CSCI B659: Machine Learning through Approximate Inference in Graphical Models
Spring 2023
Roni Khardon
Assignment 3

In this assignment you will implement and test a simple version of normalizing flows in 2D, following the outline described in class. The assignment is due by Saturday 3/25, 11:59pm on canvas as a ZIP file. Submission instructions below.

Forward and Backward Flows and Density Estimation

1 level forward flow: maps z to x, using parameters w, b (where z, x, w, b are all points in 2D space) and mask m which is 0 or 1. The forward flow $x = f_{w,b,m}(z)$ sets $x[m] = z[m]$ and $x[1-m] = z[1-m] * \theta_0 + \theta_1$ where $\theta = \theta(z[m]) = e^{tanh(z[m]*w+b)}$ is also a 2D vector and the (exponentiation, tanh, and addition) operations in its definition are performed element-wise, whereas the product multiplies a scalar by a vector.
Multiple levels of forward flow: For depth d, define the mask to be a vector of length d alternating between 0 and 1. For example, for $d = 3$, $m = [0, 1, 0]$. The mask pattern will be fixed for this assignment. Let W and B be matrices of dimension $d \times 2$, and let $W[i], B[i]$ refer to the i'th row. Then starting at z_0 we have $z_{i+1} = f_{W[i],B[i],m[i]}(z_i)$ and the final output is z_d. Denote this by $z_d = f^{(d)}(z_0)$
Dataset and inverse flow: Given a dataset of values $Z_0 = \{z_0 \sim \mathcal{N}(0, 1)\}$ the forward flow is applied to each example to get $X = Z_d = \{f^{(d)}(z_0) | z_0 \in Z_0\}$. As discussed in class, the inverse flow can be computed because $z[m]$ is preserved and θ only depends on $z[m]$. The inverse flow can be similarly applied to the entire dataset X to produce a set of initial values Z_0.
Density Estimation: To perform density estimation we recall that $\ln q(z_d) = \ln q(z_0) - \sum_{i=0}^{d-1} \ln \theta(z_i)_0$. Here $\theta(z_i)_0$ is the first entry of θ at the i'th level, i.e., the coefficient of the linear term in the definition of $x[1-m]$. Given a dataset X we need to compute the inverse flow to recover all the $z_i(x)$ for each $x \in X$. This allows us to compute the log likelihood by calculating $\ln p(z_0)$ and the corresponding values of $\theta(z_i)_0$ for each $x \in X$. We then optimize the parameters W, B using our favorite gradient based optimizer.

Implementation

Note: There are many demonstrations for normalizing flows on the internet. For this assignment I am expecting you to implement the code yourself from basic principles. If you consult any on line code in the process, please make sure to cite the source.

We will need to use automatic differentiation for the optimization. Hence it makes sense to use the pytorch demo given for BBVI/Reparameterization as a template for this portion.[1]

Task 1

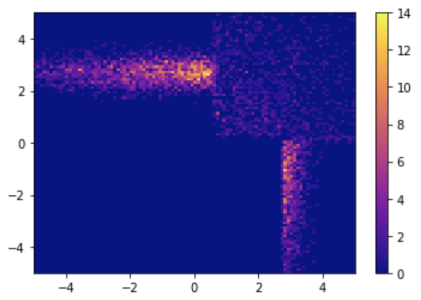
First implement forward and backward flow and make sure that they indeed form a bijection. For this part, first demonstrate that the code is correct by testing it on the following values: $d = 2$,

[1] We do not need reparameterization or specially defined backward functions for this assignment so other tricks from that document are not needed.

$W = [[-1., 1.], [1., -1.]]$, $B = [[1., 1.], [1., 1.]]$, the initial sample data is $Z_0 = [[1., 2.], [3., 4.]]$. With these values the forward flow produces $X = [[3.0866, 4.6222], [8.5234, 4.2419]]$, and inverse flow should reporoduce Z_0. Then draw 5000 independent examples for Z_0 and provide a visualization of the generated distribution. showing both steps of the 2 step flow for both forward and backward computations to illustrate visually that the bijection holds.

You may use any visualization you find informative. For example you can use a scatter plot color histogram. To illustrate, here is an example color histogram of data1.csv discussed below. If s holds your data then this can be produced with `matplotlib` using

```
plt.hist2d(s[:,0],s[:,1], bins=100, range=[[-5,5],[-5,5]], cmap='plasma')
cb = plt.colorbar()
```



Task 2

Next implement the density estimation procedure as outlined above by extending the code for calculating the reverse flow. To optimize the parameters W, B initialize them to random values from $\mathcal{N}(0, 1)$ and set the optimization objective to be the negation of the log likelihood of X. Use the Adam optimizer with learning rate 0.3 and 1000 optimization steps, each using the entire dataset (i.e., no mini-batches are used in this assignment). These values were tuned to make sure the optimization is stable but you may change them if it helps your optimization. Please discuss this in your report.

I have provided 2 datasets (data1.csv and data2.csv) for this assignment.

(1) For data1, run the density estimator for $d = 2, 3, 4, 5$. Plot the original dataset and compare it to a generated dataset (as in task 1) for each d. Does the algorithm succeed in generating a similar distribution? Does the success depend on the random seed used for initialization?

(2) Pick the most successful variant among your trials in the previous part and provide a plot of the training set log probability as a function of iterations, as well as a color histogram of the generated distribution at 0,100,200,. . . , iterations. Does the approximate distribution improve with training?

(3) For data2, run the density estimator for $d = 5$, reporting the learning curve and quality of distribution as in the previous part. Repeat this with multiple seeds to test Plot the original dataset and compare it to the generated dataset (as in task 1). Does the algorithm succeed in generating a similar distribution? Does the success result on the random seed used for initialization?

For Extra Fun and Credit

(1) [should be easy] Above we mainly evaluate the results qualitatively. Split the data into train and test portions and evaluate test set log likelihood as a function of depth and training iterations. Do the quantitative results agree with your visual conclusions? (2) [more work] I have provided two additional datasets (twomoon.csv and ledge.csv). Our flow representation is not very expressive

and with limited run time and exploration the model fails to learn these distributions. Explore some more expressive flows or alternative optimization process to learn a generative distribution for these datasets.

Submitting Your Assignment

Please pack the following 3 items into a directory: `YourNameHW3/`, pack the directory into `YournameHW3.zip` and upload the zip file on canvas. The items are:
(1) Please write a report on the experiments and results as requested above and put this into a PDF file - please call it `report.pdf`.
(2) The code, including a README file/portion that will tel me how to run it. The code should assume that data is in sub-directory `hw3data/`. Please make sure the code is well structured and easy to read (i.e., document it as needed). This portion can be a single file or multiple files.
(3) One PDF file which includes a "printout" of: the report, the code and README so that I do not need to print many different files for grading. Please call this `YourNameHW2Everything.pdf`.

Submission notes: Given the context it would make sense to use pytorch. You may use a Jupyter notebook but this is not required.

Grading Notes: I will use item 3 as the main tool for grading so submission of the "everything" file is crucial. The assignment will be graded based on the clarity of the code, its correctness, and the presentation and discussion of the results. I may or may not run your code. Following the naming conventions will make the management of grading a lot easier so please remember to follow it.