3rd Symposium on Advances in Approximate Bayesian Inference, 2020 1–11

Argmax Flows: Learning Categorical Distributions with Normalizing Flows

Emiel Hoogeboom* E.HOOGEBOOM@UVA.NL
UvA-Bosch Delta Lab, University of Amsterdam

Didrik Nielsen* DIDNI@DTU.DK
Technical University of Denmark

Priyank Jaini
UvA-Bosch Delta Lab, University of Amsterdam

Patrick Forré
University of Amsterdam

Max Welling
UvA-Bosch Delta Lab, University of Amsterdam

Abstract

This paper introduces a new method to define and train continuous distributions such as normalizing flows directly on categorical data, for example text and image segmentation. The generative model is defined by a composition of a normalizing flow and an argmax function. To optimize this model, we dequantize the argmax using a distribution that is a probabilistic right-inverse to the argmax. This distribution lifts the categorical data to a continuous space on which the flow can be trained. We demonstrate that applying existing dequantization techniques naïvely to categorical data leads to suboptimal solutions. In addition, the model is fast both in generative (for sampling) and inference direction (for training), as opposed to autoregressive models.

1. Introduction

Typically, normalizing flows model continuous distributions. As a result, directly optimizing a flow on discrete data may lead to arbitrarily high likelihoods. An example of this phenomenon is when flows are trained on ordinal 8-bit image data. Pixels are discretely valued from 0 to 255 and need to be dequantized (Uria et al., 2013; Theis et al., 2016), *i.e.* noise is added to lift the discrete pixels into a continuous space. In their framework, dequantization is the inference distribution that is the natural counterpart of generative rounding. Even though rounding is a natural transformation to obtain ordinal discrete variables, it places an unwanted inductive bias on categorical variables as different pairs of categories can be closer or further apart.

In this paper we resolve these issues by proposing a generative model using an argmax surjection and a corresponding family of probabilistic right-inverses for these argmax surjections. Argmax surjections are a deterministic map to obtain categorical variables from a continuous representation and they do not add uncorrelated noise to the sampling proce-

* equal contribution

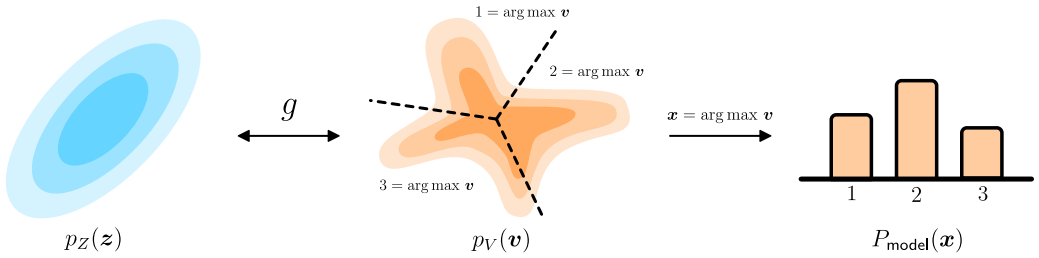


Figure 1: Overview of the generative model of argmax flow. A continuous distribution $p_V(\boldsymbol{v})$ is transformed into a categorical distribution $P_{\text{model}}(\boldsymbol{x})$ using the argmax function. In this example, the continuous distribution $p_V(\boldsymbol{v})$ is learned using a normalizing flow g that maps from a latent base distribution $p_Z(\boldsymbol{z})$.

dure. To learn the underlying density model, we parametrize a probabilistic right-inverse to the argmax surjection, referred to as *dequantization*.

2. Preliminaries and Problem Setup

Let $\mathcal{X} = \{1, 2, \cdots, K\}^d$ be a d-dimensional categorical space with probability mass function $P_{\text{data}}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{X}$ where $\boldsymbol{x} = (x_1, x_2, \cdots, x_d)$ and each x_i is a univariate categorical random variable of K categories.

Given $\mathcal{V} = \mathbb{R}^d$ and $\mathcal{Z} = \mathbb{R}^d$ with densities p_V and p_Z respectively, normalizing flows (Rezende and Mohamed, 2015) learn a bijective and differentiable transformation $g : \mathcal{Z} \to \mathcal{V}$ such that the change of variables formula gives the density at any point $\boldsymbol{v} \in \mathcal{V}$:

$$p_V(\boldsymbol{v}) = p_Z(\boldsymbol{z})) \cdot \left|\nabla_{\boldsymbol{z}} g(\boldsymbol{z})\right|^{-1}, \qquad \boldsymbol{v} = g(\boldsymbol{z}), \tag{1}$$

where p_Z can be any density (usually chosen as standard Gaussian). Thus, normalizing flows provide a powerful framework to learn *exact* density functions in an unsupervised manner. However, Equation (1) is restricted to continuous densities and cannot be applied in a straight-forward manner to discrete random variables.

Theis et al. (2016) have shown that modeling this continuous density $p(\boldsymbol{v})$ lower bounds the discrete distribution $P(\boldsymbol{x})$ for uniform distribution. Ho et al. (2019) extended this framework for any variational distribution $q(\boldsymbol{u}|\boldsymbol{x})$. In (Hoogeboom et al., 2020) it is shown that from a variational inference perspective, not only hypercubes but any partitioning of the space $\mathcal{V}$ can be optimized using this objective. Furthermore, Nielsen et al. (2020) reinterpreted the process of dequantization as a surjective transformation $f : \mathcal{X} \to \mathcal{V}$ that is deterministic in one direction (since $\boldsymbol{x} = \mathsf{round}(\boldsymbol{v})$) and stochastic in the other ($\boldsymbol{v} = \boldsymbol{x} + \boldsymbol{u}$ where $\boldsymbol{u} \sim q(\boldsymbol{u}|\boldsymbol{x})$). Using this interpretation, dequantization can be seen as a family of probabilistic right-inverses for a rounding surjection in the latent variable model given by:

$$P(\boldsymbol{x}) = \int P(\boldsymbol{x}|\boldsymbol{v}) p(\boldsymbol{v}) \, \mathrm{d}\boldsymbol{v}, \qquad P(\boldsymbol{x}|\boldsymbol{v}) := \delta\big(\boldsymbol{x} = \mathsf{round}(\boldsymbol{v})\big)$$

In this case, the density model $p(\boldsymbol{v})$ can be any distribution and is modeled using a normalizing flow. Learning proceeds by introducing the variational distribution $q(\boldsymbol{v}|\boldsymbol{x})$ that models the family of probabilistic right-inverses for rounding surjection and optimizing the

Table 1: Surjective flow layers for applying continuous flow models to discrete data. The layers are deterministic in the generative direction, but stochastic in the inference direction. Rounding corresponds to the commonly-used dequantization for ordinal data.

Layer	Generation	Inference	Applications
Rounding	$\boldsymbol{x} = \lfloor \boldsymbol{v} \rfloor$	$\boldsymbol{v} \sim q(\boldsymbol{v}\|\boldsymbol{x})$ w/ support $\mathcal{S}(\boldsymbol{x}) = \{\boldsymbol{v}\|\boldsymbol{x} = \lfloor \boldsymbol{v} \rfloor\}$	Ordinal Data e.g. images, audio
Argmax	$\boldsymbol{x} = \arg\max \boldsymbol{v}$	$\boldsymbol{v} \sim q(\boldsymbol{v}\|\boldsymbol{x})$ w/ support $\mathcal{S}(\boldsymbol{x}) = \{\boldsymbol{v}\|\boldsymbol{x} = \arg\max \boldsymbol{v}\}$	Categorical Data e.g. text, segmentation

following bound:

$$\begin{aligned} \log P(\boldsymbol{x}) &\geq \mathbb{E}_{\boldsymbol{v}\sim q(\boldsymbol{v}|\boldsymbol{x})} \left[\log P(\boldsymbol{x}|\boldsymbol{v}) + \log p(\boldsymbol{v}) - \log q(\boldsymbol{v}|\boldsymbol{x})\right] \\ &= \mathbb{E}_{\boldsymbol{v}\sim q(\boldsymbol{v}|\boldsymbol{x})} \left[\log p(\boldsymbol{v}) - \log q(\boldsymbol{v}|\boldsymbol{x})\right] \end{aligned} \tag{2}$$

Under the constraint that the support of $q(\boldsymbol{v}|\boldsymbol{x})$ is enforced to be only over the region $\mathcal{S} = \{\boldsymbol{v} \in \mathbb{R}^d : \boldsymbol{x} = \mathsf{round}(\boldsymbol{v})\}$ which ensures that $P(\boldsymbol{x}|\boldsymbol{v}) = 1$. In the next section, we will propose a novel surjective transformation called *argmax flows* that directly extends the ideas of rounding surjection to categorical random variables by designing a probabilistic right-inverse $q(\boldsymbol{v}|\boldsymbol{x})$ for a surjective transformation that maps a categorical random variable to a continuous random variable.

3. Argmax Flows

We propose a novel method to learn categorical data with continuous distributions. This method consists of two parts: (1) the *generative* model that comprises of an argmax function i.e. $\boldsymbol{x} = \arg\max \boldsymbol{v}$, and (2) an *inference* model that requires a probabilistic right inverse $q(\boldsymbol{v}|\boldsymbol{x})$ with support over the region $\mathcal{S} = \{\boldsymbol{v} \in \mathbb{R}^d : \boldsymbol{x} = \arg\max \boldsymbol{v}\}$ where $\boldsymbol{v} \sim p(\boldsymbol{v})$ can be any underlying continuous distribution. Such a latent variable model induces the following discrete distribution P_{model}:

$$P_{\mathsf{model}}(\boldsymbol{x}) := \int P(\boldsymbol{x}|\boldsymbol{v})p(\boldsymbol{v}), \qquad P(\boldsymbol{x}|\boldsymbol{v}) := \delta\big(\boldsymbol{x} = \arg\max(\boldsymbol{v})\big) \tag{3}$$

where $p(\boldsymbol{v})$ may be modelled by any continuous distribution, such as a normalizing flow. Importantly, $P(\boldsymbol{x}|\boldsymbol{v})$ denotes a Kronecker delta peak such that $P(\boldsymbol{x}|\boldsymbol{v}) = \delta(\boldsymbol{x} = \arg\max \boldsymbol{v})$. Intuitively, one can see $P(\boldsymbol{x}|\boldsymbol{v})$ as partitioning the space $\mathcal{V}$ for different values of $\boldsymbol{x}$. To be precise, we define the (elementwise) argmax operation as:

$$\arg\max : \qquad \mathbb{R}^{D\times K} \to \{1, \dots, K\}^D : \qquad \boldsymbol{v} \mapsto \Big(\underset{k\in\{1,\dots,K\}}{\arg\max}\ v_{d,k}\Big)_d, \tag{4}$$

assigning for each dimension d separately the index k_d such that $v_{d,k_d} \geq v_{d,k}$ for all $k = 1, \dots, K$. Consequently, in concise notation, the corresponding categorical variable is defined to be $\boldsymbol{x} = \arg\max \boldsymbol{v}$, where $\boldsymbol{x} \in \{1, \dots, K\}^D$. See Fig. 1 for an illustration.

The main difficulty lies in *optimizing* this generative model. Suppose one would naïvely choose any variational distribution, then a sample $\boldsymbol{v} \sim q(\boldsymbol{v}|\boldsymbol{x})$ may lead to samples where there is no probability at all because $\delta(\boldsymbol{x} = \arg\max \boldsymbol{v}) = 0$. Instead, we need to learn the probabilistic right-inverses to the generative argmax function. In other words, for any sample $\boldsymbol{v} \sim q(\boldsymbol{v}|\boldsymbol{x})$ it is desired that $\delta(\boldsymbol{x} = \arg\max \boldsymbol{v}) = 1$. Recall that under this condition, the expected lowerbound (ELBO) can be simplified as in Equation 2.

3.1. Dequantization by asymptotic thresholding

A relatively straightforward method to construct a distribution satisfying the argmax constraint, is by thresholding values using injective functions. More concretely, assume a distribution with infinite support $q(\boldsymbol{u}|\boldsymbol{x})$ where $\boldsymbol{u} \in \mathbb{R}^{D\times K}$, for example a Gaussian distribution or a normalizing flow. We can map $\boldsymbol{u}$ injectively to an argmax partition using a threshold function and $\boldsymbol{x}$. The injectivity of the map is important, because in that case the likelihood $q(\boldsymbol{v}|\boldsymbol{x})$ is easily computed via the change of variables formula $q(\boldsymbol{u}|\boldsymbol{x})\left|\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\boldsymbol{v}}\right|$. In our implementation thresholding is implemented using a softplus function such that all values are mapped below a limit T.

$$v = \text{threshold}(u, T) = -\text{softplus}(-(u - T)) + T, \quad \text{where} \quad \text{softplus}(x) = \log(1 + e^x), \tag{5}$$

for which it is guarenteed that $v \in (-\infty, T)$. In particular, the variable $\boldsymbol{u}$ is injectively mapped to $\boldsymbol{v}$ such that $v_{d,k} = u_{d,k}$ if x_d equals k and otherwise $v_{d,k} = \text{threshold}(u_{d,k}, u_{d,x_d})$ if $k \neq x_d$ where x_d is used as an index. In other words, all values except for the argmax indices are thresholded to be below the argmax values.

3.2. Alternative methods for dequantization

For a detailed description of alternative dequantization methods based on Gumbel reparametrization see Appendix A. Further, we outline a method to trade-off between symmetry and the number of dimensions in Argmax Flows, which we term Cartesian products of Argmax Flows (further details in Appendix A).

4. Experiments

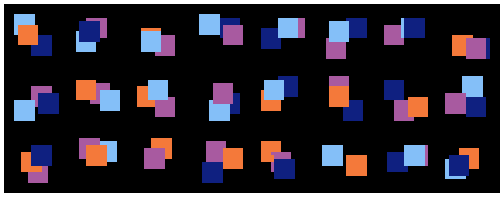
In this section we compare the performance of our method to alternative dequantization methods, a standard VAE baseline with flexible posterior and prior, and to a VAE-based latent normalizing flow approach. In the first experiment, we fit a toy 50 class problem using maximum log-likelihood using different dequantization methods. As can be seen in Table 6, our proposed Argmax based methods perform better than existing approaches

Table 2: Comparison of dequantization methods on a toy 50 class categorical distribution problem, in bits.

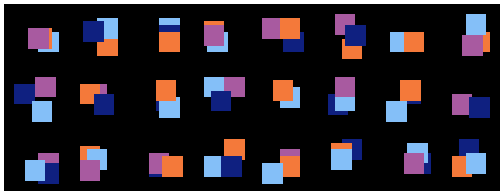
Model	ELBO	IWBO
Hypercube / Uniform (Uria et al., 2013)	9.78	7.64
Hypercube / Var. (Ho et al., 2019)	5.32	4.91
Argmax / Asymptotic thresholding (ours)	5.00	**4.82**
Argmax / Gumbel distribution (ours)	**4.86**	**4.82**
Argmax / Gumbel thresholding (ours)	4.88	**4.82**
Data Entropy	4.82	

4.1. Image data

To show that our method also generalized to images, we introduce an *unconditional* image segmentation learning experiment. In contrast with the standard setting where the

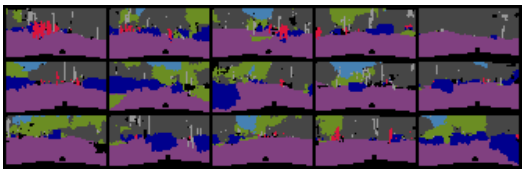


(*a*) Samples from Argmax Flow using Gumbel dequantization.

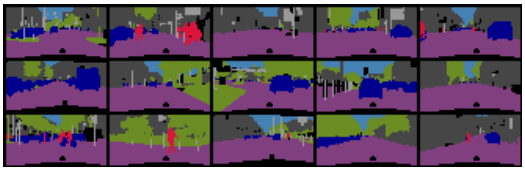


(*b*) Squares segmentation data.

Figure 2: Visualization of samples of the squares segmentation data.



(*a*) Samples from Argmax Flow using thresholding dequantization.



(*b*) Cityscapes data.

Figure 3: Note that the model is trained on segmentations *unconditionally*, that is there is no photograph which the model is conditioned on.

prediction is conditioned on photographs, this problem is designed to be more complicated by learning a distribution over the target *without* this photograph. We introduce a 5-class toy segmentation problem named "Squares" (see Figure 2). Further, we rescale segmentation maps from cityscapes to 32×64 images and use the global category for an 8-class segmentation problem. The results of these experiments are depicted in Table 3. Results depicted are the ELBO and IWBO (with 1000 samples) measured in bits per dimension. The thresholded-based argmax dequantization peforms best in terms of ELBO values, and Gumbel-thresholded dequantization performs best in terms of IWBO evaluation for cityscapes.

Table 3: Performance of different dequantization methods on squares and cityscapes dataset, in bits per dimension for the ELBO and (IWBO) in parentheses. Lower is better.

Model	Rectangles	Cityscapes
Argmax / Gumbel (ours)	0.105 (0.089)	0.307 (**0.287**)
Argmax / Threshold (ours)	**0.098** (0.088)	**0.303** (0.290)
Hypercube / Uniform (Uria et al., 2013)	0.303 (0.220)	1.011 (0.930)
Hypercube / Var. (Ho et al., 2019)	0.102 (0.088)	0.334 (0.315)
VAE (Flow prior, Flow var. posterior)	0.101 (0.089)	0.306 (0.293)

4.2. Text data

In this section we learn a normalizing flow on the text8 dataset. Instead of using a $K = 27$ argmax space, we find empirically that using a 5×2-binary space with threshold dequantization (see Appendix A) work better. Further, the same density model as proposed in

(Lippe and Gavves, 2020) is utilized. This experiment shows that Argmax Flow achieve lower bits per character (bpc) than CategoricalNF on text8 (see Table 4).

Table 4: Comparison of dequantization methods on text8 dataset, in bits. Both models use the same underlying density model from CategoricalNF. Lower is better.

Model	ELBO	IWBO
CategoricalNF (Lippe and Gavves, 2020)	1.45 bpc	-
Argmax Flow (ours)	**1.43** bpc	**1.43** bpc

5. Related Work

Deep generative models broadly fall into the categories autoregressive models ARMs (Germain et al., 2015), Variational Autoencoders (VAEs) (Kingma and Welling, 2014), Adversarial Network (GANs) (Goodfellow et al., 2014), Normalizing Flows Rezende and Mohamed (2015) and Diffusion Models (Sohl-Dickstein et al., 2015; Ho et al., 2020). Normalizing Flows and Diffusion models typically learn a continuous distribution and dequantization is required to train these methods. A large body of work is dedicated to building more expressive continuous normalizing flows (Dinh et al., 2017; Germain et al., 2015; Kingma et al., 2016; Papamakarios et al., 2017; Chen et al., 2018; Song et al., 2019; Perugachi-Diaz et al., 2020).

To learn ordinal discrete distribution, adding uniform noise in-between ordinal classes was proposed in (Uria et al., 2013) and later theoretically justified in (Theis et al., 2016). An extension for more powerful dequantization based on variational inference was proposed in Ho et al. (2019). Dequantization for binary variables was proposed in (Winkler et al., 2019).

In other works, VAEs have been adapted to learn a normalizing flow for the latent space (Ziegler and Rush, 2019; Lippe and Gavves, 2020). However, these approach typically still utilize an argmax heuristic to sample, even though this is not the distribution specified during training. Further, Tran et al. (2019) propose invertible transformations for categorical variables directly, but results on images have thus far not been demonstrated. In addition flows for ordinal discrete data (integers) have been explored in (Hoogeboom et al., 2019; van den Berg et al., 2020)

6. Conclusion

In this paper we introduce a principled method to train continuous distributions such as flows on categorical data. The generative model is defined by an argmax function which can be evaluated using a variational distribution over right-inverses. Different from other approaches is that our method does not require a stochastic decoder in the generative process, and as a result our model does not suffer from undesired uncorrelated noise in the model distribution. We demonstrate that our method performs competitively on similar approaches on unconditional image segmentation and text.

References

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pages 6572–6583, 2018.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *5th International Conference on Learning Representations, ICLR*, 2017.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889, 2015.

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

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. *36th International Conference on Machine Learning*, 2019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

Emiel Hoogeboom, Jorn W. T. Peters, Rianne van den Berg, and Max Welling. Integer discrete flows and lossless compression. In *Neural Information Processing Systems 2019, NeurIPS 2019*, pages 12134–12144, 2019.

Emiel Hoogeboom, Taco S. Cohen, and Jakub M. Tomczak. Learning discrete distributions by dequantization. *CoRR*, abs/2001.11235, 2020.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, 2014.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.

Phillip Lippe and Efstratios Gavves. Categorical normalizing flows via continuous transformations. *CoRR*, abs/2006.09790, 2020.

Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. Survae flows: Surjections to bridge the gap between vaes and flows. *CoRR*, abs/2007.02731, 2020.

George Papamakarios, Iain Murray, and Theo Pavlakou. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, pages 2338–2347, 2017.

Yura Perugachi-Diaz, Jakub M. Tomczak, and Sandjai Bhulai. i-densenets. *CoRR*, abs/2010.02125, 2020.

Danilo Rezende and Shakir Mohamed. Variational Inference with Normalizing Flows. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538. PMLR, 2015.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, 2015.

Yang Song, Chenlin Meng, and Stefano Ermon. Mintnet: Building invertible neural networks with masked convolutions. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, 2019.

L. Theis, A. van den Oord, and M. Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations*, 2016.

Dustin Tran, Keyon Vafa, Kumar Agrawal, Laurent Dinh, and Ben Poole. Discrete flows: Invertible generative models of discrete data. *ICLR 2019 Workshop DeepGenStruct*, 2019.

Benigno Uria, Iain Murray, and Hugo Larochelle. Rnade: The real-valued neural autoregressive density-estimator. In *Advances in Neural Information Processing Systems*, pages 2175–2183, 2013.

Rianne van den Berg, Alexey A. Gritsenko, Mostafa Dehghani, Casper Kaae Sønderby, and Tim Salimans. IDF++: analyzing and improving integer discrete flows for lossless compression. *CoRR*, abs/2006.12459, 2020.

Christina Winkler, Daniel E. Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *CoRR*, abs/1912.00042, 2019.

Zachary M. Ziegler and Alexander M. Rush. Latent normalizing flows for discrete sequences. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7673–7682. PMLR, 2019.

Appendix A. Further methods for dequantization

A.1. Dequantization with Gumbel distributions

An alternative method to construct $q(\boldsymbol{v}|\boldsymbol{x})$ is to first define a marginal $q(\boldsymbol{v})$ from which we can sample conditioned on the argmax $q(\boldsymbol{v}|\arg\max \boldsymbol{v} = \boldsymbol{x})$. A natural distribution to do such manipulations is the Gumbel distribution because of its favourable properties: the arg max and max are independent and the max is also distributed as a Gumbel distribution. As a result it is straightforward to sample from a Gumbel distribution conditioned on its argmax. For simplicity in notation, we will write that $\boldsymbol{v} \in \mathbb{R}^K$ and $x \in \{1, \dots, K\}$ without dimension d, as the distribution is defined independently over the dimension axis d. We first define a distribution for $\boldsymbol{v}$ as the Gumbel distribution with location parameters $\boldsymbol{\phi} \in \mathbb{R}^K$:

$$\boldsymbol{v} \sim \text{Gumbel}(\boldsymbol{\phi}) \tag{6}$$

Gumbel have the nice property that the argmax and max are independent distributions, and are accessible in closed-form. In particular, the variable $\max_i v_i$ is distributed as a Gumbel distribution itself and importantly does not depend on i:

$$\max_i v_i \sim \text{Gumbel}(\phi_{\max}) \tag{7}$$

where $\phi_{\max} = \log \sum_i \exp \phi_i$. These properties make it very easy to sample *conditionally* from the Gumbel distribution when the argmax index is given by x. Conditioned on x, the variable v_x is distributed as:

$$v_x \sim \text{Gumbel}(\phi_{\max}). \tag{8}$$

Furthermore, given this sampled maximum, the remaining indices can be directly sampled using *truncated* Gumbel distributions:

$$v_i \sim \text{TruncGumbel}(\phi_i; T) \text{ where } i \neq x \tag{9}$$

where the truncation value T is given by v_x. By computing first Equation 8 and subsequently Equation 9 we have drawn our sample $\boldsymbol{v} \sim q(\boldsymbol{v}|\boldsymbol{x})$ for which $\arg\max \boldsymbol{v} = \boldsymbol{x}$. Recall that to optimize Equation 2, the log-likelihood $\log q(\boldsymbol{v}|\boldsymbol{x})$ is also required. This can be computed in closed-form expressions using the log density functions. Another useful property of the Gumbel distribution is that its argmax is distributed as the categorical distribution $P(\arg\max \boldsymbol{v} = i) = \exp \phi_i / \sum_i \exp \phi_i$. As such, the location parameters $\boldsymbol{\phi}$ can be initialized to match the empirical distribution of the first minibatch of the data.

For completeness here follows a quick summary of Gumbel properties: To sample $g \sim \text{Gumbel}(\phi)$, sample $u \sim \mathcal{U}(0, 1)$ and compute $g = -\log(-\log(u)) + \phi$. Further the log-likelihood $\log \text{Gumbel}(g|\phi, 1) = \phi - g - \exp(\phi - g)$. To sample $g \sim \text{TruncGumbel}(\phi, 1; T)$, sample $u \sim \mathcal{U}(0, 1)$ and compute $g = \phi - \log(\exp(\phi - T) - \log(u))$. Further the log-likelihood $\log \text{TruncGumbel}(g|\phi, 1, T) = \exp(\phi - T) - \exp(\phi - g) + \phi - g$ under the condition that $g < T$ and otherwise $-\infty$.

A.2. Unifying Thresholding and Gumbel dequantization into Gumbel Thresholding

This section combines the results from the previous two sections. The key insight is that the Gumbel sampling procedures as defined above can be seen as a reparametrization of a

uniform noise distribution on $(0, 1]^d$ which is put through the inverse CDF of the Gumbel distributions. Additionally, the log-likelihoods can be seen as log-derivatives of the (forward) CDF. Consequently, instead of using uniform noise, we may learn *any* distribution on the interval $(0, 1]^d$ and reparametrize using the same inverse CDF functions, where the log Jacobian determinant is equal to the log likelihood at that point. The idea is that the smoothness of the Gumbel distribution is retained more while correlations across dimensions can still be learned by the interval distribution. Compared to the plain Gumbel dequantization, the interval noise can now be conditioned on $\boldsymbol{x}$ and can be further correlated across dimensions d, which leads to more expressive dequantization distributions. Suggestions to learn an interval distribution are (1) to learn a flow with infinite support composed with a sigmoid that injectively maps to $(0, 1)^d$ or (2) learn a flow starting from a uniform distribution that is transformed using interval preserving transformations such as splines.

A.3. Cartesian products of Argmax Flows

In the current description, Argmax flows require the same number of dimensions in $\boldsymbol{v}$ as there are classes in $\boldsymbol{x}$. To alleviate this constraint we introduce Cartesian products of argmax flows. To illustrate our method, consider a 256 class problem. One class can be represented using a single 256-onehot vector, but also using two hexadecimal numbers or alternatively using eight binary numbers. Formally, any categorical variable $\boldsymbol{x}^{(K)} \in \{1, \dots, K\}^d$ in base K can be converted to $\boldsymbol{x}^{(M)} \in \{1, \dots, M\}^{d_m \times d}$ in base $M \geq 2$, where at least $d_m = \lceil {}^M\log K \rceil$ M-categorical variables are required to model a single K-categorical variable. Then the variable $\boldsymbol{x}^{(M)}$ with dimensionality $M \cdot d_m \cdot d$ is dequantized instead of the variable $\boldsymbol{x}^{(K)}$ with dimensionality $K \cdot d$. Even though this may lead to some unused additional classes, the ELBO objective in Equation 2 remains valid and thus the model can be optimized using an M-categorical argmax flow. To illustrate the changes in dimensionality, in Table 5. Finally note that binary class problems are a special case where the variable can be straightforwardly encoded symmetrically into a single dimension. As a consequence the last row in Table 5 is a special case that can alternatively be encoded using only $d_m = 11$ dimensions.

Table 5: Example of the trade-offs when taking Cartesian products of Argmax Flows, in a hypothetical problem with $K = 2000$ classes.

M	d_m	max neighbours	max distance	total dimensions
2000	1	1999	1	$d_m \cdot M = 2000$
45	2	$M = 45$	2	$d_m \cdot M = 90$
13	3	$M = 13$	3	$d_m \cdot M = 39$
2	11	$M = 2$	11	$d_m \cdot M = 22$

A.4. Ablation study: Gumbel distribution versus Gumbel thresholding

Table 6: Ablation on cityscapes.

Model	ELBO	IWBO
Argmax / Gumbel distribution	0.365	0.341
Argmax / Gumbel thresholding	0.307	**0.287**