Normalizing Flows as Classifiers

Dejan Stancevic

University of Gothenburg

gusstancde@student.gu.se

March 13, 2024

Introduction

- Normalizing flows have various uses:
  - Generative models
  - AI explainability
- In this project, we use normalizing flows as classifiers
  - Synthetic dataset
  - Adversarial attack
  - MNIST dataset
- Based on "Semi-Supervised Learning with Normalizing Flows" paper by Izmailov et al.
- Normalizing flow models are based on RealNVP architecture
- Code is available at `https://github.com/DejanStancevic/ms-in-dnns/tree/master/Project`

Idea behind Normalizing Flows

- Find a diffeomorphism f from a data space X to a base space Z where we have a nice (Gaussian) distribution

$$f : X \to Z$$

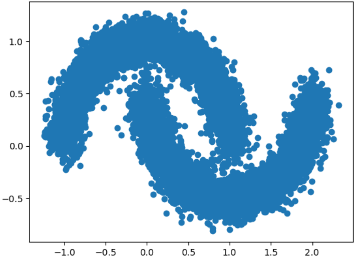
- Set the diffeomorphism to be a neural network, f_θ, and try to learn it by maximizing the likelihood

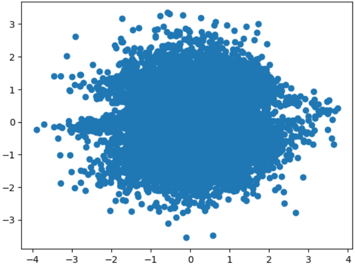
$$p_X(x) = p_Z(f_\theta(x)) \left| \det \left(\frac{df_\theta(x)}{dx} \right) \right|$$

or

$$\log(p_X(x)) = \log(p_Z(f_\theta(x)) + \log \left(\left| \det \left(\frac{df_\theta(x)}{dx} \right) \right| \right)$$

Idea behind Normalizing Flows





Idea behind Normalizing Flows as classifiers

- Introduce more Gaussians in Z so that every class k can be associated to a different Gaussian
- We still think of the diffeomorphism as a neural network, f_θ, and try to maximize the likelihood

$$\log(p_X(x|k)) = \log(p_{Z,k}(f_\theta(x)) + \log\left(\left|\det\left(\frac{df_\theta(x)}{dx}\right)\right|\right)$$

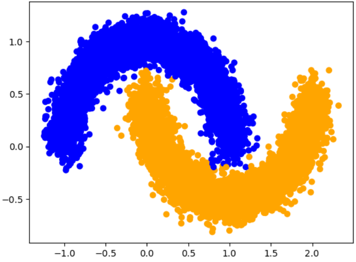
- Given $x \in X$ our model predicts

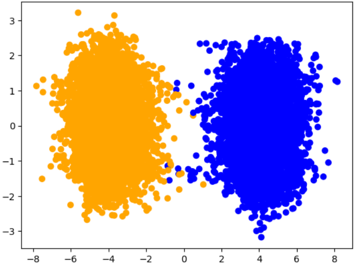
$$arg \max_k p_{Z,k}(f_\theta(x))$$

or

$$arg \max_k \frac{p_{Z,k}(f_\theta(x))}{\sum_c p_{Z,c}(f_\theta(x))}$$

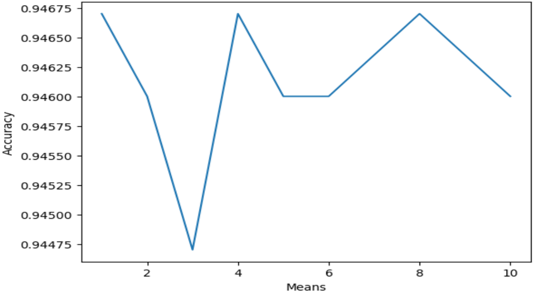
Idea behind Normalizing Flows as classifiers





Synthetic Dataset: Introduction

- The moons dataset from the sklearn library was generated with noise levels of 0.1, 0.25, and 0.5
  - 8500 training data points
  - 1500 testing data points
- Two models were compared
  - 3-layer feedforward neural network
  - Normalizing flow classifier with five coupling layers
- Gaussians with means $(\pm 4, 0)$ and identity covariance matrices



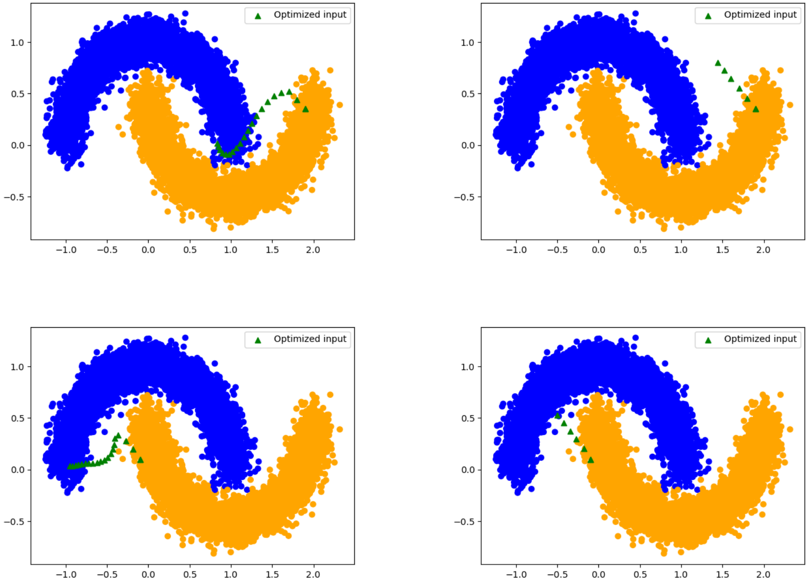
Synthetic Dataset: Results

Noise	Normalizing Flow	FNN
0.1	100 %	100 %
0.25	94.73 %	94.54 %
0.5	82.80 %	82.87 %

Adversarial Attack

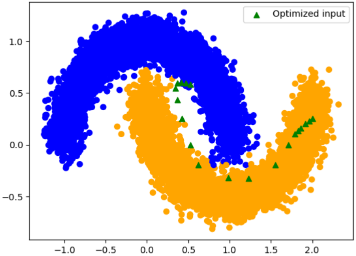
- An adversarial attack was performed on the moons dataset with a noise level of 0.1
- Standard optimization of an input to the FNN such that the predicted probability of a certain class gets bigger than 99%
- For the normalizing flow, the likelihood of an input belonging to a different class was optimized for 20 steps

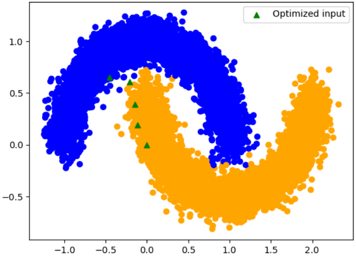
Adversarial Attack



Adversarial Attack

- When standard normalizing flow is combined with the FNN we get the best results i.e. optimized input spends the most time on the data manifold (not in the report)



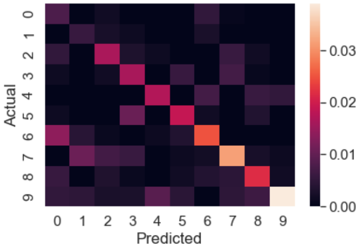


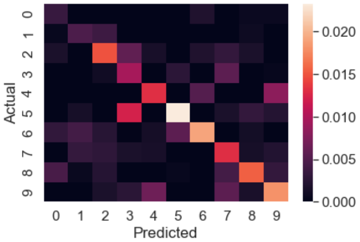
MNIST Dataset: Introduction

- Standard MNIST dataset provided by torchvision
  - 60000 training data points
  - 10000 testing data points
- Two models were compared
  - Convolutional neural network with two convolutional layers and one hidden linear layer
  - Normalizing flow classifier with six coupling layers
- Gaussians had means sampled from a uniform distribution on a box $[0,1]^{28\times 28}$ and identity covariance matrices

MNIST Dataset: Results

Normalizing Flow	CNN
98.33 %	99.13 %





Conclusion

- We showed that normalizing flows are a viable option for classification tasks
- Cons: More complex and longer training times
- Pros: Interpretability and robustness to adversarial attacks
- Further studies could delve into optimizing training procedures to improve the efficiency of normalizing flow classifiers
  - Investigate how the geometry of Gaussians in the base space affects the performance of normalizing flow classifiers.
- Exploring the performance of normalizing flows in semi-supervised and unsupervised learning settings

Questions

Thank You