SpaceGAN
A generative adversarial net for geospatial point data

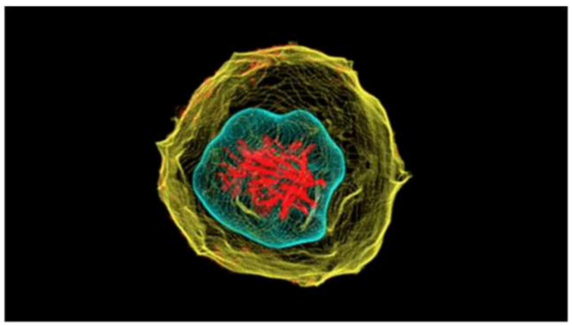
Konstantin Klemmer, **Adriano Koshiyama**, and **Sebastian Flennerhag**

Deep Learning is revolutionising science

Machine learning predicts the look of stem cells

Amy Maxmen

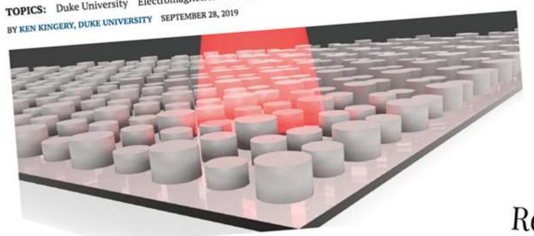
05 April 2017



New Metamaterials for Energy Harvesting Designed by Machine Learning

TOPICS: Duke University Electromagnetics Energy Materials Science

BY KEN KINGERY, DUKE UNIVERSITY SEPTEMBER 28, 2019



Research into effectiveness of AI at diagnosing disease

By Emma Morriss · September 26, 2019

Deep Learning is also entering social science

Fusing Social Networks with Deep Learning for Volunteerism Tendency Prediction

Yongpo Jia[1,2], Xuemeng Song[2], Jingbo Zhou[3], Li Liu[2], Liqiang Nie[2], David S. Rosenblum[2]

Using the TensorFlow Deep Neural Network to Classify Mainland China Visitor Behaviours in Hong Kong from Check-in Data

Shanshan Han[1], Fu Ren[1,2], Chao Wu[1], Ying Chen[1], Qingyun Du[1,2,3,4,*] and Xinyue Ye[5,*]

Predicting judicial decisions of the European Court of Human Rights: a Natural Language Processing perspective

Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preotiuc-Pietro, Vasileios Lampos

Learning to Generate Reviews and Discovering Sentiment

Alec Radford[1] Rafal Jozefowicz[1] Ilya Sutskever[1]

Topic today →

Augmenting correlation structures in spatial data using deep generative models

Konstantin Klemmer *
The Alan Turing Institute &
University of Warwick
kklemmer@turing.ac.uk

Adriano Koshiyama *
The Alan Turing Institute &
University College London
akoshiyama@turing.ac.uk

Sebastian Flennerhag
The Alan Turing Institute &
University of Manchester
sflennerhag@turing.ac.uk

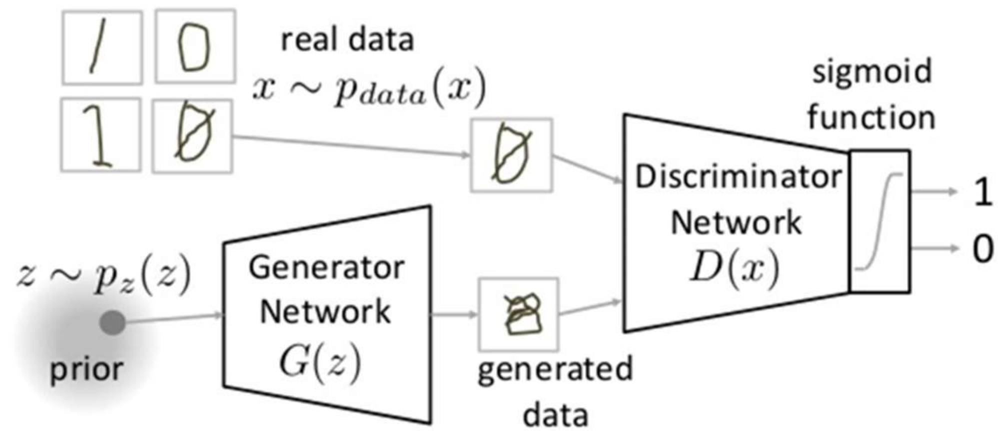
Generative Adversarial Networks

Generative Adversarial Networks - What are they?

- ➔ **Discriminator**: Train a classifier to distinguish between the two distributions using samples
- ➔ **Generator**: Train to generate samples that fool the discriminator
- ➔ **Minimax game** alternates between training discriminator and generator

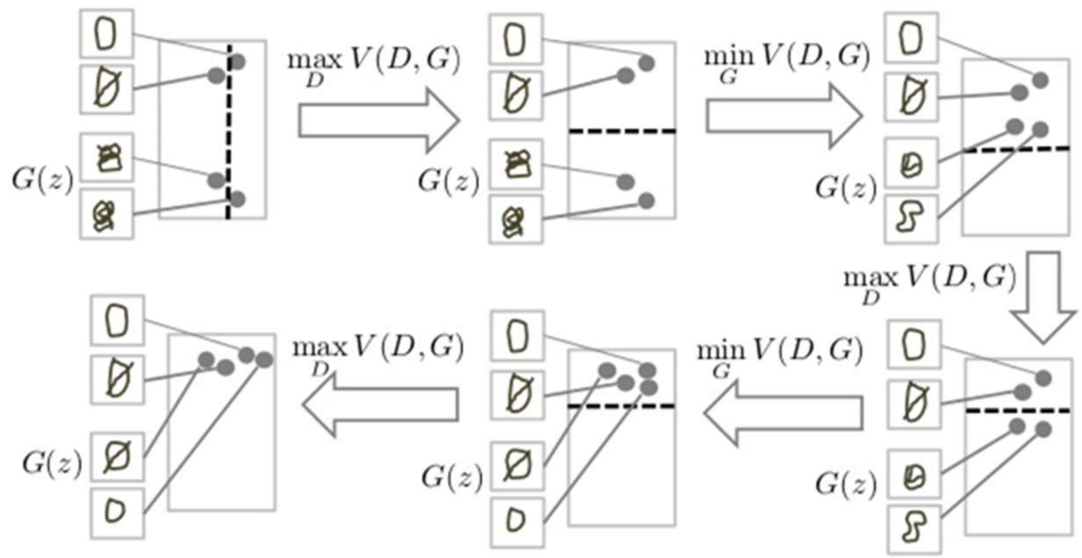
$$\min_G \max_D V(D,G)$$

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$



Generative Adversarial Networks - How to train them?

- **Comparison (Discriminator)**
  - Use a hypothesis test or comparison to build an auxiliary model to indicated how data synthetically generated differs from the observed one.
- **Learning (Generator)**
  - Adjust model parameters to better match the data distribution using the comparison.

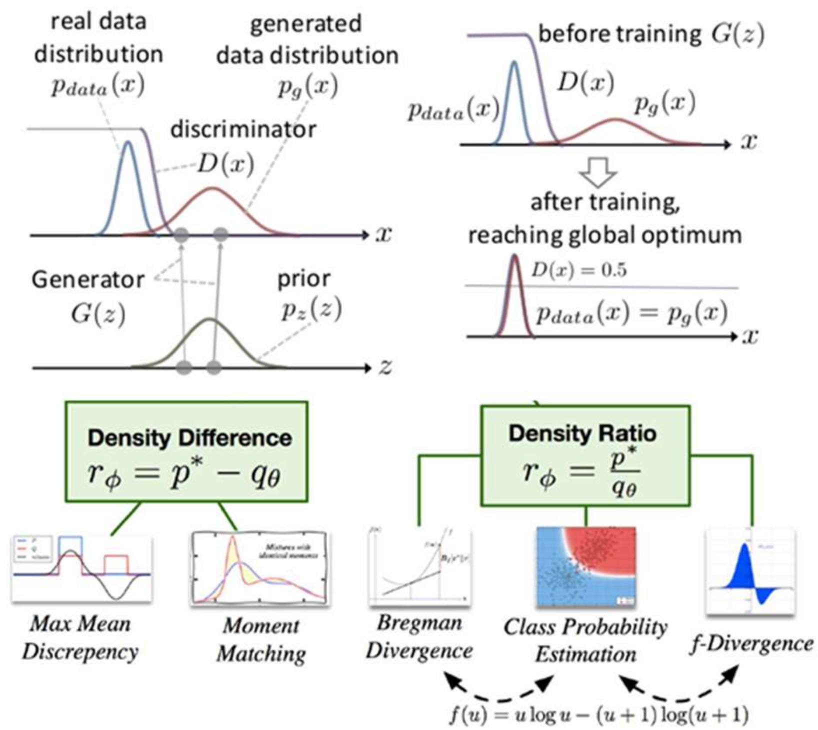


Generative Adversarial Networks - How to train them?

- ➔ Learn a generative model
  $$\mathbf{x} = G_\theta(\mathbf{z}), \mathbf{z} \sim p_{\mathbf{z}}$$
- ➔ Goal
  - ◆ Given samples $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ from the distribution $p^*(\mathbf{x})$, find θ
- ➔ How: define a joint loss function and alternate between training D and G

$$\min_G \max_D V(D, G)$$

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$



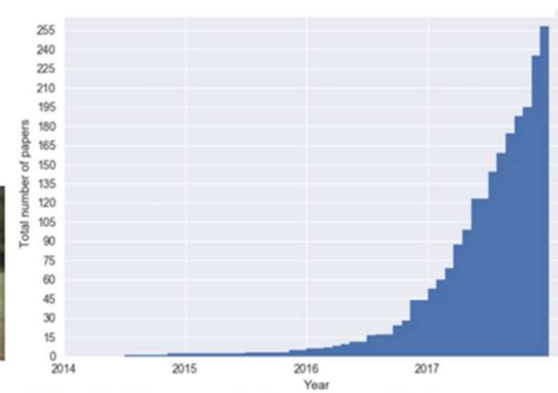
Generative Adversarial Networks - Hype or Hope?

https://github.com/junyanz/CycleGAN/blob/master/imgs/horse2zebra.gif

Examples from Ian Goodfellow: https://arxiv.org/pdf/1701.00160.pdf

Ferenc Huszar
@fhuszar

practicin' the alphabet with my son:
A is for AffGAN
B is for B-GAN
C is for Conditional GAN
D is for DCGAN
E is for EBGAN
F is for f-GAN



https://github.com/hindupuravinash/the-gan-zoo

Pseudo-code - GAN - Goodfellow et al., 2014

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**
  for k steps **do**
  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

  end for
  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

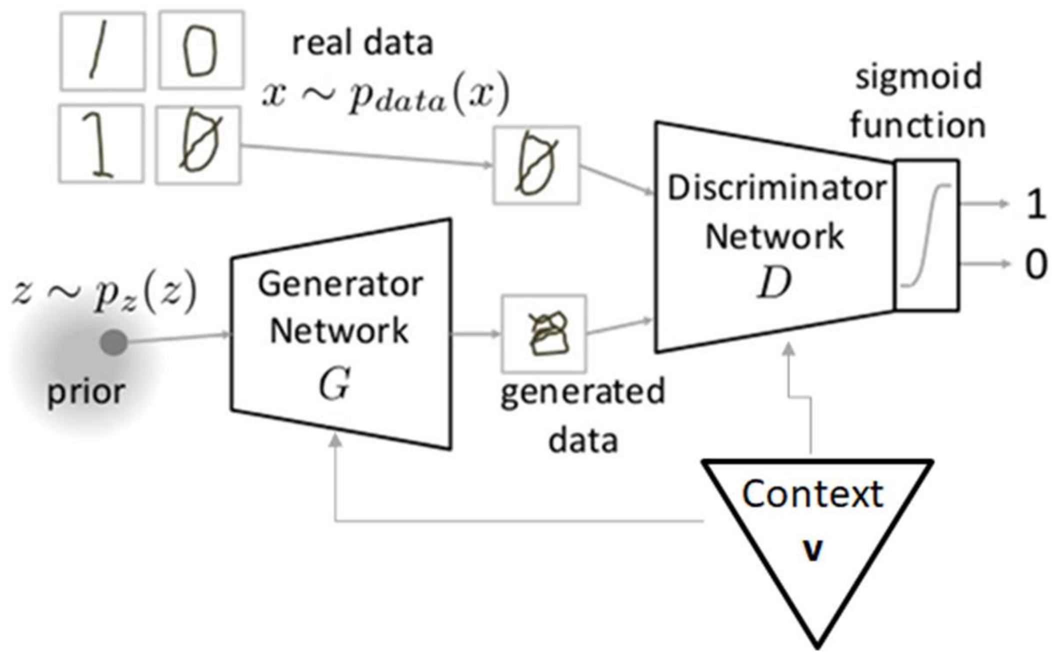
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Conditional GANs

Conditional GANs - What are they?

➔ Conditional GANs (cGANs) are an extension of a traditional GAN, when both G and D decision is based not only in noise or generated inputs, but include an additional context set $\mathbf{v}$:
  - a class label
  - historical data
  - additional features
  - neighbourhood structure



$$\min_G \max_D V(D,G) = E_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x}|\mathbf{v})] + E_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z}|\mathbf{v})))]$$

Conditional GANs - Why we need them?

➔ We inherit all the advantages of GANs
  - Data anonymization
  - Novelty detection
  - Knowledge discovery

➔ With additional bonuses of conditioning information
  - Conditional Inference: **v** can represent a current/expected market condition; such application is more appropriate in cases where the data follows a sequence (time series, text, etc.) or when the user wants to build "what-if" scenarios
  - Data augmentation: imbalanced classification, in particular to fraud detection; they are able to show that cGANs compare favourably to other traditional techniques for oversampling

SpaceGAN

SpaceGAN - Merging ML and GIS

TL;DR:

➔ We develop a novel conditional GAN that is able to augment spatial patterns in the input and can produce high-fidelity data samples with realistic representations of the underlying spatial correlation structure.

Machine Learning with Geospatial Data

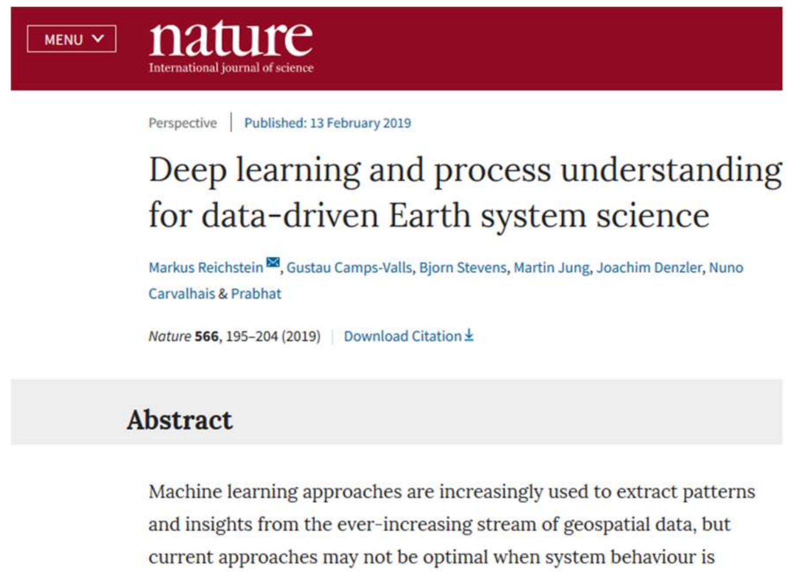
Why is it important?

- ➔ Many prime ML applications are inherently spatial in nature (e.g. autonomous cars and fleets, crop yield prediction from satellite images, hurricane trajectory forecasting,...)
- ➔ There is not so much interdisciplinary work going on yet (pointed out at the NeurIPS’18 workshop on spatio-temporal modeling)

Machine Learning with Geospatial Data

Why is it difficult?

- ➔ Data is inherently non-iid
- ➔ Many assumptions (e.g. Euclidean distances) are too simplistic
- ➔ Current methods often struggle with modeling spatial complexities (e.g. the interplay of short- versus long-distance dependencies)


MENU
nature
International journal of science

Perspective | Published: 13 February 2019

Deep learning and process understanding for data-driven Earth system science

Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais & Prabhat

Nature **566**, 195–204 (2019) | Download Citation

Abstract

Machine learning approaches are increasingly used to extract patterns and insights from the ever-increasing stream of geospatial data, but current approaches may not be optimal when system behaviour is

This is a great read on the topic!

SpaceGAN - Merging ML and GIS

What can deep learning models learn from geographic information science?

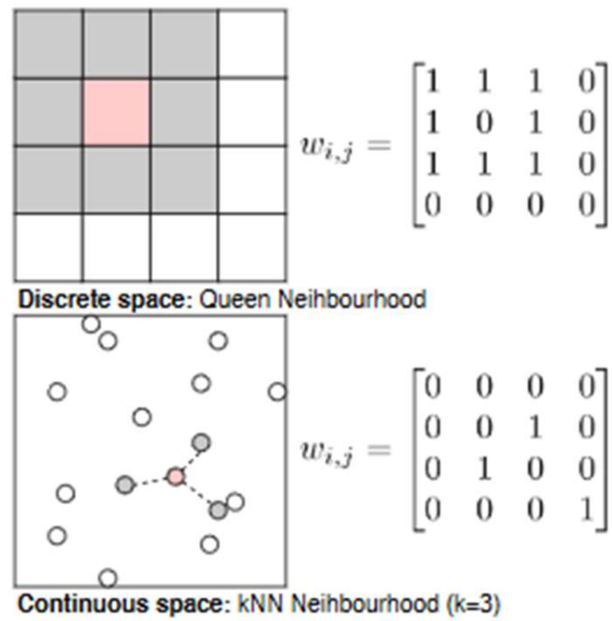
- ➔ How to include and learn from (spatial) neighbourhood information.
  - **(1) We propose a novel neighbourhood conditioning technique for GANs.**

- ➔ How to measure the models ability to capture spatial structures.
  - **(2) We evaluate and select the generator based on its ability to reproduce the observed spatial patterns.**

(1) Neighbourhood conditioning

We use the **local neighbourhood** of each datapoint as **context vector** to condition our GAN:

- ➔ Works for discrete and continuous spatial data
- ➔ Allows flexible neighbourhood definition / optimization (kNN, queen,...)
- ➔ Flexible weight choice (discrete, distance-based,...)



(2) Optimal representation of spatial structures

We measure the models **ability to replicate observed spatial patterns** throughout training and select the best one:

- ➔ GIS offer's a great tool to measure spatial autocorrelation: Moran's I
- ➔ We propose a new metric that measures the difference in observed versus augmented local spatial autocorrelation
- ➔ cGAN optimization objective remains unchanged!

Compute local Moran's I

$$I_i = (n-1)\frac{y_i - \bar{y}}{\sum_{j=1, j\neq i}^{n}(y_j - \bar{y})^2} \sum_{j=1, j\neq i}^{n} w_{i,j}(y_j - \bar{y})$$

Compute Moran's I Error (MIE)

$$MIE = \sum_{i=1}^{n} |(I(y_i) - I(\hat{y}_i))|$$

Pseudo-code - SpaceGAN

Algorithm 1 SpaceGAN Training and Selection

Require: $snap$, C, L: hyper-parameter

for number of training steps ($tsteps$) **do**

  Sample minibatch of L noise samples $\{\mathbf{z}_1, ..., \mathbf{z}_L\}$ from noise prior $p_{\mathbf{z}}(\mathbf{z})$

  Sample minibatch of L examples from $p_{data}\big((y_i, \mathbf{x}_i) \mid \mathcal{N}_i\big)$

  Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\Theta_D} \frac{1}{L} \sum_{i=1}^{L} \Big[\log D((y_i, \mathbf{x}_i) \mid \mathcal{N}_i) + \log(1 - D(G(\mathbf{z}_i \mid \mathcal{N}_i))) \Big]$$

  Sample minibatch of L noise samples $\{\mathbf{z}_1, ..., \mathbf{z}_L\}$ from noise prior $p_{\mathbf{z}}(\mathbf{z})$

  Update the generator by ascending its stochastic gradient:

$$\nabla_{\Theta_G} \frac{1}{L} \sum_{l=1}^{L} \Big[\log(D(G(\mathbf{z}_i \mid \mathcal{N}_i))) \Big]$$

Training

  if $tsteps \ \% \ snap$ **then**

    $G_k \leftarrow G$, $D_k \leftarrow D$, $MIE \leftarrow 0$ ▷ store current G, D as G_k, D_k; initiate MIE

    for C **do** ▷ draw C samples from G_k

      for $i \leftarrow 1, n$ **do** ▷ generate spatial data

        sample noise vector $\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})$

        draw $(\hat{y}_i, \hat{\mathbf{x}}_i) = G_k(\mathbf{z} \mid \mathcal{N}_i)$

      Measure *SpaceGAN* y samples spatial autocorrelation goodness-of-fit:

$$MIE \leftarrow MIE + \sum_{i=1}^{n} |(I(y_i) - I(\hat{y}_i))| \quad (4)$$

    Average of all samples: $MIE(G_k) = \frac{1}{C} MIE$

return $G := arg\min_{G_k} MIE(G_k)$, $D := arg\min_{G_k} MIE(G_k)$

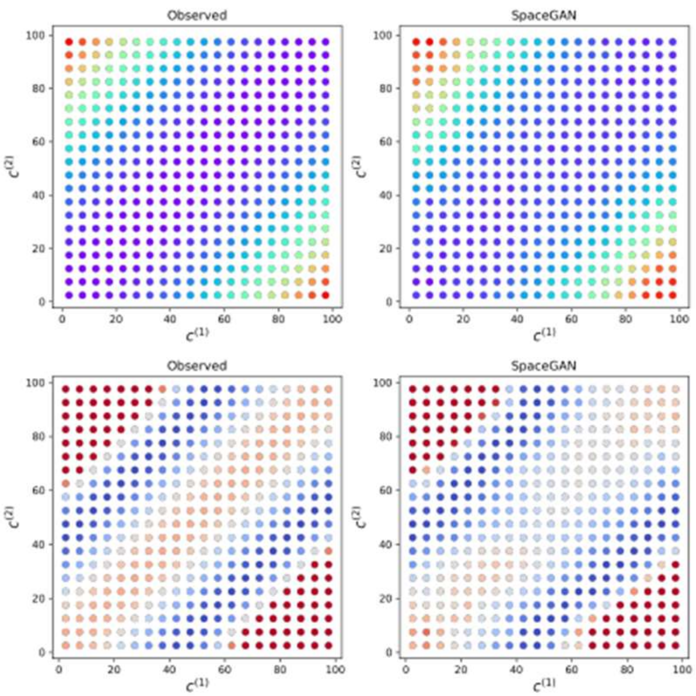
Selection

Does it work? Two experiments

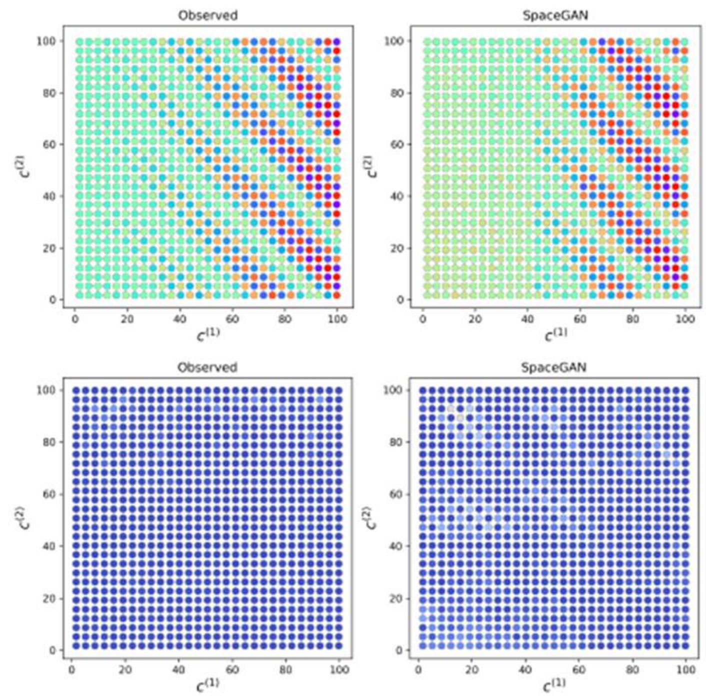
1. How does *SpaceGAN* learn spatial processes? How good is the out-of-sample extrapolation / interpolation?

2. Can *SpaceGAN* samples be used to train better predictive models?

Experiment 1

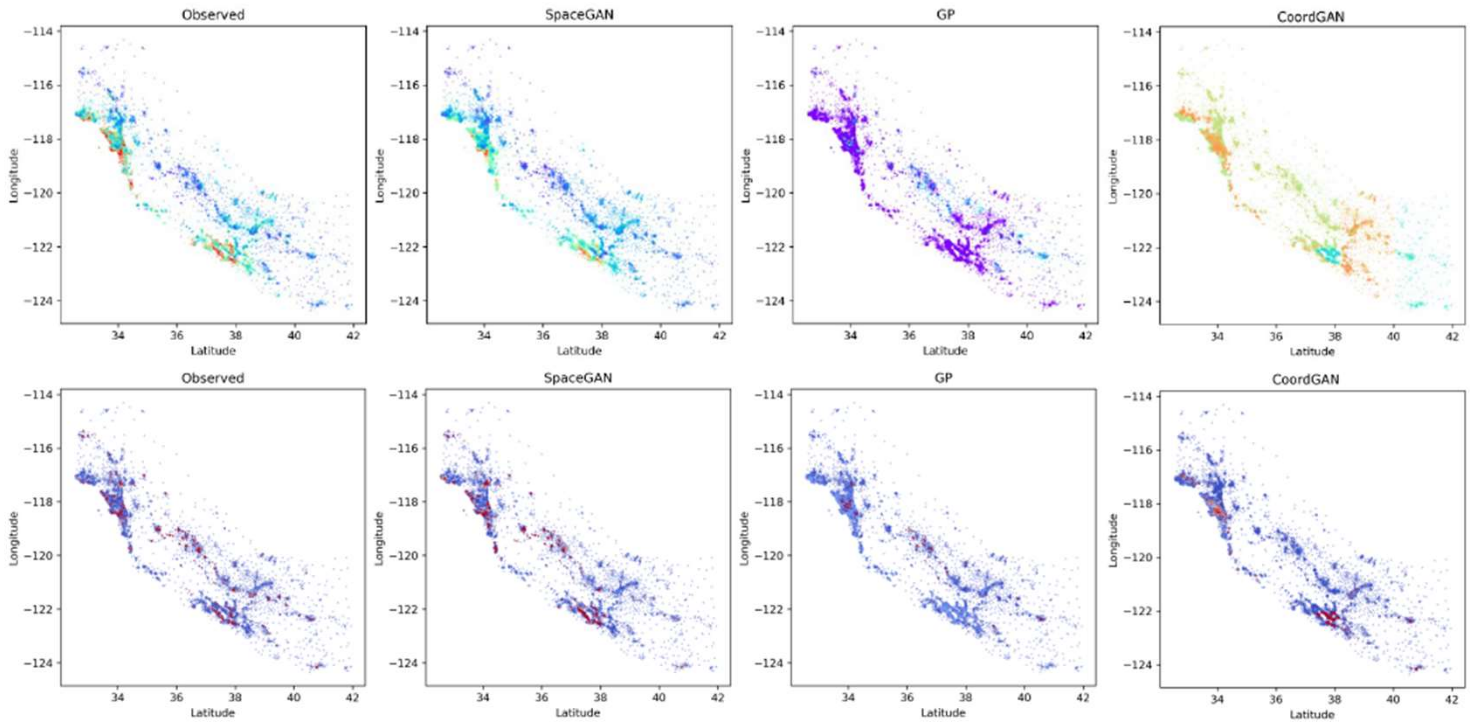


(a) Target vector y (*top*) and its Moran's I value $I(y)$ (*bottom*) of the observed and *SpaceGAN* generated data for **Toy 1**.



(b) Target vector y (*top*) and its Moran's I value $I(y)$ (*bottom*) of the observed and *SpaceGAN* generated data for **Toy 2**.

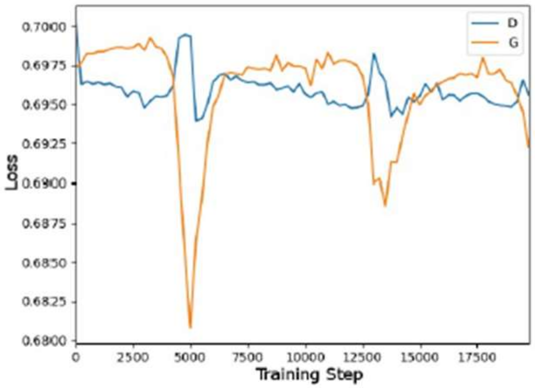
Experiment 1

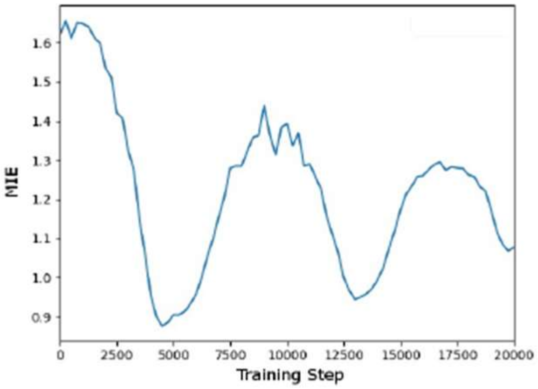


(c) Target vector y (*top*) and its Moran's I value $I(y)$ (*bottom*) of the observed data, *SpaceGAN*, Coord.GAN and GP generated samples for **California Housing 50**.

Experiment 1

- SpaceGAN outperforms competitors (GP, Coordinate GAN) at extrapolation / interpolation tasks
- The MIE works! It corresponds well to the generator loss; autocorrelation structures are captured well.
- Even more complex patterns are captured (at least partially).





Experiment 2

We use SpaceGAN-generated samples in an ensemble learning framework to generate training data!

We compare this to other methods of spatial data augmentation (spatial bootstrap, GP sampling, coordinate GAN augmentation)

Algorithm 2 "Ganning" for ensemble learning

Require: B (number of samples), M (base learner), G

for $b \leftarrow 1, B$ **do** ▷ generate spatial data
  for $i \leftarrow 1, n$ **do**
    sample noise vector $\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})$
    draw $(\hat{y}_i, \hat{\mathrm{x}}_i) = G_k(\mathbf{z} \mid \mathcal{N}_i)$
  train base learner: $M^{(b)}(\hat{\mathbf{y}}, \mathbf{x})$
return ensemble $M^{(1)}, ..., M^{(B)}$

Experiment 2

SpaceGAN (with MIE selection) outperforms competitors on almost all tasks!

The exact neighbourhood definition (size of the neighbourhood) seems to be important!

Table 2: *Experiment 2:* Prediction scores ($RMSE$) and their standard errors across 10 folds for different ensemble methods with 100 samples across the different prediction tasks.

Dataset	Model (B = 100)				
	SpaceGAN (MIE)	SpaceGAN (RMSE)	Coord.GAN (MIE)	GP	Spatial Boot
Toy 1	**0.9921**	1.1993	1.3067	1.2388	1.2013
	(0.0995)	(0.1494)	(0.1095)	(0.1490)	(0.1366)
Toy 2	**1.0097**	1.2065	1.3893	1.3135	1.2962
	(0.1092)	(0.1496)	(0.1015)	(0.1443)	(0.1413)
Cal.H. 15	**139534**	143983	200937	159340	148830
	(12026)	(10341)	(20743)	(8550)	(8660)
Cal.H. 50	**128756**	145612	171455	156814	148546
	(7463)	(7152)	(22195)	(8718)	(8611)
Inf.Mort.	7.2778	**7.2416**	7.9648	9.1793	7.3100
	(0.6518)	(0.6464)	(0.9567)	(0.5547)	(0.4685)
Election	0.1163	0.1162	0.1249	0.1330	**0.1156**
	(0.0035)	(0.0037)	(0.0072)	(0.0043)	(0.0039)

Currently ongoing & future research

- Use MIE for actual optimization (i.e. tweak the cGAN loss functions)
- Explore the importance of the neighbourhood definition and weights further
- Use SpaceGAN in more geospatial ML applications to confirm its usability
- Explore usability of the method for n-dimensional autocorrelation (e.g. spatiotemporal)
- ...and many more (we have 100000000 ideas)