Notes on "Discrete Flows: Invertible Generative Models of Discrete Data"

Xavier Garcia

June 12, 2019

The purpose of these notes is to discuss some of the concepts in "Discrete Flows: Invertible Generative Models of Discrete Data".

1 Normalizing flows on continuous data

We begin by first explaining the case for continuous variables, and point out difficulties in this setting as well as extending it to the discrete case. We begin with some data $x_1, ..., x_N$ where we view the dataset as being i.i.d. samples of some random variable X with values in $\mathbb{R}^n$.

In normalizing flows, we make the assumption that there exists an invertible smooth function $f : \mathbb{R}^n \to \mathbb{R}^n$ and an independent n-dimensional Gaussian variable Z such that $X = f(Z)$. With such an assumption, we could then write out the density of X by the change-of-variable formula:

$$p_X(x) = p_Z(f^{-1}(x))|\det Jf_{f^{-1}(x)}|^{-1}$$

where Jf is the Jacobian matrix of f. Since we don't actually know f, we assume that it belongs to (or is well-approximated by) some parametrized family of functions $f(\cdot|\theta)$, where θ denotes the parameters. We approximate the optimal parameters by the maximum likelihood estimates (MLE) i.e. minimizing the objective

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} -\log p_Z(f^{-1}(x_i)) + \log\det|Jf_{f^{-1}(x_i)}||.$$

From a computational perspective, there are two main difficulties:

1. Computing the determinant is $O(n^3)$.
2. Retaining invertibility of f with an analytic tractible Jacobian that is still flexible enough to model a large class of functions.

Both of these restrict the family of approximators and require special type of layers in order to reduce the complexity of the determinant.

2 The discrete case

The setting of this paper consist of the case where each of the n entries of X are constrainted to be in the set $\{1, ..., K\}$ for some integer K, denoting the number of classes. A natural example would be to think of X_i as ith word in a sentence, where its value represents the index of the word in some dictionary and K correspond to the size of the dictionary.

First, consider the trivial case when $n = 1$. Notice that the distribution of X is completely determined by the K parameters $p_k := \mathbb{P}(X = k)$ for $k = 1, ..., K$. We can compute the likelihood explicitly:

$$L(p_1, ..., p_K) = \sum_{k=1}^{K} \#\{i : x_i = k\} p_k$$

This quantity is maximized when

$$p_k = \frac{1}{N} \#\{i : x_i = k\}$$

for each $k \in \{1, ..., K\}$. In this case, there is no need to consider flows or some prior distribution. If $n > 1$ and the entries X_i are independent, then we can perform a similar computation to recover the density. Therefore, the case of interest is when the discrete variables are dependent on each other. Given the simplicity of the independent case, we approximate the dependent case by a function of the independent case under some suitable transformations.

One of the advantages of the discrete case is that the change-of-variable formula is much simpler. Suppose Z consists of n independent Categorical variables of K classes, and we assume that there exists a surjective function $f : \{1, ..., K\}^n \to \{1, ..., K\}^n$ such that $f(Z) = X$. In this case, the following formula holds:

$$p_X(x) = \sum_{z : f(z) = x} p_Z(z)$$

The absence of the Jacobian term makes this appealing. Moreover, if f is invertible, the expression reduces to

$$p_X(x) = p_Z(f^{-1}(x)).$$

For the remainder of these notes, we'll assume that x is a vector in $\mathbb{R}^n$ with values in $\{0, ..., K - 1\}$ and we'll denote its ith entry by x_i, discarding the previous notation where x_i denoted the ith sample.

The transformations considered in the paper all consists of one or more compositions of functions $f : \{1, ..., K\}^n \to \{1, ..., K\}^n$ of the following form:

$$f_i(z) = \mu_i + \sigma_i z_i \mod K$$

for some functions μ_i and σ_i valued in $\{0, ..., K-1\}$. We can write this compactly by

$$f(z) = \mu(z) + \sigma(z) \otimes z \mod K$$

where $\otimes$ denotes elementwise product and the understanding that μ and σ are vector-valued functions whose coordinates consist of the functions from before. The only thing that limits invertibility is the σ. One can fix this by enforcing restrictions on σ, such as $\sigma = 1$.

There are two types of families of flows that are considered:

2.1 The Autoregressive Case

In the autoregressive case, the functions μ_i and σ_i depend on first $i-1$ timesteps. More explicitly, we have:

$$\begin{aligned}
&z_1 \to \mu_1 + \sigma_1 z_1 \bmod K := \hat{x}_1 \\
&z_2 \to \mu_2(\hat{x}_1) + \sigma_2(\hat{x}_1) z_2 \bmod K := \hat{x}_2 \\
&z_3 \to \mu_3(\hat{x}_1, \hat{x}_2) + \sigma_3(\hat{x}_1, \hat{x}_2) z_3 \bmod K := \hat{x}_3 \\
&\quad ... \\
&z_n \to \mu_n(\hat{x}_1, ..., \hat{x}_{n-1}) + \sigma_n(\hat{x}_1, ..., \hat{x}_{n-1}) z_n \bmod K := \hat{x}_n
\end{aligned}$$

We can use either LSTMs or Transformers for the μ and σ.

2.2 The Bipartite Case

Borrowing from the normalizing flows literature, one can mimic the bipartite flow. In this setting, we select some subset of indices $\mathcal{J}$ and write $z = (z_{\mathcal{J}}, \tilde{z}_{\mathcal{J}})$ where $z_{\mathcal{J}}$ consists of the entries of z whose indices are in $\mathcal{J}$ and $\tilde{z}_{\mathcal{J}}$ make up the remainder. We can then define the flow explicitly as follows:

$$z_i \mapsto \begin{cases} \mu_i(z_{\mathcal{J}}) + \sigma_i(z_{\mathcal{J}}) z_j & i \in \mathcal{J} \\ z_i & \text{else} \end{cases}$$

In the bipartite case, we set $(\mu, \sigma) = (0, 1)$ for some subset of $z_1, ..., z_n$ and the rest are functions of these variables. By construction, this is fully parallelizable. To achieve higher flexibility, the paper stacks several of these flows together with varying index sets $\mathcal{J}$.

3 Training

Before we discuss the details of training, we first discuss what are the trainable parameters. There are two such sets:

1. The n parameters for the base distribution Z. (base parameters)
2. The parameters used to make μ and σ. (flow parameters)

Computing the gradient with respect to the base distribution is simple and not a problem. The trouble arises from the flow parameters, since μ and σ are

discrete-valued, hence not differentiable. To solve this problem, the paper uses a straight-through estimator approach. Let us expand on that for a moment.

In the forward pass, we will proced as if nothing is wrong. In the backwards pass, we use a soft approximation of $\mu(z)$. We can write $\mu_i(z)$ in terms of its one-hot encoded representation i.e.

$$\mu_i(z) = \text{argmax}(\theta_i(z))$$

where $\theta_i(z) \in \{0, ..., K-1\}^n$. Moreover, we can think μ as being equal to its one-hot encoded representation i.e.

$$\mu_i(z) = \text{one-hot}(\text{argmax}(\theta_i(z))).$$

We can relax this identification by replacing the composition of the one-hot and argmax operations with a softmax operation i.e.

$$\mu_i(z) = \text{softmax}(\theta_i(z)/\tau)$$

where τ is the temperature parameter. This approximation is clearly biased. One can think of the τ parameter as controlling the bias-variance tradeoff: for small τ, the softmax is very close to the one-hot composed with the argmax, but the logits will be large and hence the variance is large. Conversely, if τ is very large, the distribution is close to the uniform distribution, completely disregarding the input, hence low variance, but very high bias.