Hierarchical Few-Shot Generative Models

Giorgio Giannone
Section for Cognitive Systems
Technical University of Denmark
gigi@dtu.dk

Ole Winther
Section for Cognitive Systems
Technical University of Denmark
Department of Biology and Rigshospitalet
University of Copenhagen
olwi@dtu.dk

Abstract

A few-shot generative model should be able to generate data from a distribution by only observing a limited set of examples. In few-shot learning the model is trained on data from many sets from different distributions sharing some underlying properties such as sets of characters from different alphabets or sets of images of different type objects. We study a latent variables approach that extends the Neural Statistician [8] to a fully hierarchical approach with an attention-based point to set-level aggregation. We extend the previous work to iterative data sampling, likelihood-based model comparison, and adaptation-free out of distribution generalization. Our results show that the hierarchical formulation better captures the intrinsic variability within the sets in the small data regime. With this work we generalize deep latent variable approaches to few-shot learning, taking a step towards large-scale few-shot generation with a formulation that readily can work with current state-of-the-art deep generative models.

1 Introduction

Humans are exceptional few-shot learners able to grasp concepts and function of objects never encountered before [23]. This is because we build internal models of the world so we can combine our prior knowledge about object appearance and function to make well educated inferences from very little data [47, 25, 49]. In contrast, traditional machine learning systems have to be trained tabula rasa and therefore need orders of magnitude more data. In the landmark paper on modern few-shot learning Lake et al. [24] demonstrated how hand-written symbols from different alphabets can be distinguished one-shot, i.e. when a letter is shown for the first time.

Few-shot learning [53, 43, 10] and related approaches aiming at learning from little labelled data at test time (meta and transfer learning [19]) have recently gained new interest thanks to modeling advances, availability of large diverse datasets and computational resources. Building efficient learning systems that can adapt at inference time is a prerequisite to deploy such systems in realistic settings. Much attention has been devoted to *supervised few-shot learning*. The problem is typically cast in terms of an adaptive conditional task, where a small support set is used to condition explicitly [11] or implicitly [10] a learner, with the goal to perform well on a query set. The high-level idea is to train the model with a large number of small sets, and inject in the model the capacity to adapt to new concepts from few-samples at test time.

Comparatively less work has been developed on *few-shot adaptation in generative models* [8, 36, 2]. This is partially because of the challenging nature of learning joint distributions in an unsupervised way from few-samples and difficulties in evaluating such models. Few-shot generation has been limited to simple tasks, shallow and handcrafted conditioning mechanisms and as pretraining for supervised few-shot learning. Consequently there is a lack of quantitative evaluation and progress in the field.

5th Workshop on Meta-Learning at NeurIPS 2021, held virtually.

In this work we aim to solve these issues for few-shot generation in latent variable models. This class of models is promising because provides a principle way to include adaptive conditioning using latent variables. The setting we consider is that of learning from a large quantity of homogeneous sets, where each set is an un-ordered collections of samples of one concept or class. At test time, the model will be provided with sets of concepts never encountered during training and sets of different cardinality. We consider explicit conditioning in a hierarchical bayesian formulation, where a global latent variable carries information about the set at hand. A pooling mechanism aggregates information from the set using a hard-coded (mean, max) or learned (attention) operator. The conditioning mechanism is implemented in a shallow or hierarchical way: the hierarchical approach helps to learn better representations for the input set and gradually merges global and local information between the aggregated set and samples in the set. To handle input sets of any size the mechanism has to be permutation invariant and non-parametric. Conditional hierarchical model can naturally represent families of distributions where each conditioning set-level latent variable defines a different generative model. Learning a full distribution over the input set increases the flexibility of the model. Our contributions are the following:

1. We study latent variable models in the few-shot generation scenario. We perform **quantitative evaluation** of previous proposed methods.
2. We explore forward and iterative **sampling strategies** for the marginal and the predictive distributions implicitly defined by few-shot generative models.
3. We organize previously proposed latent variable models in the general class of hierarchical few-shot generative models, increasing the input set expressivity through convolutions, a **learnable aggregation** mechanism and **hierarchical inference**.

The paper is organized as follow: in Section 2 we summarize the Neural Statistician (NS). In Section 3 we extend the NS proposing a new class of hierarchical models - Hierarchical Few-Shot Generative Models (HFSGM). In this section we also discuss sampling strategies and aggregation mechanisms for the input set. In Section 4 we present empirical results on benchmark datasets, focusing on I) few-shot generative capacities varying the input set cardinality, II) strategies for unconditional, conditional and refined sampling and III) transfer properties on datasets of increasing complexity. In Section 5 we survey related work and in Section 6 we conclude this work.

2 Neural Statistician

In this section we present the modeling background for the proposed few-shot generative models. The Neural Statistician (NS, [8]) is a latent variable model for few-shot learning. Based on this model, many other approaches have been developed [12, 2]. The NS is a hierarchical model where two collections of latent variable are learned: a *task-specific* summary statistic c with prior $p(c)$ and a *per-sample* latent variable $\mathbf{z}$ with prior $p(\mathbf{z}|c)$:

$$p(X) = \int p(c) \prod_{s=1}^{S} \left[\int p(x_s|\mathbf{z}_s, c) p(\mathbf{z}_s|c) d\mathbf{z}_s \right] dc \,, \tag{1}$$

where $X = \{x_1, \ldots, x_S\}$ assuming the data set size is S and $p(\mathbf{z}|c)$ in general is a hierarchy of latent variables [44, 29]: $p(\mathbf{z}|c) = p(z_L|c) \prod_{l=1}^{L-1} p(z_l|z_{l+1}, c)$ with $\mathbf{z} = \{z_1, \ldots, z_L\}$.
In the original NS model, the authors factorize the lower-bound wrt c and $\mathbf{z}$ as:

$$q(c, \mathbf{Z}|X) = q(c|X) \prod_{s=1}^{S} q(\mathbf{z}_s|c, x_s) \,, \tag{2}$$

where $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_S\}$ and $q(\mathbf{z}_s|x_s, \mathbf{c})$ is in general a hierarchy of latent variables. In the NS the moments of the conditioning distribution over c are computed using a simple sum/average based formulation $r = \sum_{s=1}^{S} h(x_s)$,; and then r is used to condition $q(c|r)$. This idea is simple and straightforward, enabling efficient learning and a clean lower-bound factorization. But it has limitations: I) with such formulation the model expressivity and capacity to extract information from the set are limited. II) The invariance of $q(c|X)$ with respect set permutation is achieved by simple aggregation. The pooling mechanism assumes strong homogeneity in the context set and the generative process. However the model formulation allows more advanced invariant aggregations based on attention [51] and graph approaches [52]).

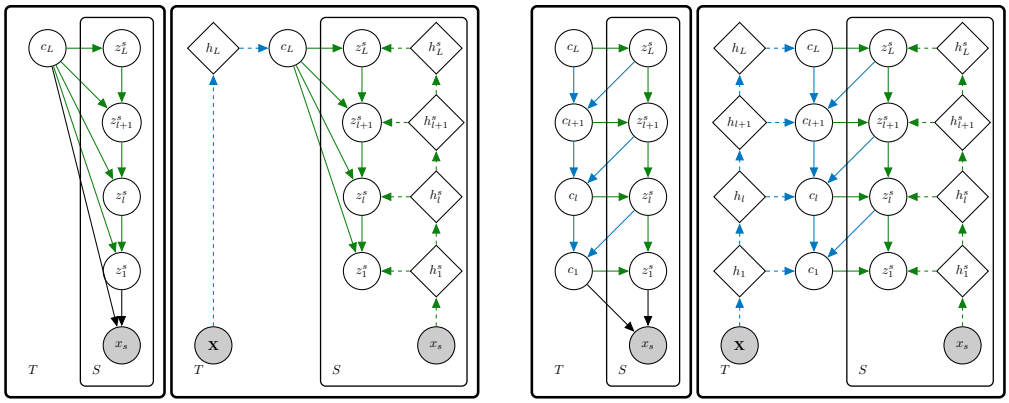

Figure 1: Generation and inference for a Neural Statistician (left) and a Hierarchical Few-Shot Generative Model (right). The generative model is composed by two collections of hierarchical latent variables, $\mathbf{c}$ for the sets $X = \{x_s\}_{s=1}^{S}$ and $\mathbf{z}_s$ for the samples x_s. The generative process is repeated S times and the full model is run on T different sets or tasks. The two variables are learned jointly, increasing expressivity and improving sampling. The generative and inference models can share parameters.

In the general case we want to be able to learn more expressive few-shot generative models able to deal with variety and complexity in the conditioning set. To go beyond these models in the next section we propose: **I)** a learnable non-parametric representation for the context set; and **II)** a hierarchical merging of information between conditioning $\mathbf{c}$ and sample $\mathbf{z}$ representations.

3 Hierarchical Few-Shot Generative Models

Notation.

Top Prior $\mathbf{c}$:	$p_\theta(c_L)$	Top Posterior $\mathbf{c}$:	$q_\phi(c_L \mid X)$
Top Prior $\mathbf{z}$:	$p_\theta(z_L \mid c_L)$	Top Posterior $\mathbf{z}$:	$q_\phi(z_L \mid c_L, x)$
Prior $\mathbf{c}$:	$p_\theta(c_l \mid c_{l+1}, Z_{l+1})$	Posterior $\mathbf{c}$:	$q_\phi(c_l \mid c_{l+1}, Z_{l+1}, X)$
Prior $\mathbf{z}$:	$p_\theta(z_l \mid z_{l+1}, c_l)$	Posterior $\mathbf{z}$:	$q_\phi(z_l \mid z_{l+1}, c_l, x)$
Observation $\mathbf{x}$:	$p_\theta(x \mid z_{1:L}, c_{1:L})$	Set Representation:	h_l

Our goal is to learn a generative model over sets, i.e. unordered collections of observations, able to generalize to new datasets given few samples. In this model, $\mathbf{c}$ is a collection of latent variables that represent a set X. We learn a posterior over $\mathbf{z}$ conditioned on $\mathbf{c}$, able to generate samples accordingly to queries x_s. The model has to be expressive: I) *hierarchical* - to increase the functions that the model can represent and improve the joint merging of set-level information $\mathbf{c}$ and sample information $\mathbf{z}$, and II) *non-parametric* - to handle input sets of any size and complexity, improving the way the model extracts and organizes information in the conditioning set. A fundamental difference between our proposal and previous models is the intrinsically hierarchical inference procedure over $\mathbf{c}$ and $\mathbf{z}$.

Generative Model. The generative model factorizes the joint distribution $p(X, \mathbf{Z}, \mathbf{c})$. In particular can be written as:

$$p(X|\mathbf{Z}, \mathbf{c})p(c_L)\left[p(Z_L|c_L)\prod_{l=1}^{L-1} p(Z_l, c_l|Z_{l+1}, c_{l+1})\right] \tag{3}$$

where $Z_l = \{z_l^1, \ldots, z_l^S\}$ are latent for layer l in the model and sample s in the input set $X = \{x_s\}_{s=1}^{S}$; c_l is a latent for layer l in the model and the input set X. We factorize the likelihood and prior terms over the set as:

$$p(X|\mathbf{Z}, \mathbf{c}) = \prod_{s=1}^{S} p(x_s|\mathbf{z}_s, \mathbf{c})$$

$$p(Z_l, c_l|Z_{l+1}, c_{l+1}) = \prod_{s=1}^{S} p(z_l^s|z_{l+1}^s, c_l)\; p(c_l|c_{l+1}, Z_{l+1}).$$

In this formulation the context $p(c_l|c_{l+1}, Z_{l+1})$ is a distribution of the previous context representation c_{l+1} and the previous latent representation for the samples Z_{l+1} as illustrated in Figure 1.

Approximate Inference. Learning in the model is achieved by amortized variational inference [20, 18]. The hierarchical formulation leverages structured mean-field approximation to increase inference flexibility. The approximate posterior parameterizes a joint distribution between latent representation for context and samples. We factorize $q(\mathbf{c}, \mathbf{Z}|X)$ following the generative model:

$$q(c_L|X)\left[q(Z_L|c_L, X)\prod_{l=1}^{L-1} q(Z_l, c_l|Z_{l+1}, c_{l+1}, X)\right] \tag{4}$$

where we factorize the posterior using a top-down inference formulation [44], merging top-down stochastic inference with bottom-up deterministic inference from the data. We factorize the posterior terms over the set as:

$$q(Z_l, c_l|Z_{l+1}, c_{l+1}, X) = \prod_{s=1}^{S} q(z_l^s|z_{l+1}^s, c_l, x_s)\; q(c_l|c_{l+1}, Z_{l+1}, X).$$

Lower-bound. In latent variable models [37, 22] we maximize a *per-sample* lower-bound over a large dataset. In few-shot generative models we maximize a lower-bound over a large collection of *small* sets. This detail is important because, even if the dataset of sets is iid by construction, learning in the few-shot scenario relies explicitly on common structure within a small set. For example, each small set can be an unordered collection of observations for a specific class or concept, like a character or face; and we rely on common structure between these observations using aggregation and conditioning on $\mathbf{c}$. The variational lower bound for $\log p(X)$ is obtained using the variational distribution $q = q(\mathbf{c}, \mathbf{Z}|X)$:

$$\begin{aligned}\log p(X) \geq\; & \mathbb{E}_q\left[\sum_{s=1}^{S}\log p(x_s|\mathbf{z}_s, \mathbf{c})\right] + \\ & \mathbb{E}_q\left[\sum_{l=1}^{L-1}\sum_{s=1}^{S}\log\frac{p(z_l^s|z_{l+1}^s, c_l)}{q(z_l^s|z_{l+1}^s, c_l, x_s)}\right] + \mathbb{E}_q\left[\sum_{s=1}^{S}\log\frac{p(z_L^s|c_L)}{q(z_L^s|c_L, x_s)}\right] + \\ & \mathbb{E}_q\left[\sum_{l=1}^{L-1}\log\frac{p(c_l|c_{l+1}, Z_{l+1})}{q(c_l|c_{l+1}, Z_{l+1}, X)}\right] - \mathbb{KL}(q(c_L|X), p(c_L))\;.\end{aligned} \tag{5}$$

The lower-bound can be split in three main components: an expected log likelihood term, divergences over $\mathbf{Z}$ and over $\mathbf{c}$. The final per-sample loss for T sets of size S then is $\mathcal{L} = \mathbb{E}_{p(\mathcal{X}_S)} l(X) = 1/N \sum_{t=1}^{T} l(X_t)$, where $l(X_t)$ is the negated lower-bound for set X_t and $N = T\,S$.

3.1 Sampling

We may be interested in either sampling unconditionally $x \sim p(x)$ or from the predictive distribution $x \sim p(x|X)$. Unconditional sampling may be performed exactly using the generative model as illustrated in Figure 1: first sample the hierarchical prior $\mathbf{z}, \mathbf{c} \sim p(\mathbf{z}, \mathbf{c})$ and then sample the likelihood $x \sim p(x|\mathbf{z}, \mathbf{c})$. Conditional sampling $x \sim p(x|X)$ can be done approximately using the variational posterior $q(\mathbf{Z}, \mathbf{c}|X)$ as a replacement for the exact posterior $p(\mathbf{Z}, \mathbf{c}|X)$ as outlined in Algorithm 1. In the single pass approach (adapted from the NS) a sample from:

$$\int p(x|\mathbf{z}, \mathbf{c})p(\mathbf{z}, \mathbf{c}_{<L}|c_L)q(c_L|X)d\mathbf{z}d\mathbf{c} \approx p(x|X) \tag{6}$$

is generated. This approach is not ideal because $\mathbf{c}_{<L}$ is only modeled jointly with the latent $\mathbf{z}$ for the new sample while omitting the latent $\mathbf{Z}$ for X. We can introduce this dependence in a Markov chain approach adapted from the missing data imputation framework proposed in Appendix F of [37]. We augment

Algorithm 1: Sampling

Data: Input set X;
single pass
$c_L \sim q(c_L|X)$;
$\mathbf{z}, \mathbf{c}_{<L} \sim p(\mathbf{z}, \mathbf{c}_{<L}|c_L)$;
$x \sim p(x|\mathbf{z}, \mathbf{c})$;
repeat
  $\tilde{X} = [X, x]$;
  $\tilde{Z}, \mathbf{c} \sim q(\tilde{Z}, \mathbf{c}|\tilde{X})$;
  $\tilde{X}' \sim p(\tilde{X}'|\tilde{Z}, \mathbf{c})$;
  $x = x'$;
until converged;
return x;

X with the sample x generated in the single pass approach: $\tilde{X} = [X, x]$. From this we can construct a distribution $\hat{p}(\tilde{Z}, \mathbf{c}|\tilde{X})$, where $\tilde{Z} = [Z, z]$ is the corresponding augmented latent. From this distribution and the likelihood we can construct a transition kernel:

$$\hat{p}(x'|x, X) = \int p(\tilde{X}'|\tilde{Z}, \mathbf{c})\hat{p}(\tilde{Z}, \mathbf{c}|\tilde{X})dX'd\tilde{Z}d\mathbf{c} \, .$$

If $\hat{p}(\tilde{Z}, \mathbf{c}|\tilde{X}) = p(\tilde{Z}, \mathbf{c}|\tilde{X})$ then we can show that $p(x|X)$ is an eigen-distribution for the transition kernel and thus sampling the transition kernel will under mild conditions converge to a sample from $p(x|X)$. There are several ways to construct $\hat{p}(\tilde{Z}, \mathbf{c}|\tilde{X})$ from the hierarchical variational and prior distributions. One possibility is shown in Algorithm 1. Alternatively, one may sample $c_L, \tilde{Z}_L \sim q(c_L, \tilde{Z}_L|\tilde{X})$, generate the remainder of the latent from the prior hierarchy and new samples X', x' from the likelihood. If the variational is exact, these approaches are equivalent and exact. In the experimental section we report results for the different approaches.

3.2 Learnable aggregation (LAG)

A central idea in few-shot generative models is to condition the generative model with a permutation invariant representation of the input set. For a NS such operator is a hard-coded per-set statistic. This approach [8, 12] maps each sample in X independently using $\{h(x_s)\}_{s=1}^S$ and then aggregating $r_L = \sum_{s=1}^S h(x_s)$ to generate the moments for $q(c_L|r_L)$. This idea is simple and effective when using homogeneous and small sets for conditioning. Another choice is a relation network [46, 39] followed by aggregation. However, the adaptation capacities of the model are a function of how we represent the input set and a more expressive learnable aggregation mechanism can be useful. In the general scenario, a few-shot generative model should be able to extract information from any conditioning set X in terms of variety and size. In this paper we consider a multi-head attention-based learnable aggregation (LAG) inspired by [27] that can be used in each block of the hierarchy over c (Figure 3). Using LAG we can account for statistical dependencies in the set, handle variety between samples in the set and generalize better to input set dimensions. In the experimental section we provide extensive empirical analysis of how using LAG improves the generative and transfer capacity of the model.

Algorithm 2: LAG

Data: Input set $X = \{x_s\}_{s=1}^S$;
compute
$\{h(x_s)\}_{s=1}^S$;
$r = 1/S \sum_{s=1}^S h(x_s)$;
$\alpha(r, x_s) \propto \text{sim}(q(r), k(h_s))$;
aggregate
$r_{\text{LAG}} = \sum_{s=1}^S \alpha(r, x_s)\, v(h_s)$;
sample
$c \sim q(c|r_{LAG})$;

4 Experiments

In this section we discuss experimental setup and results for our model. In particular for all the models our interests are: **I)** Quantitative evaluation of few-shot generative capacities. **II)** Conditional sampling from the model. **III)** Transfer of generative capacities to new datasets and input set size.

We perform experiments on binarized Omniglot [24] and CelebA [28]. We perform quantitative evaluation on Omniglot, MNIST [26], DOUBLE-MNIST [45], TRIPLE-MNIST [45] and CelebA. We follow the approach proposed in [8] for set creation. We create a large collection of small sets, where each set contains all the occurrences for a specific class or concept in a dataset: a character for Omniglot; the face of an identity for CelebA. Both datasets contain thousands of characters/identities with an average number of 20 occurrences per class. For this reason they are a natural choice for few-shot generation. Then we split the sets in train/val/test sets. Using these splits we can dynamically create sets of different dimensions (2-5-10 samples per set from the same class), generating a new collection of training sets at each epoch during training. For training we use the episodic approach proposed in [53]. Our architecture is a close approximation of [8]. We describe relevant details in Appendix C. We use a VAE - which does not explore set information - and the Neural Statistician (NS) as baselines. We propose two main model variants. Such variants are characterized by different design choices for the conditioning mechanism. A Convolutional Neural Statistician (CNS) where the latent space is shaped with convolutions at a given resolution; and a Convolutional Hierarchical Few-shot Generative Model (CHFSGM) where an additional hierarchy over $\mathbf{c}$ is employed. For all the models we consider standard aggregation mechanism (MEAN, MAX pooling) and learnable ones (LAG). Each input set is a homogeneous (one concept or class) collection of samples. During training the input set size is always between 2 and 10.

Generative. With these experiments we test the generalization properties of the model on few-shot generation. We explore the behavior of the model increasing the input size. Evaluation of generative properties is performed using the lower-bound and approximating the log-marginal likelihood with 1000 importance samples. In Table 1 we compare generalization in Omniglot on disjoint classes with a set size of 5. Our focus is on improving the model through $\mathbf{c}$. In Table 1, NS-MEAN is the original

Table 1: Generalization on disjoint Omniglot classes trained on set size 5 for a VAE, NSs with mean/max/learnable aggregation (MEAN/MAX/LAG) convolutional variants (C) and for a HFSGM with a hierarchy over c. We report minus the 1-sample lower-bound (NELBO), minus expected log likelihood (minus term 1 in Equation (5)), $\mathbf{z}$ KL terms (minus term 2 and 3), $\mathbf{c}$ KL terms (minus term 4 and 5) and minus the 1k importance sample lower bound (MLL).

MODEL	AGG	NELBO	NLL	KLz	KLc	MLL	PARAMS (M)
VAE	-	102.54	68.49	34.05	-	97.47	7.5
NS	MAX	97.52	66.82	26.29	4.41	90.23	7.9
NS	MEAN	97.52	67.04	25.75	4.73	90.21	7.9
NS	MEAN	96.28	65.64	25.95	4.87	90.07	14.9
CNS	MEAN	93.71	59.83	28.97	5.22	88.15	7.3
CNS (ours)	LAG	93.24	58.97	29.23	5.04	88.21	7.4
CHFSGM (ours)	MEAN	**92.58**	59.04	28.72	4.81	**87.37**	9.2
CHFSGM (ours)	LAG	**92.59**	**58.14**	30.00	4.44	**87.43**	9.5

NS with mean aggregation. The NS-MAX is the same model with max aggregation. NS-LAG uses an attention-based learnable aggregation, that builds an adaptive aggregation mechanism for each set. Then we report results for a hierarchical formulation over $\mathbf{c}$ with CHFSGM and we use the same two aggregation mechanisms. The proposed methods improve perform in terms of likelihood and reconstruction with a gain in performance for the learnable aggregation mechanism and the hierarchical formulation for $\mathbf{c}$. In Figure 2 we show the models behavior increasing the input set

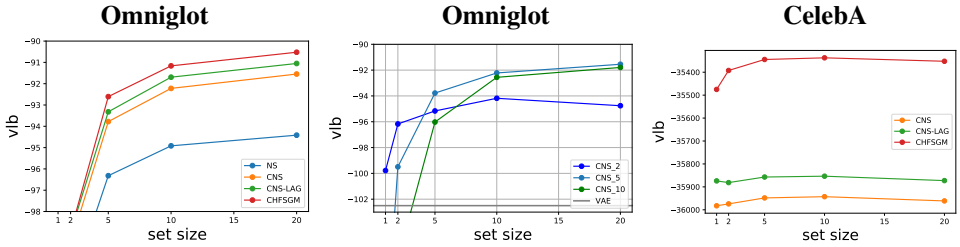

Figure 2: Set Cardinality. All models are trained with homogeneous (one concept) sets.
(Left): Lower-bounds for models trained with input set size 5 varying the test set cardinality from 1 to 20 on Omniglot. All the models improve the generative properties increasing the input set size. We see how the convolutional latent space is fundamental for performance improvement, increasing the expressivity of the model. Adding learnable attention and hierarchical inference over $\mathbf{c}$ improves the generative metrics in a monotonic way, showing how our proposed models can effectively adapt to different input set size.
(Center): Lower-bounds for models trained with varying input set size (2, 5, 10). We notice how training a CNS with input set size 2 gives better performance than a VAE for size 1 and tends to plateau and slightly decrease performance for larger context (10-20). The model trained with input set size 10 under-performs for small context size. The model trained with input set 5 gives us the best balance between expressivity and adaptability to different input set size.
(Right): Lower-bounds for models trained with input set size 5 varying the test set cardinality from 1 to 20 on CelebA. All the models struggle with set size larger than 10 because of the large variety in each set (age, perspective, general look). We can see that learnable aggregation and hierarchical inference both help in modeling the dataset.

cardinality at test time. All the models are trained with input sets of size 5 on Omniglot. Then at test time we vary the size of the context set between 1 and 20. All the models improve performance in terms of ELBO (left plot) and learn a better posterior for $\mathbf{c}$ with $\mathrm{KL}(\mathbf{c})$ (right plot). For small input sets (1-2 samples) a VAE baseline performs better because there is no enough information to aggregate in $\mathbf{c}$. Increasing the set (2-20), the NS-style models can aggregate more and more information learning a better model for the data. This is the desired behavior for a model learning

Figure 4: Sampling. Different ways to sample the model.
(Top): Refined samples obtained using Algorithm 1. Given a small set from an unknown character (right on black background), we sample the model and then refine iteratively using the inference model. We show 20 iterations from left to right. We can see how the generative process refines its guess at each iteration improving $\mathbf{c}$ and z in a joint manner.
(Middle and Bottom): Stochastic reconstruction, input sets, conditional sampling using the NS approach, conditional sampling using refinement, and unconditional sampling (sometimes referred to as imagination). The models are trained on subsets of Omniglot and CelebA and tested on disjoint identities.

from sets and more generally from permutation invariant data. All the aggregation mechanisms (hard-coded and learned in a shallow or hierarchical fashion) respects this property and can exploit additional data in input.

Sampling. In latent variable models like VAEs we can only perform unconditional sampling. In a NS we have different ways of sampling as explained in Section 3.1. In particular two main sampling approaches can be used: I) conditional sampling, where we sample the predictive distribution $p(x|X)$ relying on the inference model. II) unconditional sampling, sampling $p(X)$ through $c \sim p(c)$ and then $p(z|c)$. In Figure 4 we show (from left to right) stochastic reconstructions, input sets, samples obtained sampling like in a NS, refined samples, and unconditional samples. For simple characters there is almost no difference between conditional and refinement sampling. However when the model is challenged with a new complex character (third and ninth row) the refinement procedure greatly improves the adaptation capacities and visual quality of the generated samples. In the rightmost column we see fully unconditional samples. The model, given a set representation $\mathbf{c}$, generates consistent symbols, corroborating the assumption that the model

Figure 3: Attention. Sample formation in a Convolutional Neural Statistician with learnable aggregation. For each sample (black background) we plot the attention bars over the input sets (in white background) for four different heads.

learns a different distribution for each $\mathbf{c}$, greatly increasing the model flexibility and representation capacities.

Transfer. With these experiments we explore few-shot generation in the context of transfer learning. We use the same models we trained on Omniglot and we test on unseen classes in a different dataset. We use MNIST test set (10 classes), DOUBLE-MNIST test set (20 classes) and TRIPLE-MNIST test set (200 classes). The datasets increase in complexity and in size. We expect relatively good performance on the simple one and worse performance on the more complex one. We resize all the datasets to 28x28 pixels using BOX resizing. In Table 2 we report likelihoods for input set 5 on the three datasets. Our models perform better on all three datasets. Attention-based aggregation is essential for good performance on few-shot transfer. The same general behavior can be seen in Figure 5 where we explore transfer increasing the input set size. Again our models perform better than the baselines and attention-based aggregation is important for good performance on concepts from different datasets. In Appendix B, we report out of distribution classification performance

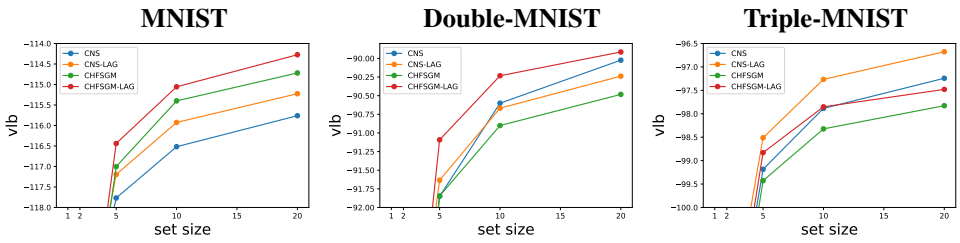

Figure 5: Transfer. Model trained on Omniglot with set size 5 and tested on MNIST, DOUBLE-MNIST and TRIPLE-MNIST (from left to right) with different set size. We can see how our models perform better than a plain CNS. In particular models with learnable aggregation (LAG) can adapt better to the new datasets.

Table 2: Transfer. We test the model transfer capacities on scenarios of increasing complexity, using a subset of disjoint classes from simple out-distribution on MNIST and more challenging out-distribution generalization on MNIST variants with 20 and 200 classes. Lower is better. Models trained on Omniglot with input set size 5.

MODEL	AGG	MNIST	DOUBLE-MNIST	TRIPLE-MNIST
VAE	-	124.71	98.13	104.75
NS	MEAN	119.92	96.28	100.71
CNS	MEAN	115.76	91.85	97.24
CNS (ours)	LAG	115.22	91.63	**96.68**
CHFSGM (ours)	MEAN	114.72	91.84	97.83
CHFSGM (ours)	LAG	**114.27**	**91.09**	97.48

for the three approaches described in Section 3. The models are trained on Omniglot for context size 5 and tested on MNIST also for context size 5 (so in total 50 data points) without adapting the distributions to the MNIST data.

5 Conclusion

Leveraging recent advances in deep latent variable models, we propose a new class of hierarchical latent variable models for few-shot generation. We ground our formulation in hierarchical inference and a learnable non-parametric aggregation. We show how simple hierarchical inference is a viable adaptation strategy. We perform extensive empirical evaluation in terms of generative metrics, sampling capacities and transfer properties. The proposed formulation is completely general and we expect there is large potential for improving performance by combining it with state of the art VAE architectures. Since benchmarks often come with grouping information, using a hierarchical formulation is a generic approach to improve generative capabilities.

Acknowledgement

We would like to thank Marco Ciccone, Andrea Dittadi, Pierluca D'Oro, Søren Hauberg, Valentin Liévin, Didrik Nielsen, Timon Willi, Max Wilson for insightful comments and useful discussions.

References

[1] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*, pages 3981–3989, 2016.

[2] S. Bartunov and D. Vetrov. Few-shot generative modelling with generative matching networks. In *International Conference on Artificial Intelligence and Statistics*, pages 670–678, 2018.

[3] B. Bloem-Reddy and Y. W. Teh. Probabilistic symmetry and invariant neural networks. *arXiv preprint arXiv:1901.06082*, 2019.

[4] B. Bloem-Reddy and Y. W. Teh. Probabilistic symmetries and invariant neural networks. *Journal of Machine Learning Research*, 21(90):1–61, 2020.

[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[6] R. Child. Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

[7] B. De Finetti. 9. on the condition of partial exchangeability. In *Studies in Inductive Logic and Probability Volume 2*, pages 193–206. University of California Press, 2020.

[8] H. Edwards and A. Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.

[9] S. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, et al. Neural scene representation and rendering. *Science*, 360 (6394):1204–1210, 2018.

[10] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[11] M. Garnelo, D. Rosenbaum, C. J. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. J. Rezende, and S. Eslami. Conditional neural processes. *arXiv preprint arXiv:1807.01613*, 2018.

[12] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

[13] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.

[14] A. Graves, J. Menick, and A. van den Oord. Associative compression networks for representation learning. *CoRR*, abs/1804.02476, 2018. URL `http://arxiv.org/abs/1804.02476`.

[15] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[16] L. B. Hewitt, M. I. Nye, A. Gane, T. Jaakkola, and J. B. Tenenbaum. The variational homoencoder: Learning to learn high capacity generative models from few examples. *arXiv preprint arXiv:1807.08919*, 2018.

[17] S. Hochreiter, A. S. Younger, and P. R. Conwell. Learning to learn using gradient descent. In *International Conference on Artificial Neural Networks*, pages 87–94. Springer, 2001.

[18] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[19] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[20] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[21] H. Kim, A. Mnih, J. Schwarz, M. Garnelo, A. Eslami, D. Rosenbaum, O. Vinyals, and Y. W. Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019.

[22] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[23] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the annual meeting of the cognitive science society*, volume 33, 2011.

[24] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[25] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

[26] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[27] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753. PMLR, 2019.

[28] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[29] L. Maaløe, M. Fraccaro, V. Liévin, and O. Winther. Biva: A very deep hierarchy of latent variables for generative modeling. *arXiv preprint arXiv:1902.02102*, 2019.

[30] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

[31] B. Oreshkin, P. Rodríguez López, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in Neural Information Processing Systems*, 31:721–731, 2018.

[32] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. *arXiv preprint arXiv:1709.07871*, 2017.

[33] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[34] S. Ravi and A. Beatson. Amortized bayesian meta-learning. In *International Conference on Learning Representations*, 2018.

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

[36] S. Reed, Y. Chen, T. Paine, A. v. d. Oord, S. Eslami, D. Rezende, O. Vinyals, and N. de Freitas. Few-shot autoregressive density estimation: Towards learning to learn distributions. *arXiv preprint arXiv:1710.10304*, 2017.

[37] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[38] D. J. Rezende, S. Mohamed, I. Danihelka, K. Gregor, and D. Wierstra. One-shot generalization in deep generative models. *arXiv preprint arXiv:1603.05106*, 2016.

[39] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018.

[40] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

[41] J. Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[42] J. Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879, 1992.

[43] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

[44] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pages 3738–3746, 2016.

[45] S.-H. Sun. Multi-digit mnist for few-shot learning, 2019. URL `https://github.com/shaohua0116/MultiDigitMNIST`.

[46] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[47] J. B. Tenenbaum. *A Bayesian framework for concept learning*. PhD thesis, Massachusetts Institute of Technology, 1999.

[48] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[49] T. D. Ullman and J. B. Tenenbaum. Bayesian models of conceptual development: Learning as building models of the world. *Annual Review of Developmental Psychology*, 2:533–558, 2020.

[50] A. Vahdat and J. Kautz. NVAE: A deep hierarchical variational autoencoder. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[52] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 1(2), 2017.

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

[54] M. Wu, K. Choi, N. D. Goodman, and S. Ermon. Meta-amortized variational inference and learning. In *AAAI*, pages 6404–6412, 2020.

[55] J. Xu, J.-F. Ton, H. Kim, A. R. Kosiorek, and Y. W. Teh. Metafun: Meta-learning with iterative functional updates. *arXiv preprint arXiv:1912.02738*, 2019.

[56] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.

A Related Work

Learning from Sets. In recent years a large corpus of work studied the problem of learning from sets [56], and more generally learning in exchangeable deep models [7, 4, 3]. These models can be formulated in a variety of ways, but they all have in common a form of permutation invariant aggregation (or pooling mechanism) over the input set. Deep Sets [56] formalized the framework of exchangeable models. The Neural Statistician [8] was the first model proposing to learn from sets in the variational autoencoder framework and used a simple and effective mean pooling mechanism for aggregation. The authors explored the representation capacities of such model for clustering and few-shot supervised learning. Generative Query Networks performs neural rendering [9] where the problem of pooling views arises. The Neural process family [12, 21], where a set of point is used to learn a context set and solve downstream tasks like image completion and few-shot learning. Set Transformers [27] leverages attention to solve problems involving sets. PointNet [33] models point clouds as a set of points. Graph Attention Networks [52] aggregate information from related nodes using attention. Associate Compression Network [14] can be interpreted in this framework, where a prior for a VAE is learned using the top-knn retrieved in latent space. In this work we build on ideas and intuitions in these works, with a focus on generative models for sets.

Few-Shot Generative Models. Historically the machine learning community has focused its attention on supervised few-shot learning, solving a classification or regression task on new classes at test time given a small number of labeled examples. The problem can be tackled using metric based approaches [53, 43, 31], gradient-based adaptation [10], optimization [35] and more. More generally, the few-shot learning task can be recast as bayesian inference in hierarchical modelling [13, 34]. In such models, typically parameters or representation are conditioned on the task, and conditional predictors are learned for such task. In [55] an iterative attention mechanism is used to learn a query-dependent task representation for supervised few-shot learning. Modern few-shot generation in machine learning was introduced in [24]. The Neural Statistician [8] is one of the first few-shot learning models in the context of VAEs. However the authors focused on downstream tasks and not on generative modeling. The model has been improved further increasing expressivity for the conditional prior using powerful autoregressive models [16], a non-parametric formulation for the context [54] and exploiting supervision [12]. [38] proposed a recurrent and attentive sequential generative model for one-shot learning based on [15]. Powerful autoregressive decoders and gradient-based adaptation are employed in [36] for one-shot generation. The context $\mathbf{c}$ in this model is a deterministic variable. In GMN [2] a variational recurrent model learns a per-sample context-aware latent variable. Similar to our approach, GMN learns a non-parametric context, learning an attention based kernel that can handle generic datasets. However the context-aware representation scales linearly with the input size, there is no separation between global and local information in latent space, The input set is processed in an arbitrary autoregressive order, and not in a permutation invariant manner. They use recurrent models where we use hierarchical models. Finally, recent large-scale autoregressive language models [5] exhibit non-trivial few-shot capacities.

Adaptation as Inference. A large corpus of works has been proposed for learning to learn techniques and fast adaptation in multitask learning [41, 42, 17, 48, 1]. Currently gradient-based adaptation [10] dominates the field of meta-learning. As shown by [13], such approaches can be seen as maximum a posteriori inference in a hierarchical model, where the global parameters of a model are adapted by few-steps of gradient descent on a support set. In this setting, meta-learning is a local weight update around a global set of parameters. Following these ideas, [34] proposed a fully amortized bayesian formulation, where the posterior distribution is adapted by gradient-based optimization and outputs a distribution over per-task model parameters. It is not obvious that few-shot generalization can be achieved through hierarchical amortized inference. In this work we focus mostly on exploring adaptation through the lens of hierarchical inference using amortized variational Bayes [12]. However the problem can be approached in a completely different way, and in particular in terms of gradient-based weight updates [10]. The outer and inner loop present in such methods can be seen as the conditioning $q(\mathbf{c}|X)$ and hierarchical inference steps $q(\mathbf{z}|x, \mathbf{c})$ in our formulation. The parallel between hierarchical inference and gradient-based adaptation is clearer when we partition the task dataset in a support set for conditioning $q(\mathbf{c}|X_s)$ and a query set for inference and generation $q(\mathbf{Z}_q|X_q, \mathbf{c})$. Each inference layer in the hierarchy can be seen as changing the base conditional distribution. A relevant difference between the approaches is that we adapt representations and latent variables, where gradient-based adaptation is typically focused on the model parameters.

Code : `https://github.com/georgosgeorgos/hierarchical-few-shot-generative-models`
Page : `https://georgosgeorgos.github.io/hierarchical-few-shot-generative-models/`

Notation :

- **Bold** $\rightarrow$ **all layers**
- CAPITALIZED $\rightarrow$ ALL SAMPLES
  - **Bold** CAPITALIZED: **all layers**, ALL SAMPLES $\mathbf{Z} = \{z_l^s\}_{s=1,l=1}^{S,L}$.
  - **Bold**: **all layers**, one sample $\mathbf{z}_s = \{z_l^s\}_{l=1}^{L}$.
  - CAPITALIZED: one layer, ALL SAMPLES $Z_l = \{z_l^s\}_{s=1}^{S}$.

Top Prior $\mathbf{c}$:	$p_\theta(c_L)$	Top Posterior $\mathbf{c}$:	$q_\phi(c_L \mid X)$
Top Prior $\mathbf{z}$:	$p_\theta(z_L \mid c_L)$	Top Posterior $\mathbf{z}$:	$q_\phi(z_L \mid c_L, x)$
Prior $\mathbf{c}$:	$p_\theta(c_l \mid c_{l+1}, Z_{l+1})$	Posterior $\mathbf{c}$:	$q_\phi(c_l \mid c_{l+1}, Z_{l+1}, X)$
Prior $\mathbf{z}$:	$p_\theta(z_l \mid z_{l+1}, c_l)$	Posterior $\mathbf{z}$:	$q_\phi(z_l \mid z_{l+1}, c_l, x)$
Observation $\mathbf{x}$:	$p_\theta(x \mid z_{1:L}, c_{1:L})$	Set Representation:	h_l

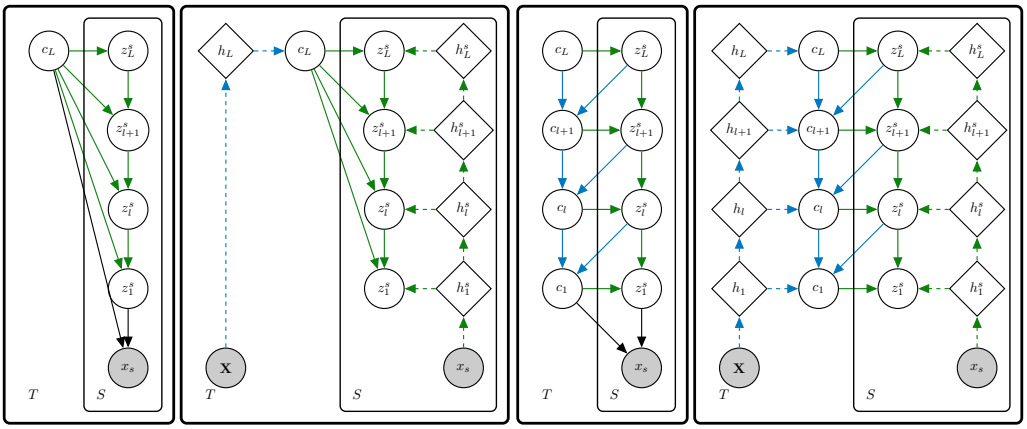

Figure 6: Generation and Inference for a Neural Statistician (left) and a Hierarchical Few-Shot Generative Model (right)

B Model Derivation

In this section we derive the generative model, inference model and lower-bound for a Basic Neural Statistician (bNS), a Neural Statistician (with hierarchy over z, this is the model used as baseline in the paper) (NS), and a Hierarchical Few-Shot Generative Model (HFSGM) with hierarchy over z and c. Doing so we can underline similarities and differences among the formulations.

Per-Set Marginal. Our goal is to model a distribution $p(X) = \prod_{s=1}^{S} p(x_s)$ where $X\{x_s\}_{s=1}^{S}$ is in general a small set ranging from 1 to 20 samples. We sample these sets from a common process. The Neural Statistician [8] introduces a global latent variable c for a set X:

$$p(X) = \int p(X, c)\, dc = \int p(c)\, p(X|c)\, dc = \int p(c) \prod_{s=1}^{S} p(x_s|c)\, dc. \tag{7}$$

Then we can introduce a per-sample latent variable z:

$$\begin{aligned} p(X) = \int p(c) \left[\int p(X, Z|c) dZ \right] dc &= \int p(c) \prod_{s=1}^{S} \left[\int p(x_s, z_s|c) dz_s \right] dc \\ &= \int p(c) \prod_{s=1}^{S} \left[\int p(x_s|z_s, c) p(z_s|c) dz_s \right] dc, \end{aligned} \tag{8}$$

where $X = \{x_s\}_{s=1}^{S}$ is a set of images, c is a latent variable for the set, $Z = \{z_s\}_{s=1}^{S}$ are latent variable for the samples in the set. The formula above is the basic marginal for all the NS-like model.

B.1 Generative Model

We can think of the model as composed of three components: a hierarchical prior over z, a hierarchical prior over c, and a render for x. For each hierarchical prior, the top distribution is not autoregressive and act as an unconditional prior for c and a conditional prior for z. In the following all the latent distributions are normal distributions with diagonal covariance. Given the hierarchical nature of our model, these assumptions are not restrictive, because from top to down in the hierarchy we build an expressive structured mean field approximation. The decoder is Bernoulli distributed for binary datasets. In the following equation we use $Z = \{z_s\}_{s=1}^{S}$, $\mathbf{Z} = \{\mathbf{z}_s\}_{s=1}^{S}$, $\mathbf{z}_s = \{z_l^s\}_{l=1}^{L}$, $\mathbf{c} = \{c_l\}_{l=1}^{L}$. Each of these equation can be written per-set or per-sample (similar to Eq.2 in the Appendix). We choose to write everything in a compact format writing per-set equations.

bNS. In this setting both z and c are shallow latent variables.

$$p(X, Z, c) = p(X|Z, c)p(c)\Big[p(Z|c)\Big] \tag{9}$$

NS. z is a hierarchy.

$$p(X, \mathbf{Z}, c) = p(X|\mathbf{Z}, c)p(c) \left[p(Z_L|c) \prod_{l=1}^{L-1} p(Z_l|Z_{l+1}, c) \right] \tag{10}$$

HFSGM. Both **z** and **c** are a hierarchy. We increase the model flexibility.
Using such hierarchy we can:

- learn a structured mean field approximation for c;
- jointly learn c and Z, informing different stages of the learning process;
- incrementally improve c through layers of inference.

$$\begin{aligned} p(X, \mathbf{Z}, \mathbf{c}) &= p(X|\mathbf{Z}, \mathbf{c})p(c_L) \left[p(Z_L|c_L) \prod_{l=1}^{L-1} p(Z_l, c_l|Z_{l+1}, c_{l+1}) \right] \\ p(Z_l, c_l|Z_{l+1}, c_{l+1}) &= p(Z_l|Z_{l+1}, c_l)\; p(c_l|c_{l+1}, Z_{l+1}). \end{aligned} \tag{11}$$

B.2 Inference Model

We learn the model using Amortized Variational Inference. In a NS-like model, inference is intrinsically hierarchical: the model encodes global set-level information in c using $q(c|X)$.

bNS.

$$q(c, Z|X) = q(c|X)\Big[q(Z|c, X)\Big] \tag{12}$$

NS.

$$q(c, \mathbf{Z}|X) = q(c|X)\Big[q(\mathbf{Z}|c, X)\Big] \tag{13}$$

HFSGM.

$$\begin{aligned} q(\mathbf{c}, \mathbf{Z}|X) &= q(c_L|X)\left[q(Z_L|c_L, X)\prod_{l=1}^{L-1} q(Z_l, c_l|Z_{l+1}, c_{l+1}, X)\right] \\ q(Z_l, c_l|Z_{l+1}, c_{l+1}, X) &= q(Z_l|Z_{l+1}, c_l, X)\; q(c_l|c_{l+1}, Z_{l+1}, X). \end{aligned} \tag{14}$$

B.3 Lower-bound

We learn the model by Amortized Variational Inference similarly to recent methods. However we are in presence of two different latent variables, and we need to lower-bound wrt both.

bNS.

$$\begin{aligned} \log p(X) \geq\; & \mathbb{E}_{q(c,Z|X)}\left[\log \frac{p(X, Z, c)}{q(c, Z|X)}\right] = \\ & \mathbb{E}_q\left[\sum_{s=1}^{S} \log p(x_s|z_s, c)\right] + \mathbb{E}_q\left[\sum_{s=1}^{S} \log \frac{p(z_s|c)}{q(z_s|c, x_s)}\right] - \mathbb{KL}(q(c|X), p(c)) = -\mathcal{L}(X). \end{aligned} \tag{15}$$

NS.

$$\begin{aligned} \log p(X) \geq\; & \mathbb{E}_{q(c,\mathbf{Z}|X)}\left[\log \frac{p(X, \mathbf{Z}, c)}{q(c, \mathbf{Z}|X)}\right] = \\ & \mathbb{E}_q\left[\sum_{s=1}^{S} \log p(x_s|\mathbf{z}_s, c)\right] + \mathbb{E}_q\left[\sum_{l=1}^{L-1}\sum_{s=1}^{S} \log \frac{p(z_l^s|z_{l+1}^s, c)}{q(z_l^s|z_{l+1}^s, c, x_s)}\right] + \mathbb{E}_q\left[\sum_{s=1}^{S} \log \frac{p(z_L|c)}{q(z_L|c, x_s)}\right] \\ & - \mathbb{KL}(q(c|X), p(c)) = -\mathcal{L}(X). \end{aligned} \tag{16}$$

HFSGM.

$$\begin{aligned} \log p(X) \geq\; & \mathbb{E}_{q(\mathbf{c},\mathbf{Z}|X)}\left[\log \frac{p(X, \mathbf{Z}, \mathbf{c})}{q(\mathbf{c}, \mathbf{Z}|X)}\right] = \\ & \mathbb{E}_q\left[\sum_{s=1}^{S} \log p(x_s|\mathbf{z}_s, \mathbf{c})\right] + \mathbb{E}_q\left[\sum_{l=1}^{L-1}\sum_{s=1}^{S} \log \frac{p(z_l^s|z_{l+1}^s, c_l)}{q(z_l^s|z_{l+1}^s, c_l, x_s)}\right] + \mathbb{E}_q\left[\sum_{s=1}^{S} \log \frac{p(z_L|c_L)}{q(z_L|c_L, x_s)}\right] + \\ & \mathbb{E}_q\left[\sum_{l=1}^{L-1} \log \frac{p(c_l|c_{l+1}, Z_{l+1})}{q(c_l|c_{l+1}, Z_{l+1}, X)}\right] - \mathbb{KL}(q(c_L|X), p(c_L)) = -\mathcal{L}(X). \end{aligned} \tag{17}$$

B.4 Loss

The final loss for all the models is computed per-sample. Given a distribution of T sets (or tasks) of size S, the training loss is: $L = \frac{1}{ST}\sum_T \mathcal{L}(X_t)$.

B.5 Evaluation

For VAEs we evaluate the models approximating the log marginal likelihood using S importance samples:

$$\text{MLL}(x) = \log \frac{1}{IS} \sum_{is=1}^{IS} \frac{p(x, z_s)}{q(z_{is}|x)} \quad z_{is} \sim q(z|x). \tag{18}$$

For hierarchical models like the NS we use:

$$\text{MLL}(X) = \log \frac{1}{IS} \sum_{is=1}^{IS} \frac{p(X, Z_{is}, c_{is})}{q(Z_{is}, c_{is}|X)} \quad Z_{is} \sim q(Z|c, X), c_{is} \sim q(c|X). \tag{19}$$

B.6 Learnable Aggregation

In this subsection we describe more explicitly the aggregation mechanism. Given a set X, embeddings for samples in the set $h_s = f_\phi(x_s)$, and aggregated statistics $r = \frac{1}{S}\sum_{s=1}^{S} h_s$, we can compute attention weights for a NS-LAG as follows:

$$\begin{aligned} \alpha(r, h_s) &= \frac{\exp(\text{sim}(q(r), k(h_s)))}{\sum_{s'} \exp(\text{sim}(q(r), k(h_{s'})))} \\ r_{LAG} &= \sum_{s=1}^{S} \alpha(r, h_s)\ v(h_s) \\ q_\phi(c|X) &= \mathcal{N}(c|\mu(r_{LAG}), \Sigma(r_{LAG})), \end{aligned} \tag{20}$$

where q, k and v are linear layers and sim is the dot-product scaled by the square root of the representations dimensionality. This approach is inspired by [27] and resembles self-attention with a fundamental difference: instead to map from S samples to S samples, we map from S samples to a per-task learnable aggregation. The query input is a handcrafted aggregation (mean, max pooling). We improve the query scoring the handcrafted aggregation with the samples in the set.

For a full HFSGM-LAG, we can similarly write:

$$\begin{aligned} r &= \frac{1}{S} \sum_{s=1}^{S} f_\phi(h_s, z_s, c) \\ \alpha(r, h_s, z_s, c) &= \frac{\exp(\text{sim}(q(r), k(h_s, z_s, c)))}{\sum_{s'} \exp(\text{sim}(q(r), k(h_{s'}, z_{s'}, c)))} \\ r^l_{LAG} &= \sum_{s=1}^{S} \alpha(r, h_s, z_s, c)\ v(h_s, z_s, c) \\ q_\phi(c_l|c_{l+1}, Z_{l+1}, X) &= \mathcal{N}(c_l|\mu(r^l_{LAG}), \Sigma(r^l_{LAG})), \end{aligned} \tag{21}$$

B.7 Sampling Algorithms

Algorithm 3: Conditional Sampling NS

Data: Input set X;
$c \sim q(c|X)$;
$\mathbf{z} \sim p(\mathbf{z}|c)$;
$x \sim p(x|\mathbf{z}, c)$;
return x;

Algorithm 4: Refined Sampling NS

Data: Input set X;
single pass
$c \sim q(c|X)$;
$\mathbf{z} \sim p(\mathbf{z}|c)$;
$x \sim p(x|\mathbf{z}, c)$;
repeat
  $\tilde{X} = [X, x]$;
  $\tilde{Z}, c \sim q(\tilde{Z}, c|\tilde{X})$;
  $\tilde{X}' \sim p(\tilde{X}'|\tilde{Z}, c)$;
  $x = x'$;
until converged;
return x;

Algorithm 5: Conditional Sampling HFSGM

Data: Input set X;
single pass
$c_L \sim q(c_L|X)$;
$\mathbf{z}, \mathbf{c}_{<L} \sim p(\mathbf{z}, \mathbf{c}_{<L}|c_L)$;
$x \sim p(x|\mathbf{z}, \mathbf{c})$;
return x;

Algorithm 6: Refined Sampling HFSGM

Data: Input set X;
single pass
$c_L \sim q(c_L|X)$;
$\mathbf{z}, \mathbf{c}_{<L} \sim p(\mathbf{z}, \mathbf{c}_{<L}|c_L)$;
$x \sim p(x|\mathbf{z}, \mathbf{c})$;
repeat
  $\tilde{X} = [X, x]$;
  $\tilde{Z}, \mathbf{c} \sim q(\tilde{Z}, \mathbf{c}|\tilde{X})$;
  $\tilde{X}' \sim p(\tilde{X}'|\tilde{Z}, \mathbf{c})$;
  $x = x'$;
until converged;
return x;

C Additional Experiments

C.1 Generative Metrics

In the paper we follow the approach suggested in the NS for sets creation. We report additional experiments on the original Omniglot test set in Table 3. In this setting test time sets are from new characters and unknown alphabets. This setting is even harder and a good generalization benchmark for our approach. We use the preprocessed and statically binarized version proposed in [2]. Again our approaches using learnable aggregation and hierarchical inference perform consistently better in terms of likelihood and reconstruction.

Table 3: Generative Metrics. Omniglot testing on new characters from *unknown* alphabets. Static binarization. All models share the same architecture.

MODEL	AGG	NELBO	NLL	KLz	KLc	MLL	PARAMS (M)
VAE	-	102.56	54.86	47.71	-	91.90	7.5
NS	MEAN	91.98	47.88	39.00	5.10	80.85	14.9
CNS	MEAN	84.96	39.90	39.72	5.35	75.32	7.3
CNS (Ours)	LAG	83.67	37.97	40.55	5.16	74.87	7.4
CHFSGM (Ours)	MEAN	**83.03**	**35.95**	42.54	4.54	**74.01**	9.2
CHFSGM (Ours)	LAG	83.80	36.86	42.34	4.60	**74.15**	9.5

C.2 KLs vs Input Set Cardinality

In Figure 7 we report KLs behaviour varying the input set dimension from 1 to 20 on Omniglot.

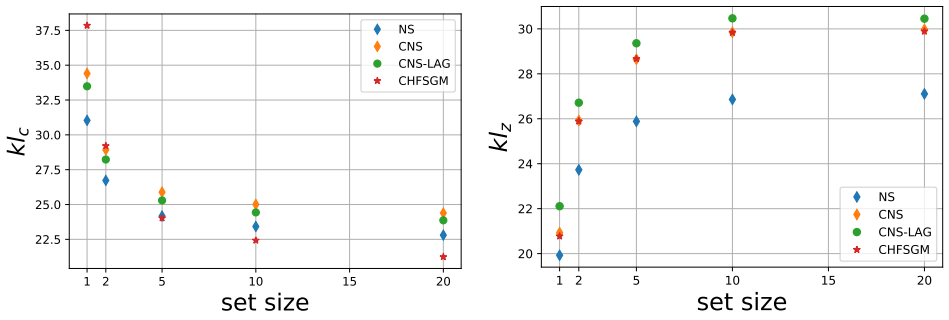

Figure 7: Set Cardinality. Lower-bound and average KL c increasing the set cardinality. Models trained with homogeneous (one concept) sets. All the models improve the generative properties with more data available and learn a better posterior for c.

C.3 Classification

In this paper our main interest is in improving few-shot generation specifically through the lens of hierarchical inference and learnable aggregation. However is natural to ask how the model perform on downstream tasks, in particular on the supervised few-shot learning task. Here we focus on a simple experiment: we train the models on Omniglot and test on MNIST without any form of adaptation. We consider different ways to approximate a classifier.

Adaptation-free Bayes classifier. We can use the fitted generative model as part of a few-shot Bayes classifier: $p(y|x, X) = \frac{p(y)p(x|X_y)}{\sum_{y'} p(y')p(x|X_{y'})}$, where $X = \{X_1, \dots, X_C\}$ is the set of datasets for the C classes. In Appendix B we investigate two approaches that approximate the predictive distribution $p(x|X_y)$ and one based on $q(c|X)$:

I) ELBO difference: $\log p(x|X_y) = \log p(x, X_y) - \log p(X_y) \approx \text{ELBO}(x, X_y) - \text{ELBO}(X_y)$.

II) Equation (6): Sample $q(c_L|X_y)$ and hierarchy $p(\mathbf{z}, \mathbf{c}_{<L}|c_L)$ and evaluate $p(x|\mathbf{z}, \mathbf{c})$ and

III) The classification approach of [8]: $\text{argmax}_y \, \mathbb{KL}(q(c_y|X_y), q(c|x))$.
In Table 4 we report preliminary results for a simple few-shot classification task.

Table 4: Metrics on few-shot classification. Models trained on Omniglot and tested on binarized MNIST. Input set dimension 5. For consistency with the KL classifier used in NS, we use only one layer of posterior, the one closer to the data.

	AGG	$\mathbb{ELBO}[x\|X]$	$\mathbb{KL}[q_X, q_x]$	$\mathbb{E}_q[\log p(x\|c,z)]$
NS	MEAN	0.46	0.73	0.74
CNS	MEAN	0.41	**0.76**	**0.76**
CNS	LAG	0.33	0.75	**0.76**
CHFSGM	MEAN	**0.52**	0.71	0.75
CHFSGM	LAG	0.29	0.71	0.74

C.4 Layer-wise KL contribution

In Figure 8 and 9 we plot the KL contribution for baselines and our models through the hierarchy for Omniglot and CelebA. We notice how models with hierarchy over c and learnable aggregation can better distribute information in latent space.

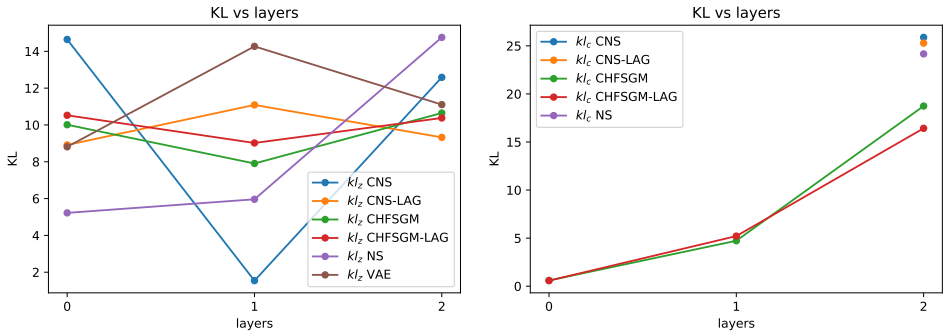

Figure 8: KL per layer for Omniglot

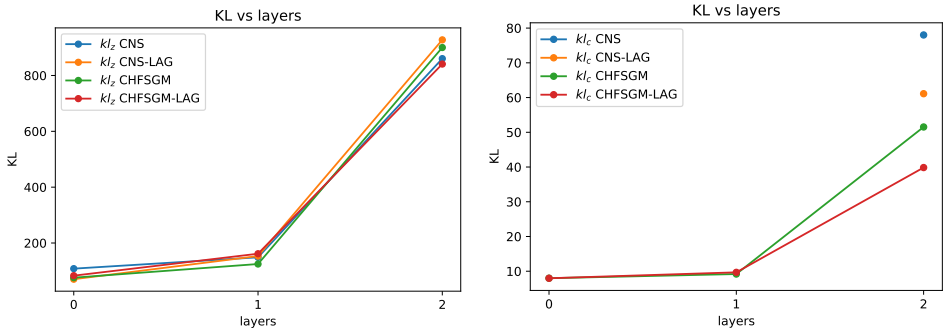

Figure 9: KL per layer for CelebA

C.5 In-homogeneous Sets

In the paper we considered only small homogeneous sets, i.e. one class (character, face, concept) per set, where the set has dimension between 2 and 20. In Figure 10 we test the model behavior when presented with an in-homogeneous input set, i.e. sets with multiple characters.

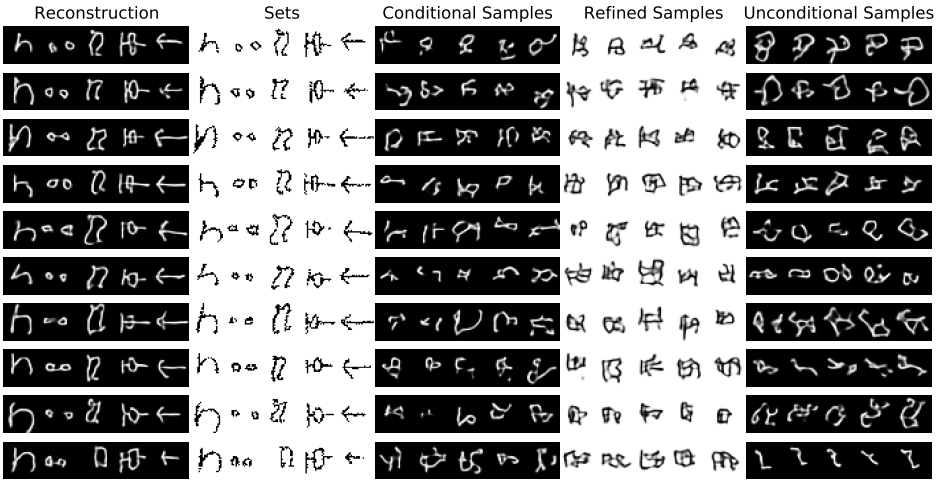

Figure 10: In-homogeneous Sets. Sampling for a CNS-LAG trained on Omniglot. The model is presented with 5 classes per set at test time.

C.5.1 Qualitative Transfer

Another interesting property for a NS-like model is the possibility to condition on samples and classes from different datasets. In the paper we reported quantitative evaluation for generative metrics on transfer. In Figure 11 we report qualitative results on MNIST using a CHFSGM-LAG. We see that the posterior over c can extract relevant information from the set and sample consistently with the conditioning set. The refinement process still helps to improve sample consistency and quality.

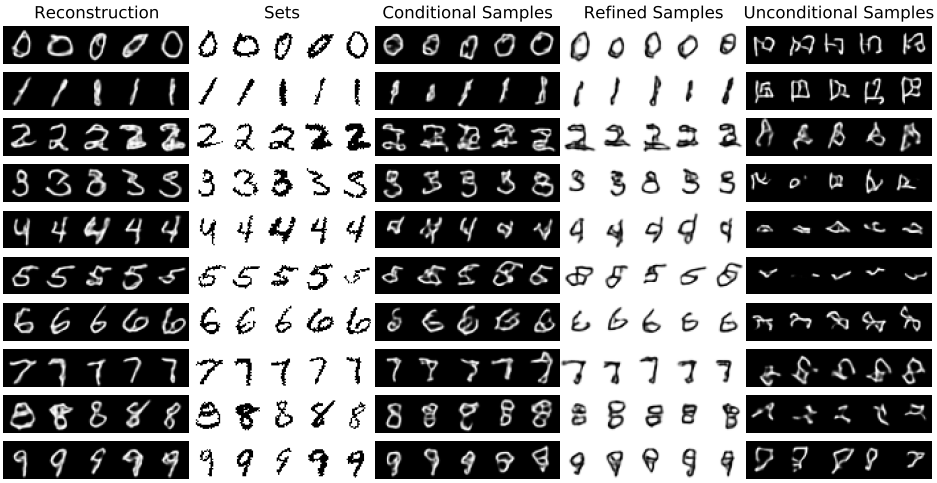

Figure 11: Transfer Sets. Sampling for a CHFSGM with LAG trained on Omniglot and tested on MNIST.

C.6 Additional Samples

In Figure 12 13 and we show stochastic reconstruction, input sets, conditional sampling using the NS approach, conditional sampling using mcmc refinement, and unconditional sampling (sometimes referred as imagination). The model is trained on binarized Omniglot and tested on disjoint classes. We can see how the refinement is particularly effective with complex signs and concept, better extracting the global structure of the concept at hand.

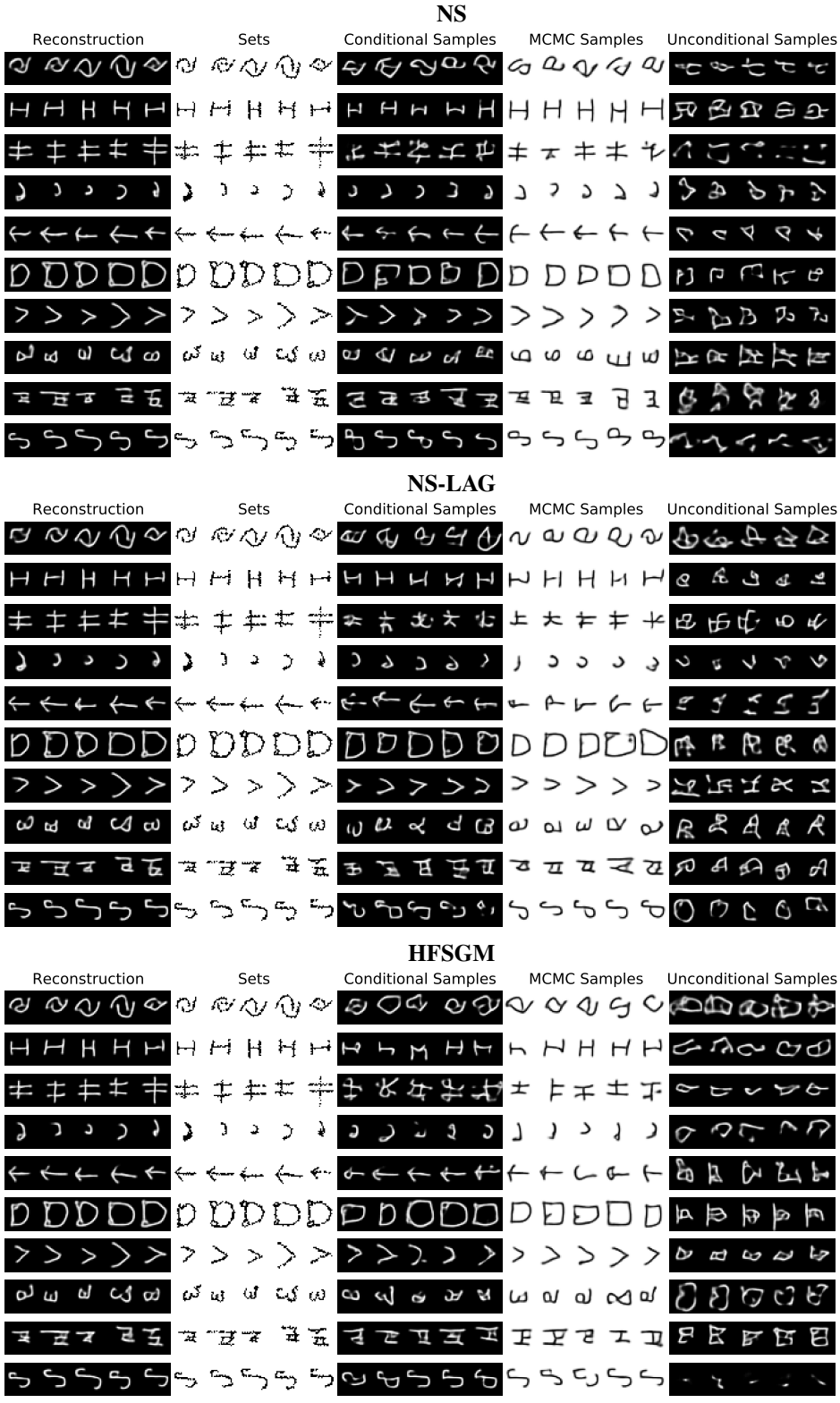

Figure 12: Omniglot Sampling for models without convolutional latent space.

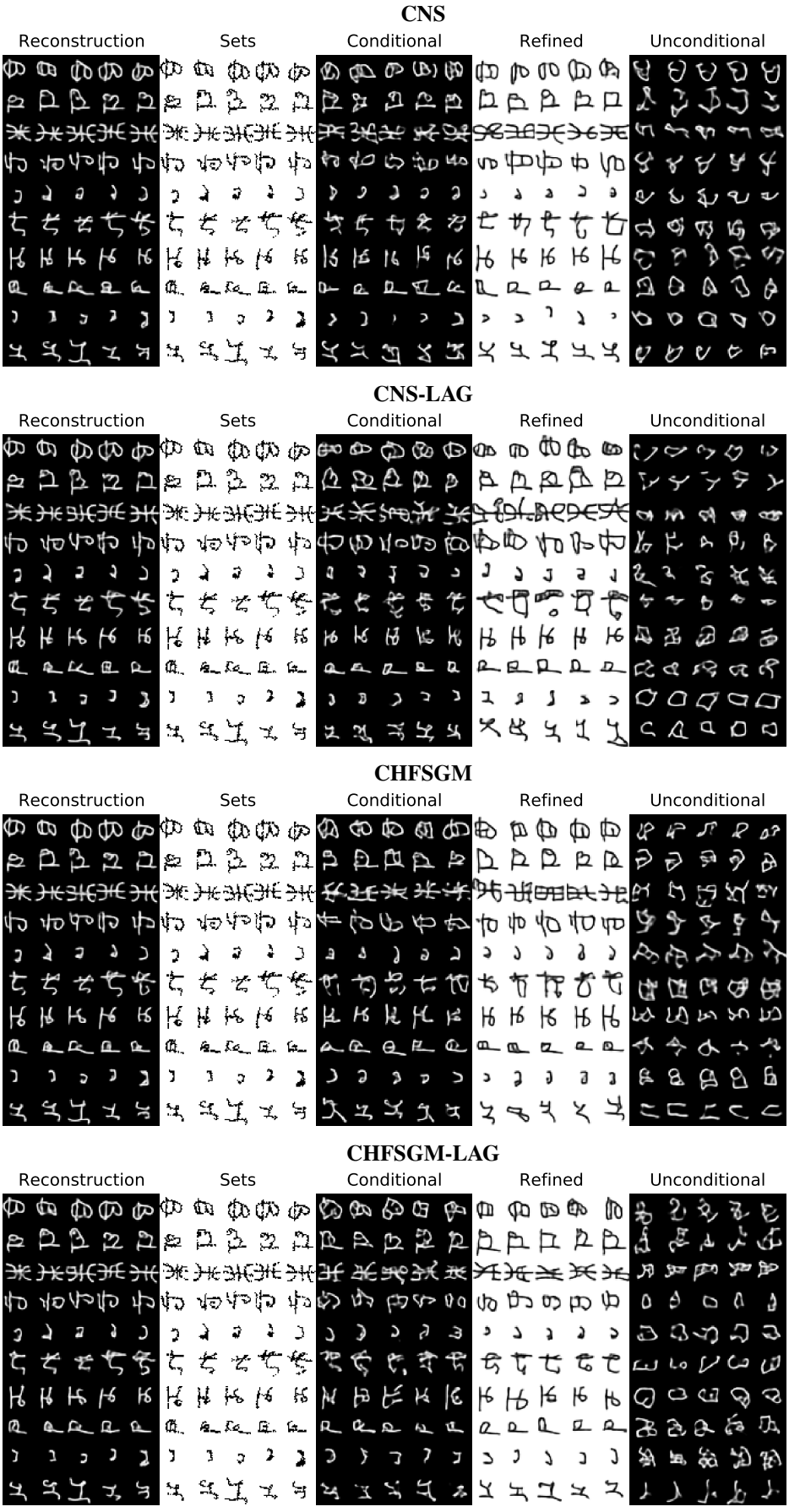

Figure 13: Omniglot Sampling. Sampling for models with convolutional latent space.

Table 5: Relevant Hyperparameters

	Omniglot	CelebA
Number classes	1623	6349
Classes Train	1000	4444
Classes Val	200	635
Classes Test	423	1270
α	1÷2	1÷2
α step	0.5÷0.98	0.5÷0.98
Batch norm	True	True
Batch size	100	50
Channels latent c	64	64
Channels latent space	128	128÷256
Channels latent z	32	32
Classes per set	1	1
Epochs	400	600
Heads	4	4
Input set size	2÷20	2÷20
Learning rate	0.001	0.0001
Schedule	plateau/step	-
Likelihood	Bernoulli	Discretized Mixture Logistics
Loss	VLB	VLB
Optimizer	Adam	Adam
Residual layers	3	3
Resolution latent space	4	4
Stochastic layers	3	3÷9
Weight decay	0.00001	0.00001

D Details

The base model is a close approximation of the Neural Statistician adapted from: `https://github.com/conormdurkan/neural-statistician`. The images are encoded using a shared encoder with 3x3 convolutions plus batch normalization. resolution is halved using stride 2. The decoder is the same, with resolution doubled using transposed convolutions. More powerful and expressive decoders can be employed [30, 40] or multi-resolution deep latent variable models [29, 50, 6]. However our goal is to improve c for a generic architecture. We do not use sample dropout (removing random samples from the input set and use the set statistics as additional features). For our model the latent space is convolutional with a resolution of 4. Latent space with multiple resolutions in the latent space can be used to improve sample quality and learn from high-dimension images. However to reduce training time and isolate the source of complexity in the model (how to improve c) we decided to use a latent space with single resolution. This approach simplifies merging information between c, z and x.

The loss is a weighted negative lower-bound: $-L = \mathrm{VLB}(\alpha) = (1+\alpha)\mathrm{REC} + (\mathrm{KL}_z + \mathrm{KL}_c)/(1+\alpha)$ where alpha is annealed decreasing at each epoch $\alpha = \alpha * \alpha_{step}$, with $\alpha_{step} < 1$ at the beginning of training. This re-weighting tends to magnify the importance of the likelihood term and reduce the risk of posterior collapse at the beginning of training. Learning is slower but typically the model and posterior learned are better.

Modulation. The way we condition the prior and generative model greatly enhances the adaptation capacities of the model. Other than conditioning the representations through direct concatenation or summation, we can condition directly the features and activations in the residual blocks. FiLM is a modulation module used in transfer learning and large-scale class conditional generation. Given an input c and a feature map f with H channels, FiLM [32] learns an affine transformation with $2H$ modulation parameters $(\gamma_h, \beta_h)_{h=1}^{H}$ for each input c:

$$\mathrm{FiLM}(z, c) = \gamma(c) f(z) + \beta(c).$$

This affine transformation can be applied in different part of the prior and generative model. In the main paper we use a simplification of such approach, applying c to z using only the bias term,

i.e. $\gamma(c) = I$ and $\beta(c) = c$. For simple tasks with a unique concept, like sets of characters and sets of faces, this approach is good enough. For more complex sets with multiple concepts or more variability, a more expressive adaptive conditioning mechanism based on FiLM or attention can be useful.