1 Interpretability

1. Visualizing learned features using t-SNE. t-distributed Stochastic Neighbor Embedding (t-SNE) is a method for visualizing high dimensional data in low-dimensional space, such that points close toghether in the high-dimensional space remain close in the low-dimensional visualization. Propose a method for visualizing features learned at various depth in a deep learning model, and then compare your approach to the one described at https://cs.stanford.edu/people/karpathy/cnnembed/.

2. Testing with CAVs. There is a risk when working with CAVs that you learn a totally meaningless concept – the procedure returns a CAV even if you defined a concept using totally random features. Define a statistic based on the CAV scores for class k and layer l by

$$\frac{\#\{S_{C,k,l}(x_i) > 0\}}{\#\{\text{examples in class } k\}}$$

which measures the fraction of samples in class k which are positively activated by the given concept. Propose a statistical test for finding out whether this fraction is meaningfully large; i.e., that it is larger than you would have if you had used totally random images to define a (totally meaningless) concept.

2 GANs

1. Using the figure on the first formulation slide, come up with a visual interpretation of the change of variables formula, which says that if $x \xrightarrow{f} y$ and if $x \sim p(x)$, then $y \sim p\left(f^{-1}(y)\right)\left|\frac{df}{dx}\right|^{-1}$.

2. GANs and VAEs are both generative models in the sense that you can sample new data from them. One however allows you to sample latent encodings z for any x of interest, and the other does not, which is which?

3. Verify the density ratio estimation claim from the lecture that $\frac{p^*(x|y=1)}{q_\theta(x)} = \frac{p(y=1|x)}{p(y=0)}\frac{1-\pi}{\pi}$. Hint: Use Bayes' rule.

4. Instead of a completely unsupervised GAN, you can learn to generate samples conditional on a class label y. Explain why an objective like,

$$\min_G \max_D V(D,G) := \mathbb{E}_{p_{\text{data}}}[\log D(x|y)] + \mathbb{E}_{p(z)}[\log(1 - D(G(z|y)))]$$

might be able to work (see Mirza and Osindro for an actual implementation).

3 Metalearning

1. Identify some contexts where metalearning could be applied in practice. Are there limitations in the metalearning setup that make it less useful in scenarios you think of?

2. In transfer learning, you may choose to fine tune the lower layer weights on your new task, rather than simply copying the original features verbatim. If this is your goal, how should you choose your learning rates for the low-level features, versus the new high-level weights?

3. For k-nearest neighbors, larger k reduces variance but increases bias – it controls model complexity. In the nearest neighbors metalearner, we aren't using nearest neighbors direction, but some smoothed-out version of it. How might you control model complexity for this alternative version of nearest neighbors?

4. How would you adapt the ordinary classification-based nearest neighbors metalearner to work with continuous y_i instead?

4 Bayesian Deep Learning

1. Assignments in mixture of Gaussians. Suppose x_i is drawn from a mixture of two gaussians, which have parameters $\left(\mu_1, \sigma_1^2\right) = (0, 1)$ and $\left(\mu_2, \sigma_2^2\right) = (2, 1)$. Show that $p\left(z = 1 | x = 1\right) = \frac{1}{2}$ and $p\left(z = 1 | x = 0\right) = \frac{1}{1+\exp(-2)} \approx 0.881$. In general the posterior is Bernoulli with probability $\varphi\left(x\right)$ of assigning to class 1. Can you find a formula for $\varphi\left(x\right)$ that applies to general (or multivariate?) μ_k, Σ_k?

2. More general reparameterization. We saw that the Gaussian distribution $\mathcal{N}\left(x | \mu, \sigma^2 I\right)$ can be reparameterized as $g_{\mu,\sigma}\left(x\right) = \mu + \sigma \bigodot \epsilon$, where $\epsilon \sim \mathcal{N}\left(\epsilon | 0, I\right)$ doesn't depend on any parameters. This trick actually applies for a variety of other distributions, which this exercise explores.

    - Random Variable generation using inverse CDFs[1]. Suppose that Z, which we assume is one-dimensional, has CDF $F\left(z\right)$. Let $U \sim \text{Unif}\left(0, 1\right)$. Verify that the transformation of u defined by $F^{-1}\left(U\right)$ has CDF $F\left(z\right)$, and so has the same distribution as Z.
    - Argue that whenever the CDF of the density $q_\varphi\left(z | x\right)$ is known, this allows for a version of the reparameterization trick.
    - Suppose that $Z \sim \lambda$, meaning that it has CDF function $F\left(z\right) = 1 - \exp\left(-\lambda z\right)$. How can you simulate this?
    - Can you think of downsides of this approach?

[1] Cumulative Distribution Functions

3. Amortization vs. Approximation gaps. Recall that in the derivation of the ELBO, we had an expression like

$$\log p_\theta(x) = \mathbb{E}_q[\log p_\theta(x|z)] - D_{KL}(q(z|x)||p(z)) + D_{KL}(q(z|x)||p(z|x))$$

and we dropped the last term from the optimization, because it is intractable. We now study the role of that term when proposing variational families.

- In the usual VAE, we set $q_\varphi(z|x) = \mathcal{N}\left(z|\mu_\varphi(x), \sigma^2_\varphi(x) I\right)$; i.e., a diagonal gaussian.. Suppose you had approximated it instead by a Gaussian with general $\Sigma(x)$. What effect would this have on $D_{KL}(q_{\varphi^*}(z|x)||p(z|x))$, when considering the best possible q_{φ^*} from either of these two (diagonal or dense covariance) variational families?
- Let $\hat{\varphi}$ be the parameters of the fitted inference network after optimizing the ELBO. Express the final inference quality $D_{KL}(q_{\hat{\varphi}}(z|x)||p(z|x))$ as,

$$D_{KL}(q_{\varphi^*}(z|x)||p(z|x)) + (D_{KL}(q_{\hat{\varphi}}(z||x)) - D_{KL}(q_{\varphi^*}(z|x))).$$

  The first term (outside parenthesis) is called the "approximation gap," and refers to the difference between the true posterior and the best possible element of the variational approximation, while the second term (in parenthesis) is called the "amortization gap," and refers to the difference between the best possible approximation within the family and what is actually found by the network. In light of this discussion, why might we choose not to proceed with the full $\Sigma(x)$ parameterization in the previous part?
- A normalizing flow is a sequence of transformations to a simple variable that results in a variable with a more complicated density, but one which can still be written in closed form, using the change of variables formula. For example, you might iteratively apply $f(z) = z + u\sigma\left(w^T z + b\right)$ to what is intiially a simple (say gaussian z), since this transformation is easy to differentiate (which is the only thing you need to apply the change of variables formula). Does this proposed procedure reduce the approximation or amortization gap?
- What are some general strategies for reducing the amortization gap?