The Variational Homoencoder: Learning to learn high capacity generative models from few examples SUPPLEMENTARY MATERIAL

1 Constrained Posterior Approximation

In a VAE, use of a recognition network encourages learning of generative models whose structure permits accurate amortised inference. In a VHE, this recognition network takes only a small subsample as input, which additionally encourages that the true posterior $p(c|X)$ can be well approximated from only a few examples of X. For a subsample $D \subset X$, $q(c; D)$ is implicitly trained to minimise the KL divergence from this posterior *in expectation* over possible sets X consistent with D. For a data distribution p_d we may equivalently describe the VHE objective (Equation **??**) as

$$\mathop{\mathbb{E}}_{p_d(D)} \mathop{\mathbb{E}}_{p_d(X|D)} \left[\mathop{\mathbb{E}}_{x \in X} \Big[\log p(x) \Big] - \frac{1}{|X|} \mathrm{D}_{KL} \Big[q(c; D) \parallel p(c|X) \Big] \right] \tag{1}$$

Note that the variational gap on the right side of this equation is itself bounded by:

$$\mathop{\mathbb{E}}_{p_d(X|D)} \mathrm{D}_{KL} \Big[q(c; D) \parallel p(c|X) \Big] \geq \mathrm{D}_{KL} \Big[q(c; D) \parallel \mathop{\mathbb{E}}_{p_d(X|D)} p(c|X) \Big] \geq 0 \tag{2}$$

The left inequality is tightest when $p(c|X)$ matches $p(c|D)$ well across all X consistent with D, and exact only when these are equal. We view this aspect of the VHE loss as regulariser for constrained posterior approximation, encouraging models for which the posterior $p(c|X)$ can be well determined by sampled subsets $D \subset X$. This reflects how we expect the model to be used at test time, and in practice we have found this ‘loose’ bound to perform well in our experiments. In principle, the bound may also be tightened by introducing an auxiliary inference network (see Supplementary Material 2) which we leave as a direction for future research.

2 Tightened variational bound

The likelihood lower bound in the VHE objective may also be tightened by introduction of an auxiliary network $r(D; c, X)$, trained to infer which subset $D \subset X$ was used in q. This meta-inference approach was introduced in Salimans et al. (2015) to develop stochastic variational posteriors using MCMC inference. Applied to Equation **??**, this yields a modified bound for the VHE objective

$$\log p(\mathcal{X}) \geq \sum_{x \in \mathcal{X}} \mathop{\mathbb{E}}_{\substack{q'(D; X_{(x)}) \\ q(c; D)}} \left[\log p(x|c) - \frac{1}{|X_{(x)}|} \log \frac{p(c) r(D; c, X_{(x)})}{q'(D; X_{(x)}) q(c; D)} \right] \tag{3}$$

where $q'(D; X)$ describes the stochastic sampling procedure for sampling $D \subset X$, which indeed may itself be learned using policy gradients.

We have conducted preliminary experiments using fixed q' and a simple functional form $r(D; c, X) = \prod_i r(d_i; c, X) \propto \prod_i \big[f_\psi(c) \cdot \xi_{d_i} \big]$, learning parameters ψ and embeddings $\{\xi_d : d \in \mathcal{X}\}$; however, on the Omniglot dataset we found no additional benefit over the strictly loose bound (Equation **??**). We attribute this to the already high similarity between elements of the same Omniglot character class, allowing the approximate posterior $q(c; D)$ to be relatively robust to different choices of D. However, we expect that the gain from using such a tightened objective may be much greater for domains with lower intra-class similarity (e.g. natural images), and thus suggest the tightened bound of Equation 3 as a direction for future research.

3 Variational Bound for Hierarchical Models

The resampling trick may be applied iteratively, to construct likelihood bounds over hierarchically organised data. Expanding on Equation **??**, suppose that we have collection of datasets

$$\mathbf{X} = \mathcal{X}_1 \sqcup \mathcal{X}_2 \sqcup \ldots \sqcup \mathcal{X}_N \tag{4}$$

For example, each $\mathcal{X}$ might be a different alphabet whose latent description a generates many character classes X_i, and for each of these a corresponding latent c_i is used to generate many images x_{ij}. From this perspective, we would like to learn a generative model for alphabets $\mathcal{X}$ of the form

$$p(\mathcal{X}) = \int p(a) \prod_{X_i \subset \mathcal{X}} \int p(c|a) \prod_{x \in X_i} p(x|c, a) \mathrm{d}c \mathrm{d}a \tag{5}$$

Reapplying the same trick as before yields a bound taken over all elements x:

$$\begin{aligned} \log p(\mathbf{X}) \geq \sum_{x \in \mathbf{X}} \mathop{\mathbb{E}}_{\substack{D^a \subset \mathcal{X}_{(x)} \\ D^c \subset X_{(x)}}} \Bigg[& \mathop{\mathbb{E}}_{\substack{q_a(a|D_1) \\ q_c(c|D_2, a)}} \log p(x|c) \\ & - \frac{1}{|\mathcal{X}_{(x)}|} \mathrm{D}_{KL}\big(q_a(a|D^a) \parallel p(a)\big) \\ & - \frac{1}{|X_{(x)}|} \mathrm{D}_{KL}\big(q_c(c|D^c, a) \parallel p(c|a)\big) \Bigg] \end{aligned}$$

This suggests an analogous *hierarchical resampling* procedure: Summing over every element x, we can bound the log likelihood of the full hierarchy by resampling subsets D^c, D^a, etc. at each level to construct an approximate posterior. All networks are trained together by this single objective, sampling x, D^a and D_c for each gradient step. Note that this procedure need only require passing sampled elements, rather than full classes, into the upper-level encoder q_a.

4 Results for Simple 1D Distributions

With a Neural Statistician model, under-utilisation of latent variables is expected to pose the greatest difficulty either when $|D|$ is too small, or the inference network q is insufficiently expressive. We demonstrate on simple 1D distributions that a Variational Homoencoder can bring improvements under these circumstances. For this we created five datasets as follows, each containing 100 classes from a particular parametric family, and with 100 elements sampled from each class.

1. **Gaussian**: Each class is Gaussian with μ drawn from a Gaussian hyperprior (fixed σ^2).
2. **Mixture of Gaussians**: Each class is an even mixture of two Gaussian distributions with location drawn from a Gaussian hyperprior (fixed σ^2 and separation).
3. **von Mises**: Each class is von Mises with μ drawn from a Uniform hyperprior (fixed κ).
4. **Gamma**: Each class is Gamma with fixed β, and with α drawn from a Uniform hyperprior.
5. **Discrete**: Each class is Uniform on a subset of $\{1,\ldots,8\}$, either *1-4*, *5-8*, *odd* or *even*.

For each dataset, we then trained models using a variety of values for $|D|$, restricting the inference network $q(c; D)$ to a simple linear map with Gaussian output. In each case the generative model $p(x|c)$ was set to the correct parametric family, with parameters learned as a linear function of c. All models were built in Torch 7 (**?**) and optimised using Adam (**?**) for 200 epochs. To aid optimisation we used an additional 50 epochs for KL annealing, and used training error to select the best parameters from 3 independent training runs.

Our results show that, when $|D|$ is small, the Neural Statistician often places little to no information in $q(c; D)$ (Figure 1, top row). Our careful training suggests that this is not an optimisation difficulty, but is core to the objective as in **?**. In these cases a VHE better utilises the latent space, leading to improvements in both few-shot generation (by conditional NLL) and classification. Importantly, this is achieved while retaining good likelihood of test-set classes, typically matching or improving upon that achieved by a Neural Statistician (including a standard VAE, corresponding to $|D| = 1$).

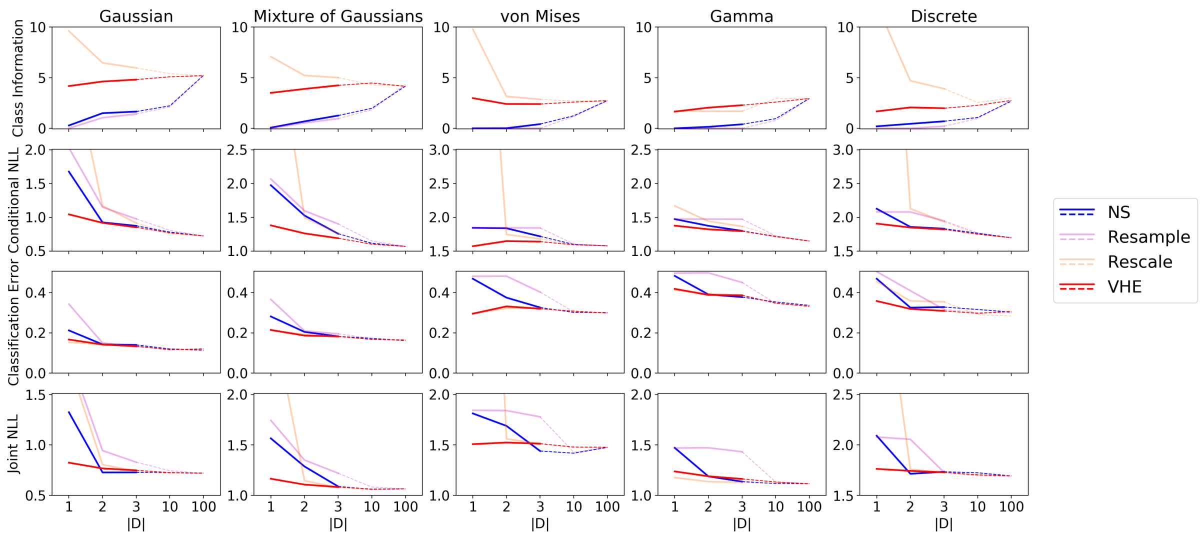


Figure 1: Comparison of models trained simple 1D distributions using various alternate objectives, including the Neural Statistician. $|D|$ is the number of encoder inputs during training. *Top row*: Mean encoded information $D_{KL}[q(c;D) \parallel p(c)]$; *Second row*: $|D|$-shot generation loss $-\mathbb{E}_{c\sim q(c;D)} \log p(x'|c)$; *Third row*: $|D|$-shot binary classification error, by minimising conditional NLL *Bottom row*: Joint NLL (per element) of full test set, calculated by importance weighting on 200 samples from $q(c;X)$;

5 PixelCNN Omniglot Architecture

5.1 Methodology

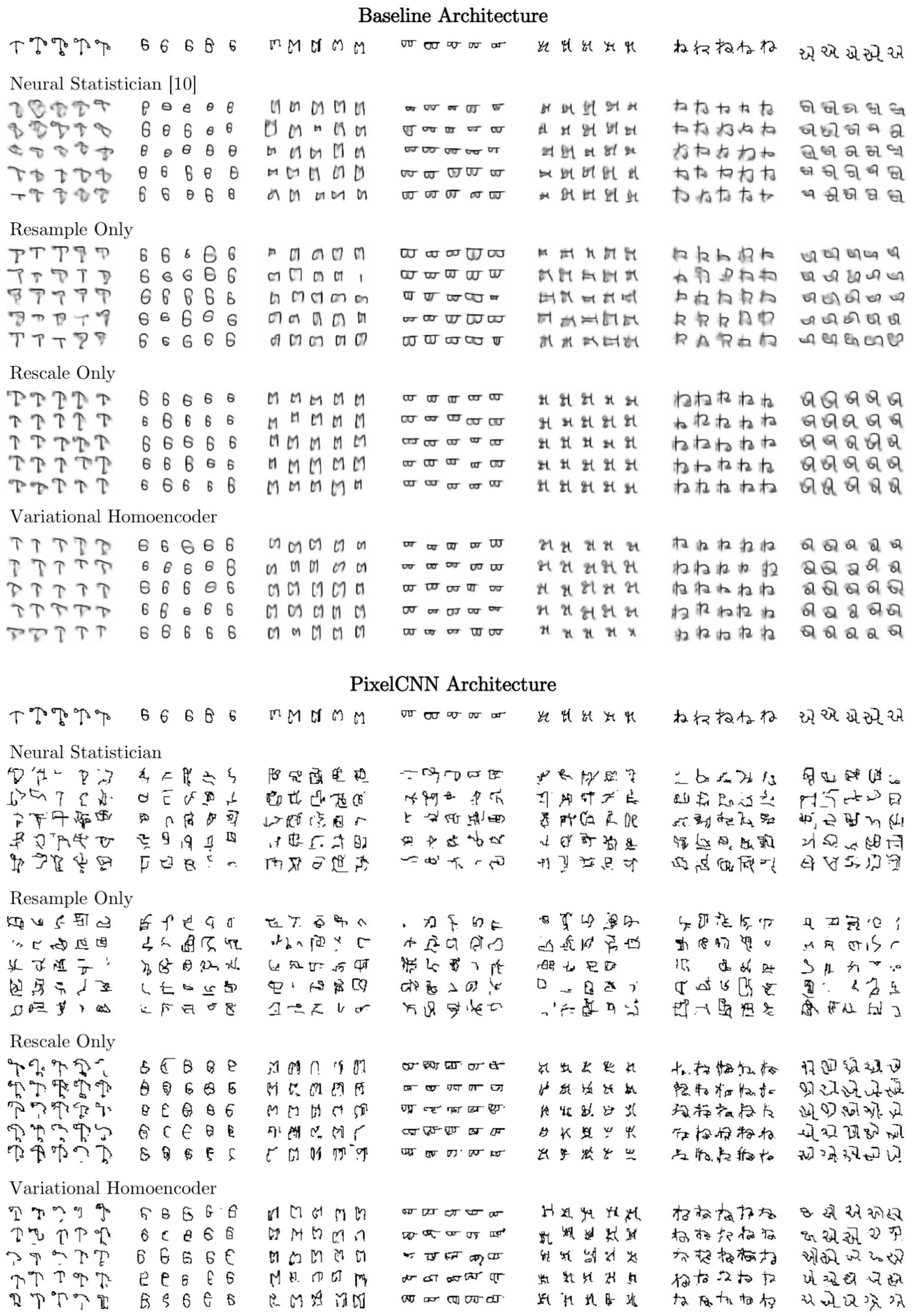
Our architecture uses a 8x28x28 latent variable c, with a full architecture detailed below. For our classification experiments, we trained 5 models on each of the objectives (VHE, Rescale only, Resample only, NS). Occasionally we found instability in optimisation, causing sudden large increases in the training objective. When this happened, we halted and restarted training. All models were trained for 100 epochs on 1000 characters from the training set (the remaining 200 have been used as validation data for model selection). Finally, for each objective we selected the parameters achieving the best training error.

Note that we did not optimise or select models based on classification performance, other than through our development of our model's architecture. However, we find that classification performance is well correlated the generative training objective, as can be seen in the full table of results.

We perform classification by calculating the expected conditional likelihood under the variational posterior: $\mathbb{E}_{q(c;D)}\, p(x|c)$. This is approximated using 20 samples for the outer expectation, and importance sampling with $k = 10$ for the inner integral $p(x|c) = \mathbb{E}_{q(t|x)} \frac{p(t)}{q(t|x)} p(x|c,t)$

To evaluate and compare log likelihood, we trained 5 more models with the same architecture, this time on the canonical 30-20 alphabet split of Lake et al. We did not augment our training data. Again, we split the background set into training data (25 alphabets) and validation data (5) but do not use the validation set in training or evaluation for our final results. We estimate the total class log likelihood by importance weighting, using k=20 importance samples of the class latent c and k=10 importance samples of the transformation latent t for each instance.

5.2 Conditional Samples on Omniglot



5.3 Model Specification

[d] denotes a dimension d tensor. $\{t\}$ denotes a set with elements of type t. Posteriors q are Gaussian.

p(c)

A PixelCNN with autoregressive weights along only the spatial (not depth) dimensions of c. We use 2 layers of masked 64x3x3 convolutions, followed by a ReLU and two 8x1x1 convolutions corresponding to the mean and log variance of a Gaussian posterior for the following pixel.

p(t)

t: [16]   Normal(0, 1)

p(x—c,t)

c: [8x28x28], t: [16] $\mapsto$ x: [1x28x28]
PixelCNN is gated by y, and is autoregressive along only the spatial (not depth) dimensions of c. We use 2 layers of masked 64x3x3 convolutions, followed by a ReLU, a 2x1x1 convolution and a softmax, corresponding to a Bernoulli distribution on the following pixel.

q(c;D)

D: $\{$[1x28x28]$\}$ $\mapsto$ c: [8x28x28]

Input	**Operation**	**Output**
D	STNq	Y: $\{$[1x28x28]$\}$
Y	Mean	y: [1x28x28]
y	16x28x28 Conv	mu: [8x28x28], logvar: [8x28x28]

q(t;x)

x: [1x28x28] $\mapsto$ t: [16]
[H]

Spatial Transformer STNq

x: [1x28x28] $\mapsto$ y: [1x28x28]

6 Hierarchical Omniglot Architecture

We extend the same architecture described in Appendix B of (**?**), with only a simple modification: we introduce a new latent layer containing a 64-dimensional variable a, with a Gaussian prior. We give $p(c|a)$ the same functional form as $p(z|c)$, and give $q(a|D^a)$ the same functional form as $q(c; D^c)$ using the shared encoder.

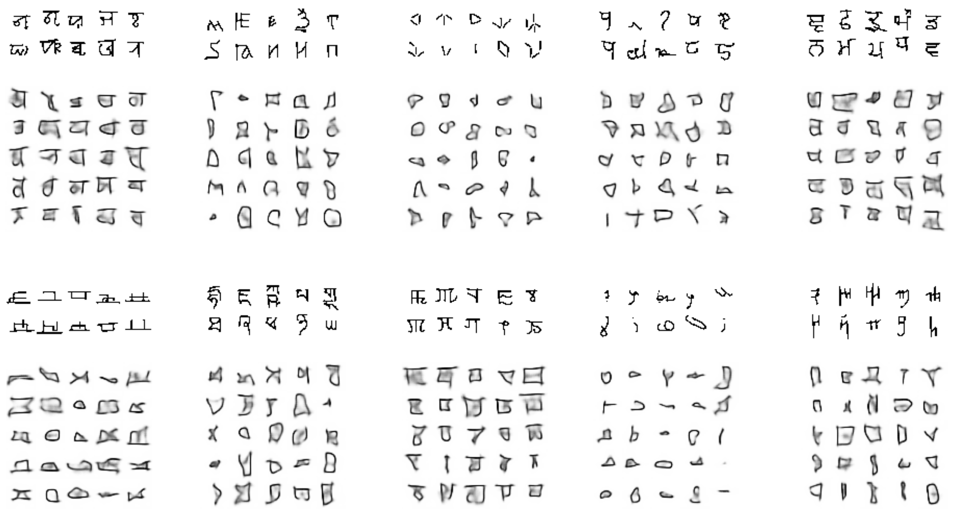


Figure 2: 10-shot alphabet generation samples from the hierarchical model.

7 Conditional Samples on Faces Dataset

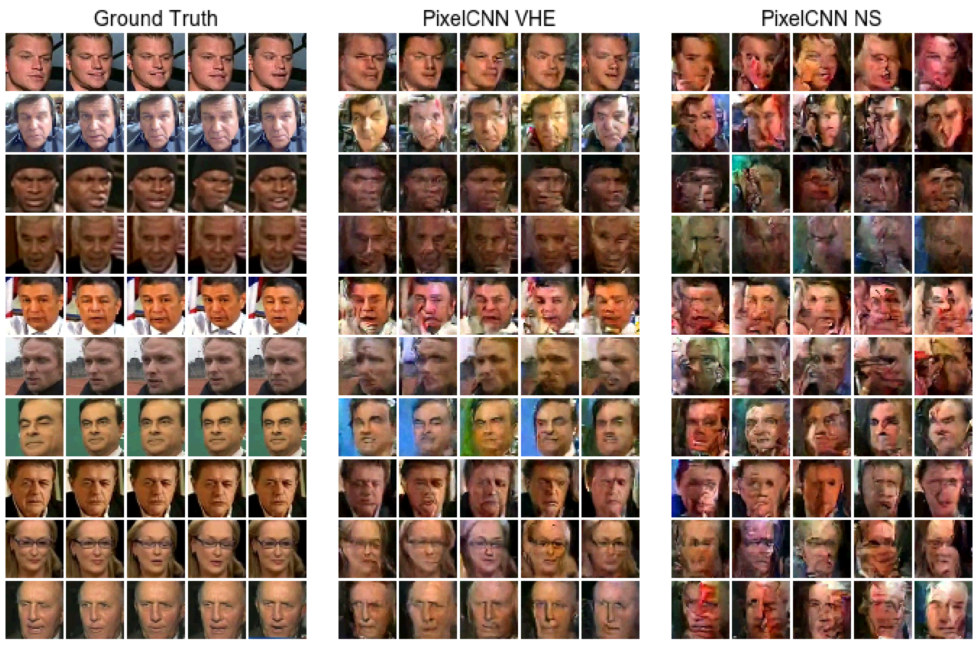


Figure 3: 5-shot samples of YouTube faces generated using PixelCNN architectures

8 Conditional Samples on Silhouettes Dataset

We created a VHE using the same deconvolutional architecture as applied to omniglot, and trained it on the Caltech-101 Silhouettes dataset. 10 object classes were held out as test data, which we use to generate both 1-shot and 5-shot conditional samples.