KANERVA++: EXTENDING THE KANERVA MACHINE WITH DIFFERENTIABLE, LOCALLY BLOCK ALLOCATED LATENT MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Episodic and semantic memory are critical components of the human memory model. The theory of complementary learning system (McClelland et al., 1995) suggests that the compressed representation produced by a serial event (episodic memory) is later re-structured to build a more generalized form of re-usable knowledge (semantic memory). In this work we develop a new principled hierarchical Bayesian memory allocation scheme that bridges the gap between episodic and semantic memory via a heirarchal latent variable model. We take inspiration from traditional heap allocation and extend the idea of locally contiguous memory to the Kanerva Machine, enabling a novel differentiable block allocated latent memory. In contrast to the Kanerva Machine, we simplify the process of memory writing by treating it as a fully feed forward deterministic process, relying on the stochasticity of the read key distribution to distribute information within the memory. We demonstrate that this allocation scheme improves performance in conditional image generation, resulting in new state-of-the-art likelihood values on binarized MNIST (**45.64 nats/image**), binarized Omniglot (**71.64 nats/image**) as well as presenting competitive performance on CIFAR10, DMLab Mazes, Celeb-A and ImageNet32$\times$32.

1 INTRODUCTION

Memory is a central tenant in the model of human intelligence and is crucial to long term reasoning and planning. Recent work has shown that memory augmented neural networks drastically improve the performance of generative models (Wu et al., 2018a;b), language models (Weston et al., 2015), meta-learning (Santoro et al., 2016), long term planning (Graves et al., 2014; 2016) and sample efficiency in reinforcement learning (Zhu et al., 2019).

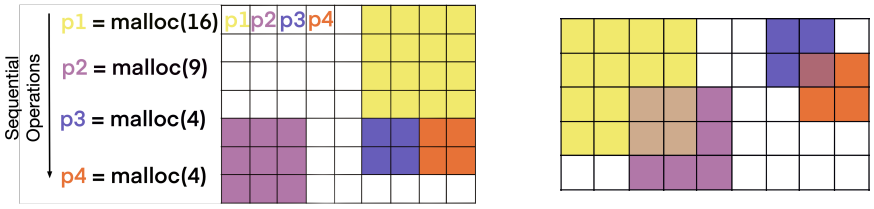


Figure 1: Example final state of a traditional heap allocator (Marlow et al., 2008) (*Left*) vs. K++ (*Right*); final state created by sequential operations listed on the left. K++ uses a key distribution to stochastically point to a memory sub-region while Marlow et al. (2008) uses a direct pointer.

Inspired by the traditional (computer-science) memory model of heap allocation (Figure 1-*Left*), we propose a novel differentiable memory allocation scheme called Kanerva ++ (K++), that learns to compress an episode of samples, referred to by the set of pointers $\{p1, ..., p4\}$ in Figure 1, into a latent multi-dimensional memory (Figure 1-*Right*). The K++ model infers a key distribution as a proxy to the pointers used in Marlow et al. (2008) and is able to embed similar samples to an overlapping latent representation space, thus enabling it to be more efficient on compressing input distributions. In this work we focus on applying this novel memory allocation scheme to latent variable generative models, where we improve the memory model in the Kanerva Machine (Wu et al., 2018a;b).

2 RELATED WORK

Variational Autoencoder Variational autoencoders (VAE) (Kingma & Welling, 2014) are a fundamental part of the modern machine learning toolbox and have wide ranging applications from generative modeling (Kingma & Welling, 2014; Kingma et al., 2016; Burda et al., 2016), learning graphs (Kipf & Welling, 2016), medical applications (Sedai et al., 2017; Zhao et al., 2019) and video analysis (Fan et al., 2020). As a latent variable models, VAEs infer an approximate posterior over a latent representation Z and can be used in downstream tasks such as control in reinforcement learning (Nair et al., 2018; Pritzel et al., 2017). VAEs maximize an evidence lower bound (ELBO), $\mathcal{L}(X, Z)$, of the log-marginal likelihood, $\ln p(X) > \mathcal{L}(X, Z) = \ln p_\theta(X|Z) - \mathcal{D}_{KL}(q_\phi(Z|X)||p_\theta(Z))$. The produced variational approximation, $q_\phi(Z|X)$, is typically called the encoder, while $p_\theta(X|Z)$ comes from the decoder. Methods that aim to improve these latent variable generative models typically fall into two different paradigms: learning more informative priors or leveraging novel decoders. While improved decoder models such as PixelVAE (Gulrajani et al., 2017) and PixelVAE++ (Sadeghi et al., 2019) drastically improve the performance of $p_\theta(X|Z)$, they suffer from a phenomenon called posterior collapse (Lucas et al., 2019), where the decoder can become almost independent of the posterior sample, but still retains the ability to reconstruct the original sample by relying on its auto-regressive property (Goyal et al., 2017a).

In contrast, VampPrior (Tomczak & Welling, 2018), Associative Compression Networks (ACN) (Graves et al., 2018), VAE-nCRP (Goyal et al., 2017b) and VLAE (Chen et al., 2017) tighten the variational bound by learning more informed priors. VLAE for example, uses a powerful auto-regressive prior; VAE-nCRP learns a non-parametric Chinese restaurant process prior and VampPrior learns a Gaussian mixture prior representing prototypical virtual samples. ACN on the other hand takes a two stage approach: by clustering real samples in the space of the posterior and using these related samples as inputs to a learned prior, ACN provides an information theoretic alternative to improved code transmission. Our work falls into this latter paradigm: we parameterize a learned prior by reading from a common memory, built through a transformation of an episode of input samples.

Memory Models Inspired by the associative nature of biological memory, the Hopfield network (Hopfield, 1982) introduced the notion of content-addressable memory, defined by a set of binary neurons coupled with a Hamiltonian and a dynamical update rule. Iterating the update rule minimizes the Hamiltonian, resulting in patterns being stored at different configurations (Hopfield, 1982; Krotov & Hopfield, 2016). Writing in a Hopfield network thus corresponds to finding weight configurations such that stored patterns become attractors via Hebbian rules (Hebb, 1949). This concept of memory was later extended to a distributed, continuous setting in Kanerva (1988) and to a complex valued, convolutional binding mechanism by Plate (1995). The central difference between an associative memory model (Hopfield, 1982; Kanerva, 1988) and (Plate, 1995) is that the latter decouples the size of the memory from the input word size.

Most recent work with memory augmented neural networks treat memory in a slot-based manner (closer to the associative memory paradigm), where each column of a memory matrix, M, represents a single slot. Reading memory traces, z, entails using a vector of addressing weights, r, to attend to the appropriate column of M, $z = r^T M$. This paradigm of memory includes models such as the Neural Turing Machine (NTM) (Graves et al., 2014), Differentiable Neural Computer (DNC) (Graves et al., 2016) [1], Memory Networks (Weston et al., 2015) and Variational Memory Addressing (VMA) (Bornschein et al., 2017). VMA differs from DNC, NTM and Memory Networks by taking a stochastic approach to discrete key-addressing instead of the deterministic approach of the latter models.

Recently, the Kanerva Machine (KM) (Wu et al., 2018a) and its extension, the Dynamic Kanerva Machine (DKM) (Wu et al., 2018b), interpreted memory writes and reads as inference in a generative model, wherein memory is now treated as a distribution, $p(M)$. Under this framework, memory reads and writes are recast as sampling or updating the memory posterior. The DKM model differs from the KM model by introducing a dynamical addressing rule that could be used throughout training. While providing an intuitive and theoretically sound bound on the data likelihood, the DKM model requires an inner optimization loop which entails solving an ordinary least squares (OLS) problem. Typical OLS solutions require a matrix inversion ($O(n^3)$), preventing the model

[1] While DNC is slot based, it should be noted that DNC reads rows rather than columns.

from scaling to large memory sizes. More recent work has focused on employing a product of smaller Kanerva memories (Marblestone et al., 2020) in an effort to minimize the computational cost of the matrix inversion. In contrast, we propose a simplified view of memory creation by treating memory writes as a deterministic process in a fully feed-forward setting. Crucially, we also modify the read operand such that it uses localized sub-regions of the memory, providing an extra dimension of operation in comparison with the KM and DKM models.

3 MODEL

To better understand the K++ model, we examine each of the individual components to understand their role within the complete generative model. We begin by first deriving a variational lower bound (Section 3.1), describing the optimization objective and probabilistic assumptions. We then describe the write operand (Section 3.3), the generative process (Section 3.4) and finally the read operand (Section 3.5).

3.1 PRELIMINARIES

K++ operates over an exchangeable episode (Aldous, 1985) of samples, $X = \{x_t\}_{t=1}^{T} \in \mathcal{D}$, drawn from a dataset $\mathcal{D}$, as in the Kanerva Machine. Therefore, the ordering of the samples within the episode does not matter. This enables factoring the conditional, $p(X|M)$, over each of the individual samples: $\prod_{t=1}^{T} p(x_t|M)$, given the memory, $M \sim p(M), M \in \mathbb{R}^{\hat{C}\times\hat{W}\times\hat{H}}$. Our objective in this work is to maximize the expected conditional log-likelihood as in (Bornschein et al., 2017; Wu et al., 2018a):

$$\mathcal{J} = \mathbb{E}_{p(X),\, p(M|X)} \ln p_\theta(X|M) = \int\int p(X)p(M|X)\sum_{t=1}^{T} \ln p_\theta(x_t|M)dXdM \tag{1}$$

As alluded to in Barber & Agakov (2004) and Wu et al. (2018a), this objective can be interpreted as maximizing the mutual information, $I(X;M)$, between the memory, M, and the episode, X, since $I(X;M) = H(X) + \mathcal{J} = H(X) - H(X;M)$ and given that the entropy of the data, $H(X)$, is constant. In order to actualize Equation 1 we rely on a variational bound which we derive in the following section.

3.2 VARIATIONAL LOWER BOUND

To efficiently read from the memory, M, we introduce a set of latent variables corresponding to the K addressing read heads, $Y = \{Y_t = \{y_{tk}\}_{k=1}^{K}\}_{t=1}^{T}, y_{tk} \in \mathbb{R}^3$, and a set of latent variables corresponding to the readout from the memory, $Z = \{z_t\}_{t=1}^{T}, z_t \in \mathbb{R}^L$. Given these latent variables, we can decompose the conditional, $\ln p(X|M)$, using the product rule and introduce variational approximations $q_\phi(Z|X)$ and $q_\phi(Y|X)$ via a multipy-by-one trick:

$$\begin{aligned}
\ln p(X|M) &= \ln \frac{p(X,Z,Y|M)}{p(Z,Y|X,M)} = \ln \frac{p(X|Z,M,Y)\, p(Z|M,Y)\, p(Y|M)}{p(Z|M,Y,X)\, p(Y|M,X)} \\
&\approx \mathbb{E}_{q_\phi(Z|X), q_\phi(Y|X)} \left(\ln \frac{p(X|Z,M,Y)\, p(Z|M,Y)\, p(Y)\, q_\phi(Z|X)\, q_\phi(Y|X)}{p(Z|M,Y,X)\, p(Y|M,X)\, q_\phi(Z|X)\, q_\phi(Y|X)} \right) && (2) \\
&= \mathcal{L}_T + \underbrace{\mathbb{E}_{q_\phi(Y|X)\; q_\phi(M|X,Y)} \mathcal{D}_{KL}(q_\phi(Z|X)||p(Z|M,Y,X))}_{\geq 0} && (3) \\
&\quad + \underbrace{\mathcal{D}_{KL}(q_\phi(Y|X)||p(Y|M,X))}_{\geq 0}
\end{aligned}$$

To simplify the objective, Equation 2 assumes that Y is independent from M: $p(Y|M) = p(Y)$. This decomposition results in Equation 3, which includes two KL-divergences against true posteriors, $p(Z|M,Y,X)$ and $p(Y|M,X)$. We can then train the model by maximizing the evidence lower bound (ELBO), $\mathcal{L}_T$, to the true conditional, $\ln p(X|M) > \mathcal{L}_T$:

$$\mathcal{L}_T = \underbrace{\mathbb{E}_{q_\phi(Z|X),\, q_\phi(Y|X)} \ln p_\theta(X|Z, M, Y)}_{\text{Decoder}} - \underbrace{\mathbb{E}_{q_\phi(Y|X)} \mathcal{D}_{KL}[q_\phi(Z|X)||p_\theta(Z|M, Y)]}_{\text{Amortized latent variable posterior vs. memory readout prior}} - \underbrace{\mathcal{D}_{KL}[q_\phi(Y|X)||p(Y)]}_{\text{Amortized key posterior vs. key prior}} \tag{4}$$

The bound in Equation 4 is tight if $q_\phi(Z|X) = p(Z|M, Y, X)$ and $q_\phi(Y|X) = p(Y|M, X)$. While the ELBO in Equation 4 is principled, it involves inferring the entire memory M. This prevents us from decoupling the size of the memory from inference and scaling the computation complexity based on the size of the memory. To alleviate this constraint, we assume a purely deterministic memory, $M \sim \delta[f_{mem}(f_{enc}(X))]$, built by transforming the input episode, X, via an encoder and memory transformation model, $f_{mem} \circ f_{enc}$. We also assume that the regions of memory which are useful in reconstructing a sample, x_t, can be summarized by a set of K localized contiguous memory sub-blocks:

$$\hat{M} \sim \delta\left[\{m_k = f_{ST}(M, y_{tk})\}_{k=1}^{K}\right] \tag{5}$$

$\hat{M}$ in equation 5 represents a set of K memory sub-regions, determined by the addressing key, y_{tk}, and a spatial transformer (ST) network Jaderberg et al. (2015), f_{ST} [2]. Our final optimization objective, $\mathcal{L}_T \approx \mathcal{L}_T|_{\hat{M} \approx M}$, is attained by replacing M in Equation 4 with $\hat{M}$ and is summarized by the graphical model in 2 below.

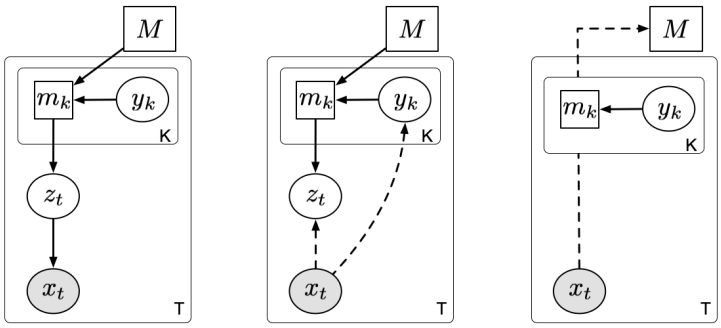
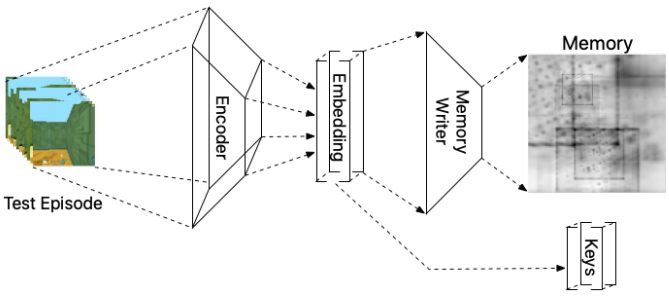


Figure 2: *Left*: Generative model. *Center*: Read inference model. *Right*: Write inference model. Dashed lines represent approximate inference, while solid lines represent computing of a conditional distribution. Squares represent deterministic nodes. Standard plate notation is used to depict a repetitive operation.

3.3 Write Model



sample episode:
$X_t = \{x_1, ..., x_T\} \sim \mathcal{D}$;
compute embedding:
$E = f_{\theta_{enc}}(X_t)$;
infer keys: $Y_t \sim$
$\mathcal{N}(\mu_{\theta_{key}}(E), \sigma^2_{\theta_{key}}(E))$;
write memory:
$M \sim \delta(f_{\theta_{mem}}(E))$

Figure 3: *Left*: Write model. *Right*: Write operation.

[2] We provide a brief review of spatial transformers in Appendix A

Writing to memory in the K++ model (Figure 3) entails encoding the input episode, $X_t = \{x_t\}_{t=1}^T$, through the encoder, $E = f_{\theta_{enc}}(X_t)$, pooling the representation over the episode and encoding the pooled representation with the memory writer, $M = f_{\theta_{mem}}(E)$. In this work we employ a Temporal Shift Module (TSM) (Lin et al., 2019), applied on a ResNet18 (He et al., 2016). TSM works by shifting feature maps of a 2d vision model in the temporal dimension. This allows the model to build richer representations of contextual features. In the case of K++, this allows the encoder to build a better representation of the memory. Using a TSM encoder over a standard convolutional stack improves the performance of both K++ and DKM, where the latter observes an improvement of 6.32 nats/image over the reported test ELBO of 77.2 nats/image (Wu et al., 2018b) for the Omniglot dataset. As far as we are aware, the application of a TSM encoder to memory models has not been explored and is a contribution of this work.

The memory writer model, $f_{\theta_{mem}}$, in Figure 3, allows K++ to non-linearly transform the pooled embedding to better summarize the episode. In addition to inferring the deterministic memory, M, we also project the non-pooled embedding, E, through a key model, $f_{\theta_{key}}$, described in Section 3.4:

$$\{Y_t\}_{t=1}^T = \mu_{\theta_{key}}(E) + \sigma^2_{\theta_{key}}(E) \odot \epsilon, \quad \{Y_t\}_{t=1}^T \in \mathbb{R}^{T\times 3}, \quad \epsilon \sim \mathcal{N}(0,1). \tag{6}$$

3.4 Sample Generation

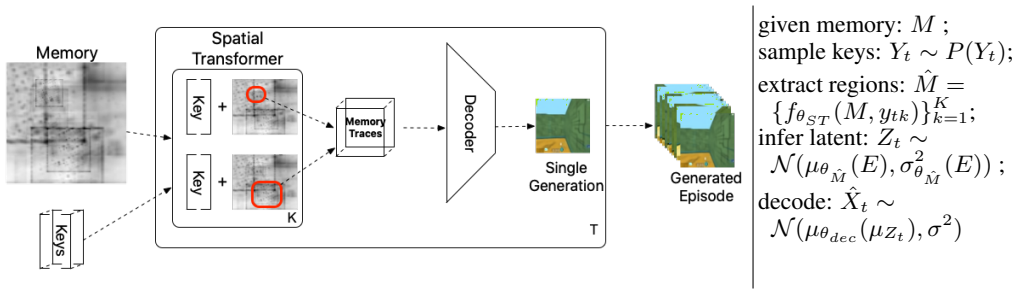


Figure 4: *Left*: Generative model. *Right*: Generative operation.

The Kanerva++ model, like the original KM and DKM models, enables sample generation given an existing memory or set of memories. K samples from the prior key distribution, $\{y_k\}_{k=1}^K \sim p(Y) = \mathcal{N}(0,1), y_k \in \mathrm{R}^3$, are used to parameterize the spatial transformer, f_{ST}, which indexes the deterministic memory, M. The result of this differentiable indexing is a set of memory sub-regions, $\hat{M}$, which are used in the decoder, $p_\theta(X|\cdot)$, to generate synthetic samples. Reading samples in this manner forces the encoder to utilize memory sub-regions that are useful for reconstruction, as non-read memory regions receive zero gradients during backpropagation.

The memory traces, $\hat{M}$, are used in the decoder training (Section 3.5) through the learned prior, $p_\theta(Z|\hat{M}, Y) = \mathcal{N}(\mu_z, \sigma_z^2)$, from Equation 4. The KL divergence, $\mathbb{E}_{q_\phi(Y|X)}\mathcal{D}_{KL}[q_\phi(Z|X)||P_\theta(Z|\hat{M}, Y)]$ constrains the optimization objective to keep the representation of the amortized approximate posterior, $q_\phi(Z|X)$, close to the memory readout representation of the learned prior, $p_\theta(Z|\hat{M}, Y)$.

3.5 Read Model

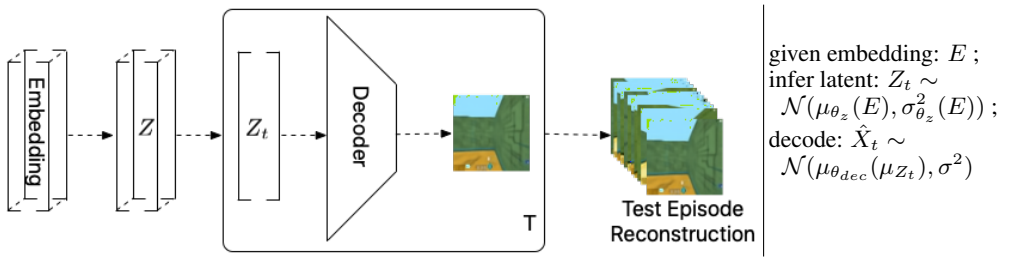


Figure 5: *Left*: Read model. *Right*: Read operation.

The K++ read model (Figure 5) is similar in structure to a standard VAE with the exception that there exist T samples per episode. As in KM and DKM, we use the mean of the latent variable, μ_{Z_t}, as input to the decoder. In contrast to (Tomczak & Welling, 2018; Graves et al., 2018; Salimans et al., 2017; van den Oord et al., 2016), we use a simple convolutional decoder for all our experiments.

4 EXPERIMENTS

We contrast K++ against state of the art vision models for likelihood estimation and present empirical results in Table 1. A 1000 sample importance weighted estimate of the likelihood (Burda et al., 2016), $\ln p_\theta(x) \approx \sum_{i=1}^{1000} \mathcal{L}_{t_i}$, (for each sample in the empirical test distribution) is used as a Monte Carlo estimate to integrate out latent variables. This allows us to fairly compare various models with different latent variable structures. Binarized datasets assume Bernoulli output distributions, while continuous values are modeled by a Gaussian output distribution with a single learnable variance term (Dai & Wipf, 2019). Model architecture and optimization details are provided in Appendix E as well as the included code. Finally, extra Celeb-A generations and test image reconstructions for all experiments are provided in Appendix B and Appendix D respectively.

Method	**Binarized MNIST (nats / image)**	**Binarized Omniglot (nats / image)**	**Fashion MNIST (bits / dim)**	**CIFAR10 (bits / dim)**	**DMLab Mazes (bits / dim)**
VAE Kingma & Welling (2014)	87.86	104.75	5.84	6.3	-
IWAE Burda et al. (2016)	85.32	103.38	-	-	-
Improved decoders					
PixelVAE++ Sadeghi et al. (2019)	78.00	-	-	2.90	-
MAE Ma et al. (2018)	77.98	89.09	-	2.95	-
DRAW Gregor et al. (2015)	87.4	96.5	-	3.58	-
MatNet Bachman (2016)	78.5	89.5	-	3.24	-
Richer priors					
Ordered ACN* Graves et al. (2018)	73.9	-	-	3.07	-
VLAE Chen et al. (2017)	78.53	102.11	-	2.95	-
VampPrior Tomczak & Welling (2018)	78.45	89.76	-	-	-
Memory models					
VMA Bornschein et al. (2017)	-	103.6	-	-	-
KM* Wu et al. (2018a)	-	68.3	-	4.37	-
DNC* Graves et al. (2016)	-	100	-	-	-
DKM* Wu et al. (2018b)	75.3	77.2	-	4.79	**2.75**
DKM w/TSM (our impl) *	51.84	70.88	-	-	-
Kanerva++ (ours)	**45.64 / 41.58**	**71.64 / 66.24**	**3.63 / 3.42**	-2344.5 / -2248.96	- / 2.89

Table 1: Test negative likelihoods. * indicates models that only provide negative ELBO values; note that $\ln p_\theta(x) > \mathcal{L}_T$. K++ formatted as {-likelihood / -ELBO} to contrast methods that only provide the lower bound. Lower is better for both cases. CIFAR10 values reported as nats/img for K++.

4.1 ITERATIVE INFERENCE

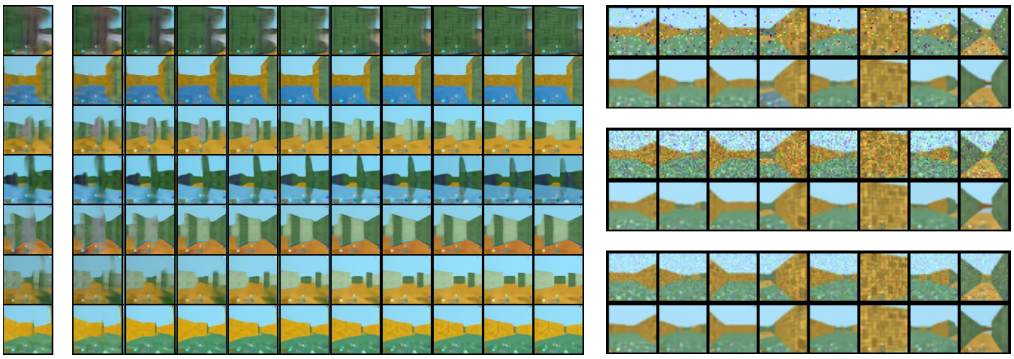


Figure 6: *Left*:Left most column: first random key generation. Following columns created by inferring previous sample through K++. *Right*: denoising of salt & pepper (top), speckle (middle) and poisson noise (bottom).

One of the benefits of K++ is that it uses the memory to learn a more informed prior by condensing the information from an episode of samples. One might suspect that, based on the dimensionality of the memory and the size of the read traces, the memory might only learn prototypical patterns, rather than a full amalgamation of the input episode. This presents a problem for generation, as described in Section 3.4 and can be observed in the left most column of Figure 6-*Left* where the first generation from a random key appears blurry. To overcome this limitation we rely on the iterative inference of Kanerva memory models (Wu et al., 2018a;b). By holding the memory, M, fixed and repeatedly inferring the latents, we are able to cleanup the pattern by leveraging the contextual information contained within the memory. This is visualized in the proceeding columns of Figure 6, where we observe a slow but clear improvement in generation quality. This property of iterative inference is one of the central benefits of using a memory model over a tradition solution like a VAE.

We also present results of iterative inference on more classical image noise distributions such as salt-and-pepper, speckle and poisson noise in Figure 6-*Right*. For each original noisy pattern (top rows) we provide the resultant final reconstruction after ten steps of cleanup. The proposed K++ is pretty robust to input noise distribution and is able to cleanup most of the patters in a semantically meaningful way.

4.1.1 IMAGE GENERATIONS

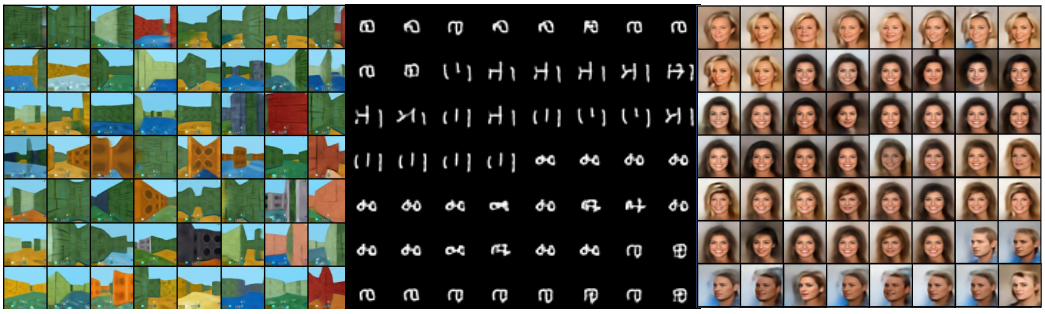


Figure 7: Key perturbed generations. *Left*: DMLab mazes. *Center*: Omniglot. *Right*: Celeb-A 64x64.

Figure 8: Random key generations. *Left*: DMLab mazes. *Center*: Omniglot. *Right*: Celeb-A 64x64.

Typical VAEs use high dimensional isotropic Gaussian latent variables ($\geq \mathbb{R}^{16}$) Burda et al. (2016); Kingma & Welling (2014). A well known property of high dimensional Gaussian distributions is that most of their mass is concentrated on the surface area of a high dimensional ball. Perturbations to a sample in an area of valid density can easily move it to an invalid density region (Arvanitidis et al., 2018; White, 2016), causing blurry or irregular generations. In the case of K++, since the key distribution, $y_t \sim p(Y), y_t \in \mathbb{R}^3$, is within a low dimensional space, local perturbations, $y_t + \epsilon, \epsilon \sim N(0, 0.1)$, are likely in regions with high probability density. We visualize this form of generation in Figure 7 for DMLab Mazes, Omniglot and Celeb-A 64x64 as well as the more traditional random key generations, $y_t \sim p(Y)$, in Figure 8.

Interestingly, local key perturbations of a trained DMLab Maze K++ model induces resultant generations that provide a natural traversal of the maze as observed by scanning Figure 7-*Left*, row by row, from left to right. In contrast, the random generations of the same task (Figure 8-*Left*) present a more discontinuous set of generations. We see a similar effect for the Omniglot and

Celeb-A datasets, but observe that the locality is instead tied to character or facial structure as shown in Figure 7-*Center* and Figure 7-*Right*. Finally, in contrast to VAE generations, K++ is able to generate sharper images of ImageNet32x32 as shown in Appendix C. Future work will investigate this form of locally perturbed generation through an MCMC lens.

4.2 Ablation: Is Block Allocated Spatial Memory Useful?

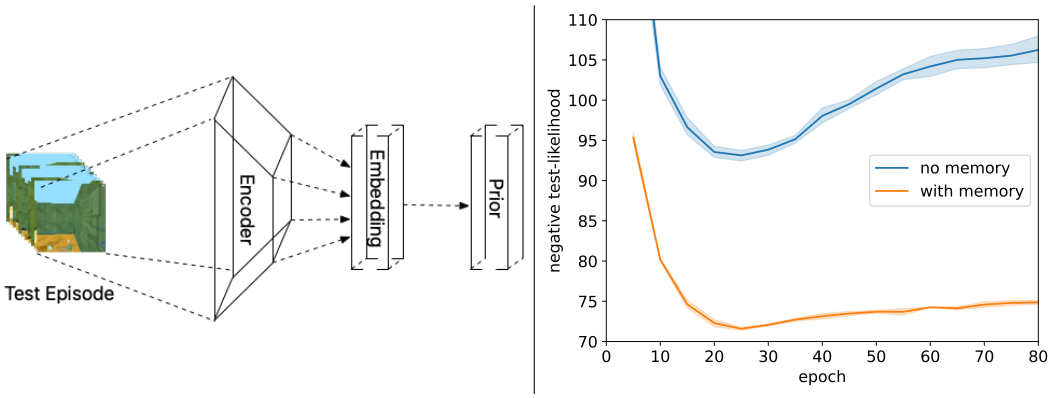


Figure 9: *Left*: Memory free write model. *Right*: Test likelihood (mean and 1 std reported).

While Figure 7 demonstrates the advantage of having a low dimensional sampling distribution and Figure 6 demonstrates the benefit of iterative inference, it is unclear whether the performance benefit in Table 1 is achieved from the episodic training, model structure, optimization procedure or memory allocation scheme. To isolate the cause of the performance benefit, we simplify the write architecture from Section 3.3 as shown in Figure 9-*Left*. In this scenario, we replace the learned memory prior, with an equivalently sized dense model that projects the embedding, E, while keeping all other aspects the same. We train both models five times with the exact same TSM-ResNet18 encoder, decoder, optimizer and learning rate scheduler. As shown in Figure 9-*Right*, the likelihood of the K++ model is **20.6 nats/image** better than the baseline model for the evaluated Omniglot dataset. This confirms that the spatial, block allocated latent memory model proposed in this work is useful when working with image distributions. Future work will explore this dimension for other modalities such as audio and text.

5 Conclusion

In this work we propose a novel block allocated memory in a generative model framework and demonstrate its state-of-the-art performance on several conditional image generation tasks. We also show that stochasticity in low-dimensional spaces produces higher quality samples in comparison to high dimensional latents typically used in VAEs. Furthermore, perturbations to the low-dimensional key generate samples with high variations. There are still however many unanswered questions: would a hard attention based solution to differentiable indexing prove to be better than a spatial transformer? What is the optimal upper bound of window read regions based on the input distribution? Future work will hopefully be able to address these lingering issues and further improve generative memory models.

References

David J Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII—1983*, pp. 1–198. Springer, 1985.

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: On the curvature of deep generative models. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

Philip Bachman. An architecture for deep, hierarchical generative models. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 4826–4834, 2016. URL http://papers.nips.cc/paper/6141-an-architecture-for-deep-hierarchical-generative-models.

David Barber and Felix V Agakov. Information maximization in noisy channels: A variational approach. In *Advances in Neural Information Processing Systems*, pp. 201–208, 2004.

Jörg Bornschein, Andriy Mnih, Daniel Zoran, and Danilo Jimenez Rezende. Variational memory addressing in generative models. In *Advances in Neural Information Processing Systems*, pp. 3920–3929, 2017.

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1509.00519.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *ICLR*, 2017.

Bin Dai and David P. Wipf. Diagnosing and enhancing VAE models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=B1e0X3C9tQ.

Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

Yaxiang Fan, Gongjian Wen, Deren Li, Shaohua Qiu, Martin D Levine, and Fei Xiao. Video anomaly detection and localization via gaussian mixture fully convolutional variational autoencoder. *Computer Vision and Image Understanding*, pp. 102920, 2020.

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.

Anirudh Goyal Alias Parth Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. Z-forcing: Training stochastic recurrent networks. In *Advances in neural information processing systems*, pp. 6713–6723, 2017a.

Prasoon Goyal, Zhiting Hu, Xiaodan Liang, Chenyu Wang, and Eric P Xing. Nonparametric variational auto-encoders for hierarchical representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5094–5102, 2017b.

Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: training imagenet in 1 hour. *CoRR*, abs/1706.02677, 2017c. URL http://arxiv.org/abs/1706.02677.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

Alex Graves, Jacob Menick, and Aaron van den Oord. Associative compression networks for representation learning. *arXiv preprint arXiv:1804.02476*, 2018.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning*, pp. 1462–1471, 2015.

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taïga, Francesco Visin, David Vázquez, and Aaron C. Courville. Pixelvae: A latent variable model for natural images. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BJKYvt5lg.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Donald Olding Hebb. *The organization of behavior: a neuropsychological theory*. J. Wiley; Chapman & Hall, 1949.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pp. 2017–2025, 2015.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Pentti Kanerva. *Sparse distributed memory*. MIT press, 1988.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014.

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

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pp. 4743–4751, 2016.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016. URL http://arxiv.org/abs/1611.07308.

Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. In *Advances in neural information processing systems*, pp. 1172–1180, 2016.

Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7083–7093, 2019.

Hanxiao Liu, Andrew Brock, Karen Simonyan, and Quoc V. Le. Evolving normalization-activation layers. *CoRR*, abs/2004.02967, 2020. URL https://arxiv.org/abs/2004.02967.

Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

James Lucas, George Tucker, Roger B. Grosse, and Mohammad Norouzi. Understanding posterior collapse in generative latent variable models. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=r1xaVLUYuE.

Xuezhe Ma, Chunting Zhou, and Eduard Hovy. Mae: Mutual posterior-divergence regularization for variational autoencoders. In *International Conference on Learning Representations*, 2018.

Adam Marblestone, Yan Wu, and Greg Wayne. Product kanerva machines: Factorized bayesian memory. *Bridging AI and Cognitive Science Workshop. ICLR*, 2020.

Simon Marlow, Tim Harris, Roshan P James, and Simon Peyton Jones. Parallel generational-copying garbage collection with a block-structured heap. In *Proceedings of the 7th international symposium on Memory management*, pp. 11–20. ACM, 2008.

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.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pp. 9209–9220, 2018. URL http://papers.nips.cc/paper/8132-visual-reinforcement-learning-with-imagined-goals.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims (eds.), *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pp. 807–814. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/432.pdf.

Tony A Plate. Holographic reduced representations. *IEEE Transactions on Neural networks*, 6(3): 623–641, 1995.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2827–2836. PMLR, 2017. URL http://proceedings.mlr.press/v70/pritzel17a.html.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=Hkuq2EkPf.

Hossein Sadeghi, Evgeny Andriyash, Walter Vinci, Lorenzo Buffoni, and Mohammad H. Amin. Pixelvae++: Improved pixelvae with discrete prior. *CoRR*, abs/1908.09948, 2019. URL http://arxiv.org/abs/1908.09948.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BJrFC6ceg.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap. Meta-learning with memory-augmented neural networks. In Maria-Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1842–1850. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/santoro16.html.

Suman Sedai, Dwarikanath Mahapatra, Sajini Hewavitharanage, Stefan Maetschke, and Rahil Garnavi. Semi-supervised segmentation of optic cup in retinal fundus images using variational autoencoder. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 75–82. Springer, 2017.

Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of residual networks using large learning rates. *CoRR*, abs/1708.07120, 2017. URL http://arxiv.org/abs/1708.07120.

Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pp. 1214–1223, 2018.

Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with pixelcnn decoders. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 4790–4798, 2016. URL http://papers.nips.cc/paper/6527-conditional-image-generation-with-pixelcnn-decoders.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1410.3916.

Tom White. Sampling generative networks. *arXiv preprint arXiv:1609.04468*, 2016.

Yan Wu, Greg Wayne, Alex Graves, and Timothy Lillicrap. The kanerva machine: A generative distributed memory. *ICLR*, 2018a.

Yan Wu, Gregory Wayne, Karol Gregor, and Timothy Lillicrap. Learning attractor dynamics for generative memory. In *Advances in Neural Information Processing Systems*, pp. 9379–9388, 2018b.

Yuxin Wu and Kaiming He. Group normalization. *Int. J. Comput. Vis.*, 128(3):742–755, 2020. doi: 10.1007/s11263-019-01198-w. URL https://doi.org/10.1007/s11263-019-01198-w.

Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

Qingyu Zhao, Nicolas Honnorat, Ehsan Adeli, Adolf Pfefferbaum, Edith V Sullivan, and Kilian M Pohl. Variational autoencoder with truncated mixture of gaussians for functional connectivity analysis. In *International Conference on Information Processing in Medical Imaging*, pp. 867–879. Springer, 2019.

Guangxiang Zhu, Zichuan Lin, Guangwen Yang, and Chongjie Zhang. Episodic reinforcement learning with associative memory. In *International Conference on Learning Representations*, 2019.

A SPATIAL TRANSFORMER REVIEW

Indexing a matrix, $M[x : x + \Delta x, y : y + \Delta y]$, is typically a non-differentiable operation since it involves hard cropping around an index. Spatial transformers (Jaderberg et al., 2015) provide a solution to this problem by decoupling the problem into two differentiable operands:

1. Learn an affine transformation of coordinates.
2. Use a differntiable bilinear transformation.

The affine transformation of source coordinates, $\begin{bmatrix} i^s \\ j^s \end{bmatrix}$, to target coordinates, $\begin{bmatrix} i^t \\ j^t \end{bmatrix}$ is defined as:

$$\begin{bmatrix} i^t \\ j^t \end{bmatrix} = \begin{bmatrix} s & 0 & x \\ 0 & s & y \end{bmatrix} \begin{bmatrix} i^s \\ j^s \\ 1 \end{bmatrix} = \begin{bmatrix} y_0 & 0 & y_1 \\ 0 & y_0 & y_2 \end{bmatrix} \begin{bmatrix} i^s \\ j^s \\ 1 \end{bmatrix} \tag{7}$$

Here, the affine transform, $\theta = \begin{bmatrix} s & 0 & x \\ 0 & s & y \end{bmatrix}$ has three learnable scalars: $\{s, x, y\}$ which define a scaling and translation in i and j respectively. In the case of K++, these three scalars represent the components of the key sample, $\{y_0, y_1, y_2\} \in \mathbb{R}^3$ as shown in Equation 7. After transforming the co-ordinates (not to be confused with the actual data), spatial transformers learn a differentiable bilinear transform which can be interpreted as learning a differentiable mask that is element-wise multiplied by the original data, M:

$$\sum_{n=-1}^{J} \sum_{r=-1}^{J} \left(M_{nr}^{c} \max(0, 1 - |i_{nr}^{t} - r|) \max(0, 1 - |j_{nr}^{t} - n|) \right) \tag{8}$$

Consider the following example where $\theta = \begin{bmatrix} 0.5 & 0 & 0.3 \\ 0 & 0.5 & 0.5 \end{bmatrix}$; this parameterization differntiably extracts the region shown in Figure 10-*Right* from Figure 10-*Left*:

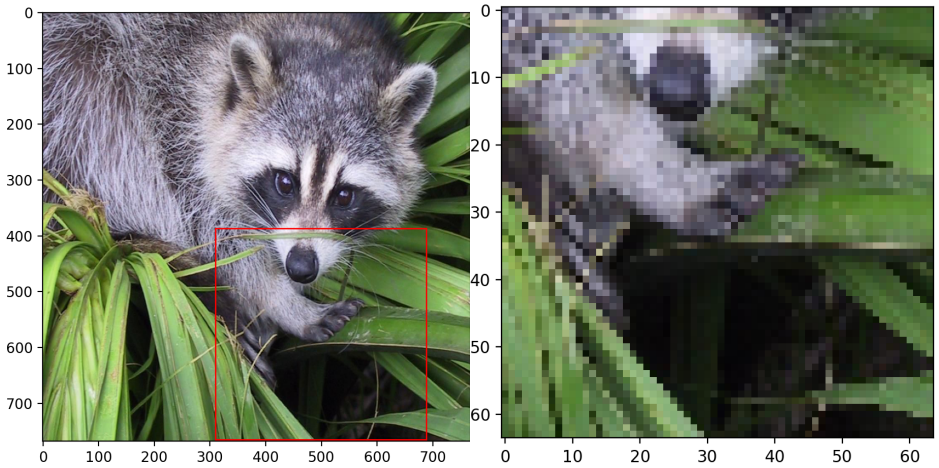


Figure 10: Spatial transformer example. *Left*: original image with region inlaid. *Right*: extracted grid.

The range of values for $\{s, x, y\}$ is bound between $[-1, 1]$, where the center of the image is $[0, 0]$.

B CELEB-A GENERATIONS

Figure 11: Random key Celeb-A generations.

We present random key generations of Celeb-A 64x64, trained without center cropping in Figure 11.

C VAE VS. K++ IMAGENET32X32 GENERATIONS

Figure 12: ImageNet32x32 generations. *Left*: VAE; *Right*: K++.

Figure 12 shows the difference in generations of a standard VAE vs. K++. In contrast to the standard VAE generation (Figure 12-*Left*), the K++ generations (Figure 12-*Right*) appear much sharper, avoiding the blurry generations observed with standard VAEs.

D TEST IMAGE RECONSTRUCTIONS

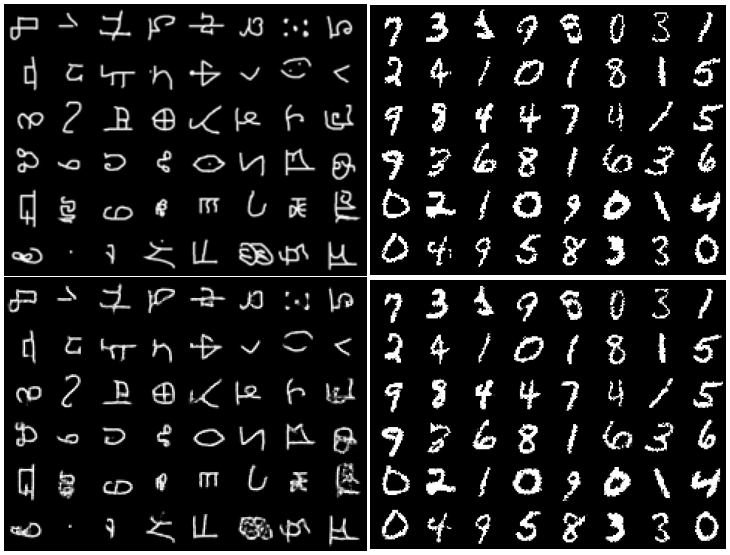


Figure 13: Binarized test reconstructions; top row are true samples. *Left*: Omniglot; *Right*: MNIST.

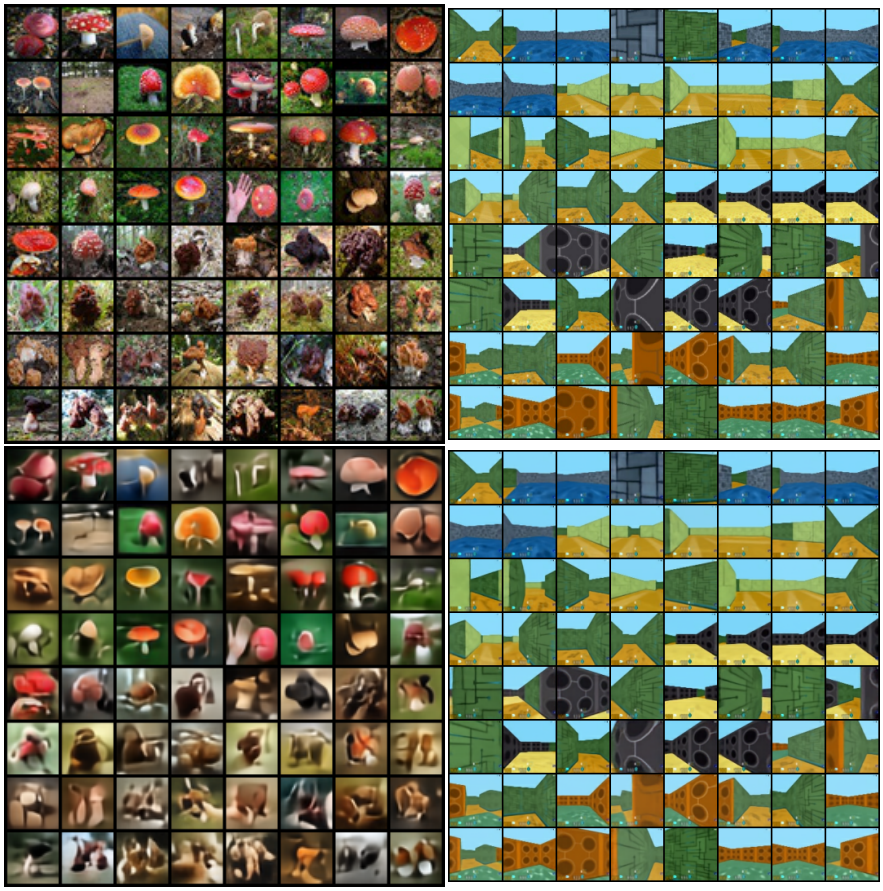


Figure 14: Test set reconstructions; top row are true samples. *Left*: ImageNet64x64. *Right*: DMLab Mazes.

Figure 15: Test set reconstructions; top row are true samples. Celeb-A 64x64.

E MODEL ARCHITECTURE & TRAINING

Encoder: As mentioned in Section 3.3, we use a TSM-Resnet18 encoder with Batchnorm (Ioffe & Szegedy, 2015) and ReLU activations (Nair & Hinton, 2010) for all tasks. We apply a fractional shift of the feature maps by 0.125 as suggested by the authors.

Decoder: Our decoder is a simple conv-transpose network with EvoNormS0 (Liu et al., 2020) inter-spliced between each layer. Evonorms0 is similar in stride to Groupnorm (Wu & He, 2020) combined with the swish activation function (Ramachandran et al., 2018).

Optimizer & LR scheule: We use LARS (You et al., 2017) coupled with ADAM (Kingma & Ba, 2014) and a one-cycle (Smith & Topin, 2017) cosine learning rate schedule (Loshchilov & Hutter, 2017). A linear warm-up of 10 epochs (Goyal et al., 2017c) is also used for the schedule. A weight decay of $1e-3$ is used on every parameter barring biases and the affine terms of batchnorm. Each task is trained for 500 or 1000 epochs depending on the size of the dataset.

Dense models: All dense models such as our key network are simple three layer deep linear dense models with a latent dimension of 512 coupled with spectral normalization (Miyato et al., 2018).

Memory writer: $f_{\theta_{mem}}$ uses a deep linear conv-transpose decoder on the pooled embedding, E with a base feature map projection size of 256 with a division by 2 per layer. We use a memory size of $\mathbb{R}^{3\times64\times64}$ for all the experiments in this work.

Learned Prior: $p_\theta(Z|\hat{M}, Y)$ uses a convolutional encoder that stacks the K read traces, $\{f_{ST}(M, y_{tk})\}_{k=1}^{K}$, along the channel dimension and projects it to the dimensionality of Z.

In practice, we observed that K++ is about 2x as fast (wall clock) compared to DKM.