Conditional BRUNO: A Neural Process for Exchangeable Labelled Data

Iryna Korshunova[a,*], Yarin Gal[b], Arthur Gretton[c,**], Joni Dambre[a,**]

[a] *Ghent University, Technologiepark-Zwijnaarde 126, 9052 Ghent, Belgium*
[b] *University of Oxford, OX1 3QD Oxford, UK*
[c] *University College London, Gatsby Unit, 25 Howland Street, W1T 4JG London, UK*

Abstract

We present a neural process which models exchangeable sequences of high-dimensional complex observations conditionally on a set of labels or tags. Our model combines the expressiveness of deep neural networks with the data-efficiency of Gaussian processes, resulting in a probabilistic model for which the posterior distribution is easy to evaluate and sample from, and the computational complexity scales linearly with the number of observations. The advantages of the proposed architecture are demonstrated on a challenging few-shot view reconstruction task which requires generalization from short sequences of viewpoints, and a contextual bandits problem.

Keywords: exchangeability, meta-learning, conditional density estimation

1. Introduction

Exchangeability is an implicit assumption underlying many machine learning algorithms. It entails that any re-ordering of a finite sequence of observations is equally likely. As a consequence, it allows to reason about future observations based on the behaviour of the previous ones. Owing to de Finetti's theorem, the exchangeability property is a cornerstone of Bayesian statistics as it facilitates inference and parameter learning in probabilistic models.

*Corresponding author.
**Joint senior authorship.
Email address: iryna.korshunova@ugent.be (Iryna Korshunova)

Some problems can be explicitly formulated in terms of modelling exchangeable data. For instance, few-shot concept learning can be seen as learning to complete short exchangeable sequences [1], where it is natural to assume no inherent ordering in the observations coming from the same concept. BRUNO (Bayesian recurrent neural model) [2] follows this explicit approach by modelling autoregressive distributions of an exchangeable process $p(\boldsymbol{x}_n|\boldsymbol{x}_{1:n-1})$. This was proven to be an efficient way of doing both few-shot image generation and classification within one model.

In this work, we extend the idea of BRUNO to the conditional case, where we wish to model the distribution $p(\boldsymbol{x}_n|\boldsymbol{h}_n, \boldsymbol{x}_{1:n-1}, \boldsymbol{h}_{1:n-1})$, where $\boldsymbol{h}_i$ are vector valued labels or tags associated with observations $\boldsymbol{x}_i$. As an example, conditional BRUNO can be used in the task of generating new viewpoints of a scene while given a few images of that scene under different camera positions.

Formally, a stochastic process $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots$ is said to be exchangeable if for all n and all permutations π

$$p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = p\left(\boldsymbol{x}_{\pi(1)}, \ldots, \boldsymbol{x}_{\pi(n)}\right), \tag{1}$$

i.e. the joint probability remains the same under any permutation of the sequence.

The intimate connection between exchangeability and Bayesian statistics is due to de Finetti's theorem, which states that every exchangeable process is a mixture of i.i.d. processes,

$$p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \int p(\boldsymbol{\theta}) \prod_{i=1}^{n} p(\boldsymbol{x}_i|\boldsymbol{\theta}) d\boldsymbol{\theta}, \tag{2}$$

where $\boldsymbol{\theta}$ is a parameter vector (finite or infinite-dimensional) conditioned on which, the $\boldsymbol{x}_i$'s are i.i.d. [3].

This theorem gives two ways of defining models of exchangeable sequences. One is via explicit Bayesian modelling: define a prior $p(\boldsymbol{\theta})$, a likelihood $p(\boldsymbol{x}_i|\boldsymbol{\theta})$ and calculate the posterior in Eq. 2 directly. Here, the difficulty is the intractability of the posterior as it requires an integration over the parameter $\boldsymbol{\theta}$.

A common solution is to use a variational approximation. The neural statistician [4] implements this approach by building upon a variational autoencoder model (VAE) [5].

A second way is to *construct* an exchangeable process while modelling its autoregressive distributions $p(\boldsymbol{x}_n|\boldsymbol{x}_{1:n-1})$ without referring to the underlying Bayesian model. BRUNO [2] proposes a design for doing so. It consists of two components: **(a)** a bijective mapping that transforms an intricate input space $\mathcal{X}$ into a Gaussian latent space $\mathcal{Z}$, and **(b)** a collection of exchangeable Gaussian processes ($\mathcal{GP}$s) defined in the latent space $\mathcal{Z}$. Using deep neural networks to implement the bijection $f : \mathcal{X} \mapsto \mathcal{Z}$ allows to model complex and high-dimensional inputs, for example, images. At the same time, the construction of BRUNO guarantees that the process in $\mathcal{X}$ is exchangeable, and thus the model performs an exact, albeit implicit, Bayesian inference in the space $\mathcal{X}$.

A natural extension when building exchangeable models would be to have a conditional process with two associated sequences: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ and $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots$. For instance, when $\boldsymbol{x}_i$ is an image, $\boldsymbol{h}_i$ could be a vector of descriptive labels or tags associated with this image. By analogy with Eq. 1, the exchangeability property becomes:

$$p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n|\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n) = p\left(\boldsymbol{x}_{\pi(1)}, \ldots, \boldsymbol{x}_{\pi(n)}|\boldsymbol{h}_{\pi(1)}, \ldots, \boldsymbol{h}_{\pi(n)}\right). \tag{3}$$

To have a valid stochastic process, we also need a consistency property as imposed by the Kolmogorov extension theorem [6]:

$$p(\boldsymbol{x}_{1:m}|\boldsymbol{h}_{1:m}) = \int p\left(\boldsymbol{x}_{1:n}|\boldsymbol{h}_{1:n}\right) d\boldsymbol{x}_{m+1:n} \text{ for } 1 \leq m < n. \tag{4}$$

To our best knowledge, Bayesian theory does not have an established proof of de Finetti's theorem for conditional probabilities. In other words, it remains a conjecture that conditions in Eq. 3 and Eq. 4 guarantee that one can represent the process as a mixture of conditionally i.i.d. models as given in Eq. 5 or

equivalently, in Eq. 6.

$$p(\boldsymbol{x}_{1:n}|\boldsymbol{h}_{1:n}) = \int p(\boldsymbol{\theta}) \prod_{i=1}^{n} p(\boldsymbol{x}_i|\boldsymbol{h}_i, \boldsymbol{\theta}) d\boldsymbol{\theta} \tag{5}$$

$$p(\boldsymbol{x}_n|\boldsymbol{h}_n, \boldsymbol{x}_{1:n-1}, \boldsymbol{h}_{1:n-1}) = \int p(\boldsymbol{\theta}|\boldsymbol{x}_{1:n-1}, \boldsymbol{h}_{1:n-1}) p(\boldsymbol{x}_n|\boldsymbol{h}_n, \boldsymbol{\theta}) d\boldsymbol{\theta} \tag{6}$$

However, for processes where $\boldsymbol{x}_i$ and $\boldsymbol{h}_i$ take values from a finite set, this theorem is proven in the field of quantum physics [7]. Though, it is yet unclear how to extend these results to continuous variables.

Relying on the conditional version of de Finetti's theorem, neural processes [8] take an approach that is similar to the neural statistician's. It extends the VAE model to handle collections of $(\boldsymbol{x}_i, \boldsymbol{h}_i)$ input pairs in a permutation-invariant way while optimizing a variational lower bound on $p(\boldsymbol{x}_n|\boldsymbol{h}_n, \boldsymbol{x}_{1:n-1}, \boldsymbol{h}_{1:n-1})$. Versa [9] also follows the idea of approximating the aforementioned posterior predictive distribution, though they use a training procedure that differs from standard variational inference. Both models achieve permutation invariance of $p(\boldsymbol{\theta}|\boldsymbol{x}_{1:n}, \boldsymbol{h}_{1:n})$ by using instance-pooling operations, e.g. the mean over representations of $(\boldsymbol{x}_i, \boldsymbol{h}_i)$ pairs.

An alternative method that does not require approximations of the right-hand side of Eq. 6, is to use the idea of BRUNO and construct a process that satisfies Eq. 3 and Eq. 4 directly. In the next section, we show how this can be done by slightly modifying the architecture of BRUNO, thus leading to conditional BRUNO, which we will further refer to as C-BRUNO.

2. Method

We begin this section with an overview of the mathematical tools needed to construct our model: first the exchangeable Gaussian processes; and then a proposed conditional version of Real NVP - a deep, stably invertible and learnable neural network architecture for conditional density estimation [10]. We next present C-BRUNO – an extension of the BRUNO model [2], wherein we combine the two aforementioned components. Our model is illustrated in Figure 1.

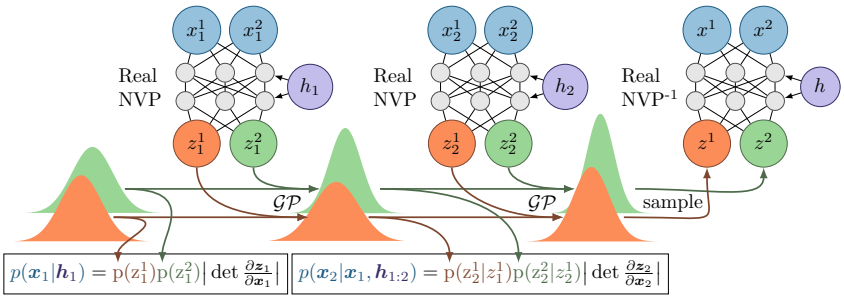

Figure 1: A schematic of C-BRUNO unrolled for two update steps and a sampling step. This illustrates how our model is able to update, evaluate and sample from the predictive distribution. Here, the two data points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are mapped into latent vectors $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$, where the mappings are conditioned on $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ respectively. The GPs' parameters are updated after processing every input. For example, after observing $\boldsymbol{z}_1$, we update the parameters of the priors $p(z_1^1)$ and $p(z_1^2)$. At every step, we can also evaluate the predictive distributions $p(\boldsymbol{x}_1|\boldsymbol{h}_1)$ and $p(\boldsymbol{x}_2|\boldsymbol{x}_1, \boldsymbol{h}_{1:2})$. Sampling from the predictive distribution is done by sampling z^1 and z^2 and then applying the inverse Real NVP mapping.

2.1. Exchangeable Gaussian processes

Consider a stochastic process $z_1, z_2, z_3, \ldots$, where for any finite number n of random variables $z_1, z_2, \ldots z_n \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with a constant mean $\boldsymbol{\mu} = (\mu, \mu, \ldots, \mu)$ and a compound symmetry covariance matrix $\boldsymbol{\Sigma}$, i.e. $\boldsymbol{\Sigma}_{ii} = v$ and $\boldsymbol{\Sigma}_{ij,i\neq j} = \rho$. To ensure that $\boldsymbol{\Sigma}$ is a positive-definite covariance matrix that complies with covariance properties of exchangeable sequences, we additionaly require that $0 \leq \rho < v$ [3]. The constructed process a special case of a Gaussian process, and it is exchangeable since the definition in Eq. 1 is satisfied.

Imposing exchangeability constraints allows to derive recurrent linear-time updates for the mean and variance parameters of the predictive distribution, i.e. $p(z_{n+1}|z_{1:n}) = \mathcal{N}(\mu_{n+1}, v_{n+1})$ for $n \geq 1$:

$$\mu_{n+1} = (1 - d_n)\mu_n + d_n z_n, \quad v_{n+1} = (1 - d_n)v_n + d_n(v - \rho), \tag{7}$$

where $d_n = \frac{\rho}{v+\rho(n-1)}$, and the prior parameters $\mu_1 = \mu$ and $v_1 = v$. Derivation of these updates are given in the Appendix B of Korshunova et al. [2].

Exchangeable $\mathcal{GP}s$ on their own have the insufficient power to model any interesting data. However, they become useful once we combine them with

deep neural networks. We will show this experimentally in Section 3, where the resulting model is able to sucessfully model complex conditional densities of image sequences.

2.2. Conditional Real NVP

Real NVP (real-valued non-volume preserving transformation) [10] is a member of the normalizing flows family of models, where some density $p(\boldsymbol{x})$ in the input space $\mathcal{X} = \mathbb{R}^D$ is transformed into the desired probability distribution $p(\boldsymbol{z})$ in the latent space $\mathcal{Z} = \mathbb{R}^D$ through a sequence of invertible mappings [11]. Real NVP is implemented as a stack of alternating coupling layers, with every layer transforming half of its input dimensions while copying the other half directly to the output:

$$\begin{cases} \boldsymbol{x}_{\text{out}}^{1:d} = \boldsymbol{x}_{\text{in}}^{1:d} \\ \boldsymbol{x}_{\text{out}}^{d+1:D} = \boldsymbol{x}_{\text{in}}^{d+1:D} \odot \exp(s(\boldsymbol{x}_{\text{in}}^{1:d})) + t(\boldsymbol{x}_{\text{in}}^{1:d}), \end{cases} \tag{8}$$

where $\odot$ is an elementwise product, and the functions s (scale) and t (translation) are usually deep neural networks. In addition to bijectivity, this design ensures the following two properties. Firstly, the inverse is easy to evaluate, i.e. the computational cost of the backward mapping $\boldsymbol{x} = f^{-1}(\boldsymbol{z})$ is the same as for the forward mapping $\boldsymbol{z} = f(\boldsymbol{x})$. Thus, sampling from Real NVP is fast compared to autoregressive flow models [12]. Secondly, computing the Jacobian determinant takes linear time in the number of input dimensions D, which allows to easily evaluate the likelihood of the inputs via the change of variables formula:

$$p(\boldsymbol{x}) = p(\boldsymbol{z}) \left| \det \left(\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} \right) \right|. \tag{9}$$

A learnable, sufficiently expressive mapping of the data to a latent space is capable of making the transformed data conform to a factorized and easy-to-model distribution. Namely, $p(\boldsymbol{z}) = \prod_{d=1}^{D} p(z^d)$, where every z^d is independent, and $p(z^d)$ is a standard distribution, commonly chosen to be Gaussian [5, 10].

When the goal is to model a conditional distribution $p(\boldsymbol{x}|\boldsymbol{h})$, we propose to make the transformations s and t dependent on $\boldsymbol{h}$. One way to achieve this is

to concatenate the features of $\boldsymbol{h}$ to the inputs of every dense and convolutional layer of the s and t networks. In this case, the Jacobian of the Real NVP mapping becomes dependent on $\boldsymbol{h}$, and the change of variables formula gives the conditional density:

$$p(\boldsymbol{x}|\boldsymbol{h}) = p(\boldsymbol{z}) \left| \det \left(\frac{\partial f(\boldsymbol{x}, \boldsymbol{h})}{\partial \boldsymbol{x}} \right) \right|. \tag{10}$$

2.3. C-BRUNO: the conditional exchangeable sequence model

We now combine Bayesian and deep learning tools from the previous sections and present our model for conditional exchangeable sequences, whose schematic is given in Figure 1.

Assume we are given exchangeable sequences $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ and $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n$, where every D-dimensional variable $\boldsymbol{x}_i$ is associated with a vector of labels or tags $\boldsymbol{h}_i$. Applying conditional Real NVP to every $(\boldsymbol{x}_i, \boldsymbol{h}_i)$ pair results in an exchangeable sequence $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n$ in the latent space where the model makes the following assumptions:

A1: dimensions $\{z^d\}_{d=1,\ldots,D}$ are independent, so $p(\boldsymbol{z}) = \prod_{d=1}^{D} p(z^d)$

A2: for each dimension d, we assume that $(z_1^d, \ldots z_n^d) \sim \mathcal{N}_n(\mathbf{0}, \boldsymbol{\Sigma}^d)$, where $\boldsymbol{\Sigma}^d$ is a $n \times n$ covariance matrix with $\boldsymbol{\Sigma}^d_{ii} = v^d$ and $\boldsymbol{\Sigma}^d_{ij, i \neq j} = \rho^d$, $0 \leq \rho^d < v^d$.

These two assumptions of C-BRUNO are identical to the ones from its unconditional counterpart [2] because the dependence on $\boldsymbol{h}$ is introduced by conditioning the Real NVP part of C-BRUNO, and thus the distribution in the latent space $\mathcal{Z}$ can remain fixed.

3. Experiments

3.1. ShapeNet view reconstruction

We consider the task of few-shot image reconstruction, where the model is required to infer how an object looks from various angles based on a small set of observed views [9]. This problem can be framed as generating samples from a predictive conditional distribution $p(\boldsymbol{x}_n|\boldsymbol{h}_n, \boldsymbol{x}_{1:n-1}, \boldsymbol{h}_{1:n-1})$, where $\boldsymbol{h}_n$ is a desired angle and $\boldsymbol{x}_{1:n-1}$ is a set of observed views associated with angles

$\boldsymbol{h}_{1:n-1}$. We use a set of 12 object categories from ShapeNetCore v2 [13] as selected by Gordon et al. [9], and train C-BRUNO to predict different views from a single shot. Namely, given a random view $\boldsymbol{x}_1$ and its angle $\boldsymbol{h}_1$, the goal is to predict N views of the same object under angles $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N$. Thus, the training objective is to maximize the likelihood of ground-truth images under the model distribution, i.e. $\mathcal{L} = \sum_{n=1}^{N} \log p(\boldsymbol{x}_n | \boldsymbol{h}_n, \boldsymbol{x}_1, \boldsymbol{h}_1)$. This loss is optimized with respect to the Real NVP parameters and variance-covariance parameters of the $\mathcal{GP}$s. We train C-BRUNO in a batch-mode on all 12 object classes at once and use the same train-test split as VERSA [9], such that the two models are comparable [1].

In Figure 2, we show samples from C-BRUNO when the model is given viewpoints of an object that was not seen during training. In the majority of cases, our samples have correct orientation and are visually sharper compared to samples from VERSA [9]. The difference between the two models increases with more uncertainty in the object's appearance or when the object is less similar to the training examples. In this case, C-BRUNO generates samples with higher variance and more inaccuracies while VERSA samples become more blurry. In Figure 2, this is illustrated for the airplane object. When a single shot is given, from which we cannot infer the wing configuration, C-BRUNO samples more diverse airplanes compared to when we condition on multiple distinctive viewpoints. With more airplane shots, the quality of VERSA samples increases as well. However, as we can see from the car example, this does not always hold, thus indicating that VERSA requires training with multiple input shots in order to match these testing conditions. C-BRUNO, on the other hand, is more agnostic to the length of sequences it is trained or tested on.

3.2. Contextual bandits

Contextual bandits constitute another task where we can apply C-BRUNO. We consider a wheel bandit problem [14] that was previously used to compare

[1] The code to reproduce our experiments is available at `github.com/IraKorshunova/bruno`.

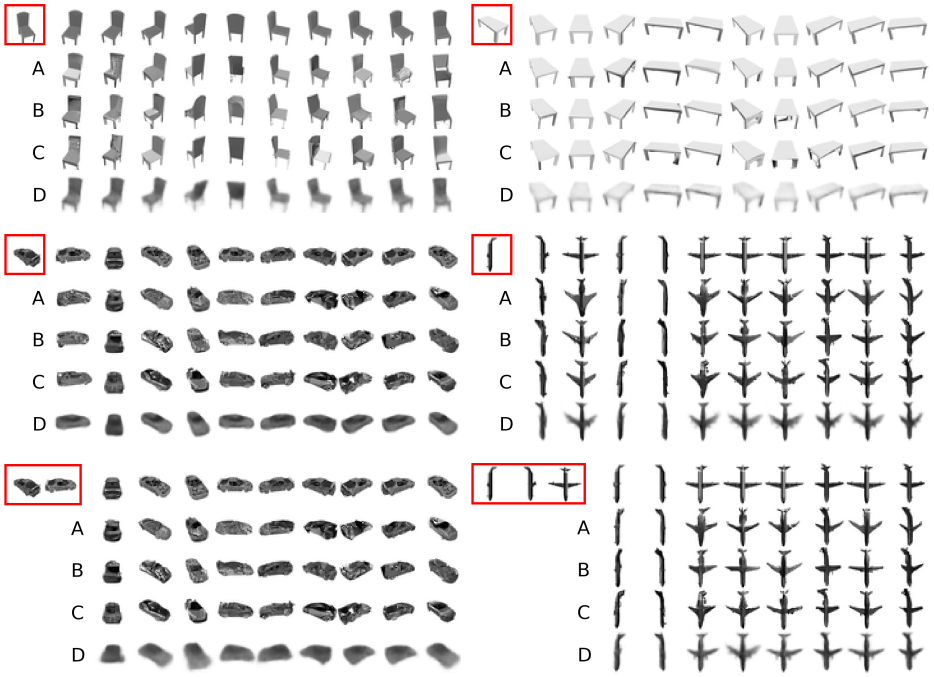

Figure 2: Few-shot C-BRUNO samples in rows A-C and VERSA samples in row D for the unseen test objects. Here, we condition on the input views $(\boldsymbol{x}_{1:n}, \boldsymbol{h}_{1:n})$ marked in red. The top row of each plot contains ground truth images, whereas the three rows A to C are the C-BRUNO samples from $p(\boldsymbol{x}|\boldsymbol{h}, \boldsymbol{x}_{1:n}, \boldsymbol{h}_{1:n})$ conditioned on a different angle $\boldsymbol{h}$ in each column.

neural processes (NPs) [8] and model-agnostic meta-learning (MAML) [15]. The task can be outlined as follows: a circle of a unit radius is partitioned into a low-reward region (blue) and 4 high-reward regions whose sizes are parameterized by δ as illustrated in Figure 3. There are five possible actions: a_1 to a_5. The first action a_1 always yields a reward r sampled from $\mathcal{N}(1.2, 0.001^2)$. The reward for other four actions depends on the location within a circle – a context point with coordinates $\boldsymbol{x} = (x_1, x_2)$. If $\|\boldsymbol{x}\| < \delta$, a_1 is the optimal action since other arms have a reward of $r \sim \mathcal{N}(1.0, 0.001^2)$. When $\|\boldsymbol{x}\| > \delta$, the optimal action is defined by the location $\boldsymbol{x}$ and it yields the reward of $\mathcal{N}(50.0, 0.001^2)$.

Similarly to NPs and MAML, we pretrain C-BRUNO on a batch of 64 sequences sampled from wheel problems with a random $\delta \sim \mathcal{U}(0, 1)$. Each element of the sequence is a tuple $(\boldsymbol{x}, a, r)$ of the context $\boldsymbol{x}$, action a and the

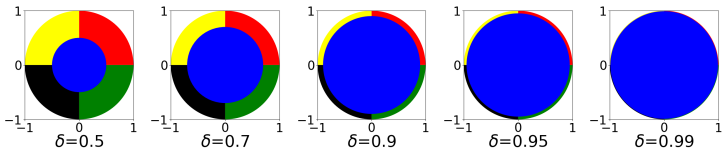

Figure 3: Context regions in the unit circle for the wheel bandits problem with varying values of the exploration parameter δ [14]. For blue, red, green, black, and yellow context regions, the optimal actions are a_1, a_2, a_3, a_4, and a_5, respectively.

reward r. Every sequence contains 562 points, where 50 points are used as targets, and we train C-BRUNO to maximize the likelihood of target rewards: $\mathcal{L} = \sum_{n=512}^{562} \log p(r_n | \boldsymbol{x}_n, a_n, \boldsymbol{x}_{1:511}, a_{1:511}, r_{1:511})$. Here, both contexts and actions constitute the conditional information previously denoted as $\boldsymbol{h}$. One issue with this approach is that modelling a scalar reward implies a one-dimensional latent space as the Real NVP architecture needs to be bijective. In this case, the inability to control the size of the latent space would limit the expressive power of the model. In practice, we found that a simple yet effective solution is to append a number of dummy dimensions to the inputs and sample their values from a uniform distribution $\mathcal{U}(0, 1)$. A similar approach was shown to be useful for other flow models, such as Neural ODEs [16]. At test time, the dummy dimensions are ignored as we are only interested in rewards sampled from $p(r | \boldsymbol{x}_{n+1}, a, \boldsymbol{x}_{1:n}, a_{1:n}, r_{1:n})$. Specifically, their average approximates the expected reward for action a at step $n + 1$ given the context $\boldsymbol{x}_{n+1}$ and the history. We choose the action that maximizes this value. Future work may want to explore how this approach relates to Thompson sampling [17].

To compare the performance of C-BRUNO to other existing models applied to the wheel bandit problem, we used the evaluation framework of by Riquelme et al. [14]. The results are given in Table 1, from which we can conclude that C-BRUNO is on par with the state-of-the-art meta-learning techniques.

δ	0.5	0.7	0.9	0.95	0.99
MAML [15]	$2.49_{\pm 0.12}$	$3.00_{\pm 0.35}$	$4.75_{\pm 0.48}$	$7.10_{\pm 0.77}$	$22.89_{\pm 1.41}$
NPs [8]	$1.04_{\pm 0.06}$	$1.26_{\pm 0.21}$	$2.90_{\pm 0.35}$	$5.45_{\pm 0.47}$	$21.45_{\pm 1.30}$
C-BRUNO	$1.32_{\pm 0.06}$	$1.43_{\pm 0.07}$	$3.44_{\pm 0.13}$	$6.17_{\pm 0.21}$	$21.52_{\pm 0.41}$

Table 1: Simple regret (mean cumulative regret in the last 500 out of 80 000 steps) for the wheel bandit problems with different values of δ. Reported values are the means and standard errors over 100 trials. The regrets are normalized with respect to the performance of the uniform method which selects each action at random with equal probability.

4. Discussion

We presented C-BRUNO – an extension of BRUNO [2] to the conditional case, which maintains its appealing properties, such as **(a)** exact likelihoods **(b)** fast sampling and inference, **(c)** no retraining or changes to the architecture at test time, and **(d)** a recurrent formulation. These features constitute a powerful meta-learning model with a flexible design. Together, BRUNO and C-BRUNO cover a broad range of meta-learning applications while performing on par with more task-specific state-of-the-art methods. In particular, this paper showed how C-BRUNO can be used for few-shot conditional image generation and contextual bandits.

BRUNO and C-BRUNO build directly on the fundamental property of exchangeability that underlies much of Bayesian statistics. They provide an alternative way to building meta-learning models by shifting to implicit inference instead of the commonly used approximate explicit Bayesian inference. BRUNO models combine exchangeable $\mathcal{GP}$s with powerful bijective feature extractors in the form of flow-based deep neural architectures. While the former component is unlikely to be improved, we expect our models to greatly benefit from the recent advances in normalizing flows, which is currently an active area of research [18, 19]. This would allow to apply our models to more challenging datasets, thus offering a simpler alternative to more complex models, for instance, Generative Query Networks [20].

Acknowledgements

We would like to thank Jonas Degrave for insightful discussions, John Bronskill for the ShapeNet dataset and for answering questions related to VERSA, Carlos Riquelme for providing crucial details regarding the contextual bandits experiments, and Ferenc Huszar for the whole idea of exchangeability via RNNs. This work was supported by the Special Research Fund of Ghent University.

References

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

[2] I. Korshunova, J. Degrave, F. Huszar, Y. Gal, A. Gretton, J. Dambre, BRUNO: A Deep Recurrent Model for Exchangeable Data, in: Advances in Neural Information Processing Systems 31, 7190–7198, 2018.

[3] D. J. Aldous, P. L. Hennequin, I. A. Ibragimov, J. Jacod, Ecole d'Ete de Probabilites de Saint-Flour XIII, 1983, Lecture Notes in Mathematics, Springer Berlin Heidelberg, 1985.

[4] H. Edwards, A. Storkey, Towards a Neural Statistician, in: International Conference on Learning Representations, 2017.

[5] D. P. Kingma, M. Welling, Auto-Encoding Variational Bayes, in: International Conference on Learning Representations, 2014.

[6] B. Øksendal, Stochastic Differential Equations: An Introduction with Applications, Hochschultext / Universitext, Springer, 2003.

[7] J. Barrett, M. Leifer, The de Finetti theorem for test spaces, New Journal of Physics 11 (3).

[8] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. M. A. Eslami, Y. W. Teh, Neural Processes, Theoretical Foundations and Applications of Deep Generative Models, ICML workshop .

[9] J. Gordon, J. Bronskill, M. Bauer, S. Nowozin, R. Turner, Meta-Learning Probabilistic Inference for Prediction, in: International Conference on Learning Representations, 2019.

[10] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density Estimation Using Real NVP, in: International Conference on Learning Representations, 2017.

[11] D. Rezende, S. Mohamed, I. Danihelka, K. Gregor, D. Wierstra, One-Shot Generalization in Deep Generative Models, in: 33rd International Conference on Machine Learning, vol. 48 of *Proceedings of Machine Learning Research*, 1521–1529, 2016.

[12] G. Papamakarios, T. Pavlakou, I. Murray, Masked Autoregressive Flow for Density Estimation, in: Advances in Neural Information Processing Systems 30, 2338–2347, 2017.

[13] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, F. Yu, ShapeNet: An Information-Rich 3D Model Repository, arXiv preprint arXiv:1512.03012 .

[14] C. Riquelme, G. Tucker, J. Snoek, Deep Bayesian Bandits Showdown: An Empirical Comparison of Bayesian Deep Networks for Thompson Sampling, in: International Conference on Learning Representations, 2018.

[15] C. Finn, P. Abbeel, S. Levine, Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks, in: 34th International Conference on Machine Learning, vol. 70 of *Proceedings of Machine Learning Research*, 1126–1135, 2017.

[16] E. Dupont, A. Doucet, Y. W. Teh, Augmented Neural ODEs, ArXiv abs/1904.01681.

[17] W. R. Thompson, On the likelihood that one unknown probability exceeds another in view of the evidence of two samples, Biometrika 25 (3-4) (1933) 285–294.

[18] W. Grathwohl, R. T. Q. Chen, J. Bettencourt, D. Duvenaud, Scalable Reversible Generative Models with Free-form Continuous Dynamics, in: International Conference on Learning Representations, 2019.

[19] D. P. Kingma, P. Dhariwal, Glow: Generative Flow with Invertible 1x1 Convolutions, in: Advances in Neural Information Processing Systems 31, Curran Associates, Inc., 10215–10224, 2018.

[20] S. M. A. Eslami, D. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, D. Hassabis, Neural scene representation and rendering, Science 360 (6394) (2018) 1204–1210.