A systematic study of Masked Autoregressive Flows for density estimation

Sergio Hernan Garrido Mejia[1]

1 DTU Transport

Introduction

Since their introduction in [3], Normalizing Flows (NF) have not caught the attention that other deep generative models (VAE, GANs, RBM) caught. Recent advances and applications (such as WaveFlow for speech synthesis) of flow based models show how powerful they are compared to the available alternatives. I tried to do a systematic study of NF starting by understanding their theoretical base, empirical study and recent advances using them.

Normalizing Flows

NF is a model that helps approximate rich posterior distributions using a tractable base distribution e.g. a Gaussian (or a mixture of them) and bijective (one to one) transformations. Let $u \rightarrow N(0, 1)$ and f be a bijective function such that $x = f(u)$ and $u = f^{-1}(x)$. The density of the transformed variable x is given by:

$$p(x) = p(f^{-1}(x))|det(J(f^{-1}(x))|$$

Since the distribution of x and f are defined to be tractable, we can transform the transformed distributions with as many functions as we want as long as they fulfill the bijective requirement. Furthermore, we would also require that the term $|det(J(f^{-1}(x))|$ is easily computable.

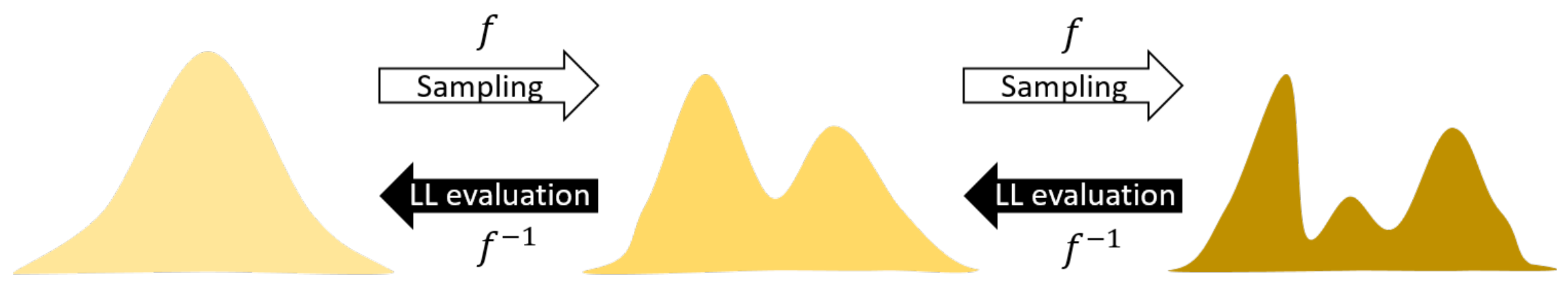

Figure 1: Normalizing Flow representation

Masked Autoencoder neural networks

The Masked Autoencoder neural networks for Density Estimation (MADE) [2], as an efficient way to do density estimation using 1) autoencoders: Neural networks that output an approximation of its input; and 2) conditional autoregression: a structure for density estimation where the probability of a random value depends only on previous variables. Mathematically $p(x) = \prod_{d=1}^{D} p(x_d|x_{<d})$ where $x_{<d}$ are all variables estimated before x. We can estimate the density of the variables with the autoregressive assumption efficiently using a mask that turns

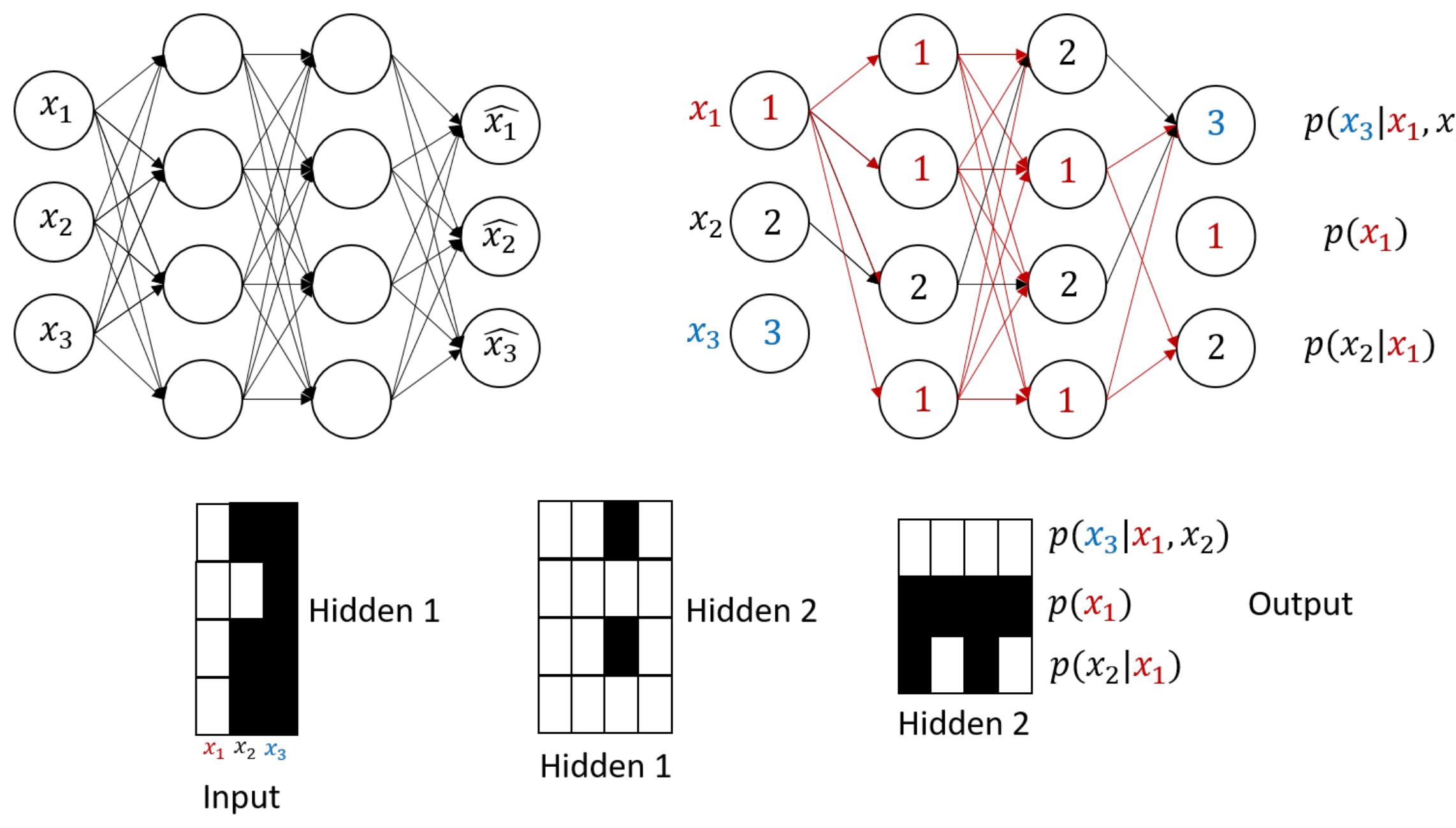

Figure 2: MADE representation. 1) The neural network on the top left is a classic autoencoder, 2) The matrices on the bottom are masks generated randomly for the autoencoder on 1), 3) The neural network on the top right is a MADE network using the architecture on 1) and the masks on 2). Note that MADE is order agnostic and connectivity agnostic so that you can create ensembles of the density estimates on parallel.

Masked Autoregressive Flow for Density Estimation

It is possible to combine the ideas from NF and MADE to create a NF were the transformations are the MADE neural networks. This was proposed in [5] as Masked Autoregressive Flows (MAF). Additional to their proposition, we use a batch normalization layer proposed in the realNVP paper [1]. The main idea of MAF is to parametrize the i^{th} conditional of the autoregressive model as: $p(x_i|x_{1:i-1}) = N(x_i|\mu_i, (exp(\sigma_i)^2))$ with $\mu_i = f_{\mu_i}(x_{1:i-1})$ and $\sigma_i = f_{\sigma_i}(x_{1:i-1})$. Define $u_i = N(0, 1)$, using the "reparametrization trick" we can rewrite $x_i = u_i * exp(\sigma_i) + \mu_i$, and $u_i = (x_i - \mu_i)exp(-\sigma_i)$. This completes our definition of the model by noting that f_{μ_i} and f_{σ_i} can be estimated by the means of a Masked Autoregressive neural network and that x_i is defined as a transformation of a random variable u_i with a tractable distribution with $log(|det(Jacobian)|) = -\sum \sigma_i$.The complete architecture of the model is as follows:

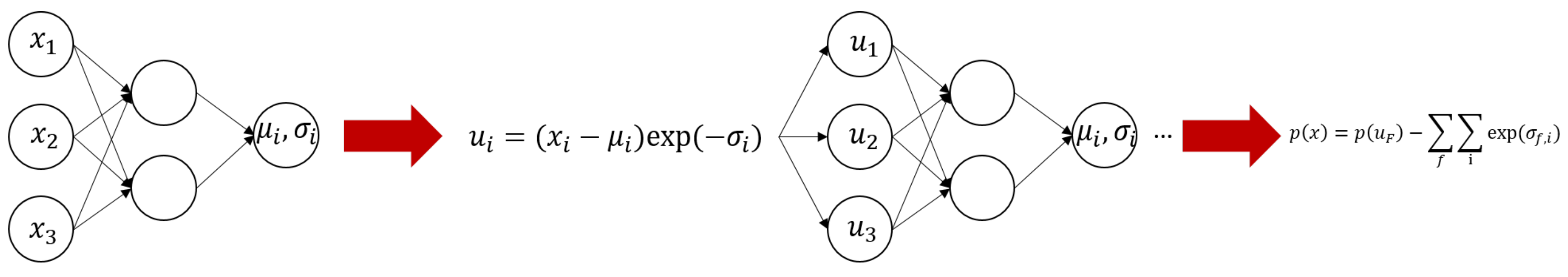
x_1
x_2
x_3
μ_i, σ_i
$u_i = (x_i - \mu_i)\exp(-\sigma_i)$
u_1
u_2
u_3
μ_i, σ_i
...
$p(x) = p(u_F) - \sum_f \sum_i \exp(\sigma_{f,i})$

Figure 3: MAF representation. Using iterative Masked Autoregressive networks as transformations into a basic distribution.

Data

The data used to test the model was criminal data from Bogota, Colombia. The data spans from 2004 and 2014 with more than 300k observations. The variables available for this data are latitude, longitude, exact time (up to seconds) and type of crime. The dataset was aggregated by hours and we took two different timeframes with a different amount of training data (all the previous data available from the same dataframe): 03/11/2004 11:00PM (200 points) and 03/11/2013 11:00PM (2000 points). The models were tested with the crimes for the same dataframe and four weeks ahead.

Model performance

Small dataset (200 training points)						
Model	AUC	1%	5%	10%	15%	20%
NF	0.8261	0.0476	0.2380	0.5714	0.6190	0.6190
KDE	0.8152	0.0476	0.2857	0.4761	0.5714	0.6190
GMM	0.8295	0.0476	0.2857	0.4285	0.6666	0.7619
Big dataset (2000 training points)						
Model	AUC	1%	5%	10%	15%	20%
NF	0.8806	0.125	0.3125	0.4375	0.625	0.875
KDE	0.9018	0.125	0.375	0.625	0.75	0.8125
GMM	0.8937	0.125	0.3125	0.4375	0.6875	0.9375

Table 1: Comparison of MAF, KDE and GMM using the Cumulative Accuracy Profile (CAP) using the number of points of the grid as the classifying parameter.

Performance

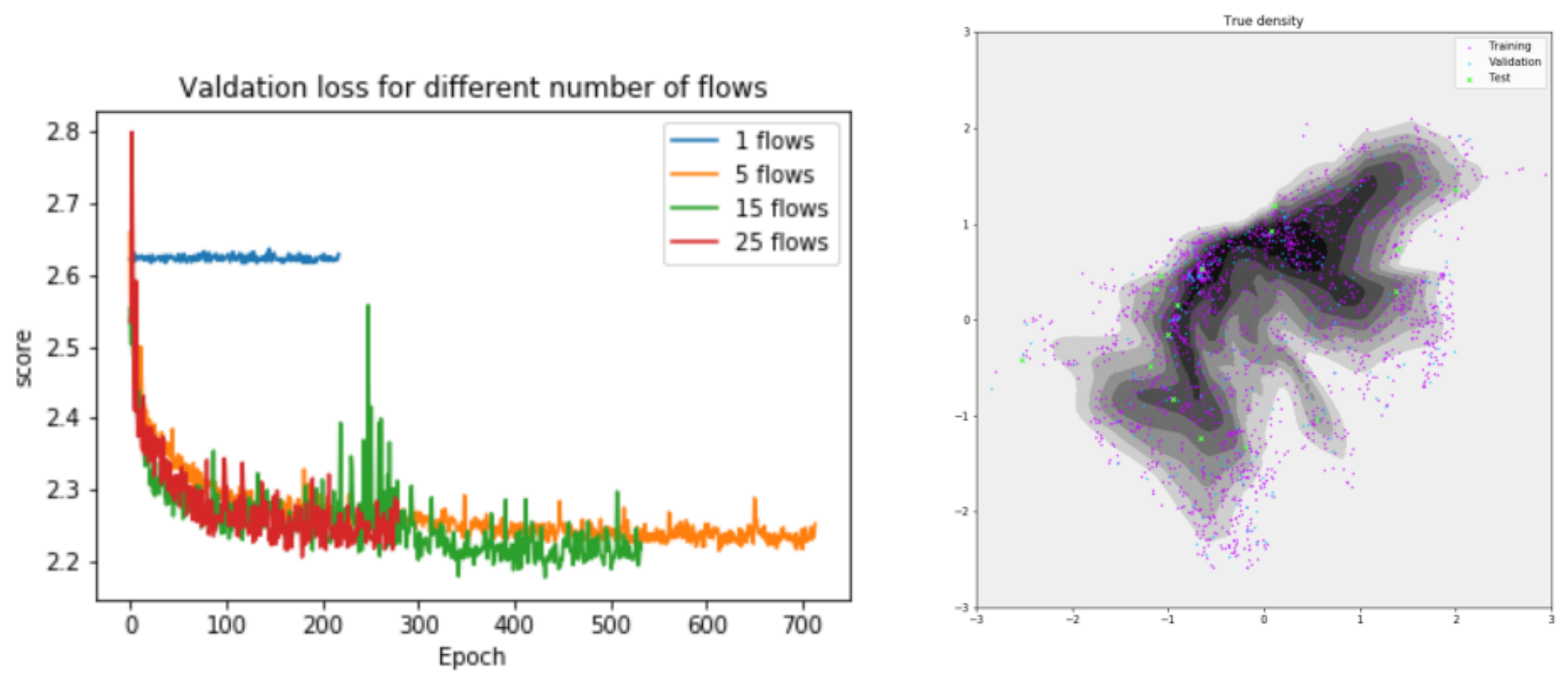

Figure 4: 1) Validation loss of the MAF model under different amount of blocks. 2) Sample density of a MAF using 25 blocks and 1024 hidden units inside the Masked Autoencoder network.

Key findings, applications and future work

Key findings:

- Training is highly unstable, same architecture can give extremely different results.
- A higher number of flows makes the training more unstable. However, a higher number of flows seems to increase the estimated density complexity and hence, fit. Note that this can be bad if you are overfitting the data.

Applications. These models are extremely flexible and can be used in a wide range of tasks that require generation of data or density estimation such as:

- Unsupervised learning.
- Anomaly detection.
- Image generation.

Future work:

- Making the training more stable, maybe through a more conscious initialization of weights, base distribution, hyperparameters, etc.
- Designing regularization for the flows, it would be interesting to explore flows in the context of bayesian neural networks, having a whole distribution for the base distribution parameters and any weights present on the estimation.
- Working with high dimensionality. This family of algorithms are designed for relatively small dimensional data, how can we make these algorithms better for high dimensional data?

Acknowledgements

I would like to thank Filipe Rodrigues for the fruitful discussions and the comments throughout the project development.

References

[1] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP. *arXiv e-prints*, art. arXiv:1605.08803, May 2016.

[2] M. Germain, K. Gregor, I. Murray, and H. Larochelle. Made: Masked autoencoder for distribution estimation. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 881–889, Lille, France, 07–09 Jul 2015. PMLR. URL `http://proceedings.mlr.press/v37/germain15.html`.

[3] D. Jimenez Rezende and S. Mohamed. Variational Inference with Normalizing Flows. *arXiv e-prints*, art. arXiv:1505.05770, May 2015.

[4] I. Kostrikov. Pytorch-flows. `https://github.com/ikostrikov/pytorch-flows`, 2018.

[5] G. Papamakarios, T. Pavlakou, and I. Murray. Masked Autoregressive Flow for Density Estimation. *arXiv e-prints*, art. arXiv:1705.07057, May 2017.

shgm@dtu.dk