Diffusion-based Deep Generative Models for Assessing Safety in Autonomous Vehicles

Connie Trojan

Supervised by: Chris Nemeth[1], Paul Fearnhead[1], Mark Bell[2]

[1]Lancaster University [2]Transport Research Laboratory

26th July 2022

The CARLA Simulator

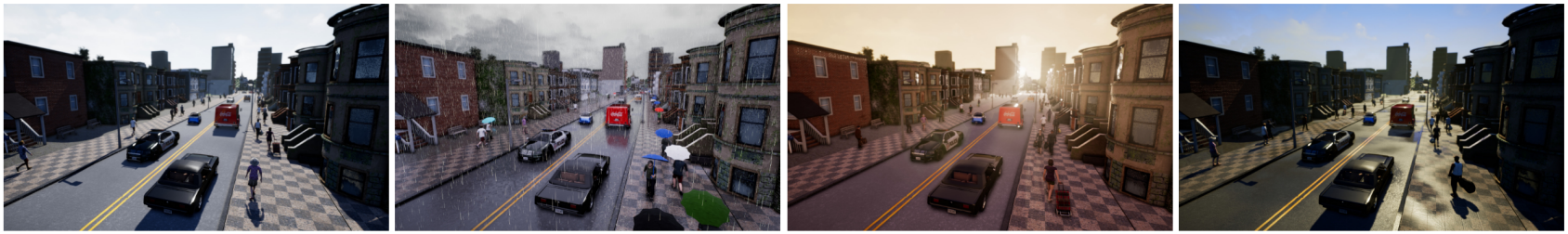


Figure: Dosovitskiy et al (2017)

- Purpose-built driving simulator with realistic physics and readouts for various sensors.
- Allows testing of autonomous AI in different scenarios, can analyse performance by varying starting parameters (e.g. weather conditions, starting speed).

Scenarios in CARLA

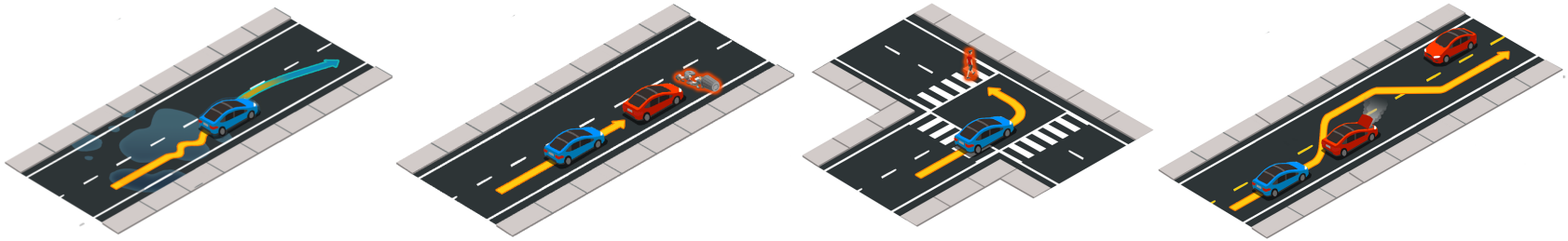


Figure: https://leaderboard.carla.org/scenarios/

- A scenario in CARLA is a detailed description of the initial conditions of a scene, ie. positions and speeds of road users, road conditions etc.
- Problem 1: Recreating real scenes as scenarios in CARLA from crash data/ dashcam footage.
- Problem 2: Generating realistic data for new scenarios.

Scenarios in CARLA

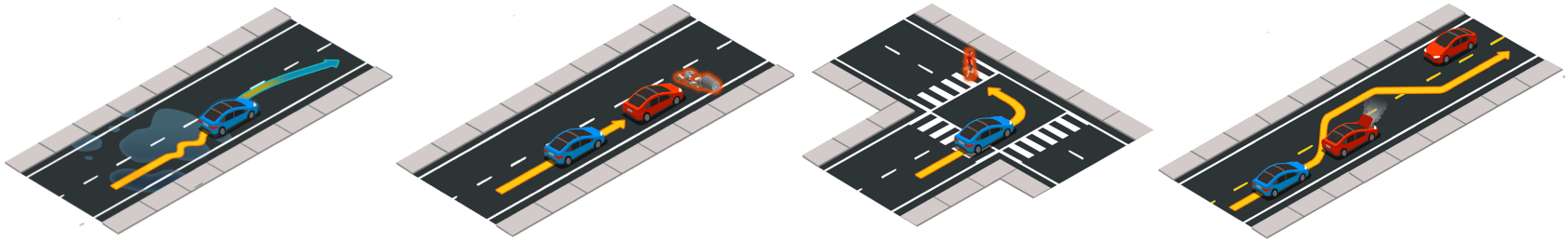


Figure: https://leaderboard.carla.org/scenarios/

- A scenario in CARLA is a detailed description of the initial conditions of a scene, ie. positions and speeds of road users, road conditions etc.
- Problem 1: Recreating real scenes as scenarios in CARLA from crash data/ dashcam footage.
- Problem 2: **Generating realistic data** for new scenarios.

Generative modelling

- Generate samples from an unknown probability distribution given a finite set of samples from the true distribution.
- Might try to directly learn the pdf of the data using maximum likelihood estimation (variational autoencoders, normalising flows). Problem: difficult to balance model complexity with tractability.
- Could define the pdf implicitly by a generative process (GANs). Problem: difficult to evaluate sample quality.

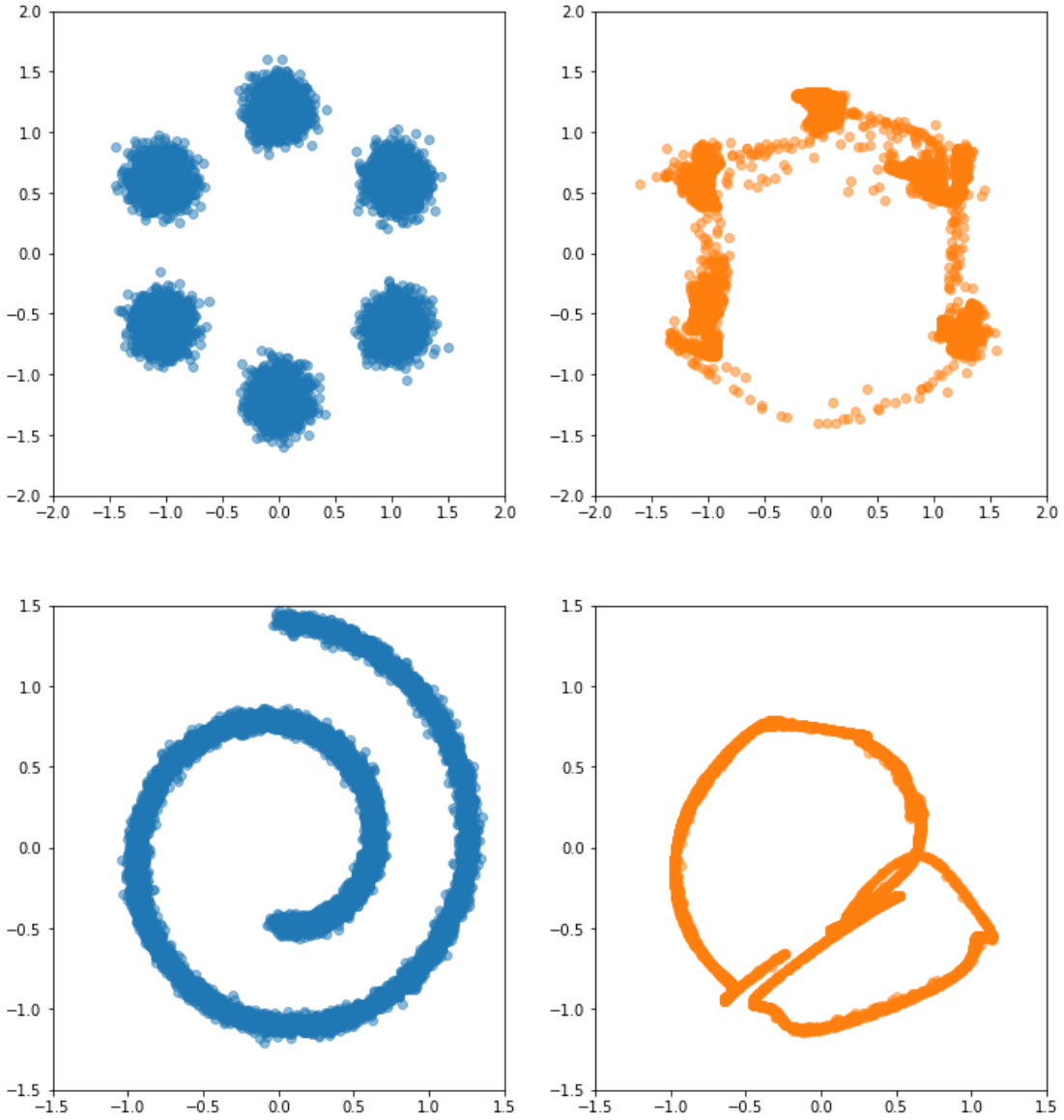
Neural Networks

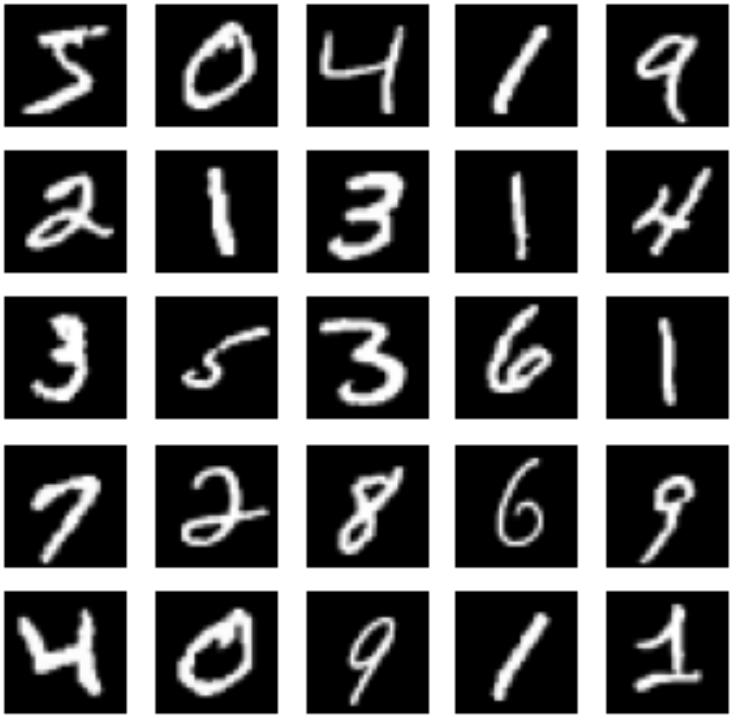
- Class of models popular for use as general function approximators.
- Composed of a sequence of linear transformations alternating with the application of a nonlinear activation function.
- Usually trained by stochastic gradient descent (SGD), requiring a differentiable objective function known as a loss function.

Generative Adversarial Networks

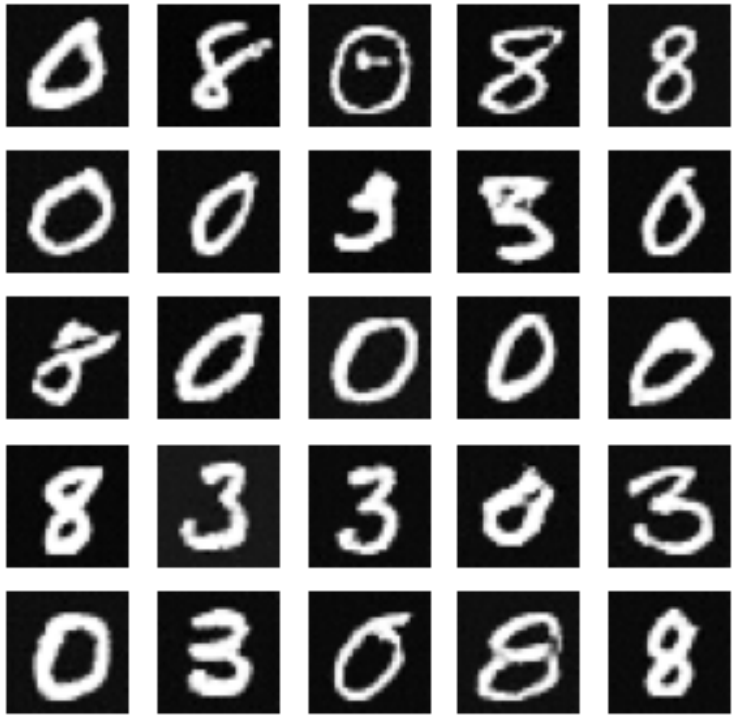
- Most well known deep generative model: **generative adversarial networks** (GANs).
- Train a generator network to generate samples from noise and a discriminator network to distinguish between real and fake samples.
- Training is a min/max problem: alternate between training the discriminator to label samples correctly and the generator to fool the discriminator.

- Good at generating realistic samples, ie. samples from areas of high probability density.
- However, adversarial training can be unstable and very time consuming.
- GANs can struggle to match the full data distribution - prone to **mode collapse** and **mode hopping**, ie. only generating samples from some parts of the distribution.
- Can improve performance by changing to an objective function with better properties.


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

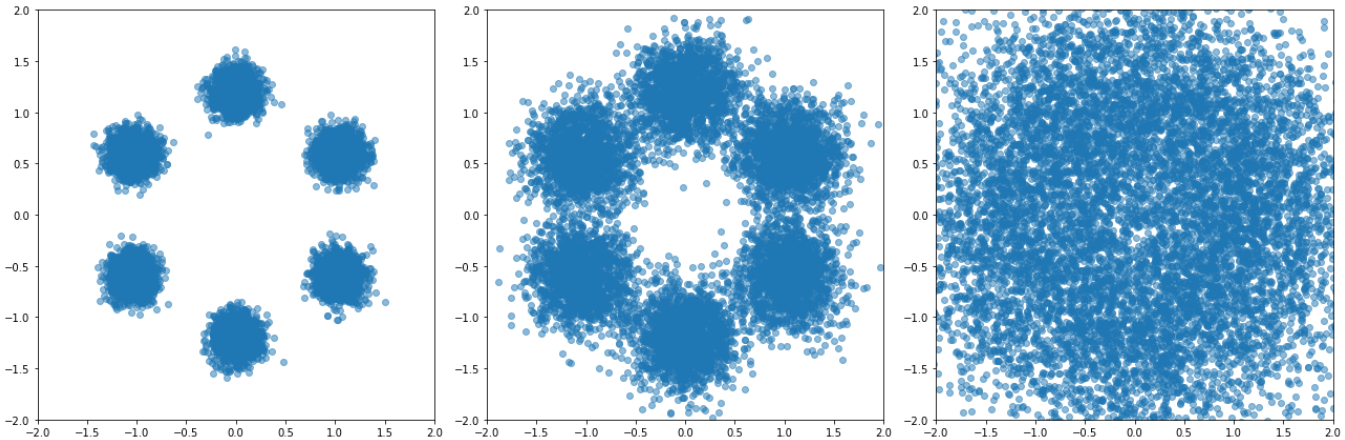


(a) MNIST digits



(b) GAN samples

Diffusion Models



- Idea: hard to generate data from noise, easy to generate noise from data.
- Define a stochastic process that gradually adds noise to the data until it resembles pure Gaussian noise, then learn the **time reversal** of this process to convert noise to data.

Diffusions

- Use a continuous time stochastic process called a **diffusion**.
- These are defined by a **stochastic differential equation**,

$$dX_t = \underbrace{f(X_t, t)dt}_{\text{drift}} + \underbrace{g(X_t, t)dW_t}_{\text{noise}}$$

- When g depends only on t, the distribution of $X_t \mid X_0$ will be Gaussian, e.g. if $dX_t = g(t)dW_t$ then $X_t \mid X_0$ has distribution $N(X_0, \int_0^t g(s)^2 ds)$ for $t > 0$.

Simulation

- **Euler approximation** - given initial value X_0, iterate:

$$X_{i+1} = X_i + f(t_i, X_i)(t_{i+1} - t_i) + g(t_i, X_i)(W_{i+1} - W_i)$$

- Here W_t is a Brownian motion, so this is equivalent to:

$$Y_{i+1} = Y_i + f(t_i, Y_i)(t_{i+1} - t_i) + g(t_i, Y_i)\sqrt{(t_{i+1} - t_i)}\, Z_{i+1}\,,$$

  with $Z_i \sim N(0, 1)$

- Typically use time intervals of constant length.
- Might require hundreds/thousands of iterations to get a good approximation.

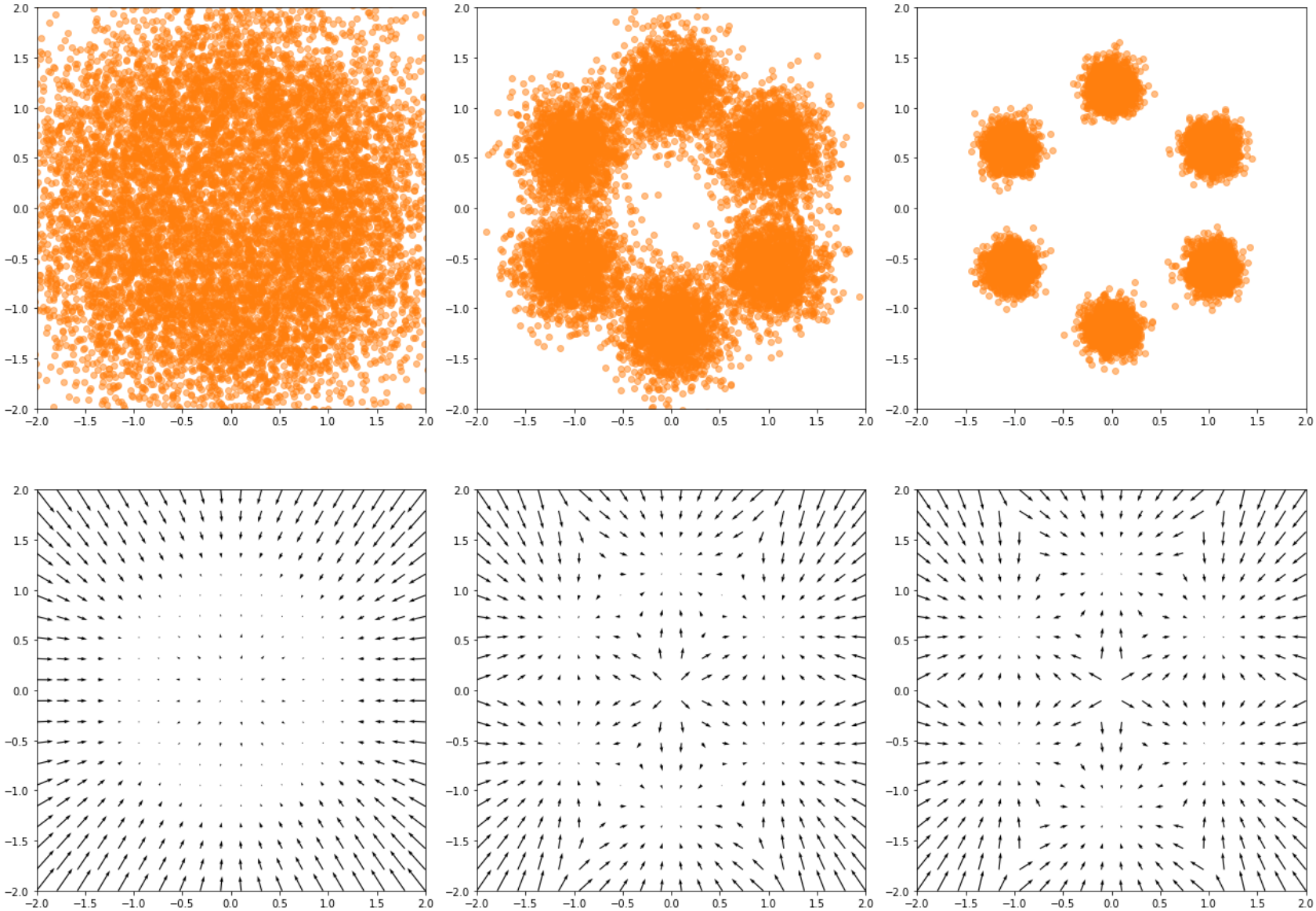
Time Reversal

- The time reversal of $dX_t = f(X_t, t)dt + g(t)dW_t$ is:

$$dX_t = \left[f(X_t, t) - g(t)^2 \nabla_x \log p_t(X_t)\right] dt + g(t)dW_t$$

- Here, the **score function** $\nabla_x \log p_t(x)$ is the gradient of the log pdf of X_t *with respect to* x.

- We know that $p(x_t|x_0)$ is Gaussian, so if we know the data distribution $p(x_0)$ then we are done.


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

Score Matching

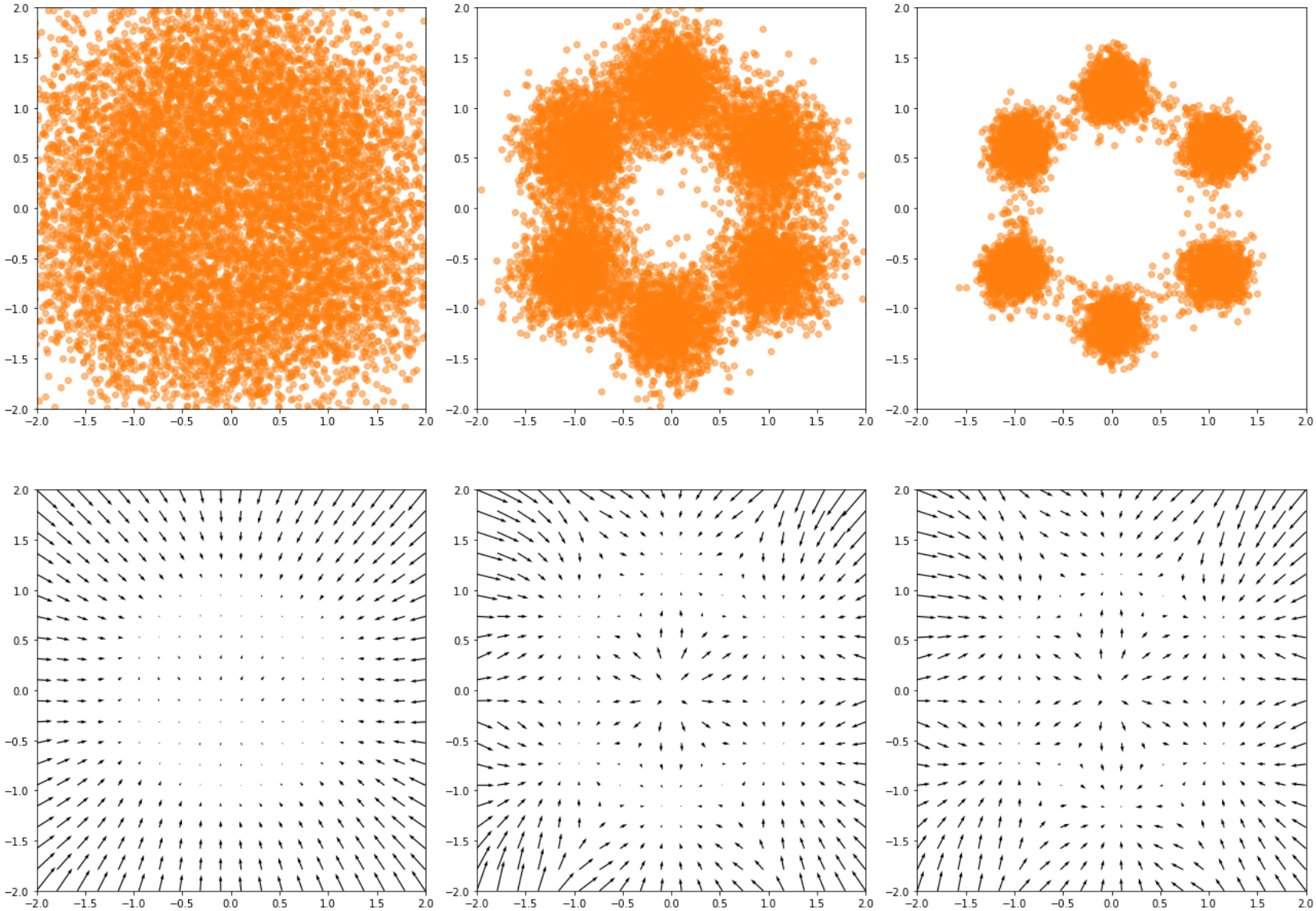
- Method for learning unnormalised probability density models by learning the score function of the data distribution.
- Want to find parameters θ minimising the MSE between the approximation $\psi(x;\theta)$ and the true score $\nabla_x p(x)$.
- **Denoising score matching**: approximate by taking samples from the data and corrupting them with Gaussian noise, learning a function that points towards the uncorrupted sample.
- Objective $\min_\theta \mathbb{E}_{p(x,\tilde{x})}\left[\frac{1}{2}\|\psi(\tilde{x};\theta) - \nabla_{\tilde{x}}\, p(\tilde{x}\,|\,x)\|^2\right]$ is equivalent to matching score of $\tilde{x}$.

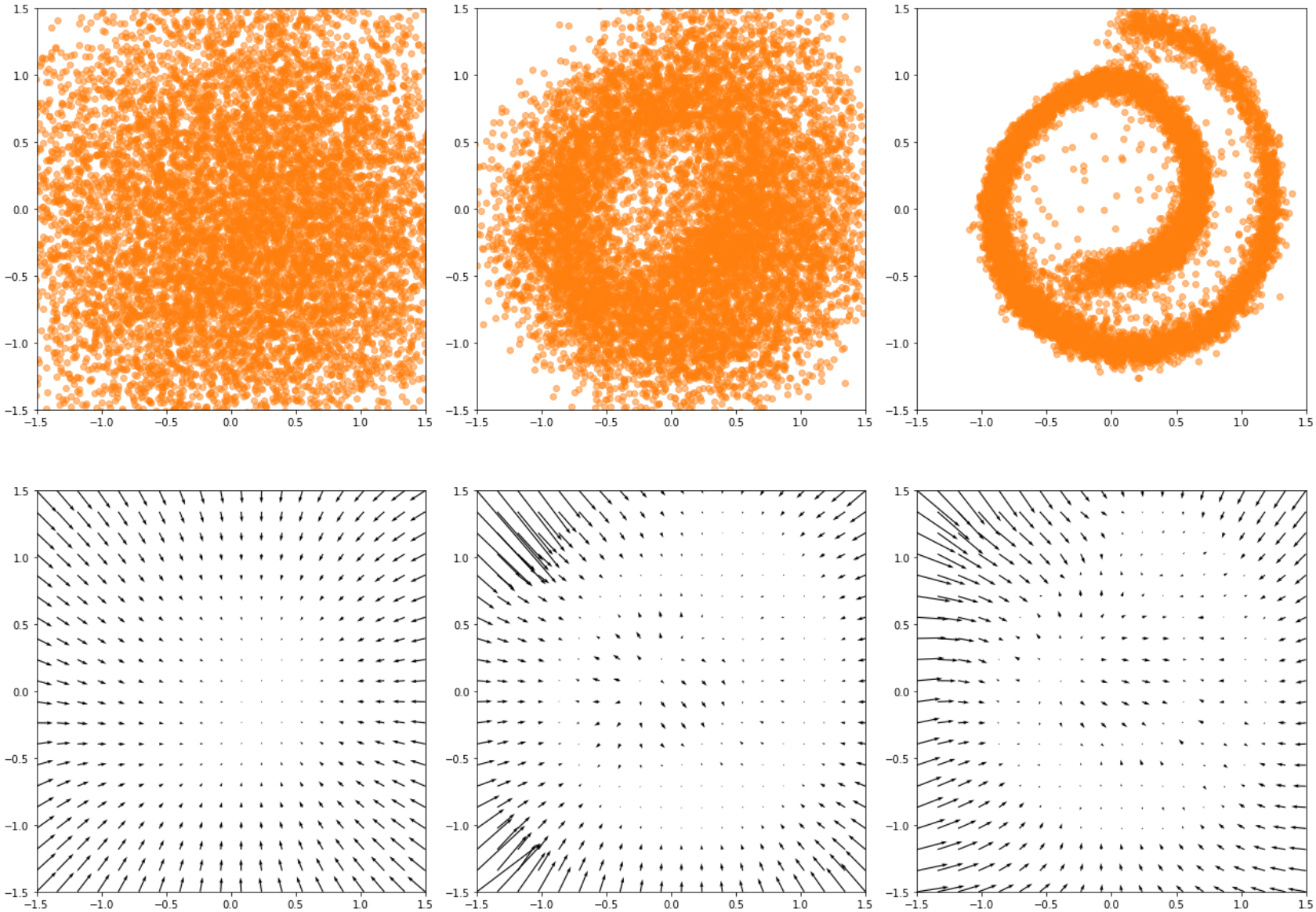
Score Matching

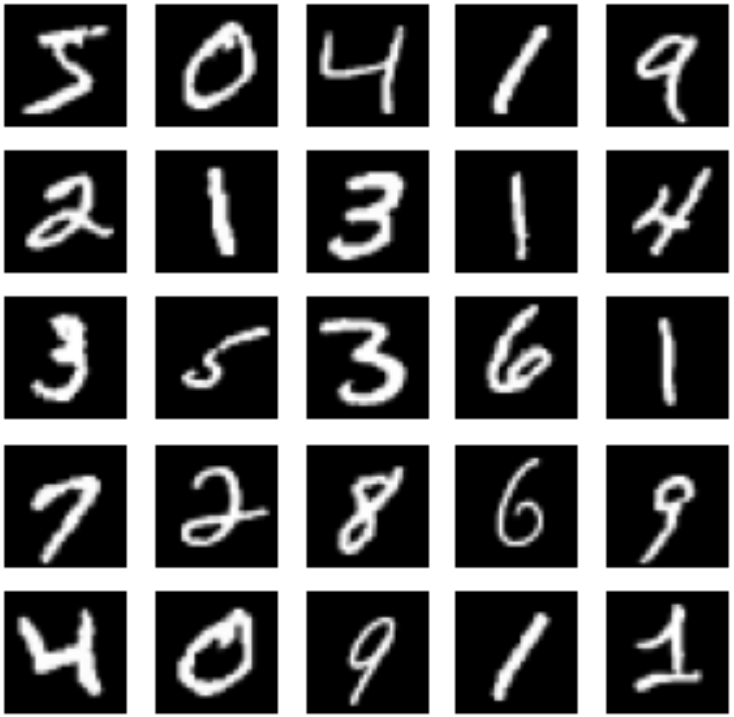
- Denoising score matching learns the score function of the data + noise, which is what we need for diffusions.
- We also need to learn a **time conditional** function to estimate the score function for different noise levels.
- e.g. for $dX_t = g(t)dW_t$, minimise:
  $$\min_\theta \mathbb{E}_t \left\{ \lambda(t)\, \mathbb{E}_{p(x_0,x_t)} \left[\frac{1}{2} \| \psi(x_t, t; \theta) - \frac{1}{\int_0^t g(s)^2 ds} (x_0 - x_t) \|^2 \right] \right\}$$
- Approximate expectations with MC sampling - sample a time uniformly from $(0, 1]$, take a sample x_0 from the data, and generate corrupted sample $\tilde{x}$ from $p(x_t \mid x_0)$.

Score Network

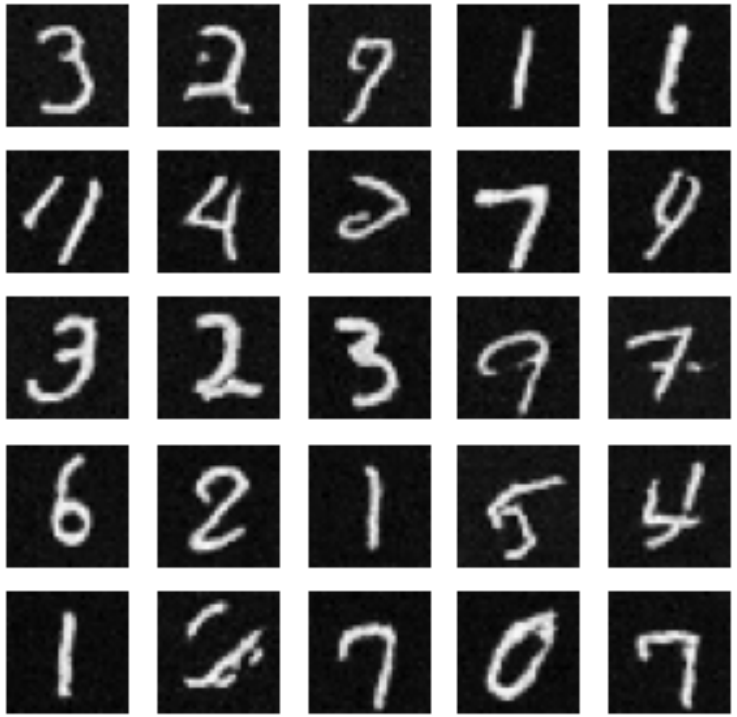
- Can in theory use any architecture as long as time is an input.
- In practise the architecture choice has a big impact on performance.
- Low dimensional distributions: simple feedforward neural network with time as an additional input to each layer.
- High dimensional/complex distributions: try to use domain knowledge - e.g. convolutional networks work well for image data.




1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5



(a) MNIST digits



(b) Diffusion samples

Summary

- Deep diffusion models learn to gradually transform noise into data via an SDE.
- They seem to be better at matching distributions/ producing diverse samples than GANs, and their training is more stable.
- **However**, sampling is slow compared to GANs as we need to simulate hundreds of steps to approximate the reverse diffusion, each of which requires evaluation of the score network.

Further Research

- Impact of choice of diffusion, improving slow sampling from diffusion models.
- Conditional generation - generating scenarios likely to result in accidents.
- Explore links to approximate Bayesian computation - CARLA as a black-box generative model.
- Potential data issues - relatively small number of existing examples, some elements of the scenario data may be discrete.

Any Questions?

References

Dosovitskiy, A. and Ros, G. and Codevilla, F. and Lopez, A. and Koltun, V. *CARLA: An Open Urban Driving Simulator*.

In Proceedings of the 1st Annual Conference on Robot Learning, volume 78 of Proceedings of Machine Learning Research, pages 1–16. PMLR, 2017.

Goodfellow, I. and Pouget-Abadie, J. and Mirza, M. and Xu, B. and Warde-Farley, D. and Ozair, S. and Courville, A. and Bengio, Y. *Generative Adversarial Nets*.

In Advances in Neural Information Processing Systems, volume 27, page 2672–268. Curran Associates, Inc., 2014.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Pool. *B. Score-based generative modeling through stochastic differential equations*.

In International Conference on Learning Representations, 2021.