Using the Gumbel softmax trick to enable mixture density networks

Sean Bittner

June 10, 2019

1 Motivation

For some applications, it's necessary to keep track of density of samples from a deep generative model. Some density preserving flexible approximations are capable of learning multi-modal distributions, but there are perhaps better achitectures than what have been used for learning in settings with suspected multi-modal structure. We introduce a simple solution called "mixture density networks."

2 Background

2.1 Normalizing flows

Normalizing flows are a class of bijective transformations with fast log-determinant jacobian computations, enabling optimizations where computation of samples from deep generative models are necessary [1]. The idea is that for a sequence of bijective transformations $g_l : \mathcal{R}^D \rightarrow \mathcal{R}^D$ to a base random variable $\omega \in \mathcal{R}^D \sim p_0$, we can compute the density of the sample $z_L = g_L \circ g_{L-1} \circ ... \circ g_1(\omega)$ using the change of variables formula.

The change of variables formula for a bijective transformation $z^{'} = f(z)$ is

$$p(z') = p(z) \left| \det \frac{\partial f^{-1}(z^{'})}{\partial z^{'}} \right| = p(z) \left| \det \frac{\partial f(z)}{\partial z} \right|^{-1}$$

We can then compute the probability of z_L using this formula for the application of each bijective function in each layer (let $z_0 = \omega$):

$$p(z_L) = p(z_0) \prod_{l=1}^{L} \left| \det \frac{\partial g_l(z_{l-1})}{\partial z_{l-1}} \right|^{-1}$$

If we wish to optimize the log density, which is most common, we have the following expression, hence the earlier point about the bijective transformations being designed to have fast log-determinant-jacobian computations.

$$\log p(z_L) = \log p(z_0) - \sum_{l=1}^{L} \log \left| \det \frac{\partial g_l(z_{l-1})}{\partial z_{l-1}} \right|$$

2.2 Discrete normalizing flows

There have been some attempts to incorporate discrete structure into density networks. One approach called "RAD" (real and discrete), partitions observations space into K groups, and assigns a unique density network to each group [2]. This seems fine, and we'll eventually compare. From first principles, it seems a bit unnecessarily complicated.

2.3 Mixture of Gaussians

Our approach employs a mixture of Gaussians at the base layer of the density network. Instead of sampling $z_0 = \omega \sim \mathcal{N}(0,1)$ which is most common, we seek to sample $z_0 \sim \text{MoG}(\alpha, \{\mu_k\}, \{\sigma_k\})$, with $\alpha \in [0,1]^K$ s.t. $\sum_k \alpha_k = 1$, $\mu_k \in \mathcal{R}^D$, and $\sigma_k \in \mathcal{R}_+^D$ (we'll use the log-sigma parameterization). The base probability of our sample z_0 would be:

$$p(z_0) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(z_0; \mu_k, \sigma_k) \tag{1}$$

3 Mixture density networks with the Gumbel softmax

For most practical purposes, we will want to reparameterize this mixture of Gaussians distribution. For example, for the purpose of doing a maximum entropy optimization [3], we wouldn't need to reparameterize the MoG to take gradients of the deep generative model with respect to its entropy, however the reparameterization becomes necessary for taking gradients with respect to the constraint violations.

3.1 Repaprameterizing a categorical using the Gumbel softmax

To reparameterize the categorical distribution of the mixture of gaussians, we will use the Gumbel softmax trick [4]. The base unparameterized random distribution is the Gumbel distribution (with parameters 0 and 1). We can generate a $g_i \sim G(0,1)$ by simulating $u_i \sim U[0,1]$ and setting $g_i = -\log(-\log(u_i))$. We can then sample an approximately one-hot vector via:

$$c_i = \frac{\exp((\log(\alpha_i) + g_i)/\tau)}{\sum_k \exp((\log(\alpha_k) + g_k)/\tau)}$$

for some $\tau > 0$.

The Gumbel softmax probability of sample c_i is then:

$$p(c = c_1, ..., c_K \mid \alpha, \tau) = \Gamma(K)\tau^{K-1} \left(\sum_{k=1}^{K} \frac{\alpha_k}{c_k^{\tau}} \right)^{-K} \prod_{k=1}^{K} \frac{\alpha_k}{c_k^{\tau+1}}$$

and the log density of c is

$$\log(p(c \mid \alpha, \tau) = \log(\Gamma(K)) + (K-1)\log(\tau) + -K \log \left(\sum_{k=1}^{L} \frac{\alpha_k}{c_k^{\tau}} \right) + \sum_{k=1}^{L} \log(\alpha_k) - (\tau+1)\log(c_k) \tag{2}$$

3.2 Repaprameterizing the mixture of Gaussians

Now that we've reparameterized the categorical distribution, we can extend this strategy to a reparameterization of the mixture of gaussians by taking a dot product of the approximately on hot vectors with the elements of the MoG μ_k's and σ_k's. Specifically, we organize the parameters into $K \times D$ matrices such that each row of $\mu \in \mathcal{R}^{K \times D}$ and $\sigma \in \mathcal{R}_+^{K \times D}$ corresponds to one of the Gaussian clusters.

For a given forward pass, we sample $c_i \sim p(c \mid \alpha, \tau)$ and $w_i \sim \mathcal{N}(0,1)$. Note here that c_i denotes the i^{th} sample in a minibatch of say M samples. We can soft-select the means and variances by taking the dot products of $c_i \in [0,1]^K$ and μ and σ.

$$\mu_i = c_i^\top \mu$$

$$\sigma_i = c_i^\top \sigma$$

We can then generate $z_{0,i}$ by calculating

$$z_{0,i} = \mu_i + \omega \cdot \sigma_i$$

The probability of a single sample from the entire mixture density network is then

$$p(z_{L,i}) = p(z_{0,i}) \prod_{l=1}^{L} \left| \det \frac{\partial g_l(z_{l-1,i})}{\partial z_{l-1,i}} \right|$$

and accordingly the log density is

$$\log p(z_{L,i}) = \log p(z_0, i) - \sum_{l=1}^{L} \log \left| \det \frac{\partial g_l(z_{l-1,i})}{\partial z_{l-1,i}} \right|$$

with

$$\log p(z_{0,i}) = \log \left(\int p(z_{0,i} \mid c_i) p(c_i) dc_i \right)$$

We can approximate $p(z_{0,i})$ by

$$p(z_{0,i}) = \int p(z_{0,i} \mid c_i) p(c_i) dc_i \approx E_{p(c_i)} \left[p(z_{0,i} \mid c_i) \right]$$

This means we will need to sample a nested minibatch to compute the base density for each sample. We can easily compute a by

$$p(z_{0,i} \mid c_i) = \frac{\mathcal{N}(\omega_i; 0, 1)}{\left| \prod_{d=1}^{D} \sigma_{i,d} \right|}$$

The full log probability of minibatch sample i is then:

$$\log(p(z_{L,i})) \approx \log \left(E_{p(c_i)} \left[p(z_{0,i} \mid c_i) \right] \right) - \sum_{l=1}^{L} \log \left| \det \frac{\partial g_l(z_{l-1,i})}{\partial z_{l-1,i}} \right|$$

Up to this point, I have been calculating joint entroy (whoops) using equation 2, although it may have been interesting to use equation 1.

Using the above strategy, we don't need to use either equations 1 or 2.

References

[1] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[2] Laurent Dinh, Jascha Sohl-Dickstein, Razvan Pascanu, and Hugo Larochelle. A rad approach to deep mixture models. *arXiv preprint arXiv:1903.07714*, 2019.

[3] Gabriel Loaiza-Ganem, Yuanjun Gao, and John P Cunningham. Maximum entropy flow networks. *arXiv preprint arXiv:1701.03504*, 2017.

[4] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.