoir sampling to select memories for storage and for the most part randomly select memories for integration with reasoning. Our main contribution instead lies in the compression of memory footprint, which has been unaddressed by recent work on deep lifelong learning.

3 LIFELONG EPISODIC COMPRESSION

Recent work on lifelong continual learning in deep neural networks (Rebuffi et al., 2017), (Lopez-Paz & Ranzato, 2017) has focused on the resource constrained lifelong learning problem and how to promote stable learning with a relatively small diversity of prior experiences stored in memory. In this work, we complete the picture by also considering the relationship to the distortion of the prior experiences stored memory. We achieve this by considering the resource constraint in scaling lifelong learning not in terms of the number of full examples stored, but instead in terms of bits of storage. This is very practical because it relates back to real footprints on computer hardware. In practice, we would like to scale to an unbounded number of examples, but must ground our work in finite time horizons. With this view in mind, what is important is to demonstrate that even if the startup cost is high, the incremental maintenance cost gets lower as the number of experiences scale.

In this paper we explore experiments on the MNIST, Omniglot, and CIFAR datasets. For MNIST and Omniglot we follow prior work and consider 28x28 images with 1 channel and 8-bits per pixel. MNIST and Omniglot images were originally larger, but others have found the down sampling to 28x28 does not effect performance of models using it to learn. Multiplying out, we find that storing full images from MNIST and Omniglot will have a cost of 8x1x28x28 = 6,272 bits per image. By the same logic, storing 32x32 CIFAR images with 3 color channels and 8-bits per pixel per channel will have a cost of 8x3x32x32 = 24,576 bits. Deep non-linear autoencoders are considered a non-linear generalization of PCA and are a natural choice for compression problems. Theoretically, an autoencoder with a representation the same size as its input should be able to copy any input by simply learning an identity transformation. An autoencoder with a continuous latent variable of size h, assuming standard 32-bit representations used in modern GPU hardware, will have a storage cost of $32h$ bits for each latent variable.

One hurdle when using a standard continuous variable autoencoder for compressing input observations is that the 32 bits used for the model parameters, that also govern the representation size, most likely exceed the minimum required precision. We propose a principled approach to guarding against this issue by leveraging the recently proposed variational autoencoder model with categorical latent variables (Jang et al., 2017; Maddison et al., 2017) to enable the model to learn a representation while explicitly considering a certain degree of storage precision. For a categorical latent variable autoencoder, we consider a bottleneck representation between the encoder and decoder with c categorical latent variables each containing a l dimensions representing a *one hot* encoding of the

categorical variable. Whereas this representation requires $S_{1\mathrm{h}} = lc$ bits to store the latent variable, with simple binary encoding of each categorical latent variable, we can store this representation with the following number of bits (Cover & Thomas, 2012):

$$S_{\mathrm{sbe}} = c \cdot \lceil \log_2(l) \rceil \,. \tag{1}$$

In Figure 2 we back up our theoretical intuition and empirically demonstrate that autoencoders with categorical latent variables can achieve significantly more storage compression of input observations at the same average distortion as autoencoders with continuous variables. More detail is provided about this experiment in Appendix A.1.

The ability of a discrete variational autoencoder to memorize inputs should be strongly related to the effective bottleneck capacity C_{ve}, which we define, for discrete latent variables, as:

$$C_{\mathrm{ve}} = \log_2 l^c \,. \tag{2}$$

3.1 Incremental Storage Resource Constraints

First, let us consider the dynamics of balancing resources in a simple setting where we have an incremental storage constraint for new incoming data without regard for the size of the model used to compress and decompress recollections. We refer to the total storage constraint over all N incoming examples as γ and the average storage rate limit as γ/N. We can then define ρ as the probability that an incoming example is stored in memory. Thus, the expected number of bits required per example stored is ρS_{sbe}, assuming simple binary encoding. If we treat ρ as fixed, we can then define the following optimization procedure to search for a combination of c and l that maximizes capacity while fulfilling an incremental resource storage constraint:

$$\begin{aligned} \underset{c,l}{\text{maximize}} \quad & C_{\mathrm{ve}} \\ \text{subject to} \quad & \rho S_{\mathrm{sbe}} \leq \frac{\gamma}{N} \,, \end{aligned} \tag{3}$$

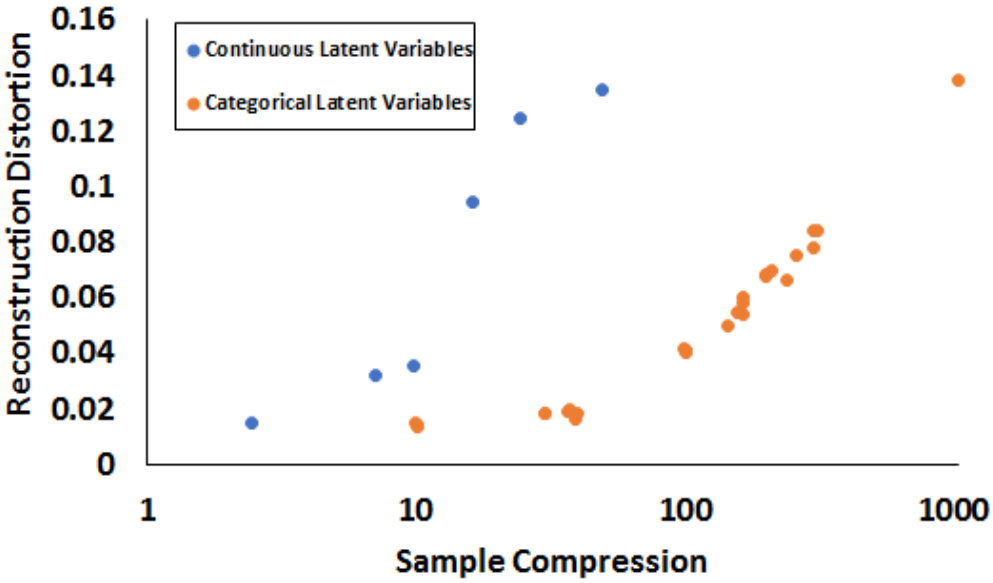

Figure 2: A comparison of the relationship between average reconstruction L1 distance on the MNIST training set and sample compression for both continuous latent variable and categorical latent variable autoencoders.

which yields the approximate solution $C_{\text{ve}} \simeq \frac{\gamma}{N\rho}$. As seen in equation 3, there is an inherent tradeoff between between the diversity of experiences we store governed by ρ and the distortion achieved that is related to the capacity. The optimal tradeoff is likely problem dependent. Our work takes a first step at trying to understand this relationship. For example, we demonstrate that in the most resource constrained settings, deep neural networks can see improved stabilization by allowing some degree of distortion in favor of an increased ability to capture the diversity in the data at the same incremental resource constraint.

3.2 Full Storage Resource Constraints

In some ways, the incremental storage constraint setting described in the previous section is not the most rigorous setting when comparing lossy compression to lossless compression where a subset of full inputs are selected. Another important factor is the number of parameters in the model $|\theta|$ used for compression and decompression. $|\theta|$ generally is also to some degree a function of c and l. For example, in most of our experiments, at the input of the decoder, we use a one hot representation of the categorical variables and use the same number of hidden variables cl at each layer as used in the bottleneck layer, which, in a fully connected layer, yields $|\theta|(c, l) \propto (cl)^2$. As such, we can revise equation 3 to handle a more rigorous constraint for optimizing a discrete latent variable autoencoder architecture:

$$\begin{aligned} \underset{c,l}{\text{maximize}} \quad & C_{\text{ve}} \\ \text{subject to} \quad & \rho S_{\text{sbe}} + |\theta|(c, l) \leq \gamma/N, \end{aligned} \tag{4}$$

While this setting is more rigorous when comparing to lossless inputs, it is a somewhat harsh restriction with which to measure lifelong learning systems. This is because it is assumed that the compression model parameters should be largely transferrable across tasks. To some degree, these parameters can be viewed as a sunk cost from the standpoint of continual learning. In our experiments, we also look at transferring these representations from related tasks to build a greater understanding of this tradeoff.

4 Stabilizing Incremental Lifelong Learning with a Dual-Purpose Recollection Buffer

In *hippocampal memory index theory* there are two central capabilities to model: pattern completion and pattern separation. In this section we will first discuss the architecture of our proposed recollection generator used for pattern completion and then explain how it is combined with a buffer used for pattern separation to stabilize training of the generator itself as well as the accompanying reasoning model.

4.1 Continuous Latent Variable Autoencoders

Deep autoencoders, sometimes referred to as autoassociators, are trained to first compress and then reconstruct approximations of the input. By providing autoassociation capabilities, autoencoders excel at pattern completion. We can think of every autoencoder has having two major components called the encoder and decoder which should have specialized architectures tuned to the problem of interest.

$$z = \text{encoder}(x) \tag{5}$$

$$\hat{x} = \text{decoder}(z) \tag{6}$$

In the standard formulation, the representation learned by the encoder z is generally modeled with continuous variables.

4.2 CATEGORICAL LATENT VARIABLE AUTOENCODERS WITH THE GUMBEL-SOFTMAX

In order to model an autoencoder with discrete latent variables we follow the success of recent work (Jang et al., 2017; Maddison et al., 2017) and leverage the Gumbel-Softmax function. The Gumbel-Softmax function leverages the Gumbel-Max trick (Gumbel, 1954; Maddison et al., 2014) which provides an efficient way to draw samples z from a categorical distribution with class probabilities p_i:

$$z = \text{one_hot}(\underset{i}{\text{argmax}}[g_i + \log(p_i)]) \tag{7}$$

In equation 7, g_i,...,g_d are samples drawn from Gumbel(0,1), which is calculated by drawing u_i from Uniform(0,1) and computing g_i=-log(-log(u_i)). The one_hot function quantizes its input into a one hot vector. The softmax function is used as a differentiable approximation to argmax , and we generate d-dimensional sample vectors y with temperature τ in which:

$$y_i = \frac{\exp((g_i + \log(p_i))/\tau)}{\sum_{j=1}^{d} \exp((g_j + \log(p_j))/\tau)} \tag{8}$$

The Gumbel-Softmax distribution is smooth for $\tau > 0$, and therefore has a well-defined gradient with respect to the parameters p. During forward propogation of our categorical autoencoder, we send the output of the encoder through the sampling procedure of equation 7 to create a categorical variable. However, during backpropogation we replace non-differentiable categorical samples with a differentiable approximation during training as the Gumbel-Softmax estimator in equation 8. Although past work (Jang et al., 2017; Maddison et al., 2017) has found value in varying τ over training, we still were able to get strong results keeping τ fixed at 1.0 across our experiments.

4.3 RECOLLECTION BUFFER DESIGN

Now that we have discussed our method for achieving pattern completion with a variational autoencoder, we will explain how we also preserve pattern separation by storing discrete latent codes to be used for recalling specific past experiences. An illustration of our proposed approach for creating incremental lifelong learning training batches is depicted in Figure 3. A recollection buffer is maintained that stores a diversity of latent codes corresponding to the autoencoder's compressed representation when the input was first experienced. When a new real input is received, we select codes from the recollection buffer (i.e. by randomly sampling) and pass them through the decoder of the autoencoder using a frozen set of decoder parameters. Unfortunately, like the reasoning model, the autoencoder should naturally suffer from catastrophic forgetting. As such, our recollection buffer is also used to stabilize the autoencoder itself. In our experiments, we will demonstrate that, in resource constrained settings, using lossy self generated samples can become a more effective stabilizer for learning than storing a smaller diversity of lossless training examples for rehearsal. The period of freezing/refreshing parameters in the decoder is in essence a hyperparameter of our model. Past work on catastrophic forgetting has considered freezing parameters after tasks. In this work, we utilize a more general strategy where we freeze decoder parameters right before each incoming experience and train multiple gradient descent iterations over randomly selected recollection batches before moving on to the next experience.

4.4 INTEGRATION WITH THE REASONING MODEL

With some simple extensions, our recollection buffer described above can also work to stabilize a reasoning model. As opposed to the autoassociation model, the reasoning model is not only concerned with input observation of an experience, but also with reasoning about the optimal action to take in response. In the supervised learning setting explored in this work, we can keep a memory of the class index and task index associated with each input as they are already in a very compact representation. Another alternative is to not store anything related to the target in the buffer and use a frozen version of the reasoning model as an approximate target.

5 EVALUATION SETTINGS

In this section we will empirically support the methods proposed in the previous sections by applying them to the popular MNIST digit recognition dataset (Lecun et al., 1998) as well as multi-task splits of the CIFAR-100 image recognition dataset (Krizhevsky, 2009) considering each of the 20 course grained label to be a task, and the Omniglot character recognition dataset (Lake et al., 2011) considering each of the 50 alphabets to be a task.

Across all of our experiments, our generator model is a discrete latent variable convolutional variational autoencoder including three convolutional layers in the encoder and three deconvolutional layers in the decoder. In all of our knowledge distillation experiments, we report an average result over 5 runs. More details can be found for all of our experiments in Appendix A.

5.1 GENERAL PURPOSE KNOWLEDGE TRANSFER EXPERIMENTS

We will now empirically demonstrate our proposed recollection generator's ability to transfer knowledge from a teacher neural network to a student model. In our experiments, we train a teacher model with a LeNet (LeCun et al., 1998) convolutional neural network (CNN) architecture on the popular MNIST benchmark, achieving 99.29% accuracy on the test set. Alongside the teacher model, we train a generator model with discrete latent variables. Each model is trained for 500 epochs. During

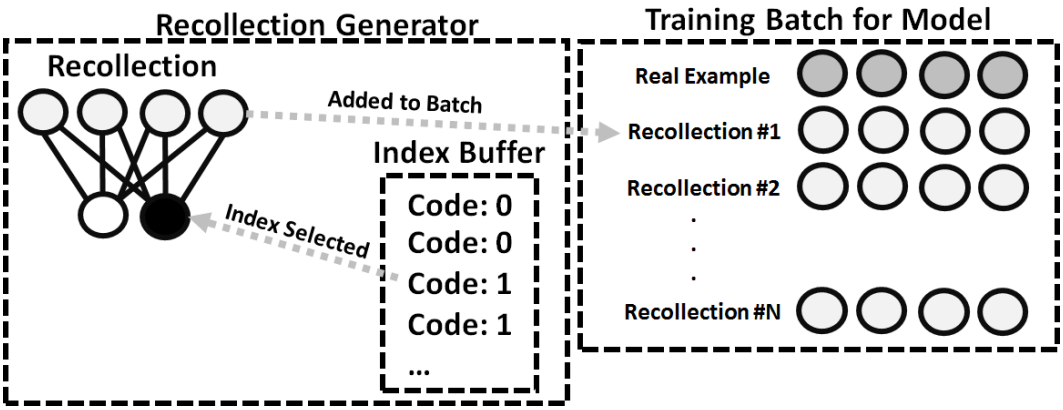

Figure 3: An illustration of how our proposed recollection generator is used to produce recollections that are interleaved with real incoming examples for training incremental lifelong learning models.

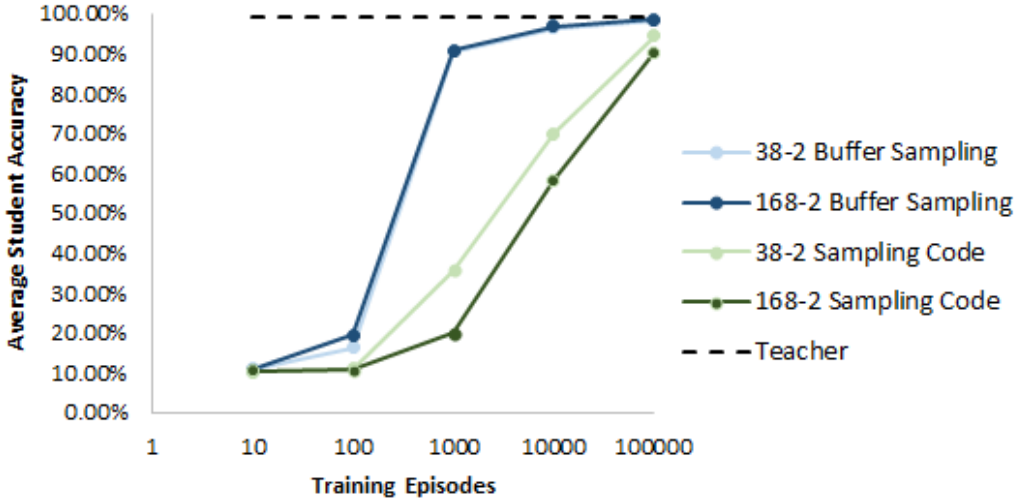

Figure 4: Comparison of generative transfer learning performance using a CNN teacher and student model on MNIST while using code sampling and recollection buffer sampling.

the final pass through the data, we forward propogate through each training example and store the latent code in a recollection buffer, which eventually grows to a size of 50,000. After training is complete, the recollection buffer is used as a statistical basis for sampling diverse recollections to train a student network. A logical and effective strategy for training a student model is to sample randomly from this buffer and thus capture the full distribution.

First, we will empirically make the case for our proposal of episodic storage to augment a variational autoencoder model for knowledge transfer. In Figure 4 we look at two discrete latent variable models with 168 2d variables and 38 2d variables, comparing performance for sampling from an episodic buffer with standard random sampling of the variables with no explicit episodic memory. We see that much more efficient knowledge transfer can be achieved by sampling from episodic storage. It is important to note that this advantage is not simply a result of increased storage used by the recollection buffer. The smaller model with episodic storage achieves better transfer performance than the larger model does when sampling the latent variable space. This is despite the fact that the larger model's parameter storage is 7.4x more than the total storage of parameters and buffer combined for the smaller model. Actually, if anything we are seeing the opposite behavior. Interestingly, we are observing decreased transfer ability with the larger model. We hypothesize that this because although the reconstruction modeling power of a discrete autoencoder should increase with capacity, there is a tension between increasing the capacity and then increasing the dimensionality of the latent space to search through. Very large latent spaces should in turn have large regions which they can model that are not statistically representative of the data distribution seen. By providing the variational autoencoder with an episodic buffer we are able to get the best of both worlds: the ability to increase the size of the representation used for modeling and the ability to efficiently transfer from it in a way that is representative of the prior distribution of inputs.

In the top half of Table 1 we further validate the effectiveness of the random sampling strategy for our discrete autoencoder based compression technique by comparing it to some additional baselines of interest. As baselines we consider training with the the same number of randomly sampled real examples, using real input and the teacher's knowledge distillation output, and using random sampling to select a subset of lossless memories. When training with a large number of memories for a more complete knowledge transfer, the recollection compression clearly shows dividens over random sampling baselines. This is impressive particularly because these results are for the stricter resource constraint setting discussed in section 3.2 and on a per sample basis the compression is actually 37x (168 2d variables), 101x (62 2d variables), and 165x (38 2d variables) to account for the autoencoder model capacity. We also would like to validate these findings in a more complex setting for which we consider distillation with outputs from a 50 task Resnet-18 teacher model that gets 94.86% accuracy. We test out performance after 100,000 training episodes and find that with real examples we achieve 93.04%, but sampling diversity restricts learning significantly, for example, achieving 27.16% accuracy with 10% sampling, 8.88% with 2% sampling, and 5.99% with 1% sampling. In contrast lossy compression is much more effective, for example, achieving 42.98% accuracy for 50x total resource compression, and 27.80% for 100x compression.

We would like to maximize the efficiency of distilling knowledge from a teacher model to a student model. This motivates the automated curriculum learning setting as recently explored for multi-task learning in (Graves et al., 2017) or rather automated generative curriculum learning in our case. We tried some simple reinforcement learning solutions with rewards based on (Graves et al., 2017) but were unsuccessful in our initial experiments because of the difficulty of navigating a complex continuous action space. We also tried an active learning formulation proposed for GANs to learn the best latent code to sample (Zhu & Bento, 2017). Unfortunately, we had limited success with this strategy as well as it tends to learn to emphasize regions of the latent space that optimize incorrectness, but no longer capture the distribution of inputs. Inspired by this finding, we instead employ a biased sampling strategy we refer to as *active sampling* where we randomly sample k latent codes and choose the one that is most difficult for the current student for backpropogation by cheaply forward propogating through the student for each. We set k to 10 in our experiments so the sampling roughly equates to sampling once from the most difficult task for the student model at each point in time. In the bottom half of Table 1, we demonstrate the superior performance of the proposed active sampling strategy relative to random sampling when the number of training examples is at least 1000.

Our main motivation for enabling general purpose knowledge distillation is for occasions where we would like to change the architectural form of our knowledge over time, not keep it constant.

Episodes	Real Data	10% Sample	2% Sample	1% Sample	Real Input Soft Target	10x Compress	50x Compress	100x Compress
10	10.43	9.94	11.07	10.70	10.07	10.65	10.99	**13.89**
100	19.63	18.16	22.82	22.35	**25.32**	19.34	16.20	21.06
1000	90.45	88.88	90.71	89.93	**91.01**	90.66	90.52	90.03
10000	97.11	96.83	95.98	94.97	**97.42**	96.77	96.37	95.65
100000	98.51	97.99	96.14	94.92	**98.63**	98.59	98.17	97.75
10	10.31	11.64	**12.73**	11.65	11.81	9.95	11.04	10.19
100	24.62	15.34	**22.74**	18.41	18.55	14.80	19.13	22.57
1000	91.98	91.60	**94.20**	93.45	92.31	93.45	93.40	92.97
10000	98.55	97.99	96.56	94.47	98.56	**98.61**	98.20	97.53
100000	99.15	98.21	96.13	94.53	**99.19**	99.18	98.76	98.25

Table 1: Generative knowledge distillation experiments with a CNN student on MNIST. The top half of the table represents results with random sampling and the bottom half of the table represents results with active sampling.

Episodes	Real Data	10% Sample	2% Sample	1% Sample	Real Input Soft Target	10x Compress	50x Compress	100x Compress
10	13.64	**17.04**	14.57	15.13	15.87	16.70	11.80	14.66
100	36.37	37.04	38.35	34.04	38.56	37.16	40.09	**42.31**
1000	80.54	79.08	78.18	77.76	80.00	**80.72**	80.00	77.75
10000	91.04	90.84	88.38	86.83	90.86	**91.37**	90.60	90.46
100000	96.66	95.02	91.61	88.97	96.60	**96.71**	96.24	95.22
10	21.82	21.22	21.28	**21.85**	15.82	15.76	18.34	17.47
100	50.22	54.67	**54.97**	53.61	46.76	43.42	44.29	45.74
1000	84.21	84.46	85.37	**87.68**	86.14	83.35	87.66	85.71
10000	95.91	94.66	91.77	89.19	**96.48**	95.96	95.85	94.51
100000	97.98	95.02	91.61	88.97	**98.12**	98.07	97.67	96.91

Table 2: Generative knowledge distillation experiments with a MLP student on MNIST. The top half of the table represents results with random sampling and the bottom half of the table represents results with active sampling.

In Table 2 we consider distillation from our LeNet CNN teacher model to a multi-layer perceptron (MLP) student with two hidden layers of 300 hidden units. For MLPs we again see the same story in that active sampling improves distillation performance, our recollection compression is comparable to the performance of real examples while using much less storage, and lossy compression scales much better than sampling lossless inputs.

5.2 Lifelong Learning without Forgetting Experiments

To assess the ability of our recollection generator to stabilize incremental lifelong learning without forgetting, we follow recent work (Lopez-Paz & Ranzato, 2017; Rebuffi et al., 2017) using an incremental learning split of CIFAR-100. It consists of 20 tasks made up of the course grained super class labels in CIFAR-100, which each contain 5 sub-classes. The tasks are learned in the predefined sequence incrementally one example at a time. In recent efforts on this dataset, the smallest number of examples stored in memory considered was 200, allowing to memorize on average two lossless example from each class in the dataset. We will explore resource constrained settings like this and ideally even smaller incremental resource constrained setting to test out the efficacy of our recollection generator. We model our experiments after (Lopez-Paz & Ranzato, 2017) and use a Resnet-18 base model. All of our models store examples in their buffer following the reservoir sampling procedure.

In Table 3 we consider performance on incremental CIFAR-100 with a very small incremental resource constraint, including a couple of settings with even less incremental memory than the number of classes. A primary baseline of ours for comparison is rehearsals which follows the same structure as our model, but with lossless recollections as opposed to lossy recollections. It performs relatively well when the number of examples is greater than the number of classes, but otherwise suffers from a biased sampling towards a subset of classes. This can be seen by the decreased performance for buffer sizes of 10 and 50 from what is achieved with no buffer at all learning online. Consistently we see that tuning our recollection generator to lossy recollections with reasonably sized diversity results in improvements over real examples at the same incremental resource cost. To further validate our method against another dataset, we turned to Omniglot and learning in an incremental 50 task setting. Omniglot is more challenging for resource constrained incremental learning than

Model	Effective Buffer Size	Recollections	Retention
Online Resnet-18	0	0	33.3
Resnet-18 Rehearsals	10	10	29.4
	50	50	33.4
	200	200	43.0
Recollection Generator	10	5000	**38.9**
	50	5000	**47.9**
	200	5000	**51.6**

Table 3: Lifelong learning retention results on incremental CIFAR-100 for low effective buffer sizes with an incremental storage resource constraint.

Model	Recollections	Retention
Resnet-18 Rehearsals	200	43.0
GEM (Lopez-Paz & Ranzato, 2017)	200	48.7
iCaRL (Rebuffi et al., 2017)	200	43.6
76 2d Categorical Variables - No Transfer	1392	43.7
76 2d Categorical Variables - CIFAR-10 Transfer	1392	**49.7**

Table 4: Effective buffer size of 200 for full storage on iCIFAR-100 with one step gradient step per example.

CIFAR-100 because it contains more tasks and fewer examples of each class. With an incremental resource constraint of 10 full examples, a Resnet-18 model with rehearsals achieves 3.6% final retention accuracy (online learning produces 3.5% accuracy). In contrast our recollection generator achieves 5.0% accuracy. For an incremental resource constraint of 50 full examples, rehearsals achieves 4.3% accuracy which is further improved to 4.8% accuracy by taking three gradient steps per new example. The recollection generator once again achieves better performance with 9.3% accuracy at one step per example and 13.0% accuracy at three steps per example.

Given our results so far, it may be fair to criticize our introduction of autoencoder parameters and question whether these incremental savings will truly be realized in practice. On the other hand, to demonstrate performance with a very small total resource constraint on a single dataset, an incredibly small autoencoder would then be needed to learn from scratch a function for the very complex input space. We demonstrate in Table 4 that transfer learning provides a solution to this problem. By transferring in background knowledge we are able to perform much better at the onset with a small autoencoder. We explore the total resource constraint of 200 examples that is the smallest explored in (Lopez-Paz & Ranzato, 2017) and demonstrate that we are able to achieve state of the art results by initializing only the autoencoder representation with one learned on CIFAR-10. CIFAR-10 is drawn from the same larger database as CIFAR-100, but is non-overlapping. We should emphasize that the technique proposed in our paper is some sense orthogonal to and possibly complimentary to approaches to utilizing these episodic examples explored in (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017). However, we provide them for comparison to highlight that the gains seen by leveraging efficient lossy recollections for this problem outweigh those related to making more sophisticated use of the recollections.

6 Conclusion

In this work we have proposed a discrete latent variable autoencoder model for generative knowledge transfer and stabilizing lifelong learning without forgetting. We demonstrated that autoencoders with discrete latent variables are capable of far more sample compression than continuous latent variable models. We also highlighted that variational autoencoders equipped with lossy experience storage are capable of multiple orders of magnitude faster knowledge distillation in comparison to traditional code sampling. We consistently find lossy compression to be more effective than lossless episodic sampling. However, they are also complimentary in a resource constrained lifelong learning setting, and we get best results in the human realistic setting of having both lossy and sampled memories. For lifelong learning it is important to efficiently scale resources for a very large number of example. Our initial very promising results using transfer learning demonstrate that the incremental memory footprint for new memories can decrease for a lifelong learning agent over time. This is a key ingredient to scaling deep lifelong learning systems and is a promising direction for future studies to expand upon.

REFERENCES

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. *arXiv preprint arXiv:1611.06194*, 2016.

Shuang Ao, Xiang Li, and Charles X Ling. Fast generalized distillation for semi-supervised domain adaptation. 2017.

Cristian Bucilu, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541. ACM, 2006.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997. doi: 10.1023/A:1007379606734. URL http://dx.doi.org/10.1023/A:1007379606734.

Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*, 2015.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Marcus Frean and Anthony Robins. Catastrophic forgetting in simple networks: an analysis of the pseudorehearsal solution. *Network: Computation in Neural Systems*, 10(3):227–236, 1999.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ben Goodrich and Itamar Arel. Neuron clustering for mitigating catastrophic forgetting in feedforward neural networks. In *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on*, pp. 62–68. IEEE, 2014.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. *arXiv preprint arXiv:1704.03003*, 2017.

Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*. Number 33. US Govt. Print. Office, 1954.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *ICLR*, 2017.

Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. *arXiv preprint arXiv:1703.03129*, 2017.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, pp. 201611835, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Mandar Kulkarni, Kalpesh Patil, and Shirish Karande. Knowledge distillation using unlabeled mismatched images. *arXiv preprint arXiv:1703.07131*, 2017.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *ICLR*, 2017.

Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the Cognitive Science Society*, volume 33, 2011.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

Zhizhong Li and Derek Hoiem. Learning without forgetting. In *European Conference on Computer Vision*, pp. 614–629. Springer, 2016.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continuum learning. *NIPS*, 2017.

Chris J Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *Advances in Neural Information Processing Systems*, pp. 3086–3094, 2014.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *ICLR*, 2017.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher-student curriculum learning. *arXiv preprint arXiv:1707.00183*, 2017.

James L McClelland, Bruce L McNaughton, and Randall C O'reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Jacob MJ Murre. Learning and categorization in modular neural networks. 1992.

Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *arXiv preprint arXiv:1705.09847*, 2017.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, and Christoph H Lampert. icarl: Incremental classifier and representation learning. *CVPR*, 2017.

Matthew Riemer, Elham Khabiri, and Richard Goodwin. Representation stability as a regularizer for improved text analytics transfer learning. *arXiv preprint arXiv:1704.03617*, 2017.

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

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Sungho Shin, Kyuyeon Hwang, and Wonyong Sung. Generative knowledge transfer for neural language models. 2017.

Timothy J Teyler and Pascal DiScenna. The hippocampal memory indexing theory. *Behavioral neuroscience*, 100(2):147, 1986.

Timothy J Teyler and Jerry W Rudy. The hippocampal indexing theory and episodic memory: updating the index. *Hippocampus*, 17(12):1158–1169, 2007.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *Advances in neural information processing systems*, pp. 640–646, 1996.

Tao Wei, Changhu Wang, Yong Rui, and Chang Wen Chen. Network morphism. In *International Conference on Machine Learning*, pp. 564–572, 2016.

Jia-Jie Zhu and Jose Bento. Generative adversarial active learning. *arXiv preprint arXiv:1702.07956*, 2017.

c	l	Compression	Distortion
6	20	241.8673588	0.0660938
10	20	145.1204153	0.049652
6	16	261.3333333	0.0754573
12	10	157.3383444	0.0549662
10	14	164.7337884	0.0540967
24	3	164.8829756	0.0598787
38	2	165.0526316	0.0578548
6	2	1045.333333	0.138312
40	3	98.92978536	0.0415754
20	2	313.6	0.0844624
8	6	303.2926009	0.0842347
12	6	202.1950672	0.0675599
30	2	209.0666667	0.0695824
24	6	101.0975336	0.0406478
4	37	300.991273	0.077945
8	15	200.6710915	0.0681162
48	10	39.3345861	0.0164898
209	8	10.00318979	0.0145487
12	37	100.3304243	0.0399604
313	4	10.01916933	0.0141956
392	3	10.09487606	0.0134821
50	18	30.08207581	0.0185929
168	2	37.33333333	0.0195544
108	3	36.64066124	0.0189382
62	2	101.1612903	0.0407307
208	2	30.15384615	0.0183247
68	5	39.723579	0.0184864

Table 5: This table provide more specifics about the discrete latent variable architectures involved in Figure 2.

h	Compression	Distortion
1	49	0.135196
2	24.5	0.124725
3	16.33333333	0.0947032
5	9.8	0.0354035
7	7	0.031808
20	2.45	0.0149272

Table 6: This table provide more specifics about the continuous latent variable architectures involved in Figure 2.

A Additional Details on Experimental Protocol

Each convolutional layer has a kernel size of 5. As we vary the size of our categorical latent variable across experiments, we in turn model the number of filters in each convolutional layer to keep the number of hidden variables consistent at all intermediate layers of the network. In practice, this implies that the number of filters in each layer is equal to $cl/4$. We note that the discrete autoencoder is stochastic, not deterministic and we just report one stochastic pass through the data for each experimental trial.

A.1 Distortion as a Function of Compression Experiments

More detail about the architecture used in these experiments are provided for categorical latent variables in Table 5 and for continuous latent variables in Table 6. For each architecture we ran with a learning rate of 1e-2, 1e-3, 1e-4, and 1e-5, reporting the option that achieves the best training distortion. For the distortion, the pixels are normalized by dividing by 255.0 and we take the mean over the vector of the absolute value of the reconstruction to real sample difference and then report the mean over the samples in the training set.

A.2 MNIST GENERATIVE DISTILLATION EXPERIMENTS

For all of our distillation experiments we ran the setting with a learning rate of 1e-3 and 1e-4, reporting the best result. We found that the higher learning rate was beneficial in setting with a low number of examples and the lower learning rate was beneficial in setting with a larger number of examples. The categorical latent variable autoencoders explored had the following representation sizes: 168 2d variables for 10x compression, 62 2d variables for 50x compression, and 38 2d variables for 100x compression. For our code sampling baselines, we used the numpy random integer function to generate each discrete latent variable.

A.3 OMNIGLOT GENERATIVE DISTILLATION EXPERIMENT

The learning rate for the Resnet-18 reasoning model was 1e-4 in our experiments. Our trained discrete autoencoder models were of the following representation sizes: 32 variables of size 2 for 100x compression, 50 variables of size 2 for 50x compression, and 134 variables of size 2 for 10x compression.

A.4 INCREMENTAL RESOURCE CONSTRAINT EXPERIMENTS

Our categorical latent variable autoencoders had the following sizes for incremental CIFAR-100: 48 2d variables for an effective buffer size of 10, 244 2d variables for an effective buffer size of 50, and 620 3d variables for an effective buffer size of 200. The reasoning model was trained with a learning rate of 1e-3 in all of our experiments. The learning rate for the autoencoder was 1e-4 for the buffer size of 200 and 1e-3 for other buffer sizes.

For incremental Omniglot our learning rate was set to 1e-3. For the effective buffer size of 50 experiments, we leveraged a categorical latent variable autoencoder with 312 2d variables. For the effective buffer size of 10 experiments, we utilized a categorical latent variables consisting of 62 2d variables.

A.5 INCREMENTAL CIFAR-100 TOTAL RESOURCE CONSTRAINT EXPERIMENTS

During the transfer learning experiments from CIFAR-10, a learning rate of 1e-3 was used for the Resnet-18 reasoning model and a learning rate of 3e-4 was used for the discrete autoencoder generator. For the experiment without transfer learning, we instead used a higher learning rate of 1e-3 for the autoencoder.