GFlowOut: Dropout with Generative Flow Networks

Dianbo Liu [1 2] **Moksh Jain** [1 3] **Bonaventure F. P. Dossou** [1 4 5] **Qianli Shen** [6] **Salem Lahlou** [1 3] **Anirudh Goyal** [7] **Nikolay Malkin** [1 3] **Chris C. Emezue** [1 8] **Dinghuai Zhang** [1 3] **Nadhir Hassen** [1 3] **Xu Ji** [1 3] **Kenji Kawaguchi** [6] **Yoshua Bengio** [1 3 9]

Abstract

Bayesian inference offers principled tools to tackle many critical problems with modern neural networks such as poor calibration and generalization, and data inefficiency. However, scaling Bayesian inference to large architectures is challenging and requires restrictive approximations. Monte Carlo Dropout has been widely used as a relatively cheap way to approximate inference and estimate uncertainty with deep neural networks. Traditionally, the dropout mask is sampled independently from a fixed distribution. Recent research shows that the dropout mask can be seen as a latent variable, which can be inferred with variational inference. These methods face two important challenges: (a) the posterior distribution over masks can be highly multi-modal which can be difficult to approximate with standard variational inference and (b) it is not trivial to fully utilize sample-dependent information and correlation among dropout masks to improve posterior estimation. In this work, we propose GFlowOut to address these issues. GFlowOut leverages the recently proposed probabilistic framework of Generative Flow Networks (GFlowNets) to learn the posterior distribution over dropout masks. We empirically demonstrate that GFlowOut results in predictive distributions that generalize better to out-of-distribution data and provide uncertainty estimates which lead to better performance in downstream tasks.

[1]Mila Quebec AI Institute [2]Broad Institute of MIT and Harvard [3]University of Montreal [4]McGill University [5]Lelapa AI [6]National University of Singapore [7]Google DeepMind [8]Technical University of Munich [9]CIFAR AI Chair. Correspondence to: Dianbo Liu <dianbo.liu@mila.quebec>, Moksh Jain <moksh.jain@mila.quebec>.

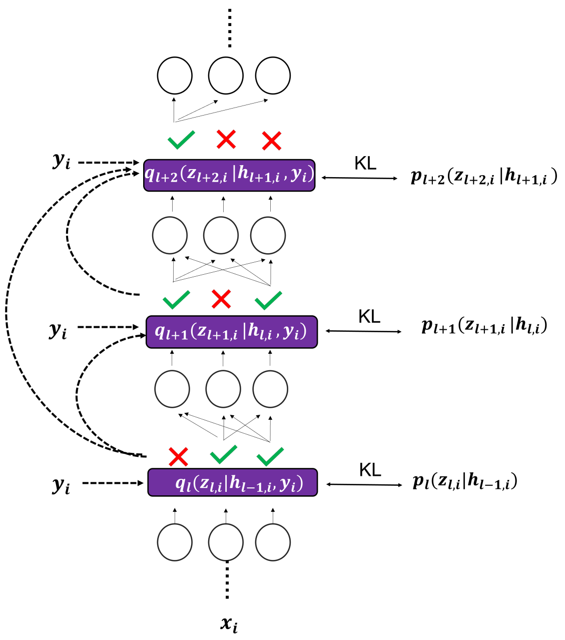
Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

A key shortcoming of modern deep neural networks is that they are often overconfident about their predictions, especially when there is a distributional shift between train and test dataset (Daxberger et al., 2021; Nguyen et al., 2015; Guo et al., 2017). In risk-sensitive scenarios such as clinical practice and drug discovery, where mistakes can be extremely costly, it is important that models provide predictions with reliable uncertainty estimates (Bhatt et al., 2021). Bayesian inference offers principled tools to model the parameters of neural networks as random variables, placing a prior on them and inferring their posterior given some observed data (MacKay, 1992; Neal, 2012). The posterior captures the uncertainty in the predictions of the model and also serves as an effective regularization strategy resulting in improved generalization (Wilson & Izmailov, 2020; Lotfi et al., 2022). In practice, exact Bayesian inference is often intractable and existing Bayesian deep learning methods rely on assumptions that result in posteriors that are less expressive and can provide poorly calibrated uncertainty estimates (Ovadia et al., 2019; Fort et al., 2019; Foong et al., 2020; Daxberger et al., 2021). In addition, even with several approximations, Bayesian deep learning methods are often significantly more computationally expensive and slower to train compared to non-Bayesian methods (Kuleshov et al., 2018; Boluki et al., 2020).

Gal and Ghahramani (2016) show that deep neural networks with dropout perform approximate Bayesian inference and approximate the posterior of a deep Gaussian process (Damianou & Lawrence, 2013). One can obtain samples from this predictive distribution by taking multiple forward passes through the neural network with independently sampled dropout masks. Due to its simplicity and minimal computational overhead, dropout has since been used as a method to estimate uncertainty and improve robustness in neural networks. Different variants of dropout have been proposed and can be interpreted as different variational approximations to model the posterior over the neural network parameters (Ba & Frey, 2013; Kingma et al., 2015; Gal et al., 2017; Ghiasi et al., 2018; Fan et al., 2021; Pham & Le, 2021).

There are a few major challenges in approximating the



y_i
$q_{l+2}(z_{l+2,i}\,|h_{l+1,i}, y_i)$
KL
$p_{l+2}(z_{l+2,i}\,|h_{l+1,i})$
y_i
$q_{l+1}(z_{l+1,i}\,|h_{l,i}, y_i)$
KL
$p_{l+1}(z_{l+1,i}\,|h_{l,i})$
y_i
$q_l(z_{l,i}|h_{l-1,i}, y_i)$
KL
$p_l(z_{l,i}|h_{l-1,i})$
x_i

Figure 1. In this work, we propose a Generative Flow Network (GFlowNet) based binary dropout mask generator which we refer to as **GFlowOut**. Purple squares are GFlowNet-based dropout mask generators parameterized as multi-layer perceptrons. $z_{i,l}$ refers to dropout masks for data point indexed by i at layer l of the model. $h_{i,l}$ refers to activations of the model at layer l given input x_i. $q(\cdot)$ are auxiliary variational functions used and adapted only during model training, in which the posterior distribution over dropout masks is conditioned implicitly on input covariates (x_i) and directly on the label (y_i) of the data point to make the estimation easier. $p(\cdot)$ are mask generation functions used at test time, which are only conditioned on x_i and trained by minimizing the Kullback–Leibler(KL) divergence with $q(\cdot)$. In addition, both $q(\cdot)$ and $p(\cdot)$ conditions explicitly on dropout masks of all previous layers.

Bayesian posterior over model parameters using dropout: (1) the multimodal nature of the posterior distribution makes it difficult to approximate with standard variational inference (Gal & Ghahramani, 2016; Le Folgoc et al., 2021), which assumes factorized priors; (2) dropout masks are discrete objects making gradient-based optimization difficult (Boluki et al., 2020); (3) variational inference methods can suffer from high gradient variance resulting in optimization instability (Kingma et al., 2015); (4) modeling dependence between dropout masks from different layers is non-trivial.

The recently proposed Generative Flow Networks (GFlowNets) (Bengio et al., 2021a;b) frame the problem of generating discrete objects as a control problem based on the sequential construction of discrete components. GFlowNets learn probabilistic policies that sample objects proportional to a reward function (or exp(-energy)). They have demonstrated better generalization to multimodal distributions (Nica et al., 2022) and have lower gradient variance compared with policy gradient-based variational methods (Malkin et al., 2023), making it an interesting choice for posterior inference for dropout.

Contributions. In this work, to address the limitations of standard variational inference, we develop a GFlowNet-based binary dropout mask generator which we refer to as *GFlowOut*, to estimate the posterior distribution of binary dropout masks. GFlowOut generates dropout masks for a layer, conditioned on masks generated for the previous layer, therefore accounting for inter-layer dropout dependence. Furthermore, the GFlowOut estimator can be conditioned on the data point: GFlowOut improves posterior estimation here by utilizing both input covariates and labels in the training set of supervised learning tasks via an auxiliary variational function. To investigate the quality of the posterior distribution learned by GFlowOut, we design empirical experiments, including evaluating robustness to distribution shift during inference, detecting out-of-distribution examples with uncertainty estimates, and transfer learning, using both benchmark datasets and a real-world clinical dataset.

2. Related work

2.1. Dropout as a Bayesian approximation

Deep learning tools have shown tremendous power in different applications. However, traditional deep learning tools lack mechanisms to capture the uncertainty, which is of crucial importance in many fields. Uncertainty quantification (UQ) is studied extensively as a fundamental problem of deep learning and a large number of Bayesian deep learning tools have emerged in recent years. For example, Gal & Ghahramani (2016) showed that casting dropout in deep learning model training is an approximation of Bayesian inference in deep Gaussian processes and allows uncertainty estimation without extra computational cost. Kingma et al. (2015) proposed variational dropout, where a dropout posterior over parameters is learned by treating dropout regularization as approximate inference in deep models. Gal et al. (2017) developed a continuous relaxation of discrete dropout masks to improve uncertainty estimation, especially in reinforcement learning settings. Lee et al. (2020) introduced "meta-dropout", which involves an additional global term shared across all data points during inference to improve generalization. Xie et al. (2019) replaced the hard dropout mask following a Bernoulli distribution with the soft mask following a beta distribution and conducted the optimization using a stochastic gradient variational Bayesian algorithm to control the dropout rate. Boluki et al. (2020) combined a model-agnostic dropout scheme with variational auto-encoders (VAEs), resulting in semi-implicit VAE models. Instead of using mean-field family for variational inference, Nguyen et al. (2021) utilized a structured representation of multiplicative Gaussian noise for better posterior estimation. More recently, Fan et al. (2021) developed "contextual dropout", which optimizes variational objectives in a sample-dependent manner and, to the best of our knowledge, is the closest approach to GFlowOut in the literature. GFlowOut differs from contextual dropout in several aspects. First, both methods take trainable priors into account, but GFlowNet also takes into account priors that depend on the input covariate of each data point. Second, the variational posterior of contextual dropout only depends on the input covariate (x), while in GFlowOut, the variational posterior is also conditioned on the label y, which provides more information for training. Third, within each neural network layer, the mask of contextual dropout is conditioned on previous masks implicitly, while the mask of GFlowOut is conditioned on previous masks explicitly by directly feeding previous masks as inputs into the generator, which improves the training process. Finally, instead of a REINFORCE-based gradient estimator used for contextual dropout training, GFlowOut employs powerful GFlowNets for the variational posterior.

2.2. Generative flow networks

Generative flow networks (GFlowNets) (Bengio et al., 2021a;b) are a family of probabilistic models that amortizes sampling discrete compositional objects proportionally to a given unnormalized density function. GFlowNets learn a stochastic policy to construct objects through a sequence of actions akin to deep reinforcement learning (Sutton & Barto, 2018). GFlowNets are trained so as to make the likelihood of reaching a terminating state proportional to the reward. Recent works have shown close connections of GFlowNets to other generative models (Zhang et al., 2022a) and to hierarchical variational inference (Malkin et al., 2023). GFlowNets achieved great empirical success in learning energy-based models (Zhang et al., 2022b), small-molecule generation (Bengio et al., 2021a; Nica et al., 2022; Malkin et al., 2022; Madan et al., 2023; Pan et al., 2023), biological sequence generation (Malkin et al., 2022; Jain et al., 2022; Madan et al., 2023), and structure learning (Deleu et al., 2022). Several training objectives have been proposed for GFlowNets, including Flow Matching (FM) (Bengio et al., 2021a), Detailed Balance (DB) (Bengio et al., 2021b), Trajectory Balance (TB) (Malkin et al., 2022), and the more recent Sub-Trajectory Balance (SubTB) (Madan et al., 2023). In this work, we use the Trajectory Balance (TB) objective.

3. Method

In this section, we define the problem setting and mathematical notations used in this study, as well as describe the proposed method, GFlowOut, for dropout mask generation in detail.

3.1. Background and notation

Dropout. In a vanilla feed-forward neural network (MLP) with L layers, each layer of the model has weight matrix w_l and bias vector b_l. It takes as input activations h_{l-1} from previous layer with layer index $l-1$, and computes as output $h_l = \sigma(w_l h_{l-1} + b_l)$ where σ is a non-linear activation function. Dropout consists of dropping out units from the output of a layer. Formally this can be described as applying a sampled binary mask $z_l \sim p(z_l)$ on the output of the layer $h_l = z_l \circ \sigma(w_l h_{l-1} + b_l)$, at each layer in the model. In regular random dropout, z_l is a collection of i.i.d. $\mathrm{Bernoulli}(r)$ variables, where r is a fixed parameter for all the layers. Recently, several approaches have been proposed to learn $p(z_l)$ along with the model parameters. In these approaches, z is viewed either as latent variables or part of the model parameters. We consider two variants for our proposed method: GFlowOut where the dropout masks z are viewed as *sample dependent* latent variables, and ID-GFlowOut, which generates masks in a *sample independent* manner where z is viewed as a part of the model parameters shared across all samples. Next, we briefly introduce GFlowNets and describe how they model the dropout masks z given the data.

GFlowNets. Let $G = (\mathcal{S}, \mathcal{A})$ be a directed acyclic graph (DAG) where the vertices $s \in \mathcal{S}$ are *states*, including a special initial state s_0 with no incoming edges, and directed edges $(s \to s') \in \mathcal{A}$ are *actions*. $\mathcal{X} \subseteq \mathcal{S}$ denotes the terminal states, with no outgoing edges. A complete trajectory $\tau = (s_0 \to \ldots s_{i-1} \to s_i \cdots \to z) \in \mathcal{T}$ in G is a sequence of states starting at s_0 and terminating at $z \in \mathcal{X}$ where each $(s_{i-1} \to s_i) \in \mathcal{A}$. The forward policy $P_F(-|s)$ is a collection of distributions over the children of each non-terminal node $s \in \mathcal{S}$ and defines a distribution over complete trajectories, $P_F(\tau) = \prod_{(s_{i-1} \to s_i) \in \tau} P_F(s_i | s_{i-1})$. We can sample terminal states $z \in \mathcal{X}$ by sampling trajectories following P_F. Let $\pi(x)$ be the marginal likelihood of sampling terminal state x, $\pi(z) = \sum_{\tau = (s_0 \to \cdots \to z) \in \mathcal{T}} P_F(\tau)$. Given a non-negative reward function $R : \mathcal{X} \to \mathbb{R}^+$, the learning problem tackled in GFlowNets is to estimate P_F such that $\pi(z) \propto R(z),\ \forall z \in \mathcal{X}$. We refer the reader to Bengio et al. (2021b); Malkin et al. (2022) for a more thorough introduction to GFlowNets.

We adopt the Trajectory Balance (TB) (Malkin et al., 2022) parameterization, which includes $P_F(-|-;\phi), P_B(-|-;\phi)$, and Z_γ, where ϕ and γ

are the learnable parameters. The backward policy P_B is a distribution over parents of every noninitial state, and Z is an estimate of the partition function.

Within the context of generating dropout masks, a complete dropout mask $x \in \mathcal{X}$ is a binary vector of dimension M, where M is the number of units in the neural network, i.e. $\mathcal{X}$ is equal to $\{0,1\}^M$. A partially constructed mask $s \in \mathcal{S}$ is a binary vector of dimension $m < M$ representing the mask for a set of initial layers in the model, and an action consists of appending the mask for the subsequent layer to this vector. That is, each action in the sequence samples the mask for an entire layer (in parallel), conditioned on the masks for the previous layers.

In the next section, we formally describe how GFlowNets can be used for generating dropout masks, as well as practical implementation details.

3.2. GFlowOut

We consider a generative model of the form $p(x,y,z) = p(x)p(z|x)p(y|x,z)$, where x is the input data with corresponding label y and z is a local discrete latent variable representing the sample-dependent dropout mask, along with a dataset of observations $D = \{(x_i,y_i)\}_{i=1}^N$. GFlowOut learns to approximate the posterior $p(z|x,y)$ using the given dataset D. In a supervised learning task where the goal is to learn the predictive distribution $p(y|x)$, with the assumed generative model above, the following variational bound can be derived:

$$\log \prod_{i=1}^{N} p(y_i|x_i) \tag{1}$$

$$= \log \prod_{i=1}^{N} \sum_{z_i \in \mathcal{X}} p(z_i|x_i) p(y_i|x_i, z_i)$$

$$= \log \prod_{i=1}^{N} \sum_{z_i \in \mathcal{X}} p(z_i|x_i) \frac{q(z_i|x_i,y_i)}{q(z_i|x_i,y_i)} p(y_i|x_i, z_i)$$

$$= \sum_{i=1}^{N} \log \mathop{\mathbb{E}}_{q(z_i|x_i,y_i)} \left[\frac{p(z_i|x_i)}{q(z_i|x_i,y_i)} p(y_i|x_i,z_i) \right] \tag{2}$$

$$\geq \sum_{i=1}^{N} \mathbb{E}_{q(z_i|x_i,y_i)} \left[\log \frac{p(z_i|x_i)}{q(z_i|x_i,y_i)} p(y_i|x_i,z_i) \right]$$

$$= \sum_{i=1}^{N} \Big[\mathbb{E}_{q(z_i|x_i,y_i)} [\log p(y_i|x_i,z_i)] - \mathrm{KL}(q(z_i|x_i,y_i) \| p(z_i|x_i)) \Big] \tag{3}$$

where $p(z_i|x_i)$ is part of the generative process and $q(z_i|x_i,y_i)$ is the variational distribution used to approximate the posterior of z_i. To improve the efficiency of training and fully utilize the information available in each

Algorithm 1 GFlowOut

The whole system has the following 3 components:

- *Backbone Model* (eg, classifier) neural network $p(y_i|x_i,z_i;\theta)$.This algorithm section is written assuming $p(y_i|x_i,z_i;\theta)$ is an MLP with L hidden layers, but it can be easily extended to other architectures
- *GFlowNet* $q(z_i|x_i,y_i;\phi)$ which approximates the posterior distribution over dropout masks z_i conditioned on both x_i and y_i from the data point. Its tempered version $q^{\sim}(z_i|x_i,y_i;\phi)$ is used for dropout mask sampling during training.
- $p(z_i|x_i;\xi)$, which generates dropout mask distribution only conditioned on x_i, is optimized by minimizing KL divergence with $q(z_i|x_i,y_i;\phi)$ and is used for dropout mask sampling during test time.

$q(z_i|x_i,y_i;\phi)$ and $p(z_i|x_i;\xi)$ are all implemented as groups of MLPs, one MLP for each layer l and they do not share parameters with each other nor between different layers.
Next, we explain how dropout masks are generated and how $p(y_i|x_i,z_i;\theta)$, $q(z_i|x_i,y_i;\phi)$ and $p(z_i|x_i;\xi)$ are computed.

```
for epoch do
  for Iterate data point $x_i, y_i$ do    ▷ (batches used in actual training)
    $h_0 = x_i$
    $var1 = 0$  ▷ Variables to store probabilities from each layer
    $var2 = 0$
    for layer $l$ in $1 : L-1$ do
      Use current layer's activation and dropout masks of all previous layers for mask generation
      $h'_{l,i} = \mathrm{ReLU}(b_l + w_l h_{l-1,i})$
      $z_{i,l} \sim q_l^{\sim}(z_{i,l}|h'_{l,i}, y_i, (z_{i,j})_{j=1}^{l-1}; \phi)$ ▷ dropout masks generated
      $h_{l,i} = z_{i,l} h'_{l,i}$    ▷ apply dropout
      $var1 += \log q_l(z_{i,l}|x_i, y_i, (z_{i,j})_{j=1}^{l-1}; \phi)$    ▷ calculate log probabilities
      $var2 += \log p_l(z_{i,l}|x_i(z_{i,j})_{j=1}^{l-1}; \xi)$
    end for
    $\log q(z_i|x_i,y_i;\phi) = var1$
    $\log p(z_i|x_i;\xi) = var2$

    In the output layer, $\hat{y_i} = f_{\mathrm{out}}(b_L + w_L h_{L-1,i})$
    where $f_{out}$ is the output non-linearity of the output layer
    Update $\theta, \phi, \xi, \omega, \gamma$ using equations 4-12
  end for
end for
```

data point, we design $q(z_i|x_i, y_i)$ so that the distribution of z_i is conditioned on both x_i and y_i. As a consequence, $q(z_i|x_i, y_i)$ is not accessible during inference where y_i is not available. Instead, $p(z_i|x_i)$, which is trained by minimizing KL divergence with $q(z_i|x_i, y_i)$, is used for inference.

Parametrizing each of the terms as $p(y_i|x_i, z_i; \theta)$, $q(z_i|x_i, y_i; \phi)$ and $p(z_i|x_i; \xi)$, the goal is to maximize the lower bound derived above, which we denote as:

$$\mathcal{B}(\mathcal{D}; \theta, \phi, \xi) = \sum_{i=1}^{N} \Bigg[\mathop{\mathbb{E}}_{q(z_i|x_i,y_i;\phi)} [\log p(y_i|x_i, z_i; \theta)] \\ - \mathrm{KL}(q(z_i|x_i, y_i; \phi) \| p(z_i|x_i; \xi)) \Bigg] \quad (4)$$

The gradients of $\mathcal{B}$ with respect to its parameters are:

$$\nabla_\theta \mathcal{B}(\mathcal{D}; \theta, \phi, \xi) = \sum_{i=1}^{N} \nabla_\theta \mathbb{E}_{q(z_i|x_i,y_i)} [\log p(y_i|x_i, z_i; \theta)]$$

$$\nabla_\xi \mathcal{B}(\mathcal{D}; \theta, \phi, \xi) = \sum_{i=1}^{N} \nabla_\xi \mathbb{E}_{q(z_i|x_i,y_i)} [\log p(z_i \mid x_i; \xi)]$$

The gradient of the variational objective $\mathcal{B}$ with respect to ϕ requires a score function estimator, which is known to suffer from high gradient variance (Malkin et al., 2023). Instead of directly optimizing $\mathcal{B}$ with respect to ϕ, we first observe that $\mathcal{B}$ can be written as:

$$\mathcal{B}(\mathcal{D}) = \sum_{i=1}^{N} (\log p(y_i|x_i) - \mathrm{KL}(q(z_i|x_i, y_i) \| p(z_i|x_i, y_i))) ,$$

making $p(z_i|x_i, y_i)$, the true posterior, a target for the variational distribution $q(z_i|x_i, y_i)$. We thus propose to use a GFlowNet with the Trajectory Balance loss to train $q(z_i|x_i, y_i; \phi)$ to match its target, given by its unnormalized density $R = p(y_i|x_i, z_i)p(z_i|x_i)$. As binary dropout masks z are high-dimensional discrete objects that can be constructed sequentially, we consider them as the terminating states of a GFlowNet, and instead of learning a distribution over these terminating states directly, we exploit the DAG structure to learn a forward policy $P_F(-|-; \phi)$, for which the terminating state distribution is $q(z_i|x_i, y_i; \phi)$. The Trajectory Balance loss requires an additional parameter Z_γ, to train $q(z_i|x_i, y_i; \phi)$ ((Bengio et al., 2021b; Malkin et al., 2022)).

Corresponding trajectory balance loss for a trajectory $\tau = (s_0, ...s_L)$ w.r.t. $(Z_\gamma, P_F(-|-; \phi))$ will be

$$\mathcal{L}_{TB}(\tau, \mathcal{D}; \phi, \gamma) = \left(\log \frac{Z_\gamma \prod_{t=1}^{L} P_F(s_t|s_{t-1}; \phi)}{R} \right)^2 \quad (5)$$

where a state s_l in the GFlowNet graph refers to the set of dropout masks sampled by the GFlowNet from layer 1 to l of the model $((z_j)_{j=1}^{l})$. L is the number of layers involving dropout. s_L indicates the termination of the trajectory sampling process. $P_F(s_t|s_{t-1}; \phi)$ refers to the forward policy in GFlowNets [1]. $\log Z_\gamma = f(x_i, y_i; \gamma)$ is the partition function estimator with parameters γ conditioned on both x_i and y_i from the data point indexed by i. Its parameter γ is trained together with ϕ. R is the reward calculated from the likelihood of the data and the prior distribution of the states which are sets of dropout masks (see equation 12).

The parameters ϕ and γ are updated by taking gradient steps on $\mathcal{L}_{TB}(\tau, \mathcal{D}; \phi, \gamma)$ for τ sampled from some training policy. We choose to make the training policy a tempered version of $q(z_i|x_i, y_i; \phi)$, denoted $q^{\sim}(z_i|x_i, y_i; \phi)$. The expected gradient update is thus equal to

$$\sum_{i=1}^{N} \mathop{\mathbb{E}}_{z_i \sim q^{\sim}(\cdot|x_i,y_i;\phi)} \nabla_{\phi,\gamma} \big(\log Z_\gamma + \log q(z_i|x_i, y_i; \phi) \\ - \log R \big)^2 \quad (6)$$

where

$$\log R = \log p(y_i|x_i, z_i; \theta) + \log p(z_i|x_i, \xi).$$

During inference one estimates the posterior predictive as $p^{pred}(y_i|x_i) = \frac{1}{M} \sum_{j=1}^{M} p(y_i|x_i, z_j; \theta)$ where M different z_j are sampled from the $p(z_i|x_i; \xi)$ distribution.

Implementation details. Algorithm 1 presents a high-level overview of the GFlowOut implementation. $q(z_i|x_i, y_i; \phi)$ is implemented as a set of multiple MLPs, one for each layer in the model that requires dropout mask generation (see Figure 1). At layer l of the model, the dropout probabilities of all units in layer l are estimated in parallel conditioned on previous layer's activation $h_{l-1,i}$, the label y_i, and all dropout mask in layers before l using $q_l(z_{i,l}|h_{l-1,i}, y_i, (z_{i,j})_{j=1}^{l-1}; \phi)$ which is parameterized as an MLP. The same process is repeated for each layer in the model. In this way, dropout mask probability at layer l takes into consideration input x_i through $h_{l-1,i}$, label y_i and joint probability with all dropout mask in previous layers, but are independent of masks in the same layer. $p(z_i|x_i; \xi)$ follows the same implementation except that it is not conditioned on y_i and hence it can be used at test time for prediction. In convolutional neural networks, GFlowOut is implemented in a similar manner except that the units are dropped out channel-wise (Yang et al., 2020; Park & Kwak, 2016). Early stopping based on performance on the validation set is used to prevent overfitting. Details of the computational efficiency of GFlowOut are discussed in the Appendix.

[1]As the graph G used in this study has a tree structure, the backward policy $P_B(s_{t-1}|s_t)$ is a constant so we leave it out of the equations.

3.3. ID-GFlowOut: GFlowOut without sample-dependent information

To understand if the sample-dependent information is needed, we introduce a variant of GFlowOut that only uses sample-independent information and keeps the rest of the algorithm as close to GFlowOut as possible for comparison.

Consider a generative model of the form $p(x, y, z) = p(x)p(z)p(y|x, z)$. Given a supervised learning task $p(y|x)$, we generate a dropout mask z that is *not* conditioned on the data point. We use $q(z)$ to approximate the posterior of z which can be seen as part of the model parameters shared by all data points. The following equations can be derived:

$$\begin{aligned}\log \prod_{i=1}^{N} p(y_i|x_i) &= \log \sum_z p(z) \prod_{i=1}^{N} p(y_i|x_i, z) \\ &= \log \sum_z p(z) \frac{q(z)}{q(z)} \prod_{i=1}^{N} p(y_i|x_i, z) \\ &= \log \mathop{\mathbb{E}}_{z\sim q(z)} [\frac{p(z)}{q(z)} \prod_{i=1}^{N} p(y_i|x_i, z)] \\ &\geq \mathop{\mathbb{E}}_{z\sim q(z)} \log \left[\frac{p(z)}{q(z)} \prod_{i=1}^{N} p(y_i|x_i, z)\right] \\ = \mathop{\mathbb{E}}_{z\sim q(z)} &\left[\sum_{i=1}^{N} \log p(y_i|x_i, z)\right] - \mathrm{KL}(q(z)\|p(z)) \end{aligned} \tag{7}$$

where the same distribution $p(z)$ is shared across the whole dataset and q is conditional on the whole data set implicitly.

$$\begin{aligned}\mathcal{B}(\mathcal{D}; \theta', \phi') = \mathop{\mathbb{E}}_{q(z;\phi')} &\sum_{i=1}^{N} \log p(y_i|x_i, z; \theta') \\ &- \mathrm{KL}(q(z; \phi')||p(z))\end{aligned} \tag{8}$$

Where $\mathcal{B}$ is a lower bound. We parameterized each of the terms as $p(y|x, z; \theta')$ and $q(z; \phi')$. $p(z)$ is set as fixed prior to each unit of the dropout rate of 0.5. The gradients for stochastic optimization of θ can be obtained as

$$\nabla_\theta \mathcal{B}(\mathcal{D}; \theta', \phi', \xi') = \sum_{i=1}^{N} \mathop{\mathbb{E}}_{z\sim q(z;\phi')} \nabla_{\theta'} \log p(y_i|x_i, z; \theta') \tag{9}$$

The distribution $q(z; \phi')$ can be trained as a the policy of a GFlowNet, using a tempered version $q^{\sim}(z; \phi')$ to sample trajectories for training. The expected update direction for ϕ' and γ' can be shown to equal

$$\sum_{i=1}^{N} \mathop{\mathbb{E}}_{z_i\sim q^{\sim}(z_i;\phi')} \nabla_{\phi,\gamma'} (\log Z_{\gamma'} + \log q(z_i; \phi') - \log R)^2, \tag{10}$$

where the log-reward for is

$$\log R = N \log p(y_i|x_i, z_i; \theta') + \log p(z_i). \tag{11}$$

In ID-GFlowOut, $\log Z_{\gamma'} = \gamma'$ and does not condition on any inputs.

During inference, the posterior predictive estimate is $p^{pred}(y_i|x_i) = \frac{1}{M} \sum_{j=1}^{M} p(y_i|x_i, z_j; \theta')$ where M different z_j are sampled from the $q(z; \phi')$ distribution.

4. Experiments

In this section, we empirically evaluate GFlowOut[2] on a variety of tasks to understand its ability to generalize across different distributions and estimate uncertainties in prediction. We first evaluate the generalization performance of the posterior predictive approximated by GFlowOut on an image classification task. We also evaluate the efficacy of GFlowOut in the context of transfer learning. To understand the performance of GFlowOut when used in larger models and datasets, we conduct Visual Question Answering (VQA) experiments using Transformer architectures. Next, we evaluate the uncertainty captured by the posterior to detect out-of-distribution examples. Finally, we study a potential application of GFlowOut in a real-world clinical use case for the cross-hospital prediction of mortality in intensive care units (ICUs). We supplement these results with further analysis and additional experimental details in the Appendix.

Robustness to data distribution shift. To evaluate the robustness of GFlowOut to distribution shifts between the train and test data, we study its predictive performance on OOD examples. We conduct experiments on MNIST, CIFAR-10, and CIFAR-100 datasets with different types and levels of deformations. For MNIST, we train a two-layer MLP with 300 and 100 units respectively and evaluate predictions on MNIST images rotated by a uniformly sampled angle $(0-360°)$. Similarly, we use the ResNet-18 (He et al., 2016) models for the CIFAR-10/CIFAR-100 datasets and evaluate their robustness to distribution shifts induced by random rotations. Additionally, we consider "Snow", "Frost" and Gaussian noises image corruptions (Hendrycks & Dietterich, 2019), and analyze the robustness of models to each type of deformation applied with varying intensities. We consider both GFlowOut and ID-GFlowOut variants and as baselines use Random Dropout (Standard Bernoulli Dropout) (Hinton et al., 2012), Contextual Dropout (Fan et al., 2021) and Concrete Dropout (Gal et al., 2017). The results, as summarized in Table 1, show that models trained using GFlowOut are in general more robust to random rotations, and GFlowOut outperforms (or at least matches the performance of) baselines

[2]Code is available at `https://github.com/kaiyuanmifen/GFNDropout`

Table 1. Performance on clean and corrupted data to evaluate the robustness of models trained with different dropout methods to random image rotations at test time.

Data	Method	Acc.	Acc.(rotated)
CIFAR-10	Concrete	90.13	30.68
	Contextual	90.12	29.11
	Random	91.47	27.04
	ID-GFlowOut	**91.85**	30.57
	GFlowOut	91.52	**31.00**
CIFAR-100	Concrete	62.49	16.43
	Contextual	62.30	15.93
	Random	67.19	15.70
	ID-GFlowOut	**69.99**	16.29
	GFlowOut	69.80	**17.01**
MNIST	Concrete	97.36	66.10
	Contextual	**98.15**	66.20
	Random	87.38	43.96
	ID-GFlowOut	97.05	**70.19**
	GFlowOut	96.75	66.41

in five out of six experiments with different levels of corruption (Figure 2, Appendix Figure 3 and Figure 4). These observations suggest that models trained with GFlowOut are more robust to distribution shifts as compared to the baselines. Better generalization performance to distribution shifts indicates that GFlowOut potentially learns a better approximation of the Bayesian posterior over the model parameters.

Visual Question Answering task using transformer architecture To evaluate GFlowOut on large-scale tasks with larger models, we consider a transformer-based multi-modal architecture MCAN (Yu et al., 2019) for the Visual Question Answering (VQA) task, following Fan et al. (2021). The task involves answering a textual question related to the content of a given image. There are three types of questions in the task, namely binary yes/no questions, numerical questions, and other questions. Dropout is applied on cross-modal attention between images and texts, within data type self-attention, and the feed-forward layers after attention. Our experimental results in Table 2 suggest that GFlowOut either outperforms or matches the performance of contextual and concrete dropout when tested on generalization to noisy dataset where a Gaussian noise is added to the visual inputs (Fan et al., 2021).

Uncertainty estimation for out-of-distribution detection. Another way to evaluate the quality of the learned posterior is to analyze the uncertainty estimates on a downstream task. We consider the standard task of using uncertainty estimates for detecting out-of-distribution (OOD) examples (Nado et al., 2021). The intuition is that a well-calibrated model should produce uncertainty in predictions on OOD examples. This can be useful in cases where difficult OOD examples can be delegated to humans for more careful consideration. As in the previous experiments, we consider ResNet-18 models for CIFAR-10/CIFAR-100 classification and compute uncertainty estimates on the CIFAR-10/CIFAR-100 and SVHN (OOD) test sets. Uncertainty for prediction on each example is calculated using the Dempster-Shafer metric (Sensoy et al., 2018). For baselines, we consider Contextual Dropout and Concrete Dropout, along with standard MC Dropout and Deep Ensembles which are strong baselines for this task. We run the experiment with 5 seeds and report the mean and standard error. We study both GFlowOut with sample-dependent information and ID-GFlowOut with only sample-independent information. In Table 3, we present AUPR and AUROC for in-distribution classification (CIFAR-10 and CIFAR-100) and OOD classification (SVHN) using the uncertainty estimates from each method. We observe that GFlowOut outperforms the other dropout baselines with both CIFAR-10 and CIFAR-100 as the training dataset, indicating that sample-dependent information used in GFlowOut results in more calibrated uncertainty estimates. ID-GFlowOut performs well on CIFAR-100 but performs poorly on CIFAR-10. Results of deep ensembles, which is a widely used state-of-the-art uncertainty estimation method, is also reported in Table 3 for comparison.

Table 2. Performance on different question types in Visual Question Answering task with a Transformer-based model trained with different methods.

Method	Test Set	Acc.(All)	Acc.(Yes/No)	Acc.(Number)	Acc.(Other)
Ide-GFlowOut		66.66	84.21	48.99	**58.42**
GFlowOut		66.12	83.91	49.01	58.33
Contextual	Original	66.89	84.48	**49.04**	58.24
Concrete		**66.92**	**84.51**	48.66	58.38
Ide-GFlowOut		50.27	**74.01**	32.16	36.12
GFlowOut		**50.33**	73.31	**32.64**	**40.17**
Contextual	Noisy	49.72	73.81	32.4	35.97
Concrete		50.2	73.5	31.45	37.39

Adaptation after training on noisy data. Next, we evaluate the ability of models trained with GFlowOut to adapt quickly after being trained on noisy data. Concretely, during training we add label noise i.e., we randomly assign the labels for a fraction (30%) of points in the training set and then re-train the classifier on a small fraction of the dataset with clean labels. We adopt the same experimental setup as the previous experiments. We consider ResNet-18 models trained on CIFAR-10/CIFAR-100 datasets. As baselines, we again use Contextual Dropout and Concrete Dropout. Figure 2 shows that the models trained with GFlowOut perform adapt faster than the dropout methods we use as a baseline. We also observe that both the sample-dependent and sample-independent variants of GFlowOut achieve similar performance.

Application on real-world clinical data. We explore the

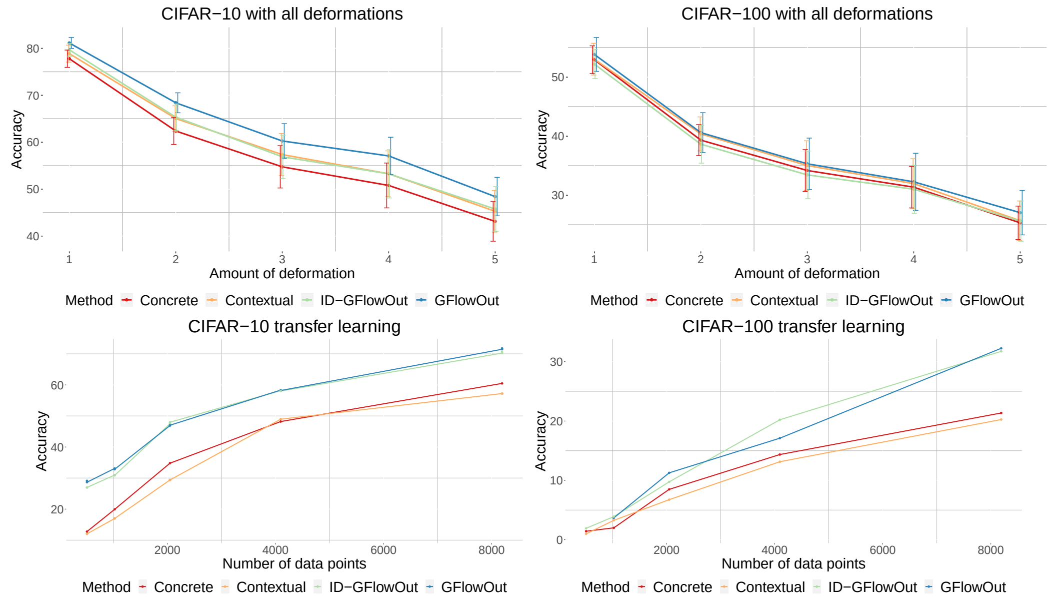


Figure 2. Top: Evaluating the robustness of ResNet-18 models, trained with different dropout methods, to different amounts of deformation including SNOW deformation, FROST deformation or Gaussian noise on CIFAR-10/CIFAR-100 at test time. See Figure 3 and 4 in Appendix for detailed results. *Bottom*: Evaluating the transfer learning performance of ResNet-18 models trained on CIFAR-10/CIFAR-100 with label noise and fine-tuned on varying amounts of clean data (i.e., without any label noise).

Table 3. Performance on the out-of-distribution (OOD) detection task indicates the posterior approximated by GFlowOut provides better uncertainty estimates.

Data	**Method**	**AUROC**	**AUPR**
CIFAR-10	Concrete	0.909± 0.021	0.872± 0.017
	Contextual	0.915± 0.011	0.874± 0.014
	MC Dropout	0.882± 0.009	0.869± 0.010
	Deep Ensembles	0.935± 0.007	0.912± 0.006
	ID-GFlowOut	0.781± 0.021	0.843± 0.018
	GFlowOut	**0.955± 0.013**	**0.924± 0.019**
CIFAR-100	Concrete	0.795± 0.017	0.691± 0.021
	Contextual	0.812± 0.013	0.728± 0.014
	MC Dropout	0.783± 0.011	0.715± 0.021
	Deep Ensembles	**0.842± 0.008**	0.731± 0.017
	ID-GFlowOut	0.819± 0.008	0.707± 0.012
	GFlowOut	**0.839± 0.010**	**0.741± 0.012**

competence of GFlowOut as a probabilistic tool for solving real-world problems. In intensive care units (ICUs), the ability to forecast the mortality of patients can help clinicians to allocate limited resources to help the individuals at the highest risk. However, to respect patient privacy, most hospitals have access only to data for a limited number of patients that is anonymized and available for training predictive models. Moreover, there are stringent regulations on the exchange of medical records among hospitals. To enable data-driven decision-making in these critical scenarios, we consider learning probabilistic classifiers for the problem of mortality predictions. We use patients' medication usage in the first 48 hrs of ICU stay to make a binary prediction of mortality during the stay. We emphasize that the predictions are meant to help decision-makers (doctors) in making clinical decisions rather than being used directly. We use a 3-layer MLP trained with 4500 patients' ICU records, including 158 deaths, from one hospital, and tested with data from another hospital (4018 patients, 147 deaths). Records of both hospitals are obtained from the eICU database (Pollard et al., 2018). This task encompasses several important challenges in applying machine learning tools to real-world tasks: (1) the underlying prediction task is extremely difficult due to limited information and complex case-specific clinical details, (2) the data distribution is severely imbalanced as deaths are rare events, (3) limited training data resulting in a complex posterior, and (4) large distribution shifts between hospitals. Overall, this cross-hospital task setup is quite challenging. Our results in Table 4, show that GFlowOut significantly outperforms the baselines, in all the metrics. While the margins may appear to be small, in the context of real-world decision-making, they can have a significant impact. The findings demonstrate GFlowOut's effectiveness in addressing risk-averse real-world problems.

Table 4. Performance of methods on cross-hospital mortality prediction on real-world clinical data from ICUs demonstrates superior performance of GFlowOut.

Method	**F1 (Macro)**	**Precision**	**Recall**
Concrete	0.528	0.524	0.641
Contextual	0.521	0.518	0.659
Random	0.49	0.5	0.499
ID-GFlowOut	0.499	0.506	0.534
GFlowOut	**0.536**	**0.528**	**0.681**

5. Conclusion

In this work, we propose GFlowOut, to learn the posterior distribution over dropout masks in a neural network. We evaluate GFlowOut on various downstream tasks such as uncertainty estimation, robustness to distribution shift, and transfer learning, using both benchmark datasets and real-world clinical datasets. Our empirical results show the favorable performance of GFlowOut over related methods like Concrete and Contextual Dropout. Future work should involve combining top-down and bottom-up dropout strategies, applying GFlowOut on larger models with complex architectures, and using it to promote exploration in RL problems where accurate estimation of posterior has shown to enhance sample efficiency (Osband et al., 2013).

Author contributions

D.L., Y.B. and K.K. initialized the project. D.L., M.J., B.D., C.E., Q.S., and S.L. contributed to the implementation and experiments of the project. D.L., M.J., and A.G. designed the experimental studies. D.L., Y.B., A.G., N.M., M.J., X.J., and S.L. contributed to the conceptualization of the project. N.M., S.L., K.K., D.Z., D.L., M.J., Q.S., N.H., and Y.B contributed to the mathematical parts of the project. D.L., A.G., and M.J. coordinated the project. Y.B. supervised the whole project and designed the whole framework. All authors contributed to the writing of the manuscript.

Acknowledgments

The authors thank CIFAR, Samsung, and IVADO for funding and NVIDIA for equipment.

References

Ba, J. and Frey, B. Adaptive dropout for training deep neural networks. *Neural Information Processing Systems (NIPS)*, 2013.

Bengio, E., Jain, M., Korablyov, M., Precup, D., and Bengio, Y. Flow network based generative models for non-iterative diverse candidate generation. *Neural Information Processing Systems (NeurIPS)*, 2021a.

Bengio, Y., Deleu, T., Hu, E. J., Lahlou, S., Tiwari, M., and Bengio, E. Gflownet foundations. *arXiv preprint arXiv:2111.09266*, 2021b.

Bhatt, U., Antorán, J., Zhang, Y., Liao, Q. V., Sattigeri, P., Fogliato, R., Melançon, G., Krishnan, R., Stanley, J., Tickoo, O., et al. Uncertainty as a form of transparency: Measuring, communicating, and using uncertainty. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 401–413, 2021.

Boluki, S., Ardywibowo, R., Dadaneh, S. Z., Zhou, M., and Qian, X. Learnable Bernoulli dropout for Bayesian deep learning. *Artificial Intelligence and Statistics (AISTATS)*, 2020.

Damianou, A. and Lawrence, N. D. Deep Gaussian processes. *Artificial Intelligence and Statistics (AISTATS)*, 2013.

Daxberger, E., Nalisnick, E., Allingham, J. U., Antorán, J., and Hernández-Lobato, J. M. Bayesian deep learning via subnetwork inference. *International Conference on Machine Learning (ICML)*, 2021.

Deleu, T., Góis, A., Emezue, C., Rankawat, M., Lacoste-Julien, S., Bauer, S., and Bengio, Y. Bayesian structure learning with generative flow networks. *Uncertainty in Artificial Intelligence (UAI)*, 2022.

Fan, X., Zhang, S., Tanwisuth, K., Qian, X., and Zhou, M. Contextual dropout: An efficient sample-dependent dropout module. *International Conference on Learning Representations (ICLR)*, 2021.

Foong, A., Burt, D., Li, Y., and Turner, R. On the expressiveness of approximate inference in Bayesian neural networks. *Neural Information Processing Systems (NeurIPS)*, 2020.

Fort, S., Hu, H., and Lakshminarayanan, B. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *International Conference on Machine Learning (ICML)*, 2016.

Gal, Y., Hron, J., and Kendall, A. Concrete dropout. *Neural Information Processing Systems (NIPS)*, 30, 2017.

Ghiasi, G., Lin, T.-Y., and Le, Q. V. Dropblock: A regularization method for convolutional networks. *Neural Information Processing Systems (NeurIPS)*, 2018.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. *International Conference on Machine Learning (ICML)*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *Computer Vision and Pattern Recognition (CVPR)*, 2016.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *International Conference on Learning Representations (ICLR)*, 2019.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Jain, M., Bengio, E., Hernandez-Garcia, A., Rector-Brooks, J., Dossou, B. F., Ekbote, C. A., Fu, J., Zhang, T., Kilgour, M., Zhang, D., Simine, L., Das, P., and Bengio, Y. Biological sequence design with gflownets. *International Conference on Machine Learning (ICML)*, 2022.

Jain, M., Lahlou, S., Nekoei, H., Butoi, V., Bertin, P., Rector-Brooks, J., Korablyov, M., and Bengio, Y. DEUP: Direct epistemic uncertainty prediction. *Transactions on Machine Learning Research (TMLR)*, 2023.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. *Neural Information Processing Systems (NIPS)*, 2015.

Kuleshov, V., Fenner, N., and Ermon, S. Accurate uncertainties for deep learning using calibrated regression. *International Conference on Machine Learning (ICML)*, 2018.

Le Folgoc, L., Baltatzis, V., Desai, S., Devaraj, A., Ellis, S., Manzanera, O. E. M., Nair, A., Qiu, H., Schnabel, J., and Glocker, B. Is MC dropout bayesian? *arXiv preprint arXiv:2110.04286*, 2021.

Lee, H. B., Nam, T., Yang, E., and Hwang, S. J. Meta dropout: Learning to perturb latent features for generalization. *International Conference on Learning Representations (ICLR)*, 2020.

Lotfi, S., Izmailov, P., Benton, G., Goldblum, M., and Wilson, A. G. Bayesian model selection, the marginal likelihood, and generalization. *International Conference on Machine Learning (ICML)*, 2022.

MacKay, D. J. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

Madan, K., Rector-Brooks, J., Korablyov, M., Bengio, E., Jain, M., Nica, A., Bosc, T., Bengio, Y., and Malkin, N. Learning GFlowNets from partial episodes for improved convergence and stability. *International Conference on Machine Learning (ICML)*, 2023.

Malkin, N., Jain, M., Bengio, E., Sun, C., and Bengio, Y. Trajectory balance: Improved credit assignment in GFlowNets. *Neural Information Processing Systems (NeurIPS)*, 2022.

Malkin, N., Lahlou, S., Deleu, T., Ji, X., Hu, E., Everett, K., Zhang, D., and Bengio, Y. GFlowNets and variational inference. *International Conference on Learning Representations (ICLR)*, 2023.

Nado, Z., Band, N., Collier, M., Djolonga, J., Dusenberry, M., Farquhar, S., Filos, A., Havasi, M., Jenatton, R., Jerfel, G., Liu, J., Mariet, Z., Nixon, J., Padhy, S., Ren, J., Rudner, T., Wen, Y., Wenzel, F., Murphy, K., Sculley, D., Lakshminarayanan, B., Snoek, J., Gal, Y., and Tran, D. Uncertainty Baselines: Benchmarks for uncertainty & robustness in deep learning. *arXiv preprint arXiv:2106.04015*, 2021.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *Computer Vision and Pattern Recognition (CVPR)*, 2015.

Nguyen, S., Nguyen, D., Nguyen, K., Than, K., Bui, H., and Ho, N. Structured dropout variational inference for Bayesian neural networks. *Neural Information Processing Systems (NeurIPS)*, 2021.

Nica, A. C., Jain, M., Bengio, E., Liu, C.-H., Korablyov, M., Bronstein, M. M., and Bengio, Y. Evaluating generalization in gflownets for molecule design. *ICLR 2022 Machine Learning for Drug Discovery workshop*, 2022.

Osband, I., Russo, D., and Van Roy, B. (more) efficient reinforcement learning via posterior sampling. *Neural Information Processing Systems (NIPS)*, 2013.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Neural Information Processing Systems (NeurIPS)*, 2019.

Pan, L., Zhang, D., Courville, A. C., Huang, L., and Bengio, Y. Generative augmented flow networks. *International Conference on Learning Representations (ICLR)*, 2023.

Park, S. and Kwak, N. Analysis on the dropout effect in convolutional neural networks. *Asian Conference on Computer Vision*, 2016.

Pham, H. and Le, Q. Autodropout: Learning dropout patterns to regularize deep networks. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2021.

Pollard, T. J., Johnson, A. E., Raffa, J. D., Celi, L. A., Mark, R. G., and Badawi, O. The eICU Collaborative Research Database, a freely available multi-center database for critical care research. *Scientific data*, 5(1):1–13, 2018.

Sensoy, M., Kaplan, L., and Kandemir, M. Evidential deep learning to quantify classification uncertainty. *Neural Information Processing Systems (NeurIPS)*, 2018.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Wilson, A. G. and Izmailov, P. Bayesian deep learning and a probabilistic perspective of generalization. *Neural Information Processing Systems (NeurIPS)*, 2020.

Xie, J., Ma, Z., Zhang, G., Xue, J.-H., Tan, Z.-H., and Guo, J. Soft dropout and its variational bayes approximation. *Machine Learning for Signal Processing (MLSP)*, 2019.

Yang, X., Tang, J., Torun, H. M., Becker, W. D., Hejase, J. A., and Swaminathan, M. Rx equalization for a high-speed channel based on bayesian active learning using dropout. *Electrical Performance of Electronic Packaging and Systems (EPEPS)*, 2020.

Yu, Z., Yu, J., Cui, Y., Tao, D., and Tian, Q. Deep modular co-attention networks for visual question answering. *Computer Vision and Pattern Recognition (CVPR)*, 2019.

Zhang, D., Chen, R. T., Malkin, N., and Bengio, Y. Unifying generative models with gflownets. *arXiv preprint arXiv:2209.02606*, 2022a.

Zhang, D., Malkin, N., Liu, Z., Volokhova, A., Courville, A., and Bengio, Y. Generative flow networks for discrete probabilistic modeling. *International Conference on Machine Learning (ICML)*, 2022b.

A. Appendix

A.0.1. Hyperparameters

Hyperparameters of the “backbone” ResNet and Transformer models were obtained from published baselines or architectures (He et al., 2016; Yu et al., 2019; Fan et al., 2021; Gal & Ghahramani, 2016; Gal et al., 2017). Several GFlowNet-specific hyperparameters are taken into consideration in this study, including the architecture of the variational function $q(\cdot)$ and its associated hyperparameters and the temperature of $q^{\sim}(\cdot)$. For ID-GFlowOut, there is an additional hyperparameter, which is the prior $p(z)$. The parameters are picked via grid search using the validation set. The temperature of $q^{\sim}(\cdot)$ is set as 2. In addition, with a 0.1 probability, the forward policy will choose a random mask set in each layer.

A.0.2. Computational efficiency

On a single RTX8000 GPU, training models with GFlowOut takes around the same time as Contextual dropout and Concrete Dropout, and around twice the time (ResNet 7 hrs and MCAN Transformer 16 hrs) as a model with the same architecture and random dropout. The three learned dropout methods have similar efficiency during inference.

A.1. Experimental details

A.1.1. Sampling dropout masks

In the forward pass during inference, 20 samples are used for each data point. In ResNet experiments, dropout masks are generated for each ResNet block. In the transformer VQA experiment, in each layer, dropout is applied to both the self-attention and the feed-forward layer.

A.1.2. Robustness to distribution shift

The performance of each method was obtained with 9 repeats of different random seeds for training. Early stop using validation set was used to prevent overfitting. VQA Transformer experiments are designed according to Yu et al. (2019).

A.1.3. OOD detection

For each data point, we take 20 forward passes and calculate the Uncertainty for prediction on each example using the Dempster-Shafer metric (Sensoy et al., 2018) and algorithm from Jain et al. (2023). The uncertainty score is used for classification of in-distribution vs. out-of-distribution data points assuming the later should have higher uncertainty.

A.1.4. Adaptation after training on noisy data

When training the model with noisy CIFAR-10/100 data, randomly picked 30% data points are assigned a random label in the whole training set. The model obtained is then fine-tuned using a small number of clean data points all with correct labels. We conducted experiments with 1000,2000,4000 and 8000 data points used for fine-tuning.

A.1.5. Real-world clinical data

The ICU dataset is a real-world dataset, containing information about the deaths or survival of 126489 patients, across 58 different hospitals, given a set of administrated drugs. The goal of this experiment is to evaluate how well our approach generalizes, in real-world settings. To imitate this, we built two sets:

- a training set that contains data points about patients from all hospitals, except the hospital with the highest number of patients (hospital ID 167). This results in a dataset with 120945 entries, which is equally partitioned (70:30 ratio) into the real training and validation sets.
- a test set that contains information about 5544 patients. As each hospital follows a specific distribution, the test set was designed to measure the OOD efficiency of GFlowOut, on the widest possible set of patients, which is a real-world scenario.

We used a 3-layer MLP with multiple Dropout options as presented in Table 4. For the evaluation, we perform 20 forward passes and take the mean of the prediction.

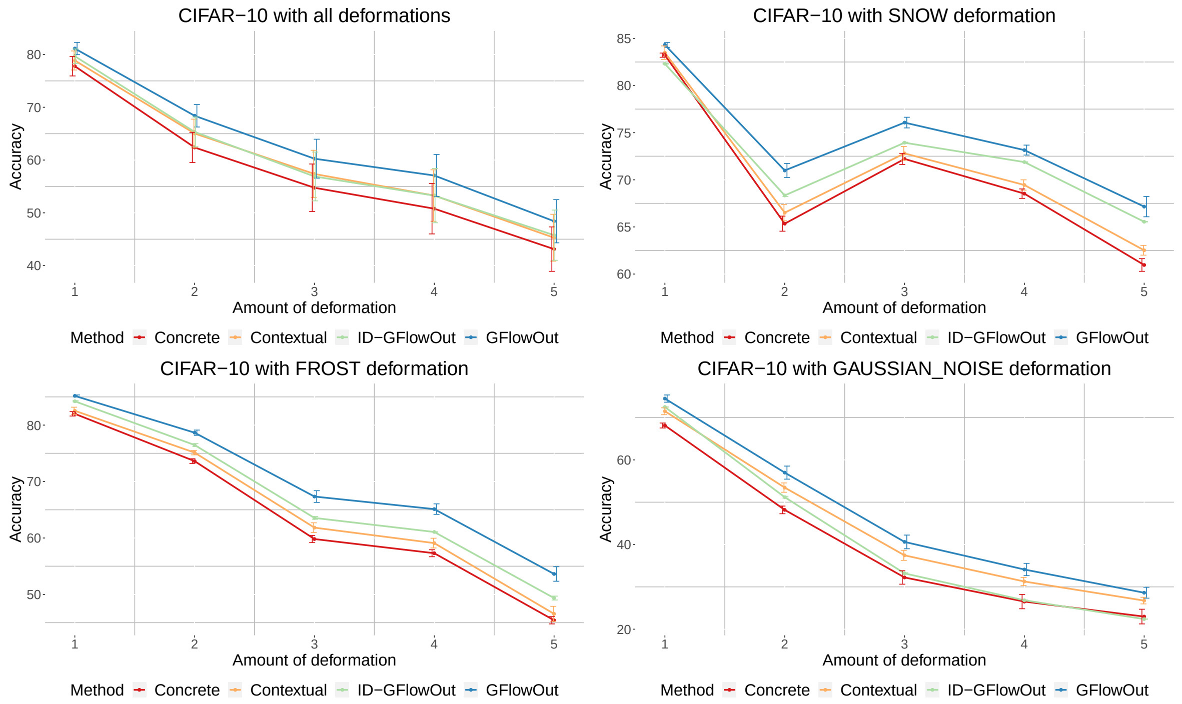


Figure 3. Robustness of ResNet-18 models trained with different dropout methods to different amounts of SNOW deformation, FROST deformation or Gaussian noise in CIFAR-10 during test time

A.2. Analysing dropout masks

Here, we analyze the behavior and dynamics of the binary masks generated by GFlowOut for data points corresponding to different labels and different augmentations. First, we want to verify that GFlowOut generates masks with probability proportional to the reward R as defined in equation (7). Our analysis shows a statistically significant correlation between the probabilities of a set of masks being generated by GFlowNet and the corresponding rewards, with correlation ≥ 0.4 and p values ≤ 0.05. Next, we want to explore whether GFlowOut generates diverse dropout masks. We take the mean dropout masks generated during inference for each data point and calculate Manhattan distances among different samples in the data set. The results are shown in Figure 6.

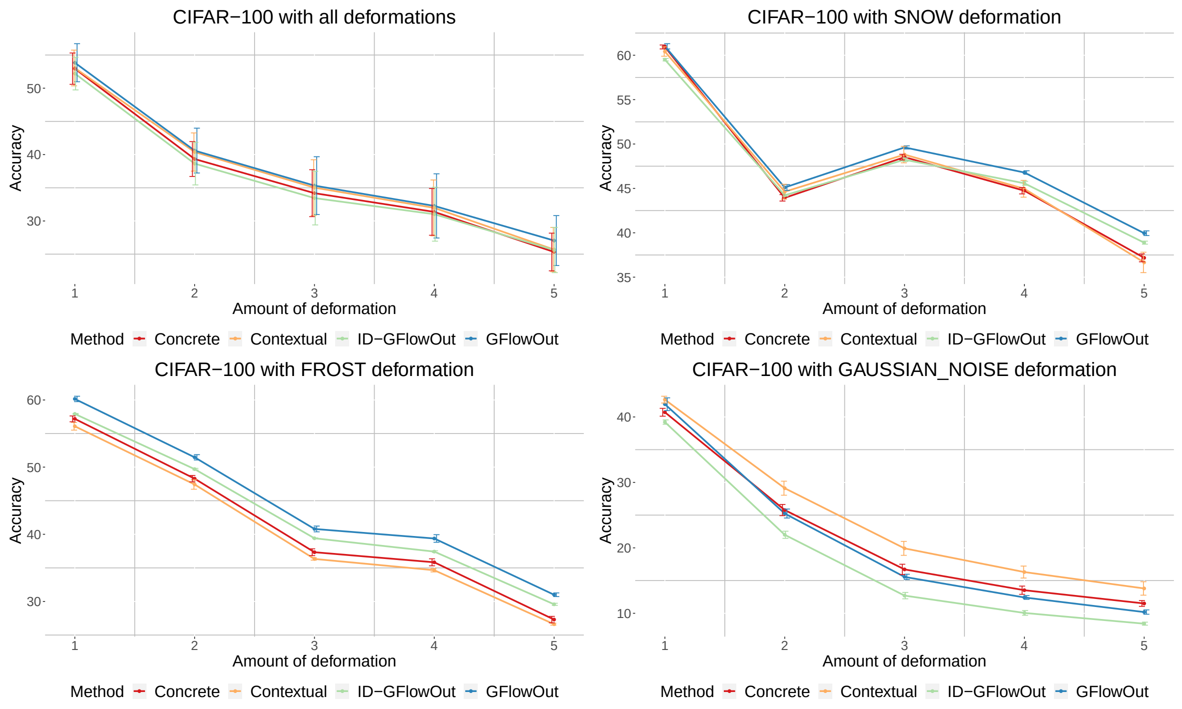


Figure 4. Robustness of ResNet-18 models trained with different dropout methods to different amounts of SNOW deformation, FROST deformation or Gaussian noise in CIFAR-100 during test time

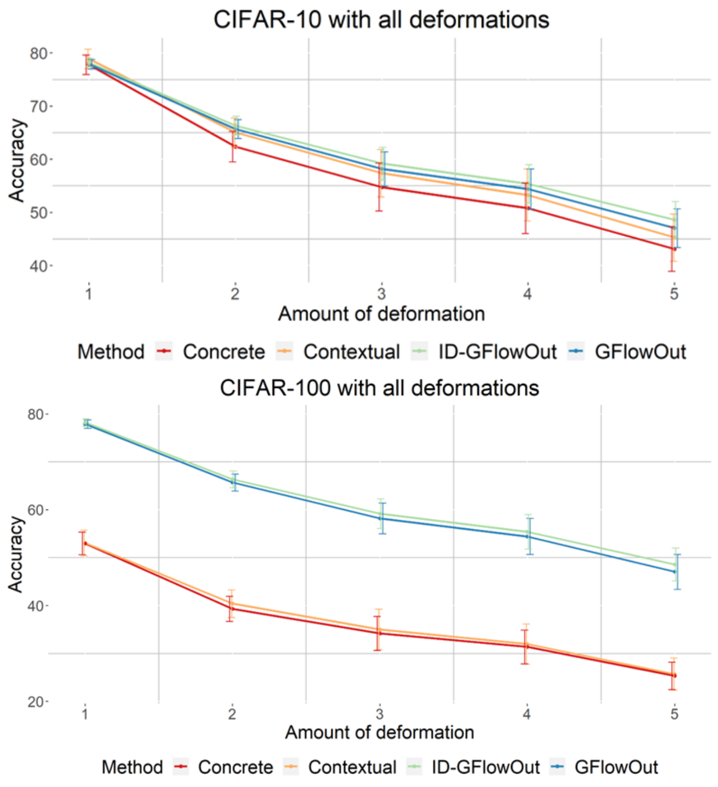


Figure 5. Robustness of ResNet-18 models trained with different dropout methods to different amounts of deformation using Resnet blocks with components in a slightly different order: BatchNorm -Conv-BatchNorm-Conv

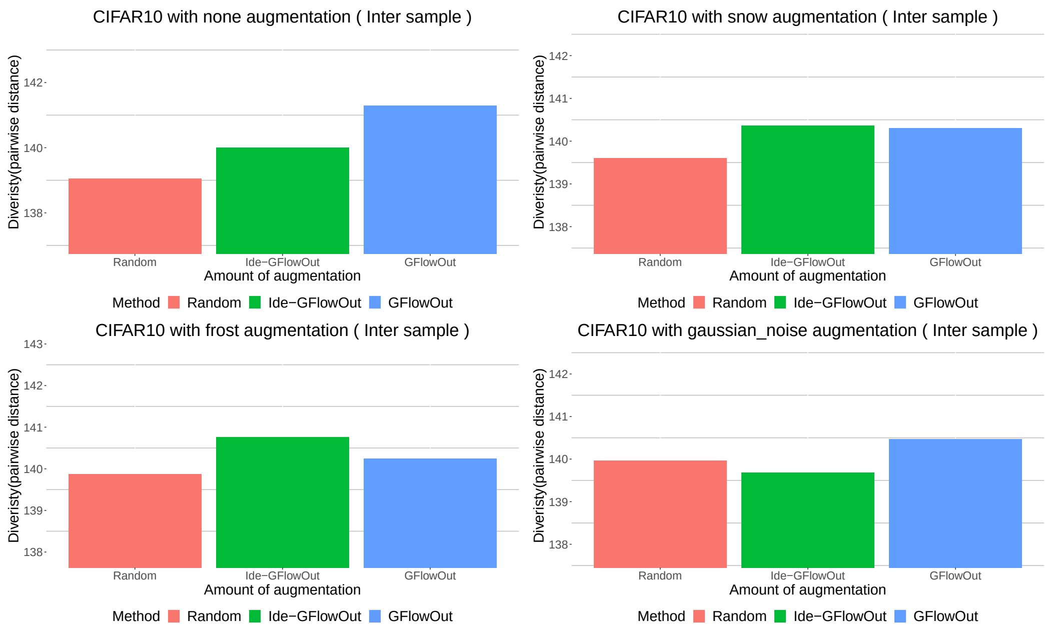


Figure 6. Diversity of binary dropout masks among different data points measured by Manhattan distance