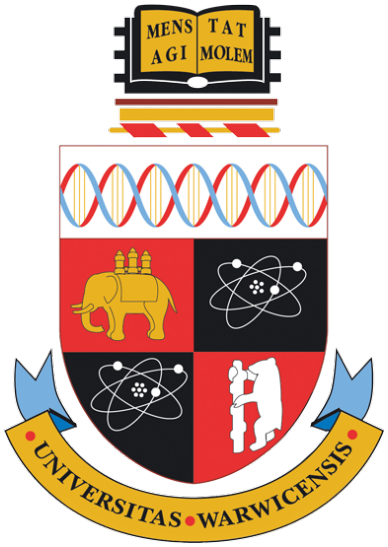
WARWICK
THE UNIVERSITY OF WARWICK



Improving Neural Networks for Geospatial Applications with Geographic Context Embeddings

by

Konstantin Klemmer

Thesis

Submitted to the University of Warwick

in partial fulfilment of the requirements

for admission to the degree of

Doctor of Philosophy

Department of Computer Science

January 2022

Contents

List of Tables

List of Figures

Acknowledgments

I want to express my deep gratitude to my supervisors, Dr Daniel Neill, Dr Hongkai Wen, and Prof Stephen Jarvis for their guidance throughout the four years of my doctoral studies. Their encouragement has pushed me to become a better researcher, and their patience and support have given me confidence in my abilities when I had doubts. I have also been fortunate to have had amazing and insightful advisors and mentors throughout my doctoral studies, including Prof Tobias Brandt, Prof Joaõ Porto de Albuquerque, Nyalleng Moorosi, Prof Xiaoxiang Zhu, Prof Theo Damoulas and Prof Beatrice Acciaio. I am immensely grateful for my friends Mike, Adriano, Sebastian, Svenja, Esmael, Fernando, Farzana, James and many more who have made the last few years exciting and taken my mind off work. I especially want to thank my partner Lauren for her kindness and patience, for long walks and discussions, for always challenging me to improve but never be too harsh on myself and for going on this adventure with me. Finally, I want to thank my family, my parents Henriette and Axel, my brother Leo, and my grandmother Jutta for their love and unconditional support for the path I chose.

Declarations

1 Publications

I hereby declare that this dissertation is original work and has not been submitted for a degree or diploma or other qualification at any other university. Parts of this dissertation have been previously published by the author in the following:

[95] Konstantin Klemmer, Adriano Koshiyama, and Sebastian Flennerhag. Augmenting correlation structures in spatial data using deep generative models. *arXiv:1905.09796*, 2019. URL `http://arxiv.org/abs/1905.09796`

[93] Konstantin Klemmer and Daniel B. Neill. Auxiliary-task learning for geographic data with autoregressive embeddings. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, 2021

[101] Konstantin Klemmer, Nathan Safir, and Daniel B Neill. Positional Encoder Graph Neural Networks for Geographic Data. *arXiv:2111.10144*, 2021. URL `https://arxiv.org/abs/2111.10144`

[101] Konstantin Klemmer, Tianlin Xu, Beatrice Acciaio, and Daniel B. Neill. SPATE-GAN: Improved Generative Modeling of Dynamic Spatio-Temporal Patterns with an Autoregressive Embedding Loss. In *AAAI 2022 - 36th AAAI Conference on Artificial Intelligence*, 2022 [1]

[1] Konstantin Klemmer and Tianlin Xu are joint first-authors of this article. Konstantin Klemmer's contributions focused on the development of the spatio-temporal autocorrelation metric (SPATE), while Tianlin Xu's contributions focused on the integration of the SPATE metric into the existing COT-GAN framework. Both wrote the manuscript and worked on the experiments.

Chapters 3, 4, 5, and 6 are based on these publications respectively and remain mostly unchanged. Changes adapted are of an editorial nature, in order to fit the individual papers as chapters into the dissertation.
Research was performed in collaboration during the development of this dissertation, but does not form part of the dissertation:

[94] Konstantin Klemmer, Tobias Brandt, and Stephen Jarvis. Isolating the effect of cycling on local business environments in London. *PLOS ONE*, 13 (12):e0209090, dec 2018. ISSN 1932-6203. doi: 10.1371/journal.pone.0209090. URL http://dx.plos.org/10.1371/journal.pone.0209090

[131] Fernando Munoz-Mendez, Ke Han, Konstantin Klemmer, and Stephen Jarvis. Community Structures, Interactions and Dynamics in London's Bicycle Sharing Network. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers - UbiComp '18*, pages 1015–1023, New York, New York, USA, 2018. ACM Press. ISBN 9781450359665. doi: 10.1145/3267305.3274156. URL http://dl.acm.org/citation.cfm?doid=3267305.3274156

[119] Man Luo, Bowen Du, Konstantin Klemmer, Hongming Zhu, Hakan Ferhatosmanoglu, and Hongkai Wen. D3P: Data-driven Demand Prediction for Fast Expanding Electric Vehicle Sharing Systems. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(1):1–21, mar 2020. ISSN 24749567. doi: 10.1145/3381005. URL https://dl.acm.org/doi/10.1145/3381005

[96] Konstantin Klemmer, Godwin Yeboah, João Porto de Albuquerque, and Stephen A Jarvis. Population Mapping in Informal Settlements with High-Resolution Satellite Imagery and Equitable Ground-Truth. In *ML-IRL Workshop, ICLR'20*, sep 2020. URL http://arxiv.org/abs/2009.08410

[100] Konstantin Klemmer, Sudipan Saha, Matthias Kahl, Tianlin Xu, and Xiao Xiang Zhu. Generative modeling of spatio-temporal weather patterns with extreme event conditioning. *ICLR'21 Workshop AI: Modeling Oceans and Climate Change (AIMOCC)*, apr 2021. URL http://arxiv.org/

abs/2104.12469

[97] Konstantin Klemmer, Daniel B. Neill, and Stephen A. Jarvis. Understanding spatial patterns in rape reporting delays. *Royal Society Open Science*, 8(2), feb 2021. ISSN 20545703. doi: 10.1098/rsos.201795. URL https://doi.org/10.1098/rsos.201795

2 Sponsorships and Grants

This research was funded by the UK Engineering and Physical Sciences Research Council, the EPSRC Centre for Doctoral Training in Urban Science (EPSRC grant no. EP/L016400/1); The Alan Turing Institute (EPSRC grant no. EP/N510129/1).

Abstract

Geospatial data sits at the core of many data-driven application domains, from urban analytics to spatial epidemiology and climate science. Over recent years, ever-growing streams of data have allowed us to quantify more and more aspects of our lives and to deploy machine learning techniques to improve public and private services. But while modern neural network methods offer a flexible and scalable toolkit for high-dimensional data analysis, they can struggle with the complexities and dependencies of real-world geographic data. The particular challenges of geographic data are the subject of the geographic information sciences (GIS). This discipline has compiled a myriad of metrics and measures to quantify spatial effects and to improve modeling in the presence of spatial dependencies. In this dissertation, we deploy metrics of spatial interactions as embeddings to enrich neural network methods for geographic data. We utilize both, *functional embeddings* (such as measures of spatial autocorrelation) and *parametric neural-network embeddings* (such as semantic vector embeddings). The embeddings are then integrated into neural network methods using four different approaches: (1) model selection, (2) auxiliary task learning, (3) feature learning, and (4) embedding loss functions. Throughout the dissertation, we use experiments with various real-world datasets to highlight performance improvements of our geographically-explicit neural network methods over naive baselines. We focus specifically on generative and predictive modeling tasks. The dissertation highlights how geographic domain-expertise together with powerful neural network backbones can provide tailored, scalable modeling solutions for the era of real-time Earth observation and urban analytics.

Acronyms

AC-GAN Auxiliary-Classifier GAN.

BCE Binary Cross Entropy.

BicInt Bicubic Interpolation.

cGAN Conditional Generative Adversarial Network.

CNN Convolutional Neural Network.

COT Causal Optimal Transport.

COT-GAN Causal Optimal Transport GAN.

DCGAN Deep-Convolutional GAN.

DEM Digital Elevation Map.

EDGAN Encoder-Decoder GAN.

EMD Earth Mover Distance.

EV Electric Vehicle.

GAN Generative Adversarial Network.

GAT Graph Attention Network.

GCN Graph Convolutional Network.

GIS Geographic Information Sciences.

GNN Graph Neural Network.

GP Gaussian Process.

GWR Geographically Weighted Regression.

IDW Inverse Distance Weighting.

iid Identically and Independently Distributed.

IoT Internet of Things.

KCN Kriging Convolutional Network.

kNN k-Nearest-Neighbor (also k-Nearest-Neighbor Classifier).

LGCP Log-Gaussian Cox Process.

LOSH Local Spatial Heteroskedasticity.

LSD Local Spatial Dispersion.

LSTM Long Short-Term Memory.

MAE Mean Absolute Error.

MAUP Modifiable Areal Unit Problem.

MIE Moran's I Error.

MLP Multi-Layer Perceptron.

MMD Maximum Mean Discrepancy.

MSE Mean Squared Error.

NLP Natural Language Processing.

NN Neural Network.

OK Ordinary Kriging.

OLS Ordinary Least Squares.

OT Optimal Transport.

PCA Principal Component Analysis.

PDE Partial Differential Equation.

PE Positional Encoder.

POI Point Of Interest.

RBF Radial Basis Function.

ReLU Rectified Linear Unit.

RMSE Residual Mean Squared Error.

SAR Spatial Autoregressive Model.

SGD Stochastic Gradient Descent.

sinkGAN Sinkhorn GAN.

SPATE Spatio-Temporal Association.

SXL Spatially-Explicit Learning.

UK Universal Kriging.

UN United Nations.

Symbols

$w_{i,j}$	The spatial weight matrix indicating neighborhood between i and j.
$E(\cdot)$	The expected value of some variable.
y	The outcome variable (scalar).
$\mathbf{c}$	A spatial context vector, i.e. point coordinates.
$\mathbf{x}$	A vector of features.
$\bar{y}$	The mean of y.
$\hat{y}$	The predicted value of y.
$I(\cdot)$	The Moran's I metric.
$\|\| \cdot \|\|$	The Euclidean distance.
β	A regression coefficient.
λ	A weight parameter, i.e. a loss weight.
ϵ	The model error term.
$m(\cdot)$	A mean function.
$k(\cdot)$	A kernel function.
l	A lengthscale parameter.
$g(\cdot)$	An activation function.
$\|\| \cdot \|\|_2^2$	The L2 norm.
$\mathcal{L}$	A loss function.
$\mathbf{z}$	A random noise vector.
G	A generator (part of a GAN). Also, a graph.
D	A discriminator (part of a GAN).
Θ	Parameters of a neural network.
Λ	Hyperparameters.
$\mathbf{v}$	A context vector (for cGANs).
MIE	The Moran's I Error.
σ	A measure of deviance or uncertainty.
V	The vertices or nodes of a graph.
E	The edges of a graph.
d_{ij}	The distance between two observations i and j.
$\mathbf{A}$	The adjacency matrix.

$\mathbf{D}$	The degree matrix.
$\mathbf{I}$	The identity matrix.
$\mathbf{H}$	Inputs and outputs of neural network layers.
$NN(\cdot)$	A neural network.
$ST(\cdot)$	A spatial transform function.
$PE(\cdot)$	A positional encoder.
$b(\cdot)$	A temporal weighting function.
$S(\cdot)$	The SPATE metric.
α	The learning rate.

"Not all those who wander are lost."

Bilbo Baggins

Chapter 1

Introduction

1.1 Motivation

1.1.1 Big Geospatial Data & Urban Analytics

"80% of all data is geographic". This sentence has been heard by many who work with geographic data and has even made its way into corporate communications of industry giants such as Esri[1]. The quote has taken an almost mythical form, with vague original attributions and no solid evidence to back it up [26]. Nevertheless, it is undeniable that a large share of the world's data can, in some shape or form, be geo-referenced; mapped onto the globe of planet Earth. Sometimes, like with the fleet of satellites monitoring our planet in near real-time, the geographic nature of the data is essential to its purpose, for example in military surveillance or flood early-warning systems. Sometimes, the geographic connection is hidden in the meta-data, for example in contactless payments or the daily interactions on our social media accounts, and only becomes relevant if we look at the aggregates.

The density, availability and pervasiveness of geographic data is particularly evident in cities [15, 91]. Public authorities deploy sensor and camera networks to measure air quality and traffic volumes, while advances in mobile computing and networking transform our phones and cars into Internet of Things (IoT) devices [150]. The data collected throughout our cities, on personal and public sensors, represent the different layers of our urban systems: This includes physical attributes like infrastructure and land use, but also non-physical characteristics like socio-economic factors or (perceived) safety. Data allows us to view cities as organisms: aggregates of their inhabitants, physical and non-physical features that, altogether, make them more than the sum of their parts [30].

Authorities around the globe are increasingly building sensing infrastructure

[1] See e.g. `https://www.esri.com/news/arcuser/0401/bunchls.html`

and make urban data available via larger open data platforms that harmonise new, IoT-driven data sources with traditional sources such as census data [14]. Geospatial information is often used for the cross-referencing step. In fact, urban open data platforms often choose to represent data as large, multi-layered city maps [83]. The increasing availability of urban big data has lead to the emergence of scientific sub-disciplines dedicated to collecting and analysing this data, as well as building end-to-end, data-driven decision support systems based on it: urban analytics and urban informatics [17]. Under this umbrella term, researchers from the computational sciences, engineering, economics, geography and many more conduct inter- and cross-disciplinary, quantitative research on cities.

This perspective is crucially needed. According to the latest United Nations (UN) report, over 55% of the global population already lives in urban areas [175]. This number rises to 75+% when looking solely at Western countries. The UN estimates that almost 70% of the world's population will live in cities in 2050. Many cities around the globe already struggle to accommodate the steady influx of people from rural areas, who seek opportunities in the urban centers. The increasing densification of urban areas, also known as urbanisation, can lead to problems ranging from public health concerns [102] to ecological crises [200] and poses critical challenges for urban authorities.

Urban analytics offers tools for data-driven knowledge discovery as well as a framework for operationalising these insights in downstream applications. Through the design of data-centric systems, we can improve the quality and efficiency of urban infrastructure and services and tackle many of the issues that arise from increasing urbanisation. In the context of urban analytics, data-centric approaches require a particular focus on the spatial nature of the data. Generally, spatial dependencies are omnipresent in urban datasets and require dedicated methodological approaches.

For example, the shared-vehicle systems that are prevalent in many cities, from bicycles to electric cars, exhibit stark demand imbalances over space and time [92, 131]. Devising systems for planning support or relocation schemes require a thorough understanding of the urban spatial structure and accounting for contextual information that characterises specific locations [119, 190]. Spatial dependencies can also complicate the discovery of causal effects, such as challenges in linking public transportation infrastructure to economic outcomes [94]. Lastly, observed spatial effects can also arise from true, non-spatial underlying effects. For example, while the reporting delays of urban rape crime appear to be spatially autocorrelated at first glance, a deeper investigation reveals this effect to stem from spatially clustered socio-economic and demographic factors [97].

Oftentimes, spatial effects and contextual factors exhibit their own intricate

dependencies. There is a need for dedicated spatial modelling techniques to scale to the dimensions of large geospatial data and support decision making in urban settings. In this dissertation, we will develop methods that are tailored to the intricacies of geospatial data using a powerful family of scalable models: artificial neural networks.

1.1.2 Machine Learning for the Geospatial Domain

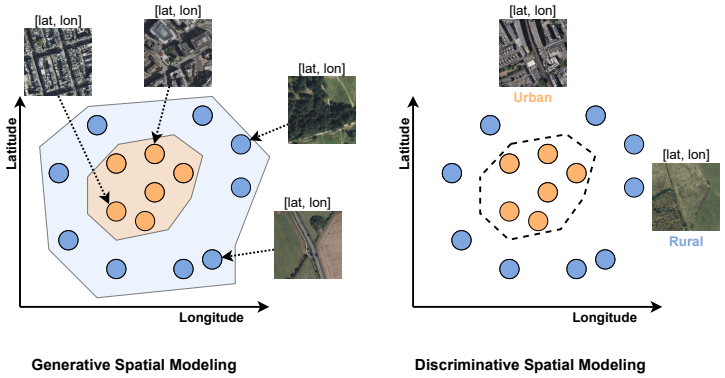


Figure 1.1: Examples of generative and discriminative spatial modelling tasks. *Left:* The objective of the generative modelling task is to generate images of urban and rural settings, conditional on the respective point location. *Right:* The objective of the discriminative modelling task is to classify an image at a given point location into the "urban" and "rural" categories.

Geography has a long tradition of modelling and empirical analysis. Ideas from the Geographic Information Sciences (GIS) and spatial statistics have sparked popular methodologies in modern machine learning, from Gaussian Processes (GPs), which were pioneered by the development of Kriging in the 1960s [123], to spatial scan statistics[105], which are currently widely used for event and pattern detection. Since the emergence of the era of deep neural networks, the relationship between the GIS and machine learning communities has been largely defined through applications of existing neural network models to geographic data. Rarely have concepts and ideas from the GIS and spatial statistics motivated methodological advancements in neural networks, whereas application areas such as computer vision, bioinformatics, and computational linguistics have strongly influenced the deep learning state-of-the-art we know today.

Some recent advances in machine learning have been exceptionally useful to the GIS community. Kernel methods such as Gaussian Processes have seen huge progress towards overcoming computational bottlenecks that stem from the complexities of working with pair-wise distance matrices. Kernel interpolation techniques and GPU acceleration have allowed GPs to easily scale

up to a million data points [55, 184]. In neural networks, the emergence of graph-based networks and particularly graph convolutions have allowed for the modelling of asymmetric and non-Euclidean spatial relationships [112], and the emergence of physics-informed deep learning has reinforced the need for neural networks to model complex spatio-temporal patterns [187].

Yet, a common observation when applying off-the-shelf neural network methods in geospatial contexts is their inadequacy for capturing spatial dependencies. For example, while neural network methods like convolutional neural networks (CNNs) enable the modelling of localized spatial effects through convolutions [69], they appear to struggle with long-range spatial dependencies [116]. Let us think of an example: In a traditional computer vision task, like the detection of a person in an image, long-distance relationships are often not very important. Instead, we aim to detect the persons outline by identifying edges–stark differences between very close pixels. In distinctly geospatial settings, like the (early) detection of a hurricane, such short-distance effects might not be sufficient. Rather, weather phenomena further away may affect the formation of a new hurricane. As such, we would require models that are able to capture both short-distance effects, long-distance effects and their interactions. This effect, which we may describe as scale-sensitivity, is only one of the distinct characteristics of geographic data. Concerns about neural networks–and deep learning specifically–in the geospatial domain have been summarised in a recent review study by Reichenstein et al. [148]. The authors highlight several challenges (which are listed below) of deep learning applications with spatial data and call for future research to improve the representation of spatial structures within these methods. While the authors focus particularly on geospatial data in the context of Earth observation and modelling, their insights are applicable to all geographic domains, from urban analytics to spatial epidemiology.

This dissertation is concerned with two types of modelling problems: *generative modelling* and *discriminative modelling* (or predictive modelling). In the broadest sense, generative models seek to learn how data is distributed in a given space, while discriminative models seek to learn the boundaries that separate data points. The difference between both paradigms in a geospatial setting is highlighted in Figure 1.1. In this example, a generative model tries to generate images that represent *rural* and *urban* settings respectively, based on a given set of geographic coordinates–the model thus has to learn to emulate the data generating process. A discriminative model on the other hand might seek to predict *rural* and *urban* setting from a given image and its geographic coordinates–the model thus has to learn a boundary between the two predetermined classes. While different in nature, both paradigms suffer from the same challenges when applied in the geospatial domain. According to the

aforementioned study by Reichenstein et al. [148], these can be summarised as follows:

- **Model interpretability:** Neural networks are black-box models and insights on how they learn and make predictions are limited. Because of their black-box nature, it is also difficult to identify causal links using neural network models. These issues are amplified in geospatial domains, in which the underlying data often exhibit spatial dependencies, which further complicate causal inference. Overcoming this challenge requires dedicated approaches, including traceable assumptions into neural networks to improve interpretability.

- **Physical constraints:** Geospatial data are often governed by underlying laws governing the system (e.g. the laws of thermodynamics), which make certain outputs implausible. Neural networks have no clear intuition for integrating these rules, which can lead to physically inconsistent outputs. For example, if we train a generative model to generate synthetic taxi drop-off locations in London, we have to integrate knowledge on the physical structure of the city into the model to prevent synthetic drop-offs (e.g. in rivers). Integrating physical constraints and geospatial dependencies into models is crucial to making them better reflect the real world.

- **Spatial complexity and scale sensitivity:** Geospatial data often exhibit dependencies like spatial autocorrelation, change patterns or dispersion. If data are also available over different time steps, complex spatio-temporal patterns can be observed. These complexities may manifest differently across different spatial scales. Neural networks are not well-equipped to handle the scale-sensitivities inherent in geospatial data. Dedicated approaches must include mechanisms to account for effects at different spatial scales.

- **Label scarcity:** For deep neural networks to learn accurate models, they require extensive amounts of training data. In many real-world applications, labels can be sparse, e.g. due to the high cost of obtaining them or privacy constraints. It is crucial to develop neural network models that can also work in scarce data environments. Spatial data augmentation and regularisation techniques can help to prevent models from overfitting due to a lack of training data.

- **Computational requirements:** Geospatial datasets can be large, which can lead to a high computational demand for processing them. They can also come in (near) real-time, which further adds complexity. Developing

neural network models that are efficient and flexible is a key challenge in geospatial machine learning. This challenge is shared with other application disciplines processing vast amounts of data, e.g. computer vision or bio-informatics.

These and other challenges have led to the emergence of "GeoAI" as a dedicated research field [78]. "GeoAI" lies at the intersection of complex methodological challenges and high-impact applications. To tackle the challenges outlined above, we must combine state-of-the art neural network modelling with domain expertise from the GIS. This is the objective of this dissertation: specifically, we will propose novel, intuitive methodological approaches for integrating GIS knowledge in the form of geographic context embeddings into neural networks for generative and discriminative modelling tasks. We understand the term "geographic context embedding" broadly as any quantitative measures of geographic context, from measures of spatial autocorrelation to vector representations. These embeddings represent domain expertise and prior information we have about the data at hand and, as such, can be beneficial to any modeling endeavour if integrated appropriately.

1.2 Contributions

The main contributions of this dissertation lie at the intersection of GIS and machine learning research. The main objective of this dissertation is to use GIS domain expertise in the form of geographic context embeddings within existing neural network to improve their performance in explicitly geospatial settings. To this end, we propose four different strategies of incorporating functional and neural-network based embeddings in neural network models: (1) model selection, (2) auxiliary task learning, (3) feature learning, and (4) embedding loss functions. Throughout a broad set of experiments, we highlight the improvements that our novel methods provide over naive, non-geographically-explicit baselines.

The specific contributions of the different technical chapters of this dissertation are as follows:

- A novel method for model selection based on the local Moran's I metric with applications to generative adversarial network (GAN) ensemble learning. **(Chapter 3)**
  - *A conditional GAN for geospatial data using spatial neighbourhood context.* We propose a novel conditional GAN that learns a conditional data distribution, depending on information about its spatial neighbourhood. This approach allows the model to capture spatial

dependencies in the data. Customizing the neighborhood definition allows conditioning at different spatial scales, one of the key challenges mentioned in section 1.1.2.

  - *A spatially-explicit selection process for generators obtained throughout a GAN training process.* We propose to use the local Moran's I measure of spatial autocorrelation to evaluate the quality of our GAN. This allows us to select the generator that is best able to reproduce observed spatial structures in new synthetic data samples.
  - *An ensemble learning method for geospatial predictive modelling tasks using GAN-generated training data.* We propose the use of synthetic data from our neighbourhood-conditioned GANs to train a set of base learners from which we can build an ensemble model. Such approaches can also help in scarce data environments, thus addressing one of the key challenges identified in section 1.1.2.

- A novel method for auxiliary task learning for neural network models based on the local Moran's I metric with applications to predictive and generative modelling. **(Chapter 4)**

  - *A multi-resolution extension of the local Moran's I metric of spatial autocorrelation.* We propose an extension of the local Moran's I to represent spatial dependencies at different spatial resolutions. This is achieved through computing the metric for inputs that are repeatedly downsampled by neighbourhood averaging. This approach again addresses the key challenge of scale-sensitivity raised in section 1.1.2.
  - *Auxiliary task learning of single- and multi-resolution local spatial autocorrelation.* We propose to use the local Moran's I and our multi-resolution extension as auxiliary tasks alongside the primary task in multi-task neural network models. This reinforces the learning of short- and long-scale spatial structures in the model.
  - *Spatially-explicit auxiliary task learning for generative and predictive neural network models.* We devise a flexible framework for integrating our auxiliary learning approach into neural networks for spatial interpolation/super-resolution and into GANs for generative modelling.
  - *Uncertainty-weighted auxiliary task GANs.* To balance the loss weights of the different tasks in our GANs, we propose the use of task-specific uncertainties. Building on work by Cipolla et al. [32], this allows for parameter-free learning of loss weights.

- A novel method for improving graph neural networks with geographic context embeddings and feature learning. **(Chapter 5)**

  - *A positional-encoder graph neural network (GNN) approach allowing for explicit learning of geospatial context.* We propose a novel GNN method including a positional encoder network that learns point coordinate embeddings based on spatial context.
  - *A novel training approach for the positional encoder.* The positional encoder is trained through backpropagation on the main GNN loss. This is facilitated by concatenating the output of the positional encoder with other node features, before feeding them through the GNN layers. This improves the models capacity for learning spatial complexities and local dependencies in the data, a key challenge outlined in 1.1.2.
  - *Training with random neighbourhoods and shuffled Moran's I auxiliary task.* We propose to train the positional-encoder GNN by sampling a subset of random point coordinates at each training step to conduct the training graph. This implies that the same point might have different neighbours at different training steps and leads to the learning of more generalisable positional encoders. This also has implications for the Moran's I auxiliary task, proposed in Chapter 4, which we use here as well. Different neighbourhoods at different training steps also lead to different Moran's I values for the same location, thus "shuffling" the metric based on random sampling. As for the multi-resolution Moran's I, this helps the learning of spatial dependencies at different scales, addressing a key challenge outlined in 1.1.2.

- A novel method for improving GANs via an embedding loss based on a novel metric of spatio-temporal autocorrelation. **(Chapter 6)**
  - *SPATE: An expansion of the Moran's I metric to the spatio-temporal domain.* We propose SPATE (spatio-temporal association), a measure of spatio-temporal autocorrelation building on the local Moran's I. We propose three approaches to calculate spatio-temporal expectations, with and without temporal weights and sequential logic. These spatio-temporal expectations are used to compute deviances between observations and their expected values, building the core of the metric.
  - *Deploying SPATE as embedding loss for video GANs.* We propose to use our SPATE metric as an embedding loss for training video GANs to improve the learning of spatio-temporal dependencies. We show how this approach fits in nicely with recent approaches to training GANs with causal optimal transport [194]. Our embedding loss can

be seen as enforcing some degree of spatio-temporal coherence in the model, thus relating to the key challenges regarding physical constraints and spatio-temporal complexity outlined in 1.1.2.

1.3 Dissertation Outline

The remainder of this dissertation is structured as follows:

- **Chapter 2** highlights relevant literature in GIS and machine learning, builds the contextual and technical background for the methodological chapters, and reviews the state of the art in machine learning for geographic data. It also presents the datasets that are used in experiments throughout the dissertation.
- **Chapter 3** proposes a GAN-based ensemble learning approach for geospatial prediction tasks. This includes a new GAN that is conditioned on spatial neighbourhood context, as well as a novel generator-selection mechanism that is based on the local Moran's I metric. This enforces the learning of spatial structures during GAN training, while assuring that spatial dependencies in the generated data are faithful to those observed in the real data.
- **Chapter 4** proposes an auxiliary task learning approach for generative and predictive modelling tasks. In this chapter, we propose a multi-resolution extension to the Moran's I measure of spatial autocorrelation to account for dependencies at different spatial scales. The Moran's I and its extensions are used as auxiliary tasks in multi-task neural network models. This approach more explicitly incorporates spatial autocorrelation into the learning process. We use task-specific uncertainties to balance the loss weights.
- **Chapter 5** proposes a new GNN approach for geospatial prediction problems. Node features are enriched with point-coordinate embeddings from a positional encoder. These embeddings can learn geographic context throughout training and provide the GNN model with an improved capacity for representing complex spatial dependencies.
- **Chapter 6** proposes SPATE, a spatio-temporal extension of the local Moran's I metric. SPATE is used to devise an embedding loss function for video GANs, explicitly incorporating the learning of spatio-temporal dependencies into the training process. The embedding loss also deploys causal optimal transport to reinforce the sequential nature of the data.

- **Chapter 7** discusses our methodological contributions with respect to existing literature and impactfull applications. We conclude the dissertation by outlining promising directions for future research.

Chapter 2

Background

In this chapter, we review the literature underpinning the main technical contributions of this dissertation. The main objective of this dissertation is to merge ideas from the GIS and machine learning communities in order to improve neural network methods for geospatial data. Accordingly, here we introduce relevant concepts from both domains: Section 2.1 introduces the relevant GIS background, focusing on metrics to capture spatial and spatio-temporal effects and traditional modelling approaches for spatial data. Section 2.2 introduces relevant machine learning concepts, focusing on predictive modelling with neural networks and generative modelling with GANs. This section also surveys state-of-the art approaches for modelling geographic data with neural networks. Lastly, it presents different approaches for integrating domain expertise into neural networks.

Section 2.3 summarises the background and contextualises the dissertation within the body of existing work.

2.1 Quantifying Spatial Effects in Geographic Data

In this section, we will introduce the history of, and important concepts from, spatial analysis and GIS. We will first build an understanding of different spatial data and neighbourhoods. We will then discuss the challenges arising in spatial modelling. The main content of this section focuses on (1) measures and metrics that capture different spatial effects (e.g. point process intensities and measures of spatial autocorrelation) and (2) modelling approaches that are tailored to spatial data (e.g. geographically weighted regression or Kriging).

2.1.1 Discrete and Continuous Geospatial Data

Geospatial data can broadly be separated into two categories: discrete and continuous geospatial data [117]. Discrete geospatial data are objects and have clearly defined, fixed locations and boundaries. This includes, for example,

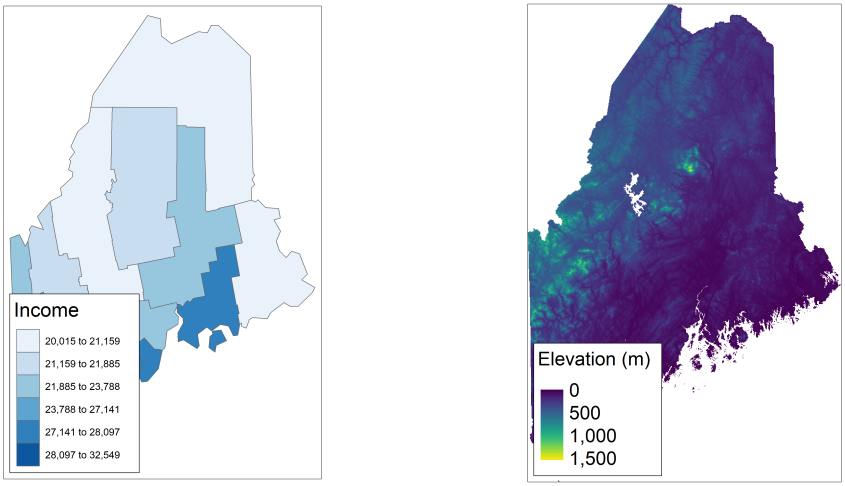


Figure 2.1: Examples of discrete and continuous spatial data. The left plot shows median incomes for the different counties in the state of Maine, USA. These counties are discrete spatial units and can be mapped as polygons. The right plot displays the elevation across the sate of Maine; a measure that can be taken at any given point location, thus representing continuous data.

census tracts, roads or the cells in a rectangular grid. Discrete geospatial data can come in the form of polygons, lines or any other objects in a predefined, discrete space. Continuous geospatial data, on the other hand, does not have clearly defined locations or boundaries. Rather, continuous geospatial data float freely in a continuous geo-space, that, for the purpose of this dissertation, is limited only by the boundaries of planet Earth. Continuous geospatial data includes surface temperatures, elevations and any other data that may be measured at any given point within our continuous space. Figure 2.1 highlights two examples for discrete and continuous spatial data.

Spatial information allows us to define whether two spatial locations are neighbours or not. Spatial neighbourhoods can be defined according to the needs of the application, but usually follow some common intuitions. For example, let us revisit the case of discrete spatial data, as e.g. the different counties in Maine shown in Figure 2.1. A given county usually borders other counties in the same state, unless it is geographically isolated (e.g. an island) or borders only non-applicable spatial objects (e.g. counties in another state). We can intuitively define the spatial neighbourhood of a county as all the counties it shares a border with. We could expand this intuition to include second-degree neighbours (neighbours of neighbours) and so forth. For continuous spatial locations, there is no straightforward definition of adjacency. Rather, for a given spatial point, we might want to define its spatial neighbourhood as all points within a certain radius around the location. Another common

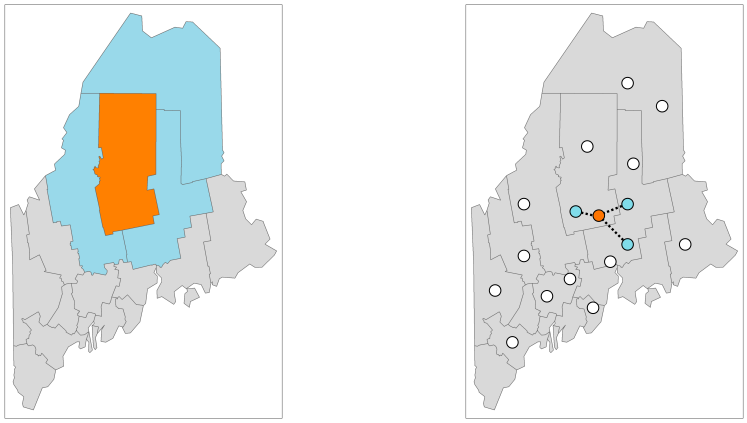


Figure 2.2: Examples of neighbourhood definitions for discrete and continuous spatial data. On the left, the orange county is neighboured by the blue counties. The neighbourhood is defined through adjacency (i.e. the orange county sharing a border with the blue counties). On the right side, the orange point location is neighboured by the blue locations. Here, neighbourhood is defined as the three closest point as measured by Euclidean distance.

option is to define the k nearest points to a location as neighbours, thus ensuring that every point has the same number of neighbours. This approach requires a measure of distance between points. One of the simplest approaches here is to use Euclidean distance–the "straight-line" distance between points. Nonetheless, many geospatial settings require distance measures that are more realistic and representative of underlying structures. For example, we can use the Haversine or "great-circle" distance to compute distances between pairs of langitude-longitude coordinates on an approximately spherical body, such as planet Earth. Other applications, such as traffic modeling, might require even more sophisticated distance measures, such as road distances. Figure 2.2 provides an example of discrete and continuous spatial neighbourhoods.

Spatial neighbourhoods are important for many applications in spatial analysis, as they allow for an explicit integration of assumptions on spatial dependencies in our models. As we show in the following sections, we might be interested in measuring the difference between on observation and its neighbours, or the difference between smaller and bigger neighbourhoods. The intuition for spatial neighbourhoods lies at the core of geographical analysis.

2.1.2 The Challenges of Spatial Data

With the definitions of spatial data, neighbourhoods, and distances set, we can now move to outlining the particular challenges that arise when working

with spatial data. We surmise these challenges from leading textbooks and perspective articles, namely Longley et al. [117], Bivand et al. [20], Goodchild [63] and following most closely the definitions given by Stewart Fotheringham and Rogerson [163]:

- **Location challenges** can arise from ill-defined or incomplete spatial information, including measures that cannot be assigned to spatial locations easily. For example, the spatial locations of all occasions of robbery can easily be mapped. However, if we try to map a different category of crime, for example corruption, this is not as easy. Does corruption occur at the home of the criminal individual? Does it occur in their office? In spatial modelling, the spatial references we include often reflect assumptions that may simplify but also distort our models. For example, when modelling individuals' income distributions, we will probably assume their home as spatial reference, rather than the location of their office or the home of their parents, even though both of these locations might be relevant to our question.

- **Scaling and aggregation challenges** can arise when we, during data collection or modelling, make assumptions on the spatial scale of a problem. Often, we are restricted by the available data. For example, while we have access to median income data for a given spatial unit (e.g. census tract), we lack access to the income of individuals who are contained within the spatial unit, to protect their privacy. Knowledge about the spatial unit does not allow us to draw conclusions regarding the individuals within that unit. This problem is also known as the "ecological fallacy" [152]. On the other hand, individual observations, when aggregated, are not necessarily a meaningful descriptor of the aggregation unit. Rather, aggregate measures are heavily affected by the shape and scale of the chosen areal unit and–depending on their definitions–can lead to information loss [135]. This issue is termed the modifiable areal unit problem (MAUP) [56, 134].

- **Dependence challenges** can arise when the values of a variable at two different locations are not independent. Tobler's first law of geography famously states that "everything is related to everything else, but near things are more related than distant things". [171] For example, the distribution of surface temperatures measured at different locations exhibits clear spatial dependence: I.e., two measures taken at stations that are nearer to each other are more likely to be similar than two measures taken at those that are more distant. Dependencies can be multi-variate and extend over more than just spatial dimensions (e.g.

exhibiting spatio-temporal dependencies).

- **Sampling challenges** stem from the aforementioned dependence of spatial data. For example, sampling data from locations nearer to each other might be unnecessary, as the values might be very similar. Nevertheless, spatial effects might not be consistent and occur only locally, which could prevent a clear rule to enable efficient, unbiased sampling. As such, spatial data often require dedicated sampling schemes [183].

- **Computational challenges** can arise when working with large spatial datasets. Spatial analysis techniques often lack linear-scaling of compute and memory requirements, as they rely on repeated up- and down-sampling, permutation testing or the creation of pairwise distance matrices. Recently, advances in distributed computing and scalable modelling techniques such as deep neural networks have highlighted promising avenues for overcoming such computational bottlenecks [196].

Having outlined the semantics of spatial data and the challenges arising from many of its use cases, we now move onto the different methods for analysing and modelling spatial data.

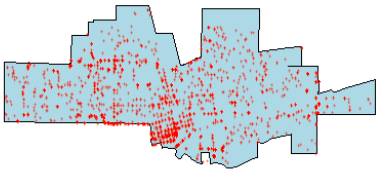
2.1.3 Analysing Spatial Point Patterns

One of the most common forms of spatial data are spatial point patterns. These are collections of events (e.g. a case of a disease) at a given spatial location. The analysis of spatial point patterns is one of the foundations of quantitative geographic analysis.

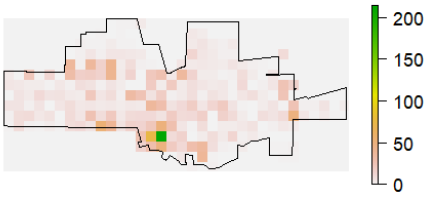
Events, Intensities and Spatial Dependencies

Spatial point patterns are collections of *events* that are registered at specific spatial locations–and potentially within spatial units that comprise the area of interest. Spatial point patterns are a common type of data in many academic disciplines. For example, in ecology, the locations of a species can be viewed as spatial point patterns. In epidemiology, spatial point patterns could manifest as the locations of patients with a specific disease. In general, a spatial point pattern can be any "stochastic mechanism which generates a countable set of events" [41] within an area of interest.

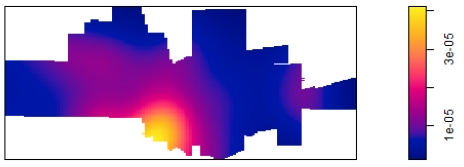
The analysis of spatial point patterns mostly revolves around identifying and describing each pattern's underlying distribution (i.e. the point-generating process) [20]. Of further interest are potential interactions between events, such as whether the occurrence of an event increases the likelihood of future events in the vicinity (in the case of a self-exciting point process). With these



(a) Original point pattern.



(b) Point density in a regular grid.



(c) Kernel density.

Figure 2.3: Example for a spatial point pattern and extracted densities / intensities. The point locations (events) represent 2661 crimes within a city boundary, following the example given by [70].

goals in mind, there are many methods to analyse spatial point patterns, the most important of which we summarise below.

Spatial dependencies are one of the key challenges outlined in the previous section. This is also an intuitive entry point for any spatial point pattern analysis: For a given spatial point pattern, we are interested to know whether the events that comprise the point pattern are distributed randomly in space or whether they follow certain (spatial) rules. For examples, trees in a forest are often distributed somewhat regularly, leaving enough space around them for other trees to grow. On the other hand, fish can often cluster around particular food sources. None of these distributions are truly random, but follow some underlying laws. Tests for complete spatial randomness are some of the most fundamental analytical tools for spatial point processes [106].

Figure 2.3 highlights an example of a spatial point pattern–the locations of crimes in a city–and its gridded and kernel densities. These densities, or intensities, calculated in given spatial units like grid cells also serve as inputs for two of the most important tools for analysing spatial point patterns and determining whether complete spatial randomness is present or not: the G function and F function:

Point Pattern Analysis: The G and F functions

The **G function** describes the distribution of all distances between an arbitrary event and its respective nearest neighbour event. Assuming complete spatial randomness, we can compute the theoretical G function and compare it to the empirical G function of our observations. The discrepancy between the observed and expected functions can help us to identify potential spatial dependencies.

The **F function**, also termed the "empty space" function, describes the distribution of all distances between an arbitrary point in our observational area and the respective nearest neighbour event. As for the G function, we can compare theoretical and empirical F functions to obtain inference on the presence of spatial dependencies.

G and F are both functions of distances d and can be plotted as such. Figure 2.4 shows example G and F functions for the crime point patterns displayed in Figure 2.3. Here, the empirical functions are shown next to their theoretical counterpart, which are the result of a homogenous Poisson process. Note that the theoretical F and G functions are identical. We can see that F and G deviate strongly from their expected lines, which gives us an indication that there is some spatial dependence (i.e., no complete spatial randomness).

We now move on to the statistical analysis of spatial point patterns. These can be broadly split into first- and second-order properties. First-order properties describe the distributions of events in space. Second-order properties

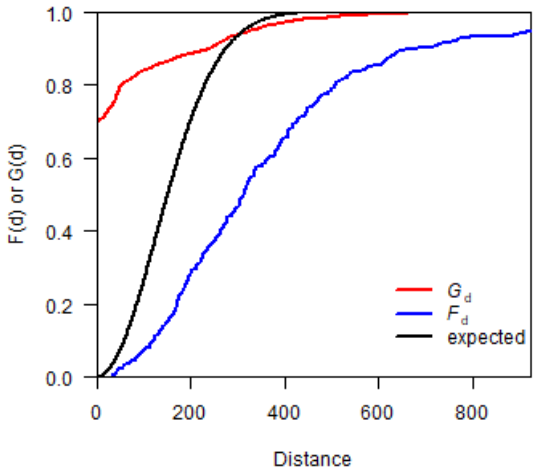


Figure 2.4: Example of the empirical G (red) and F (blue) functions (with values between 0 and 1), computed for the point pattern intensities displayed in Figure 2.3b. The expected (theoretical) G and F functions (which are identical) are given as the black line. Distance on the y-axis is given in meters.

describe interactions between events [41].

First-order properties include measures of spatial density and intensity. Intensity measures the number of events within a spatial unit (e.g. census tracts), while density also reflects the size of the spatial unit. Intensities are essential to the modelling of point processes. Poisson processes are the most common class of point processes that are used for spatial point patterns. Poisson processes assume that events occur randomly and follow their respective intensity functions. They can broadly be split into homogeneous and inhomogeneous Poisson processes.

Homogeneous Poisson processes describe point patterns with independently and uniformly distributed events. An event occurrence does not affect the probability of another event occurring in the spatial vicinity and no spatial unit has a higher probability of recording events than other units. Inhomogeneous Poisson processes allow the intensity functions to vary for different spatial units, which relaxes the restrictive assumptions of a constant intensity function. While Poisson processes are often sufficient for modelling spatial point patterns, depending on the application, point processes capable of reflecting more complex spatial (and spatio-temporal) dynamics like log-Gaussian Cox Processes may be necessary [40].

The estimation of point process intensities is a crucial part of analysing spatial point patterns. Kernel smoothing is the most commonly applied

technique here [159]. A kernel is a weight function used to estimate the density function of a given random variable (for example, the crime intensities portrayed in 2.3b). Kernel smoothing operations are characterised by two important configurations. The bandwidth of the kernel determines the degree of smoothing. The kernel function can be chosen to reflect specific assumptions or domain expertise. Beyond kernel smoothing, there exist several other, parametric methods for obtaining intensity estimates [41].

Second-order properties help with the assessment of interactions between different events in our point pattern. For example, the second-order intensity between two events describes the probability of another pair of events occurring nearby. The most common measure of second-order properties is Ripley's K function [149]. It measures the number of events occurring within a radius around a given event. Again, the discrepancy between the expected and observed K function can be used as a basis for inference, in this instance on the interaction between events. Generally, we assume that spatial effects occur only on relatively small spatial scales, which the choice of the search radius usually reflects. As such, the K function can serve as a detector of spatial clustering.

2.1.4 Traditional Metrics for Capturing Spatial and Spatio-Temporal Autocorrelation

We now move from point patterns, events distributed in space, to numerical spatial data (i.e. numerical values that can be assigned a distinct discrete or continuous location). Point process intensities in the different spatial units of a spatial point pattern comprise numerical spatial data. Another example involves the pixel values of an image that are distributed in a discrete, regular grid. The prices of different houses at their respective locations are an example for continuous numerical spatial data. Like spatial point patterns, this type of data can be found in many different scientific domains.

Spatial autocorrelation is a common phenomenon encountered in spatial data that can arise from one or more of the challenges outlined in section 2.1.2, such as the MAUP. It describes the presence of systemic spatial variation in random variables. Thus, spatial autocorrelation violates one of the key assumptions for obtaining unbiased estimators in many statistical modelling frameworks: independently and identically distributed (iid) observations. A key problem in the analysis of any spatial data is that there exist many approaches to measure spatial autocorrelation and related effects.

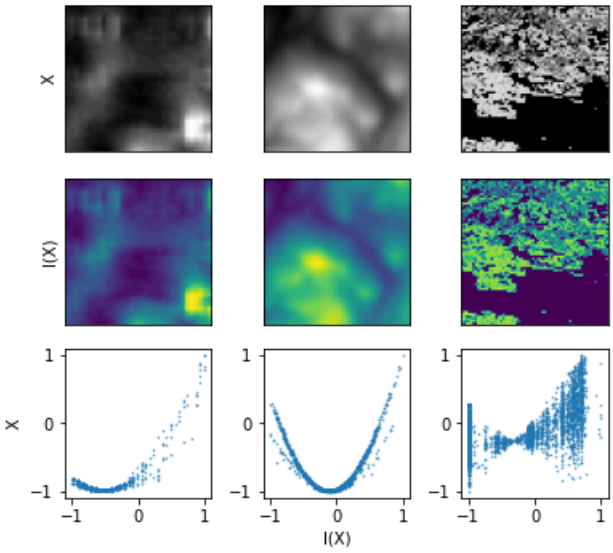


Figure 2.5: Example of the different shapes the relationship between X and $I(X)$ can take using seabed relief (column one), digital elevation map (DEM; column two) and tree canopy data (column three). Row one shows the data X plotted as images with white indicating low and black high values. Row two shows the local Moran's I of the data $I(X)$ plotted as images with yellow indicating high positive values and dark blue indicative high negative values. Row three plots X against $I(X)$ to highlight the functional relationship. All data is scaled to values between -1 and 1. The figure also highlights how the local Moran's I can serve both as a measure of spatial outliers (column one) and homogeneous spatial clusters (column three). These data are later used in experiments in chapter 4.

Moran's I

The most common of these approaches is the Moran's I measure of spatial autocorrelation. This metric was originally conceptualised by Moran [129] and popularised in the geographic domain by Anselin [2].

To compute the Moran's I metric, let us first define a vector $\mathbf{y}$ that consists of n real-valued observations x_i, referenced by an index set $\mathcal{N} = \{1, 2, \ldots, n\}$. We define the *spatial neighbourhood* of observation i to be $\mathcal{N}_i = \{j \in \mathcal{N} : w_{i,j} > 0\} = \{1, 2, \ldots, n_i\}$. Here, $w_{i,j}$ corresponds to a binary spatial weight matrix, which indicates whether any observation j is a neighbour of i. As discussed in section 2.1.1, assigning spatial neighbourhoods is an important design choice and reflects the type of spatial data (discrete or continuous) and potential prior knowledge or domain expertise on the problem at hand. For now, we assume that we have an appropriate spatial weight matrix. We can then compute the Moran's I statistic I as:

$$I = \frac{n}{W} \frac{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} w_{i,j}(y_i - \bar{y})(y_j - \bar{y})}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.1}$$

Where $\bar{y}$ is the mean of all observations $\mathbf{y}$ and $W = \sum_i^n \sum_j^n w_{i,j}$. The Moran's I measures how similar observations are compared to their spatial neighbourhood. Values of $I > 0$ imply *positive* spatial autocorrelation, while values $I < 0$ suggest negative spatial autocorrelation. The Moran's I metric can also be used to obtain a statistical test for spatial autocorrelation.

Under the null-hypothesis, that is *no spatial autocorrelation*, the expected value of the Moran's I metric is given as $E(I) = \frac{-1}{n-1}$. For large sample sizes, $E(I)$ under the null tends to 0. Using permutations, the expected and observed values of the Moran's I metric can be compared against one another to obtain z-scores and perform statistical hypothesis testing. The Moran's I measure is a global metric; thus, it returns a single value (and respective statistical measures) for all observations $\mathbf{y}$.

In some situations, we are interested in identifying local spatial effects rather than global patterns. The Moran's I metric can be expanded to obtain individual measures of *local* spatial autocorrelation I_i:

$$I_i = (n-1)\frac{y_i - \bar{y}}{\sum_{j=1, j \neq i}^{n}(y_j - \bar{y})^2} \sum_{j=1, j \neq i}^{n} w_{i,j}(x_j - \bar{y}) \tag{2.2}$$

As in Equation 2.1, which was used to compute the global Moran's I, I_i can take positive or negative values. A positive value suggests that a data point is similar to its neighbours, which could indicate latent cluster structure.

A negative value suggests that the data point is distinctly different from neighbouring data points, which could indicate a changepoint or edge. As with the global Moran's I, we can use the local Moran's I to run statistical tests for the presence of local spatial autocorrelation. While the local Moran's I statistic is closely correlated with its input, this relationship can take different forms, depending on the complexity of the inputs' spatial structure. Figure 2.5 shows some examples of spatial data and their distribution of local Moran's I values.

The local Moran's I metric represents a powerful and flexible detector of spatial outliers and spatial homogeneity. Its computation is simple and relies only on a pre-defined spatial weight matrix. As such, the local Moran's I can be seen as a simple, functional spatial context embedding. In the technical chapters of this dissertation, particularly chapter 4, we will use the local Moran's I metric to reinforce the learning of spatial autocorrelation in neural network models.

Spatial Extensions

Since its initial formulation, several extensions to the Moran's I metric have been proposed to capture different spatial effects. In this dissertation, we will limit this discussion to measures of *local* spatial effects. Specifically, we will discuss local spatial heteroskedasticity and local spatial dispersion.

Local Spatial Heteroskedasticity (LOSH) Local Spatial Heteroskedasticity (LOSH), proposed by Ord and Getis [137], assesses local deviations from global variances rather than means. It is derived from the G statistic [60, 136] and is closely related to the Moran's I. Keeping with the notation introduced for the Moran's I calculation, we can compute the LOSH metric H_i as:

$$H_i = \frac{\sum_{j=1,j\neq i}^{n} w_{i,j}(y_j - \bar{y}_j)^a}{h_1 \sum_{j=1,j\neq i}^{n} w_{i,j}} \tag{2.3}$$

where, rather than global means, we use local means that reflect spatial neighbourhoods, such that:

$$\bar{y}_j = \frac{\sum_{k=1,k\in\mathcal{N}_j}^{n_j} w_{j,k} y_k}{\sum_{k=1,k\in\mathcal{N}_j}^{n_j} w_{j,k}} \tag{2.4}$$

where k indexes second-degree neighbours of i (i.e. neighbours of j). Lastly, the mean residual of the spatial neighbourhood of i is given as:

$$h_1 = \sum_{j=1, j \neq i}^{n} (y_j - \bar{y}_j)^a \tag{2.5}$$

.

Here, the exponent a controls the type of deviance to-be-assessed, most commonly variance, i.e. $a = 2$. Under the null-hypothesis, the local variance does not deviate significantly from the global mean variance. The expected value of H_i under the null is given as $E(H_i) = 1$. Ord and Getis [137] propose a χ-squared test for statistical significance of the LOSH metric.

Local Spatial Dispersion (LSD) Local Spatial Dispersion (LSD) is a direct expansion of the LOSH metric, conceptualised by Westerholt et al. [189]. In a sense, it is the "most local" of the presented metrics, ignoring global context and focusing purely on the comparison between local dependencies and local variances.

The LSD metric can be calculated as:

$$LSD_i = \frac{\sum_{j=1, j \neq i}^{n} w_{i,j} (y_j - \bar{y}_j)^a}{h_i \sum_{j=1, j \neq i}^{n} w_{i,j}} \tag{2.6}$$

Here, we replace the mean neighbourhood residuals h_1 with an explicitly local version:

$$h_i = \frac{\sum_{j=1, j \in \mathcal{N}_i}^{n_i} (y_j - \bar{y}_j)^a}{n_i} \tag{2.7}$$

This implies that $LSD_i = H_i$ if $h_i = h_1$. The relationship of both metrics can be formulated as:

$$LSD_i = \frac{H_i h_1}{h_i} \tag{2.8}$$

Again, LSD can be used for statistical testing using a χ-squared test. Under the null hypothesis, we assume that the local spatial structure of our random variable does not affect variance. LSD, while closely related to LOSH, can be particularly useful for detection of locally contained effects that would be overlooked by the global calculation of h_1 for LOSH.

Spatio-Temporal Extensions

We also briefly want to discuss spatio-temporal autocorrelation measures. There are many approaches that aim to expand the intuition of the Moran's I metric to spatio-temporal data. Let our numerical variable $y_{i,t}$ be indexed by spatial units i and time steps t, with a total number of time steps T. Now, let us define a temporal weight matrix $b_{t,t'}$ assigning temporal neighbourhood (e.g. directly preceding and following time steps), analogous to the spatial weight matrix $w_{i,j}$. An intuitive spatio-temporal Moran's I approach is proposed by Lee and Li [108]:

$$I^{st} = \frac{nT}{W^{st}} \frac{\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} \sum_{t=1}^{T} \sum_{t'=1,t'\neq t}^{T} w_{i,j} b_{t,t'} (y_{i,t} - \bar{y})(y_{j,t'} - \bar{y})}{\sum_{i=1}^{n} \sum_{t=1}^{T} (y_{i,t} - \bar{y})^2} \tag{2.9}$$

where $W^{st} = \sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} \sum_{t=1}^{T} \sum_{t'=1,t'\neq t}^{T} w_{i,j} b_{t,t'}$. This approach uses the temporal weights b and the spatial weights w. The mean values $\bar{y} = \sum_{i=1}^{n} \sum_{t=1}^{T} y_{i,t}$ are computed over space and time. Lee and Li [108] also propose a local version of their metric, analogous to the Moran's I.

Gao et al. [53] propose a different approach that focuses on the deviance if each spatial unit's time series from the global mean time series. Their metric I^t is defined as:

$$I^t = \frac{n}{W} \frac{\sum_{i=1}^{n} \sum_{j=1,i\neq j}^{n} w_{i,j} z_i^{ts} z_j^{ts}}{\sum_{i=1}^{n} (z_i^{ts})^2} \tag{2.10}$$

Here, z_i^{ts} describes the deviation of each spatial units time series $\mathbf{y}_i = [x_{i,0}, \ldots, x_{i,T}]$ from the mean time series $\bar{\mathbf{y}}$. For details on the calculation of this deviance, please refer to the original paper [53]. The authors also expand this idea to a localised version of the metric.

Further approaches to develop measures of spatio-temporal correlation include the iterative computation of the metric over time [124] and methods that specialise in spatial point patterns [158].

2.1.5 Spatial modelling in Statistics and Econometrics

We have now obtained a broad understanding on how spatial effects in point patterns (simple locations) and numerical spatial data (i.e. locations with numerical values) are measured, as well as the most popular metrics and functions that concern these tasks.

Beyond hypothesis testing and the quantification of spatial effects, we will now look into how different scientific disciplines have traditionally approached the (predictive) modelling of spatial data. A more detailed overview on this

topic can be found in Bivand et al. [20] We will broadly split this into two categories, *spatial interpolation* and *spatial prediction*.

Spatial interpolation seeks to predict an unknown outcome at a new location from measures that are taken at existing locations. We can formulate this as a simple regression problem:

$$\hat{y}_i = f_{int}(\mathbf{c}_i) + \epsilon_i \tag{2.11}$$

Here, $\hat{y}_i$ is our predicted outcome and $\mathbf{c}_i$ represents spatial information on the new location i (e.g. its geographic coordinates), such that $\mathbf{c}_i = [lon_i, lat_i]$. $\epsilon_i = y_i - \hat{y}_i$ is the true error term of our model. The function f_{int} maps inputs to predicted outputs and can take various forms. In the case of linear regression, it is defined by regression coefficients β; for neural networks, the function is parameterised by layer weight parameters Θ. An example of this type of problem is the interpolation–and potentially extrapolation–of weather maps from a limited number of weather stations. With *spatial prediction* we describe an expansion of *spatial interpolation* where we have access to additional predictor variables at each location i. We can reflect this in the formulation of our regression problem so that:

$$\hat{y}_i = f_{pred}(\mathbf{c}_i, \mathbf{x}_i) + \epsilon_i \tag{2.12}$$

Here, $\mathbf{x}_i$ represents a vector of predictor variables. An example of this problem is the prediction of house prices at new locations, having access to a set of predictors like the age of the house or the number of bedrooms.

Simple Interpolation Approaches

Let us stick to the example of a spatial interpolation task where spatial context $\mathbf{c}_i$ is given in the form of point coordinates. The simplest way to obtain predictions is to run a simple linear model using the ordinary least squares (OLS) estimator. However, this approach has two crucial shortcomings. First, linear models are not able to model non-linearity and the complexities that are often associated with spatial interpolation tasks. Second, as mentioned in the previous sections, the presence of spatial dependencies will render our estimator biased and violate the independence assumption of the OLS estimator.

Instead, the simplest solution for spatial interpolation is often some form of distance-based averaging of known locations. Here, we will briefly discuss the popular inverse distance weighting (IDW) approach. We first define the distance weight between a known coordinate $\mathbf{c}_i$ and a new point-of-interest $\mathbf{c}_0$

as:

$$w_{dist}(\mathbf{c}_i) = ||\mathbf{c}_i - \mathbf{c}_0||^{-p} \tag{2.13}$$

where $-p$ controls the degree of distance weighting and $||\cdot||$ denotes the Euclidean distance. The outcome variable $\hat{y}$ at location $\mathbf{c}_0$ can then be calculated as a weighted average:

$$\hat{y}(\mathbf{c}_0) = \frac{\sum_{i=1}^{n} w_{dist}(\mathbf{c}_i) y(\mathbf{c}_i)}{\sum_{i=1}^{n} w_{dist}(\mathbf{c}_i)} \tag{2.14}$$

Spatial Lag and Error Models

We now consider the problem of *spatial prediction* as outlined in Equation 2.12. We can incorporate spatial autoregressive effects into regression models in different ways. Here, we focus on the *spatial lag model* and the *spatial error model*. The basis for both of these approaches is a linear regression model. Let us assume that a univariate linear model predicts an outcome y_i using a single predictor x_i. The model can be formulated as:

$$\hat{y}_i = \beta_0 + \beta_1 x_i + \epsilon_i \tag{2.15}$$

The spatial lag model includes the 'lagged' (i.e. distance-weighted) outcomes of neighbouring observations as predictors, such that:

$$\hat{y}_i = \beta_0 + \lambda w_{i,j} y_j + \beta_1 x_i + \epsilon_i \tag{2.16}$$

Where $w_{i,j}$ is the spatial weight matrix defining neighbourhood of observations i and j - the same type of matrix that is used for the computation of the Moran's I statistic. $w_{i,j}$ can also reflect distances between observations i and j, similarly to the way IDW is conducted (see Equation 2.13 and 2.14). λ is the estimated model parameter, akin to β. The inclusion of the spatial lag variable can then help to eliminate spatial autocorrelation from the model residuals, thus complying with OLS assumptions.

A second popular approach is the spatial error model, aiming to explicitly model the spatial component of the model error:

$$\hat{y}_i = \beta_0 + \beta_1 x_i + \rho w_{i,j} u_i + \epsilon_i \tag{2.17}$$

Where u_i is the spatial component of the model error and ρ the estimated coefficient. Spatial error models overcome the problem of autocorrelated residuals by estimating the spatial error component to explicitly "debias" the error term. Spatial lag and error approaches can also easily be combined:

$$\hat{y}_i = \beta_0 + \lambda w_{i,j} y_j + \beta_1 x_i + \rho w_{i,j} u_i + \epsilon_i \tag{2.18}$$

Geographically Weighted Regression

We now move onto an approach that, rather than assuming stationary model coefficients, allows them to be expressed locally, as functions of spatial coordinates. Geographically Weighted Regression (GWR), originally proposed by Brunsdon et al. [25], can help to model relationships between variables that may vary over space. This takes the form of local regression modelling, using local data subsets to fit the model. Sticking with our univariate regression example, this can be expressed as:

$$\hat{y}_i = \beta_{0i} + \beta_{1i} x_i + \epsilon_i \tag{2.19}$$

where β_{0i} and β_{1i} represent local model coefficients. To outline the estimation of the local model coefficients, it is convenient to express the model in matrix notation:

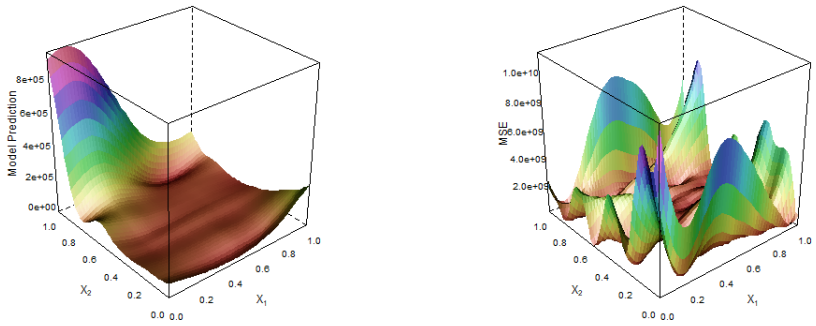
$$\hat{y}_i = \beta_{\mathbf{i}} \mathbf{x}_i + \epsilon_i \tag{2.20}$$

where $\mathbf{x}_i$ is the vector of predictor variables and β_i the vector of local regression coefficients (i.e. both are 1×1 vectors for the univariate regression model). We now estimate the local regression coefficients β_i, such that:

$$\hat{\beta}_i = [\mathbf{X}^T \mathbf{W}_i \mathbf{X}]^{-1} \mathbf{X}^T \mathbf{W}_i \mathbf{Y} \tag{2.21}$$

where $\mathbf{Y}$ is the outcome variable, a $1 \times n$ vector, $\mathbf{X} = [\mathbf{x}_0^T, \ldots, \mathbf{x}_n^T]^T$ is the design matrix of predictors, and $\mathbf{W}_i = diag[w_{i,0}, \ldots, w_{i,n}]$, corresponding to the full $n \times n$ spatial weight matrix:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \ldots & w_{1,n} \\ \ldots & \ldots & \ldots \\ w_{n,1} & \ldots & w_{n,n} \end{bmatrix} \tag{2.22}$$



(a) Prediction. (b) Error surface.

Figure 2.6: Example for Gaussian Process predictions on a $2d$ surface. Displayed are the (smooth) prediction surface and errors.

GWR can be performed through locally weighted least squares estimation. As in previous applications, the spatial weights $\mathbf{W}$ can reflect differences between point coordinates, or kernelised versions thereof.

Gaussian Processes and Kriging

To conclude this section, we discuss the most popular class of models for particularly continuous spatial data: Gaussian Processes (GPs), Kriging and the subtle differences between them. Mathematically, we will focus on GP models, as they are the far more flexible and non-parametric approach. For an in-depth discussion of GPs, please refer to Rasmussen and Williams [147]. Figure 2.6 shows an example of a Gaussian Process prediction.

Gaussian Processes can be seen as multivariate Gaussian distributions over an infinite number of jointly Gaussian variables; a Gaussian Process is a distribution over functions. A sample from a GP represents a function generating values according to some Gaussian distribution. Formally, we can define a GP regression problem as predicting the outcome $\hat{\mathbf{y}}$ as a function $f(\mathbf{x})$ of the input:

$$f(\mathbf{x}) = GP(\mathbf{m}(\mathbf{x}), \mathbf{k}(\mathbf{x}, \mathbf{x}')) \tag{2.23}$$

where $\mathbf{m}(\mathbf{x}) = E[f(\mathbf{x})]$ represents the mean function and $\mathbf{k}(\mathbf{x}, \mathbf{x}') = E[(f(\mathbf{x}) - \mathbf{m}(\mathbf{x}))(f(\mathbf{x}') - \mathbf{m}(\mathbf{x}'))^T]$ represents the covariance function. Note that the input $\mathbf{x}$ includes all available predictors, including potential spatial coordinates $\mathbf{c}$. While we introduce general-form GPs here, GP spatial interpolation represents the special case of this formulation when $\mathbf{x} = \mathbf{c}$.

The GP defines a joint Gaussian distribution $p(f|X) = \mathcal{N}(f|\mu, K)$, where $K_{i,j} = \mathbf{k}(x_i, x_j)$ and $\mu = [\mathbf{m}(\mathbf{x}_1), \ldots, \mathbf{m}(\mathbf{x}_n)]$.

For inference, the GP regression model interpolates from seen (training) to unseen (testing) data, by constructing a joint distribution. The posterior distribution then takes the form:

$$p(f_*|X, X_*, f) = \mathcal{N}(f_*|\mu_*, \Sigma_*) \tag{2.24}$$

where

$$\mu_* = \mu(X_*) + K_*^T K^{-1}(f - \mu(X)) \tag{2.25}$$

and

$$\Sigma_* = K_{**} - K_*^T K^{-1} K_* \tag{2.26}$$

Here, f are training outcome variables and f_* testing predictions. As mentioned, GPs are non-parametric models. Nevertheless, the choice of covariance function (or kernel) and its associated hyperparameters, is crucially important for modelling. The kernel function can take a myriad of forms, from linear to exponential to spectral kernels. Here, we briefly present one of the most common GP kernels, the radial basis function (RBF) kernel:

$$k_{rbf}(\mathbf{x}, \mathbf{x}') = \sigma^2 exp(-\frac{1}{2l^2}||\mathbf{x} - \mathbf{x}'||^2) \tag{2.27}$$

Here, σ^2 represents the output scale hyperparameter and l the length scale hyperparameter. Hyperparameters are optimised using maximum marginal likelihood estimation during model training.

With the general definition of GP models outlined, let us now discuss the differences between GPs and Kriging. Kriging was popularised as an interpolation technique in geostatistics by Matheron [123]. The differences between GPs and Kriging can be broadly split into four factors:

- **Bayesian vs Frequentist approach:** While GPs follow Bayesian intuition, deriving posterior distributions from informed priors and data, Kriging tackles the interpolation problem using a least squares approach. Unbiased estimators are obtained by explicitly modelling spatial covariances via variograms.

- **Predictor dimensionality:** While GPs can scale to arbitrary input dimensions using multivariate Gaussian distributions (although high-dimensional GPs are computationally expensive), Kriging was originally devised for 2 and 3-dimensional inputs. The Cokriging technique tackles this issue [147].
- **Model fitting:** As non-parametric models, GPs are substantially more flexible and able to combine and nest different kernel functions. In Kriging, it can be difficult to fit more complex variograms.
- **Design choices & hyperparameter optimisation:** While GP models require initial decision making (e.g. on kernel choices or starting values for hyperparameters) models can be fit automatically and hyperparameters can be optimised using marginal likelihood methods. Kriging, on the other hand, requires much more careful variogram design by hand to identify the optimal settings.

2.1.6 Discussion

In this section, we have outlined the specific challenges of geographic data and introduced traditional approaches for their analysis. Subsection 2.1.4 focuses on measures of spatial and spatio-temporal effects, such as autocorrelation or heteroskedasticity. These metrics, while powerful, have some limitations. First, they usually rely on some pre-defined intuition of spatial neighborhood for their calculation. This assumes either that this information is available, or the construction of approximate neighborhoods using e.g. methods such as k-nearest-neighborhoods, which can be unrealistic in some real-world settings. Second, these metrics are developed for exploratory analysis (e.g. the assessment of spatial autocorrelation in regression residuals) and there is no straightforward way of integrating them into learning algorithms. Such approaches and frameworks are necessary to utilize the domain expertise represented by these metrics in, for example, predictive modeling tasks.

Subsection 2.1.5 presents traditional approaches for modeling spatial data, from autoregressive regression models to kernel methods like GPs. And while these approaches are well established in many applied disciplines, from economics to ecology, they also have some limitations. Methods like spatial lag models or geographically weighted regressions are linear and can struggle with representing the often non-linear complexities of real-world geospatial data. GPs and Kriging, while able to model non-linear data, struggle with scaling due to their reliance on calculating expensive covariance matrices. This problem is amplified by the need for better scaling methods in the light of the growing availability of big geospatial data [148].

2.2 Neural Networks in the Geospatial Domain

As outlined in the previous section, the GIS community has found many ways to measure spatial effects and to account for these in predictive models. Let us now switch gears and look at spatial data and its challenges from a modern machine learning perspective, focusing on neural networks. First, we will introduce the two families of neural network models that are used in the remainder of this dissertation: *discriminative* (predictive) and *generative* models. Then, we will discuss how geographic data are traditionally and currently processed in neural networks. Lastly, we examine different ways of integrating domain expertise–such as our knowledge on spatial dependencies–into neural network models.

2.2.1 Predictive and Generative Modelling with Neural Networks

Neural networks (NNs) are a powerful tool for modelling high-dimensional, non-linear data. They are inspired by the signal processing over distributed nodes (neurons) of the brain. NNs consist of layers of artificial neurons, extracting higher-dimensional features from inputs by re-weighting and re-projecting them. The first layer of a NN is referred to as the *input layer*, the last layer as the *output layer* and any layers in between as *hidden layers*. The simplest form of a feed-forward neural network–unlike recurrent NNs, for instance, which have cyclic graphs–are single- and multi-layer perceptrons (MLP). Neural networks are trained via backpropagation on a set loss function, where the loss is differentiated with respect to the NN weights. The weights are then updated according to the (stochastic) gradient descent to find the loss minimum. For a detailed overview on neural networks and their optimisation, please refer to the textbook by Bishop [19].

Together, advances in NNs, gradient descent methods and automatic differentiation have revolutionised data analysis and science. NNs are not only capable of scaling to very high-dimensional data domains (e.g. high-resolution images, videos), they are also highly flexible and can process different data types (text data, image data, audio data). Specific NN architectures and operators have been developed to address different data structures, such as convolutional neural networks (CNNs) [52] for image data and long short-term memory (LSTM) networks [71] for sequential data. NNs can also be deployed for different tasks. Here, we focus on *discriminative* NNs, models that learn to predict and outcome based on a given input, and *generative* NNs, models that learn a data-generating process in order to generate synthetic data.

Neural Network Regression and Classification

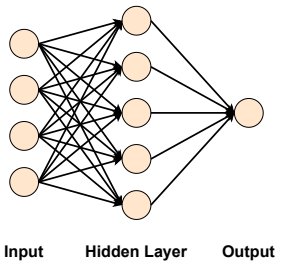


Figure 2.7: Example of a simple feed-forward neural network with four input nodes, a hidden layer, and a singular output.

Let us first outline NNs for prediction. Recall the *spatial prediction* problem outlined in Equation 2.12, but for now, let us assume there is no spatial component $\mathbf{c}$. We can define a neural network prediction model as:

$$\hat{y}_i = f_{nn}(\mathbf{x}_i) + \epsilon_i \tag{2.28}$$

Let us now assume a simple MLP with a single hidden layer. Our neural network then takes the form:

$$f_{nn} = \boldsymbol{\Theta}_2 g(\boldsymbol{\Theta}_1^T \mathbf{X} + \mathbf{b}_1) + \mathbf{b}_2 \tag{2.29}$$

Here, $\boldsymbol{\Theta}_1$ and $\boldsymbol{\Theta}_2$ are the weight parameters of the input and hidden layer, respectively. $\mathbf{b}_1$ and $\mathbf{b}_2$ represent bias added to the layers. Lastly, $g(\cdot)$ represents the activation function. This allows us to make our model–which was thus far linear–non-linear. Activation functions are an essential component for flexible neural network modelling. For example, assume a regression problem, with our target variable consisting of real numbers $y_i \in \mathbb{R}; x_i \geq 0, x_i \leq 1$. Here, we can set activation functions fit for continuous NN outputs $\mathbf{z}$ such as the identity function:

$$g(\mathbf{z}) = \mathbf{z} \tag{2.30}$$

or a hyperbolic tangent function:

$$g(\mathbf{z}) = \frac{e^{\mathbf{z}} - e^{-\mathbf{z}}}{e^{\mathbf{z}} + e^{-\mathbf{z}}} \tag{2.31}$$

In the case of a binary classification problem, with binary outcome variables $y_i \in [0, 1]$, we might require a logistic activation function:

$$g(\mathbf{z}) = \frac{1}{1 + e^{\mathbf{z}}} \tag{2.32}$$

Using a threshold (e.g. 0.5), we can then make predictions $\hat{y}_i \in [0, 1]$. We also must choose the loss function that our NN uses to optimise parameters according to the task at hand. For regression problems, we might want to use a penalised mean squared error (MSE) loss function:

$$\mathcal{L}_{MSE}(\hat{\mathbf{y}}, \mathbf{y}, \boldsymbol{\Theta}) = \frac{1}{2}||\hat{\mathbf{y}} - \mathbf{y}||_2^2 + \frac{\alpha}{2}||\boldsymbol{\Theta}||_2^2 \tag{2.33}$$

where $\boldsymbol{\Theta}$ are the NN weight parameters, $||\cdot||_2^2$ is the L2 norm, and α a non-negative hyperparameter.

In case of a binary classification problem, we might want to use a binary cross entropy (BCE) loss function, defined as:

$$\mathcal{L}_{BCE}(\hat{\mathbf{y}}, \mathbf{y}, \boldsymbol{\Theta}) = -\mathbf{y} ln \hat{\mathbf{y}} - (1 - \mathbf{y}) ln(1 - \hat{\mathbf{y}}) + \alpha||\boldsymbol{\Theta}||_2^2 \tag{2.34}$$

We can now compute the gradient of the loss function with respect to the weights $\nabla \mathcal{L}_{\mathbf{W}}$ and use it to update the weight parameters, such that:

$$\boldsymbol{\Theta}^{t+1} = \boldsymbol{\Theta}^t - \eta \nabla \mathcal{L}_{\boldsymbol{\Theta}}^t \tag{2.35}$$

where t indexes the training step and $\eta > 0$ is the learning rate, controlling the degree to which a parameter can change in one training step. NNs are trained iteratively until convergence. Figure 2.7 shows an example of a simple feed-forward neural network architecture.

Generative Adversarial Networks

Generative models aim to learn a data generating process from a set of training data. Popular classes of generative models include variational auto-encoders and generative adversarial networks (GANs), which we will extend in several applications later on in this dissertation.

GANs were originally proposed by Goodfellow et al. [64]. A GAN consists of two neural networks: a Generator (G) and a Discriminator (D). The Generator is responsible for producing a latent representations of the input, attempting

to replicate a given data generating process. It is defined as a neural network $G(\mathbf{z}, \Theta_G)$ with parameters Θ_G, mapping noise $\mathbf{z} \sim p_z(z)$ to some feature space $\mathbf{x}$ ($G : \mathbf{z} \rightarrow \mathbf{x}$). The Discriminator, a neural network $D(\mathbf{x}, \Theta_D)$, aims to probabilistically distinguish the synthetic input $\hat{\mathbf{x}}$ created by the Generator and real data $\mathbf{x} \sim p_{data}(\mathbf{x})$ ($D : \mathbf{x} \rightarrow [0, 1]$). Both networks compete in a minimax game, which improves their performance until the real and synthetic data are (ideally) indistinguishable from one another. The GAN loss function can then be expressed as:

$$\begin{aligned} \min_G \max_D \mathcal{L}_{GAN}(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})}\big[\log D(\mathbf{x})\big] + \\ \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}\big[\log(1 - D(G(\mathbf{z})))\big] \end{aligned} \tag{2.36}$$

Note that while GANs contain a Discriminator D, they belong to the family of generative models as the eventual output we are interested in is the synthetic data generated by the Generator G. The Discriminator D is merely working with pseudo-labels (for "real" and "synthetic" data respectively) and is used as counter-player for the Generator G to facilitate learning. Lastly, one of the limitations of GANs is that they can be notoriously difficult to train and require a careful balancing of generator and discriminator, through the loss function or their respective neural network architectures.

2.2.2 Modelling Geographic Data with Neural Networks

The previous subsection on neural networks for generative and predictive modelling has been kept explicitly non-spatial. We now want to discuss different approaches of integrating geospatial context into NNs. We will first assess how convolutions can process discrete spatial data distributed in regular 2 and 3-dimensional grids (e.g. images and spatial point pattern intensities). We then take a look at mechanisms for integrating spatial coordinates into neural networks.

Convolutions on Regular Grids and Graphs

Convolutions are the essential building block of convolutional neural networks. They are inspired by the way the visual cortex processes signals and convolves them into stimuli. The convolutional neurons each each process a different part of the input map, referred to as receptive field, and pass it to the following layer. For example, let our input be a set of n images of size $width \times height$ with $nchannels$ channels (e.g. three for RGB images), represented as a tensor of shape $[n, width, height, nchannels]$. A convolutional operator transforms the input into a feature map of shape $[n, width_{fm}, height_{fm}, nchannels_{fm}]$ (fm refers to feature map).

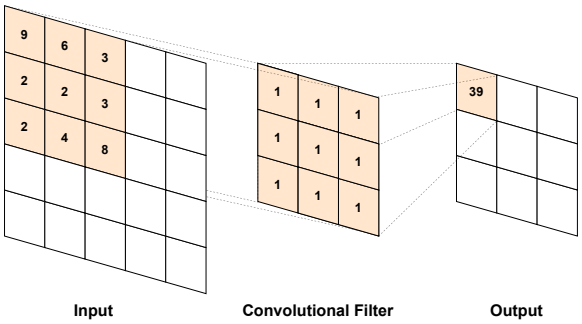


Figure 2.8: Example of a convolutional operation on an image input. The convolutional filter is a simple, equal-weight addition.

For each receptive field, a learnable spatial filter (or kernel) is deployed. The local feature map is computed as the dot product of the filter and the input. The global feature map is the combination of all local feature maps. Thus, CNNs that consist of more than one convolutional layers are locally-connected. Moreover, they explicitly exploit local correlations to learn feature maps. This is familiar to the local neighbourhood focus of for example the Moran's I metric. The kernel size of convolutional layers controls the size of each receptive field, similar to how the spatial weight matrix $w_{i,j}$ controls the neighbourhood size in the Moran's I calculation. And indeed, through enforcing sparse local connectivity, CNNs are able to account for spatial autoregressive effects.

The output size of a convolutional layers is controlled by three hyperparameters. The *depth* determines the number of neurons that connect the current layer to the input. The *stride* controls how far we move the 'window' for each receptive field. Low stride values lead to heavily overlapping receptive fields. Lastly, the *padding size* allows us to apply zero padding to the edges of the input map. We show a simple example of a convolutional operator in Figure 2.8.

Other important components of CNNs are pooling layers, which allow for a downsampling of the input, again controlled by a kernel size. Common pooling operators are max pooling and average pooling. CNNs also often deploy the rectified linear unit (ReLU) activation function between layers to avoid negative value features. ReLU is defined as:

$$g(\mathbf{z}) = max(0, \mathbf{z}) \tag{2.37}$$

While CNNs were originally devised for processing image data, their intuition for local spatial patterns makes them promising tools for geospatial

data that can be arranged in regular grids to form tensors similar to images. This includes geospatial image data such as satellite imagery, discretised points process or count data (intensities) or any other data that can be measured on such a grid, like digital elevation maps (DEM). CNNs are nonetheless limited in their capturing of spatial effects. They are restrained on learning a pre-defined local context. This has led to efforts within the computer vision community to develop methods that also account for long distance spatial effects, such as vision transformers [42], building upon a spatial extension of the attention mechanism.

CNNs can be also be used for graphical data, as proposed by Kipf and Welling [90]. Here, the intuition of receptive fields is expanded from patches of an image to local sub-graphs of the main graph. As with vision transformers, attention mechanisms have also been introduced to graphical models [178].

Embedding Point and Polygon Locations

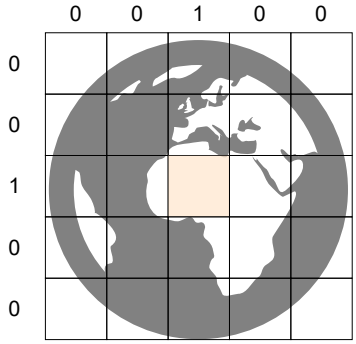


Figure 2.9: Example of a one-hot encoding of planet Earth. Earth is split into a 5×5 grid of equal-size cells. If the location of interest falls into the x and y locations of a cell (here in orange), it is represented with a value of one; otherwise, zero values are given.

Classes of geospatial data that cannot be analysed using traditional CNN implementations include data at point locations (e.g. latitude longitude co-ordinates) and in geographic polygons (e.g. census tracts). For instance, simple neural network models do not include the ability of Gaussian Processes to model spatial dependencies through covariance functions. Here, we want to discuss some existing approaches that aim to include irregular spatial data in neural networks.

One-hot Encodings **One-hot encoding** constitutes the simplest approach for integrating spatial context of any form into neural networks (and, in fact, any predictive model). For spatial polygon data, we first split our observational

area into m neighbourhoods, ensuring that each polygon belongs to at least one neighbourhood. We then create m new binary predictor variables, indicating whether an observation (i.e. polygon) belongs to a neighbourhood. For point coordinates, we first define a grid of m custom size cells covering the whole observational area. We again create m new binary predictor variables, which indicate whether a point location falls into each respective grid cell. In both approaches, the new predictor variables are added to the other features. This approach has been used to improve the classification of geo-tagged images [167].

Region Embeddings **Region embeddings** exploit pre-defined spatial regions, like census tracts, city boundaries or counties to learn context-aware embeddings representing these regions. Learning is usually facilitated through neural network encoders. This approach requires access to contextual information about a given region. For example, Fu et al. [51] use local points-of-interest (POIs) to learn region embeddings and use them to help predict geo-tagged check-ins from social media data, highlighting their value for downstream applications.

Network Embeddings **Network embeddings** aim to learn vector embeddings of graphical data (i.e. nodes and edges). Geospatial data can often be represented as graphs, with for example the road system serving as edges. A successful example of this approach is described by Wang et al. [182]: the authors propose a kernel embedding on social-media data to improve location prediction.

Point Coordinate Embeddings **Point coordinate embeddings** are dedicated methods to learn representations of geospatial point coordinates (latitude, longitude). Yin et al. [198] develop GPS2Vec, an embedding approach for latitude-longitude coordinates, by enriching one-hot encodings with spatial context (e.g. tweets and images), learned through a neural network for each grid cell separately. Mai et al. [121] develop Space2Vec, another latitude-longitude embedding that does not require further context like tweets or POIs. Space2Vec transforms the input coordinates using sinusoidal functions and then reprojects them into a desired output space using an MLP.

2.2.3 Embedding Domain Expertise into Neural Networks

As a general-purpose method, neural networks do not specialise in any particular type of data. When we want to apply NNs to a domain specific problem, we are interested in integrating any potential domain knowledge we have into the learning process. This can be done in many different ways, such as with

dedicated NN architectures, as is the case for CNNs in image processing and computer vision. For example, a common approach to dedicated data preparation in natural language processing (NLP) includes text encoding. Here, we will present approaches for integrating domain expertise into NN models, focusing on five mechanisms: (1) physics-informed neural networks, (2) model selection, (3) auxiliary tasks, (4) feature learning, and (5) embedding losses. The last four of these approaches are used to integrate GIS expertise into neural networks in the technical chapters of this dissertation.

Learning with Physics and Constraints

Physics-informed neural networks, a term coined by Raissi et al. [145], describe neural network approaches that are bound to specific, pre-defined physical constraints. They were originally devised for modelling dynamic, non-linear systems like partial differential equations (PDEs), with applications in fluid dynamics or quantum mechanics. Physics-informed approaches are inspired by shortcomings of purely data-driven approaches to extrapolate beyond the vicinity of the observations. However, when working with physical processes, we might know certain behaviour of the process, dictated by the laws of physics, that can help us with long-distance extrapolations.

For example, measures taken from a harmonic oscillator follow sine-like waves. A solely data-driven neural network might learn this process for a few time steps beyond the training data, but then start to fail. However, if we integrate our domain expertise on the physical constraints the process at hand follows into the model, we can improve predictions substantially. Practically, this can be done by appending the loss function of our NN with residuals between the known differential equations and predictions. Physics-informed NNs can be designed for any (dynamic) process for which we have access to underlying laws or constraints on the data distribution.

Model Selection

Model selection describes the process of choosing an optimal model from a set of candidate models. It also includes other model design choices that can affect the model's capability to solve the problem at hand. Model selection is conducted using a specific selection criterion, for example obtained via cross-validation. The model selection criterion can reflect specific domain knowledge about our problem. Model selection is also crucial for transfer learning–the transfer of models from one task to a different task. Here, the challenge is to select a model from a set of existing models (that are trained on known tasks and data) that extrapolates best to a new task, the characteristics of which we might know.

For an example of how model selection works in practice, and how domain expertise can be integrated, let us consider the case of model selection through cross-validation. Cross-validation is a technique that assesses how well a model generalises on unseen data. Available data is split into training and testing datasets. Let us now train 10 NN models with different random initialisation on the training data. We then select the model that performs best on the unseen training data, as evaluated by our selection criterion.

There are two ways of integrating domain expertise into this approach. First, we can define a model selection criterion that incorporates a domain-specific requirement. For example, Tanevski et al. [166] deploy such a domain specific selection criterion in their modelling of endocytosis dynamics. The second way is to adapt the sampler that creates the training and testing splits. If we have knowledge of dependencies within our data, a simple random sampler might lead to a biased model selection procedure. Rather, a sampler that accounts for such structures is needed. Roberts et al. [151] provide an overview on such approaches for spatial, temporal and hierarchical data structures.

Auxiliary Task Learning

Auxiliary learning is an approach that uses multi-task learning to improve performance on a primary task. It was originally conceptualised by Suddarth and Kergosien [164]. The authors propose to give learners "hints" related to the original task throughout training to improve training speed and model performance. This can be understood as forcing the learner (e.g. a neural network) to focus its attention on certain patterns within the data, highlighted by the auxiliary objective. It also implies that the auxiliary task must provide some meaningful embedding of the primary task.

Here, our domain expertise comes into play. Assuming that we have some extended knowledge on the environment that contextualises our primary task, we can select and design auxiliary tasks that are complementary to the main task. As such, this approach has found many use cases, especially in data-rich real-world settings. Auxiliary learning is widely used and has been particularly successful in deep reinforcement learning [48]. For example, auxiliary tasks related to image segmentation and optical flow estimation can improve the learning of how to steer a wheel [73]. Recent work has also highlighted the applicability of pixel control tasks [77] and depth estimation [130].

Auxiliary tasks can be roughly split into two categories. First, auxiliary tasks that require additional data, such as when a segmentation mask is required on top of an image label. The other category includes auxiliary tasks that require no additional data, such that the auxiliary task can be constructed from the existing data. For example, optical flow and depth maps can both be

estimated from image data (though not perfectly). Later on in this dissertation, we will use the Moran's I metric in an auxiliary task learning setting to reinforce the learning of spatial structures.

Feature Learning

Feature learning, also more broadly known as representation learning, describes models that automatically construct feature representations that are valuable to a given downstream task (e.g. prediction). Traditional approaches include clustering algorithms like k-means clustering or principal component analysis (PCA). In recent years, we have seen the emergence of neural network-based approaches for this task. Most prominently, autoencoders [104] are neural networks that aim to learn efficient, low-dimensional encodings of input data.

Feature learning has been applied to many different application domains, and in the process, often integrates specific expertise from these domains. For example, Tsai et al. [173] use feature learning approaches to identify indicator species for specific ecological habitats. Lei et al. [109] learn meaningful features from vibration signals to automatically diagnose mechanical faults.

Embedding Losses

Embedding losses describe approaches in which the loss function of a model (or components thereof) is computed on an embedding of the data. This can have desirable outcomes, such as training stability and focusing on specific patterns in the data. These embeddings can reflect domain expertise. In its technical implementation, this approach is similar to physics-informed approaches that integrate constraints into the model loss. Rather than constraints, meaningful embeddings that represent domain-specific structures in the data are used.

Embedding losses have become popular in various application domains in recent years. For example, in the computer vision community, Ghafoorian et al. [61] use embedding losses to improve GAN-based lane detection. Filntisis et al. [47] use visual-semantic embedding losses to improve predictions of bodily expressed emotions. Embedding losses have shown great potential for particularly challenging visual problems, especially those involving complex spatio-temporal dynamics.

2.2.4 Discussion

In this section, we have introduced neural networks for predictive and generative modeling. In subsection 2.2.2, we present existing approaches for the modeling of geospatial data with neural networks, from CNNs to point coordinate embeddings. However, we can run into the same issues that some of the geographic metrics presented in section 2.1.4 exhibit. Some of these approaches

also require a pre-defined neighborhood intuition. For example, the kernel used for a CNN layer makes implicit assumptions on the neighborhood-level at which we assess spatial effects. This often restricts CNNs to measuring short-distance spatial effects. One-hot encodings for point locations also require the construction of artificial areal units at a pre-defined spatial scale. The neural network embeddings presented in subsection 2.2.2 on the other hand require extensive training data, providing the geographic context needed for learning meaningful representations of a given location and its surroundings. None of the approaches presented in this section provide an explicit intuition of spatial effects such as autocorrelation or heteroskedasticity, outlined in section 2.1.4. Nonetheless, as we show in subsection 2.2.3, there are several avenues to incorporate such domain expertise into neural networks.

In summary, neural networks are a powerful family of methods that might be able to help overcome some of the challenges of traditional geospatial models, outlined in section 2.1.6. And while they lack intuition for explicitly geospatial applications, they might be augmented with with metrics such as the ones presented in subsection 2.1.4, building capacities for such tasks.

2.3 Summary and Implications

In this chapter, we have introduced relevant concepts from GIS and neural networks, which we will revisit in the following technical chapters. Most importantly, we have developed an intuition for how geospatial context and dependencies can be embedded, from measures of spatial autocorrelation to neural network-based coordinate embeddings. These embeddings of geospatial information can be broadly split into two categories.

First, we have looked into *functional embeddings*. Formally, we can define an embedding vector $\mathbf{x}_{emb}$ as a function of an input $\mathbf{x}$ and spatial context $\mathbf{c}$:

$$\mathbf{x}_{emb} = f(\mathbf{x}, \mathbf{c}, \Lambda) \tag{2.38}$$

where Λ represents potential hyperparameters of the embedding function. An example for a non-parametric functional embedding is the Moran's I metric, where spatial context $\mathbf{c}$ and design parameters Λ determine the shape of the spatial weight matrix $w_{i,j}$.

Second, we have assessed parametric neural-network embeddings. Following the intuition above, we can define these as:

$$\mathbf{x}_{emb} = f(\mathbf{x}, \mathbf{c}, \Lambda, \Theta) \tag{2.39}$$

where Θ represent model parameters of the deployed neural network. An example of a parametric neural-network embedding is the Space2Vec [121] approach discussed in section 2.2.2. Therein, point coordinates $\mathbf{c}$ are first processed in a functional transform, then inputs $\mathbf{x}$ are used to learn combined, spatial contextual features through a feed-forward neural network.

In section 2.2.3, we have discussed several approaches for integrating domain expertise into neural networks. These can help to deploy our various measures of geographic context within neural networks. This is the essence of this dissertation. We combine ideas from traditional geographical analysis with modern neural network methods to build novel, powerful and explicitly spatial models.

This combined approach allows us to benefit from each components strengths and to mitigate some of their limitations. Specifically, as section 2.1.6 mentions, traditional GIS metrics are often only deployed as exploratory tools. Their combination with neural networks allows for a seamless integration into, for example, predictive modeling. Section 2.1.6 also outlines the limited capacity of traditional geospatial modeling techniques to scale to high-dimensional data-domains and to process highly complex, non-linear data. Again, neural networks can help to overcome these issues. But as section 2.2.4 outlines, neural networks on their own are not enough as they lack intuition for dealing with explicitly spatial data. And while approaches aimed at geospatial applications exist, non of these include measures of spatial effects like autocorrelation or heteroskedasticity.

Altogether, the combined approaches presented in this dissertation also allow us to tackle some of the known challenges of spatial data, as outlined in section 2.1.2. Specifically, geographic context embeddings can help us to address **dependence challenges** by encoding information on spatial autoregressive effects and semantic context for improved modelling. On the other hand, neural networks can address **computational challenges** through their superior scaling over traditional spatial modelling approaches.

Nonetheless, some limitations remain. Setting the correct parameters (e.g. queen neighborhood, kernel size) defining spatial neighborhoods is a challenge traditional GIS metrics and some neural networks like CNNs have in common. Approaches combining both concepts will still be sensitive to this and might need to be extended to account for effects at multiple spatial scales. Another trade-off when deploying neural networks is their black-box nature. Many of the traditional spatial modeling approaches presented in section 2.1.5 are based on linear models and provide a high level of interpretability. Spatially explicit models based on neural networks might thus be inapplicable for applications that require precise insight into, for example, how predictions are made.

With the motivations lined out, we are now in a position to present the main

technical contributions of this dissertation in the following four chapters. In the remainder of this dissertation, we will use both embedding types–functional and parametric neural-network embeddings–and integrate them into neural networks for generative and predictive modelling tasks. This way, we will incorporate GIS domain expertise into the models. To facilitate this, we will consider the different approaches as outlined in section 2.2.3. Chapter 3 will assess the use of the Moran's I metric for model selection. Chapter 4 will deploy the Moran's I metric for auxiliary task learning. Chapter 5 will use a parametric neural-network embedding to improve graphical neural network models. Chapter 6 will introduce a new spatio-temporal extension of the Moran's I metric and demonstrate how it can be integrated into generative models via an embedding loss.

The four topical chapters involve applications of different methods, all of which comprise either predictive or generative neural networks. Likewise, we are working with different experimental datasets for the different methods presented throughout this dissertation. We will introduce relevant methods, applications and datasets within dedicated paragraphs in each topical chapter.

Chapter 3

Local Moran's I for Model Selection in GAN Ensemble Learners

3.1 Introduction

In this chapter, we explore how generative adversarial nets (GANs) [64] can capture spatially dependent data and how we can leverage them to learn observed spatial patterns. As they preform well on visual data, in the geospatial context GANs have been used for generating satellite imagery [115]. However, geospatial data other than image data, distributed across continuous or discrete 2-dimensional space with one or more feature dimensions (e.g., tabular data with geographic coordinates), remain mostly unexplored. While previous studies have examined GAN performance in the presence of one-dimensional autocorrelation, such as temporal point processes [191] or financial time-series [103], the multi-dimensional correlation structures in geospatial point data pose a more complex challenge.

We tackle this issue by introducing *SpaceGAN*: Borrowing well established techniques from GIS, we use spatial neighbourhood contextualisation for training conditional GANs (cGAN) and optimise cGAN selection for the best representation of the inputs local spatial autocorrelation structures. We also propose a novel stopping criterion for *SpaceGAN* training, which explicitly measures the quality of the representation of observed spatial patterns. This metric is particularly useful, as GANs often fail to converge at a stable solution throughout training. Augmented data samples from *SpaceGAN* can be used for downstream tasks, even on out-of-sample geospatial locations. We show how this can be used for prediction via an ensemble learning framework. We test our approach on synthetic and real-world geospatial prediction tasks and evaluate the results using spatial cross-validation.

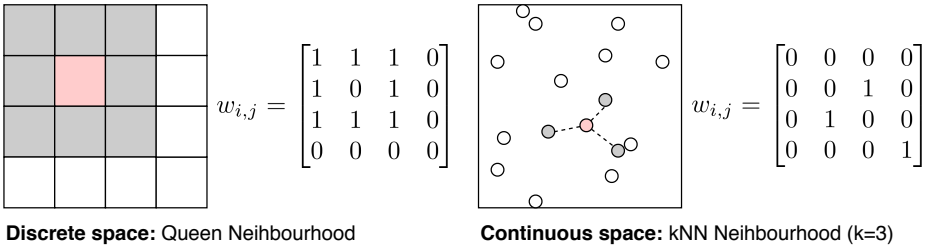


Figure 3.1: Examples of spatial weight matrices $w_{i,j}$ in the discrete and continuous case.

The main contributions of this chapter are as follows:

- We introduce *SpaceGAN*, a novel cGAN approach for geospatial data domains, using neighbourhood conditioning to capture spatial dependencies.
- *SpaceGAN* encompasses a novel stopping criterion for GAN training, measuring how well the generated data reproduces the spatial correlation patterns observed in the input.
- We introduce a novel ensemble learning method tailored to spatial prediction tasks by using *SpaceGAN* samples as training data for a set of base learners.

Across different experimental settings, we show that *SpaceGAN*-generated samples can substantially improve the performance of predictive models. As such, the results have practical implications: our proposed framework can be used to inflate low-dimensional spatial data. This allows for enhanced model training and reduced bias by compensating for a lack of training data. We thus improve generalisation performance, even when compared to existing methods for data augmentation.

The remainder of this chapter is structured as follows: section 3.2 introduces the *SpaceGAN* framework and elaborates on the technical details in respect to the cGAN architecture and spatial autocorrelation representation. In section 3.3, we evaluate *SpaceGAN* empirically using synthetic and real-world data and comparing it to existing methods for spatial data augmentation and ensemble learning. section 3.4 discusses the findings with respect to existing literature and the broader scope of the dissertation.

Algorithm 1: SpaceGAN Training and Selection

Data: p_{data} (input data), $p_{\mathbf{z}}(\mathbf{z})$ (noise prior)
Parameters: *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 $p_{data}((y_i, \mathbf{x}_i) \mid \mathcal{N}_i)$
  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:
  $\nabla_{\Theta_G} \frac{1}{L} \sum_{l=1}^{L} \Big[\log(D(G(\mathbf{z}_i \mid \mathcal{N}_i))) \Big]$
  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**
      for $i \leftarrow 1, n$ **do**
        sample noise vector $\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})$ draw
        $(\hat{y}_i, \hat{\mathbf{x}}_i) = G_k(\mathbf{z} \mid \mathcal{N}_i)$
      end
      Measure spatial autocorrelation goodness-of-fit:
      $MIE \leftarrow MIE + \sum_{i=1}^{n} |(I(y_i) - I(\hat{y}_i))|$
    end
    Average of all samples: $MIE(G_k) = \frac{1}{C} MIE$
  end
  return $G := arg\min_{G_k} MIE(G_k)$, $D := arg\min_{G_k} MIE(G_k)$
end

3.2 SpaceGAN

3.2.1 Spatial Correlation Structures

As outlined in Chapter 2.1, geospatial data exhibit inherent local interdependencies and as such an additional information layer that can be exploited. A brief example to illustrate this concept: In a typical city, when we want to estimate the price of a house, we might want to check house prices at nearby locations. If, for instance, the house is located in a rich, spatially contained neighbourhood, just knowing the price of a nearby property and without any further knowledge about the features of the house (e.g. size, age), can provide us with an informed guess. Let us formulate this intuition by first defining the i-th data point as a tuple $\mathbf{d}_i = (\mathbf{x}_i, y_i, \mathbf{c}_i)$, where $[x_i^{(1)}, ..., x_i^{(m)}] = \mathbf{x}_i \in \mathbb{R}^m$ describes a set of m features, $y_i \in \mathbb{R}$ describes the target vector and $[c_i^{(1)}, c_i^{(2)}] = \mathbf{c}_i \in \mathbb{R}^2$ describes the point coordinates in $2d$ space. The features $(\mathbf{x}, y)$ can be distributed across space randomly, or follow a—global or local—spatial process.

This can be examined by measuring the correlation of a feature with its local neighbourhood, the so called local spatial autocorrelation, which is given by the Moran's I metric [129]. While originally theorized for phenomena distributed in n-dimensional space, the concept was widely popularized in geostatistics by Luc Anselin [2]. His formalisation gives a local autocorrelation coefficient for a (numeric) vector distributed across space. While this can be applied to any vector in the feature set of $\mathbf{d}$, we will explain the concept using the target vector y here. y consists of n real-valued observations y_i referenced by an index set $N = \{1, 2, ..., n\}$. Let the neighbourhood of the spatial unit i be $\mathcal{N}_i = \{j \in N | \exists i \ni N : w_{i,j} \neq 0\}$. We can use each data points spatial coordinates $\mathbf{c}_i$ to define spatial these neighborhoods (e.g. using k-nearest-neighborhood). Let us recall the formulation of the local Moran's I metric, defined in Equation 2.2, as $I_i = I(y_i)$:

$$I_i = (n-1)\frac{y_i - \bar{y}}{\sum_{j=1, j\neq i}^{n}(y_j - \bar{y})^2} \sum_{j=1, j\neq i}^{n} w_{i,j}(y_j - \bar{y}) \tag{3.1}$$

where $\bar{y}$ represents the mean of all y_i's and $w_{i,j}$ are components of a weight matrix indicating membership of the local neighbourhood set between the observations i and j. For y_i distributed in continuous space, the weight matrix can, for example, correspond to a k-nearest-neighbourhood (kNN) with $w_{i,j} = 1$ if $j \in \mathcal{N}_i$ and $w_{i,j} = 0$ otherwise. For y_i distributed in discrete space (e.g. geospatial raster data), the weight matrix could for example correspond to a queen neighbourhood (see Figure 3.1). The Moran's I metric hence takes in a vector distributed in space and its corresponding neighbourhood structure to calculate how strongly (positively or negatively) each observation is autocorrelated with its spatial neighbourhood at any given location. Intuitively, this makes the selection of the weight matrix $w_{i,j}$, i.e. the definition of "neighbourhood", an important design choice which we have to account for when trying to augment spatial data imitating the spatial autocorrelation structures of the input. Having outlined the intuition for spatial neighbourhood and spatial autocorrelation, we now move onto data augmentation. For this, we use a popular family of generative models: GANs.

3.2.2 Spatially-conditioned GANs

Let us briefly recall GANs, as defined in Chapter 2.2.1. They deploy a Generator network (G) and a Discriminator network (D). The Generator produces synthetic data samples, while the Discriminator tries to distinguish real from fake sample. Learning is facilitated through a min-max game between both networks. Formally, let network $G(\mathbf{z}, \Theta_G)$ with parameters Θ_G, mapping noise $\mathbf{z} \sim p_z(z)$ to some feature space $\mathbf{x}$ ($G : \mathbf{z} \rightarrow \mathbf{x}$). The Discriminator, a

neural network $D(\mathbf{x}, \Theta_D)$, aims to probabilistically distinguish the synthetic input $\hat{\mathbf{x}}$ created by the Generator and real data $\mathbf{x} \sim p_{data}(\mathbf{x})$ ($D : \mathbf{x} \to [0, 1]$). Both networks compete in a min-max game, improving their performance until the maximum number of training iterations has been reached. But while GANs have been successfully applied in many areas, training them is highly non-trivial[65, 154] and remains an area of intense study [8, 181]. This is further complicated by the non-*iid* nature of geospatial data, in which learning an unconditional model would ignore inherent local dependencies. To overcome this, a sampling process taking spatial structure into account is needed, thus preserving statistical properties such as local spatial autocorrelation.

Therefore, conditional GANs (cGANs) [128] are better fit to handle context-dependent inputs, such as geospatial data. In cGANs, the input to both the generator and discriminator are augmented by a context vector $\mathbf{v}$. Typically, $\mathbf{v}$ represents a class label that we want the cGAN to generate an input for, but it can be any form of contexualisation. Formally, we can define a cGAN by including the conditional variable $\mathbf{v}$ in the original formulation so that $G : \mathbf{z} \times \mathbf{v} \to \mathbf{x}$ and $D : \mathbf{x} \times \mathbf{v} \to [0, 1]$. The minimax game between D and G is then given as $V(G, D)$:

$$\begin{aligned} \min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} \big[\log D(\mathbf{x}|\mathbf{v})\big] + \\ \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} \big[\log(1 - D(G(\mathbf{z}|\mathbf{v})))\big] \end{aligned} \tag{3.2}$$

cGANs have previously been used for spatial conditioning of image data, using pixel coordinates [74, 114]. In our formulation, this would translate to setting $\mathbf{v} = \mathbf{c}$. However, this approach is not sufficient for our problem since mere conditioning on the point coordinate alone would omit valuable information about the local neighbourhood of each point—as we show in our experiments. Instead, for each point $\mathbf{d}_i$ we are interested in capturing how its features $(\mathbf{x}_i, y_i)$ relate to those of neighbouring points $(\mathbf{x}_j, y_j) \in \mathcal{N}_i$. As such, we define the *SpaceGAN* context vector $\mathbf{v}$ of point i as $\mathbf{v} = \mathcal{N}_i$. *SpaceGAN* can thus be seen as a special instance of cGAN with spatial-neighborhood-dependent conditioning vectors.

Similarly to our intuition of spatial autocorrelation, outlined above, we assume that the features of nearby data points may offer valuable information on the point-of-interest. By conditioning each data point on all neighbouring points we allow for the learning of local patterns across the feature space. Beyond this, the versatility of constructing spatial weights $w_{i,j}$ enables experimentation with and optimisation of different spatial neighbourhood definitions. This offers a flexibility that is not provided by point coordinate conditioning.

3.2.3 Training and Selecting Generators for Spatial Data

One problem concerning GANs is that they typically fail to converge to a stable solution. To overcome this, we seek to tie training convergence to some measure of quality of the synthesised data. Accordingly, we propose to evaluate the generator performance by the faithfulness of its produced spatial patterns in relation to the true patterns observed in the input, as measured by the difference between the local spatial autocorrelation of real and synthetic data. For this, we introduce a new metric, the Mean Moran's I Error (MIE). It is defined as the mean absolute difference between the local spatial autocorrelation of the input $I(y_1, ..., y_n)$ versus that of the generated samples $I(\hat{y}_1, ..., \hat{y}_n)$:

$$MIE = \frac{1}{n} \sum_{i=1}^{n} |(I(y_i) - I(\hat{y}_i))| \tag{3.3}$$

We apply this metric for model selection (note that the MIE is not included in the model training objectives) by choosing the model with the smallest MIE, i.e. the loss of local spatial autocorrelation between real and generated $\hat{y}_1, ..., \hat{y}_n$. In our supervised learning setting, we are particularly interested in a faithful representation of the target vector and hence use $y_1, ..., y_n$ to calculate MIE. Of course, MIE can also be calculated using any other feature vector from $\mathbf{d}$. An implementation for multidimensional input is also formalised by [3] or can be achieved by averaging MIE through multiple features. It is also important to note here that a measure like MIE is somewhat domain-specific and assumes prior information on the data and task at hand, namely regarding their spatial structure. As such, it is different from general purpose goodness-of-fit measures, such as $RMSE$. To train *SpaceGAN*, we proceed as when training a normal cGAN, but include the MIE stopping-criterion. Algorithm 1 details our training procedure:

The set of user-defined hyperparameters for running *SpaceGAN Training and Selection* mainly encompass: G and D architectures, noise prior distribution $p_z(\mathbf{z})$, minibatch size L, number of training steps *tsteps*, snapshot frequency *snap*, number of samples C as well as hyper-parameters associated to the stochastic gradient optimiser (e.g., learning rate). At each training step, a minibatch of L noise inputs is sampled from the noise prior $p_z(\mathbf{z})$ and fed through the generator G to obtain synthetic data $(\hat{y}_i, \hat{\mathbf{x}}_i) = G_k(\mathbf{z} \mid \mathcal{N}_i)$. We also obtain a minibatch of size L from the real data: $(y_i, \mathbf{x}_i)$. We can then feed both real and synthetic inputs through D and update the discriminator loss by ascending stochastic gradient descent. Now another noise sample of size L is drawn from $p_z(\mathbf{z})$ and fed through G. We then feed the new synthetic data through D and update the discriminator loss by ascending stochastic gradient descent. If a snapshot training step is reached, we save store the

current discriminator and generators as D_k and G_k and compute the (averaged) MIE metric for C synthetic data samples $MIE(G_k)$. After all training steps are completed we compare the stored generators and return the one producing synthetic data with the minimum MIE error, $G := argmin_{G_k} MIE(G_k)$.

For a detailed description of the architecture and specific settings, see the experiments in Section 3 and the Appendix. Notably, our proposed stopping criterion can be seen as choosing the best member from a population of GANs acquired during training. In this way, our approach resembles "snapshot ensembling", introduced by Huang et al. [75].

3.2.4 Ganning: GAN Augmentation for Ensemble Learning

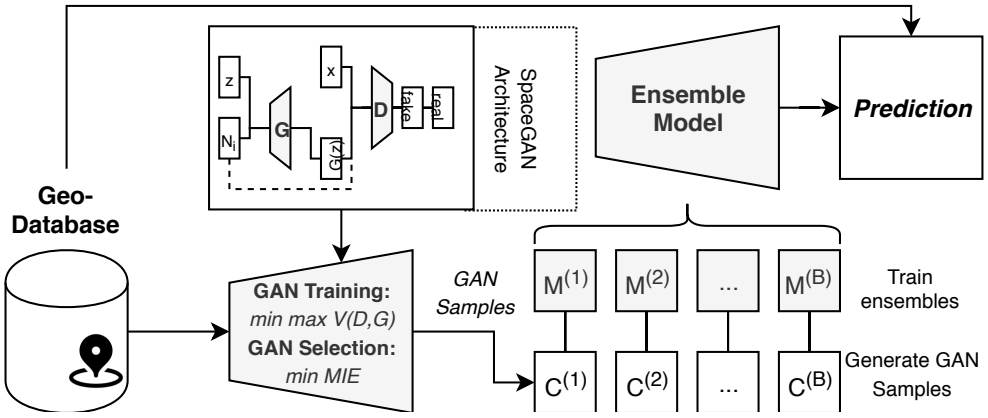


Figure 3.2: "Ganning": Re-sampling with *SpaceGAN* generated data for ensemble learning.

To highlight the practical applicability of *SpaceGAN*, we now move onto a spatial prediction (regression) task. We approach this from an ensemble learning perspective: In ensemble learning, individually "weak" base learners (e.g. Regression Trees) can be aggregated and as such outperform "strong" learners (e.g. Support Vector Machines). Traditionally, this idea includes models that make use of Bagging, Boosting or Stacking principles [43, 50]. Here, we follow Koshiyama et al. [103] and use *SpaceGAN*-generated samples as training data for the ensemble learners. This approach has not been applied to spatial data before, and since it is analogous to Bagging, we will refer to it as "Ganning" from hereon. Algorithm 2 and Figure 3.2 outline this approach.

Having fully trained and parametrised a *SpaceGAN*, we repeatedly draw B samples from the best Generator G_k (according to *MIE* selection) and train a base learner for each. After repeating this for $b = 1, ..., B$ samples we return the whole set of base models $M^{(1)}, ..., M^{(B)}$ as an ensemble. This way, we can reduce the variance of the ensemble by averaging many weakly

Algorithm 2: "Ganning" for ensemble learning

Parameters: B (number of samples), M (base learner), G
for $b \leftarrow 1, B$ **do**
  for $i \leftarrow 1, n$ **do**
    sample noise vector $\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})$ draw $(\hat{y}_i, \hat{\mathbf{x}}_i) = G_k(\mathbf{z} \mid \mathcal{N}_i)$
  end
  train base learner: $M^{(b)}(\hat{\mathbf{y}}, \mathbf{x})$
end
return ensemble $M^{(1)}, ..., M^{(B)}$

correlated predictors. Following the concept of bias-variance trade-off [43], the ensemble Mean Squared Error (MSE) decreases, particularly when low bias and high variance base learners are used. Nevertheless, there is a potential risk to this approach: Should *SpaceGAN* fail to appropriately replicate the true data generation process p_{data}, *SpaceGAN* samples might not only be more diverse, but also more "biased". Consequentially, this could lead to base learners missing obvious patters, or finding new patters that do not exist in the real data.

3.3 Experiments

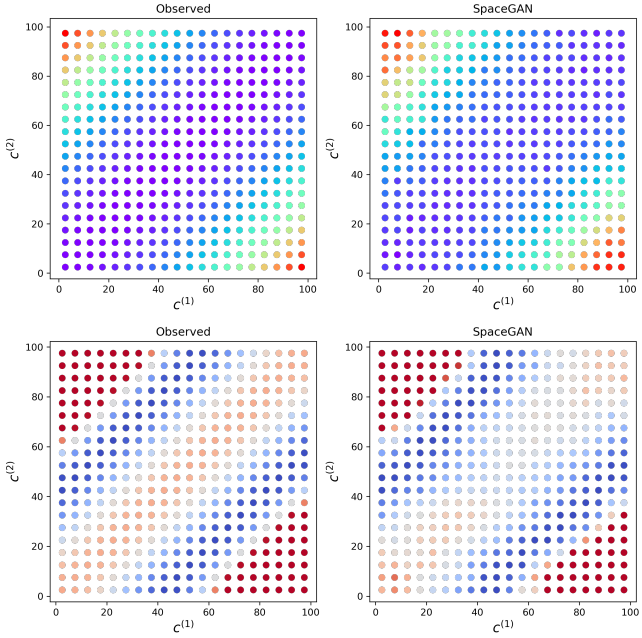
We evaluate our proposed methods in two experiments. First, we assess *SpaceGAN*'s ability to learn spatial data generating processes, including realistic representations of its internal spatial autocorrelation structure. Second, we analyse the applicability of *SpaceGAN* for spatial re-sampling usin an ensemble learning approach for spatial predictive modelling. For our experiments, we use five different datasets:

Toy 1: The data points $\mathbf{d}$ are a rectangular grid of $n = 400$ regularly distributed, synthetic point coordinates $\mathbf{c}$, a random Gaussian noise vector $\mathbf{x}$ and an outcome variable y, a simple quadratic function of the spatial coordinates $\mathbf{c}$ and random vector $\mathbf{x}$.

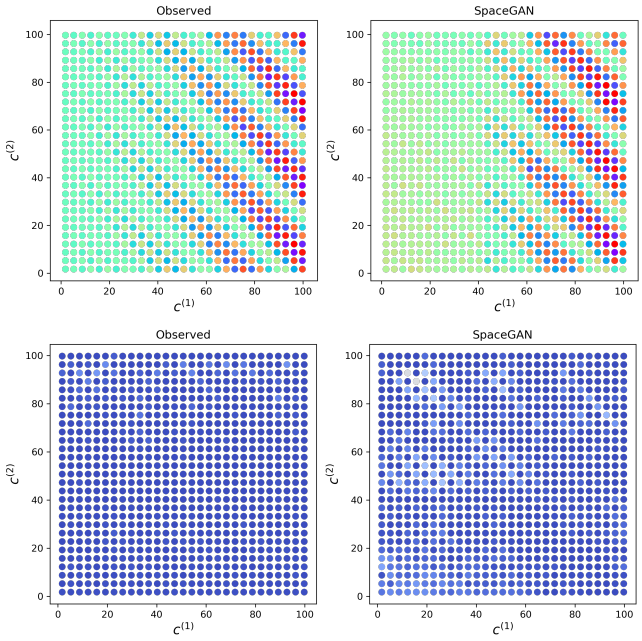
Toy 2: The data points $\mathbf{d}$ are a rectangular grid of $n = 841$ regularly distributed, synthetic point coordinates $\mathbf{c}$, a random Gaussian noise vector $\mathbf{x}$ and an outcome variable y. Here, y is a more complex combination of a π-function, a *sin*-function and a linear global pattern of $\mathbf{c}$ and $\mathbf{x}$.

California Housing: This real-world dataset describes the prices of $n = 20,640$ California houses, taken from the 1990 census. The house prices y come with point coordinates $\mathbf{c}$ and some further predictor variables $\mathbf{x}$, such as house age or number of bedrooms. The dataset was introduced by Kelley Pace and Barry [86] and, like all real-world datasets we use, is a standard example for continuous, spatially autocorrelated data.

Infant Mortality: Marshall's infant mortality in Auckland dataset, con-



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



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

Figure 3.3: *Experiment 1:* We compare the real data to *SpaceGAN* generated samples (averaged over 500 samples) showing both the target y and its Moran's I value $I(y)$. The data is synthesised out-of-sample using spatial cross-validation. For data values, lighter colors indicate higher and darker colors lower values. Values are normalized between 0 and 1.

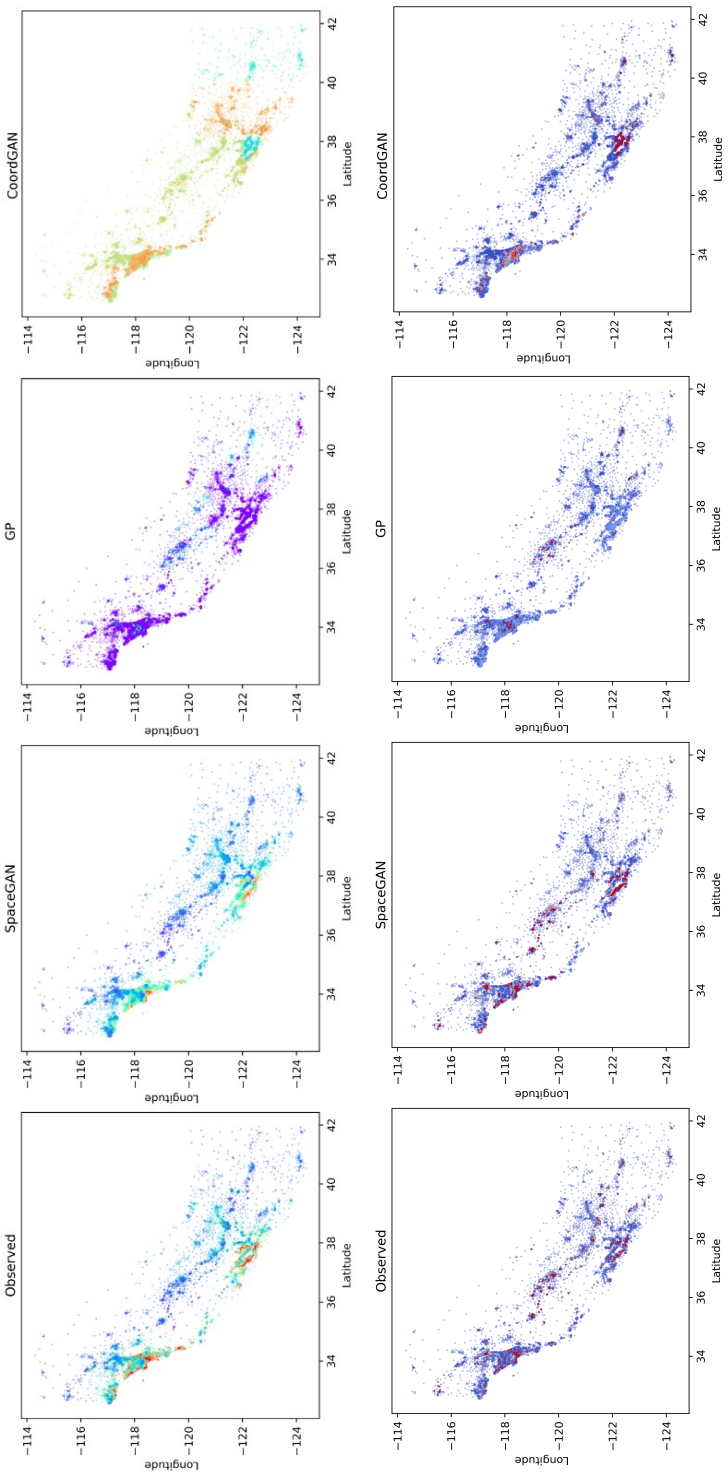


Figure 3.4: *Experiment 1:* Target vector y (*left column*) and its Moran's I value $I(y)$ (*right column*) of the observed data, *SpaceGAN*, Coord.GAN and GP generated samples for **California Housing 50**. Lighter colors indicate higher and darker colors lower values. Values are normalized between 0 and 1. Again, the data is averaged over 500 samples and synthesised out-of-sample using spatial cross-validation.

taining $n = 167$ observations, with dependent variable y infant deaths, point coordinates $\mathbf{c}$ and infant population $\mathbf{x}$ [122].

Election 1980: The turnout y of the 1980 U.S. presidential election across $n = 3107$ counties with point coordinates c and predictors (education, income p/c, homeownership) $\mathbf{x}$ [86].

More details and summary statistics of all experimental datasets can be found in the Appendix. All our experiments are conducted using 10-fold spatial cross-validation [141]. The goal of spatial cross-validation is to check for generalisability of spatial models and to avoid overfitting. In a naive cross-validation setting with spatial data, this can occur when training and test points are spatially to close. Assuming some spatial dependency between nearby points, this would roughly relate to training on the test set. Hence, we need to create a so-called buffer area around the test set within which we remove all data points from the training set. Assuming a set of data points $\mathcal{D} = (\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_n)$, we first create k spatially coherent test sets. In our case, we do this by slicing through each of the two dimensions of the coordinate space $\mathbf{c}$ five times with equal binning, thus creating 10 folds of the same width. This leaves us with a set of 10 test sets $\mathcal{D}_{test} = (\mathcal{D}^{(1)}_{test}, \mathcal{D}^{(2)}_{test}, ..., \mathcal{D}^{(10)}_{test})$. We now define the training set $\mathcal{D}^{(k)}_{train}$ as all points in set $\mathcal{D}$ which are not part of the test set $\mathcal{D}^{(k)}_{test}$ and which are not neighbouring points of the test set points, thus creating a buffer area: $\mathcal{D}^{(k)}_{train} = \mathcal{D} \notin \mathcal{D}^{(k)}_{test}, \notin \mathcal{N}_{\mathcal{D}^{(k)}_{test}}$. As a quick example, for the **California Housing 50** dataset, we would define the test set, then exclude all point which are not part of the test set, but are one of the 50-nearest-neighbours of one of the test set points. The remaining, not excluded points provide the training set. While we chose to define the buffer zone according to the neighbourhood based spatial weights matrix w, other methods such as defining a deadzone area using a radius around the test set are also applicable.

Note that for the real-world datasets, we refer to **California Housing 15 / 50** as a 15 or 50-NN implementation of the spatial cross-validation; here we also use 15 / 50-NN for *SpaceGANs* neighbourhood conditioning. For both toy datasets, we use simple queen neighbourhood (see Figure 3.1), for **Infant Mortality** 10-NNs and for **Election** 15-NNs. For a description of the specific neural network architectures for *SpaceGAN* used in the different experiments, see Appendix A.

3.3.1 *Experiment 1:* Reproducing Spatial Correlation Patterns

Our first experiment evaluates *SpaceGANs* ability to not only generate data, but also its capability of reproducing observed spatial patterns. We train *SpaceGAN* for a given experimental datasets and at each spatial location $\mathbf{c}$ return 500

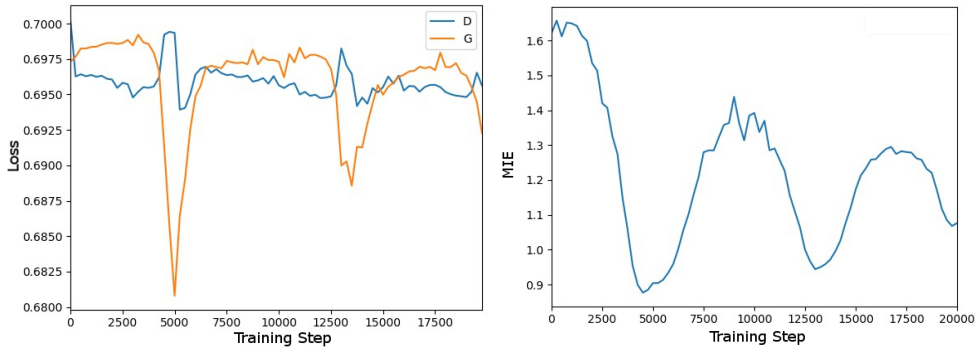


Figure 3.5: Training errors of *SpaceGAN* networks G and D (top) compared to the respective MIE criterion (bottom) across the same training process (exemplary for California Housing 50)

Table 3.1: MIE (and its standard error) between real and augmented data for *SpaceGAN*, Coord.GAN and GP implementations

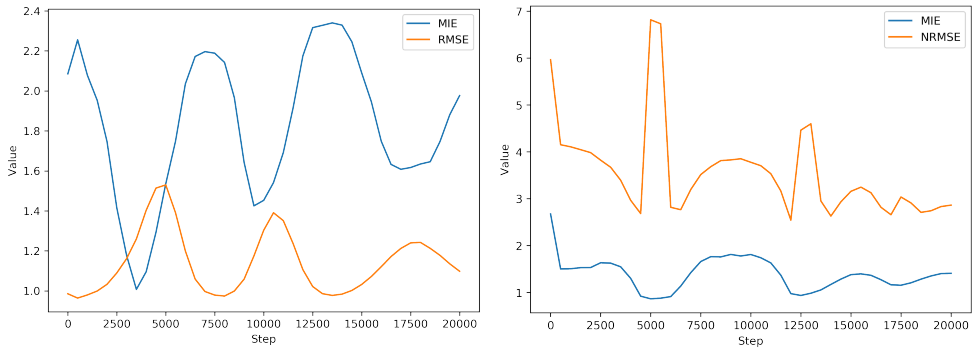
	Model		
Dataset	Coord.GAN*	GP*	SpaceGAN*
Toy 1	2.3741 (0.1716)	1.9495 (0.1750)	0.3173 (0.1791)
Toy 2	1.5728 (0.0893)	0.2195 (0.0175)	0.2141 (0.0157)
Cal.H. 15	1.3885 (0.0189)	1.9932 (0.0826)	1.1468 (0.0416)
Cal.H. 50	1.8535 (0.0249)	3.8183 (0.2072)	0.9333 (0.0288)
Inf.Mort.	0.7061 (0.1109)	1.0169 (0.0503)	0.9475 (0.0269)
Election	0.0152 (0.0068)	0.0216 (0.0080)	0.0153 (0.0073)

* - output and prediction were normalised before calculation.

samples from the generator (for examples, see Figure 3.3). Note that these results show out-of-sample extrapolations. For the dataset **Toy 1**, *SpaceGAN* is able to capture both the target vector and its spatial autocorrelation almost perfectly. In **Toy 2**, which represents a substantially more complicated pattern, we capture parts of the observed pattern seamlessly, however the spatial areas characterised by more subtle patterns are not captured fully. Nevertheless, this result shows that *SpaceGAN* also works when the spatial correlation structure is homogeneous. We now assess the real-world dataset **California Housing**. Again, *SpaceGAN* is able to capture both the target and the spatial dependencies in the data. In the real-world setting we also compare *SpaceGAN* to a Gaussian Process (GP) for data augmentation (implemented as Vanilla-GP with RBF kernel in sklearn [140]) and a point coordinate conditioned GAN (CoordGAN) [74, 114]. The last two experimental datasets, **Infant Mortality** and **Election** show *SpaceGAN* performing better than a GP but only equal to or worse than a CoordGAN.

Table 3.1 provides the MIE metric for more, extensive experiments. Note that this table shows *SpaceGAN* and CoordGAN models selected according to the minimal MIE throughout training. The GP model is trained to con-

vergence and the MIE value is computed a-posteriori. Non of the models explicitly optimises for minimal MIE, or uses it as an auxilliary loss objective. Interestingly, while we see that *SpaceGAN* doesn't always provide the lowest MIE, it still appears to learn the data generating process best, which Experiment 2 confirms. To further assess the value of MIE for model selection, we evaluate how it compares to the losses in the GAN generator G and discriminator D. We show this exemplary for the California Housing dataset: Figure 3.4 highlights how the minima in the MIE curve correspond closely to those in the generator loss. As California Housing 15 / 50 have 15 / 50-NN conditioned *SpaceGAN* models, we can also evaluate the differing lengths in the conditioning vector. We can observe that the longer conditioning vector models provides better MIE values, highlighting again the importance of the neighbourhood weight design choice.



(a) Toy 1

(b) California Housing 50

Figure 3.6: MIE and $RMSE$ evolution through a typical *SpaceGAN* training cycle, for the **Toy 1** and **California Housing 50** datasets.

3.3.2 *Experiment 2:* Data Augmentation for Predictive Modelling

Our second experiment focuses on predictive modelling. Here, we tackle regression, e.g. for the **California Housing** dataset the prediction of house prices. As outlined in section 2 and displayed in Figure 3.2, we seek to use *SpaceGAN*-generated samples in an ensemble learning setting—so called "Ganning". More specifically, we test two *SpaceGAN* configurations: First, a *SpaceGAN* using MIE as convergence criterion, second, a *SpaceGAN* using $RMSE$ for convergence, which analogously to MIE is given as

$$RMSE = \sqrt{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.4}$$

These are compared to three comparable ensemble baselines: (1) a coordin-

ate conditioned GAN with MIE selection (**CoordGAN**), (2) a GP-Bagging approach, where we draw B samples from a fully trained Gaussian Process posterior and use these to train base models for ensembling (**GP**) and (3) a traditional Bagging approach using spatial bootstrapping (**Spatial Boot**) [54]. Table 3.2 provides the out-of-sample prediction $RMSE$ values for the four approaches. We can observe that *SpaceGAN* outperforms the competitors on four of the five datasets, being only outperformed on the **Election** dataset by **Spatial Boot**.

Table 3.2: *Experiment 2:* Prediction scores ($RMSE$) and their standard errors across 10 folds for different ensemble methods with 100 samples across the different prediction tasks.

	Model (B = 100)				
Dataset	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)

Lastly, we assess *SpaceGAN* training behaviour and convergence. Figure 3.5 shows the MIE and $RMSE$ criteria throughout a typical training cycle of *SpaceGAN*, examplary for the **Toy 1** and **California Housing 50** datasets. Interestingly, for **Toy 1**, both criteria are almost antithetic, that is a local minimum for MIE convergence approximately relates to a local maximum for $RMSE$ convergence in the same training step. Moreover, $RMSE$ struggles to provide assistance for when a convergence point is reached, as it shows several local minima of approximately similar value. This point is also true for the **California Housing 50** dataset. The MIE criterion however appears to have a relatively stable minimum at the first local minimum point. Looking back to Figure 3.4, we can further see how the MIE criterion responds well to the generator network loss function, training on the **California Housing 50** dataset. We observe similar behaviour across all experimental dataset. This finding further strengthens the validity of the MIE as a model selection criterion for spatially autocorrelated data.

3.4 Discussion

We now want to contextualise our findings in relation to existing work in the field. As the academic field of machine learning advances, more and more sophisticated techniques are being developed with the aim to capture the complexity of the real world they are trying to model. This is particularly true for spatial methods, where assumptions like distributive independence or Euclidean distances restrict the performance of the most common algorithms. The motivation for this study originates from recent approaches of a more explicit modelling of spatial context within machine learning techniques. Among these are the emergence of vector embeddings for spatially distributed image data [79], the opportunities to model non-Euclidean spatial graphs using graph convolutional networks (GCNs) [39] and the modelling of spatial point processes using matrix factorisation [127]. We see *SpaceGAN* as an addition to the family of spatially explicit machine learning methods.

GAN models already have been applied to data autocorrelated in one dimensional space, e.g time series [103, 191], two dimensional space, e.g. remote sensing imagery [115, 209] and even three dimensional space, e.g. point clouds [44, 110]. However, none of this previous work used measures of local autocorrelation to improve the representation of spatial patterns. In the context of data augmentation, GANs have become a popular tool for inflating training data and increasing model robustness [22, 49, 169, 191]. However, such a method does not exist yet for $2d$ multivariate point data, where techniques such as the spatial bootstrap [23] or synthetic point generators [113, 142] are most commonly used. Spatial image data and point clouds on the other hand are often augmented using random perturbations, rotations or cropping [59, 208]. Lastly, ensemble learning is increasingly popular for spatial modelling [38], with applications ranging from forest fire susceptibility prediction [170] to class ambiguity correction in spatial classifiers [82]. Nevertheless, to our knowledge, no research has yet been conducted combining GAN augmentation and ensemble learning within a spatial data environment, highlighting the novelty of this study.

With respect to the broader scope of this dissertation, this chapter uses a popular functional embedding from the GIS, the Moran's I metric, to improve the performance of neural network models. This is facilitated through one of the avenues for integrating domain expertise into neural network models outlined in section : model selection. We empirically show performance improvements on predictive downstream tasks built on top of a GAN ensemble learner. Our approach highlights how geospatial domain expertise can be beneficial, even if it is not explicitly integrated into the model's architecture or loss function. Nonetheless, in the coming chapters we will examine more explicit geospatial

machine learning approaches.

3.5 Summary

In this chapter we introduce *SpaceGAN*, a GAN-based data augmentation method for spatial data, learning both the data generating process and its spatial dependencies through two key innovations: First, we provide a novel approach to spatially condition the GAN: Instead of conditioning on raw spatial features like point coordinates, we use the feature vectors of spatial neighbours for conditioning. This approach offers unprecedented flexibility, as it can be applied to discrete and continuous spatial data likewise. Second, we introduce a novel convergence criterion for GAN training, the MIE. This metric measures how well the generator is able to imitate the observed spatial patterns. We show that this approach succeeds at generating faithful samples in experiments using synthetic and real-world data. Turning towards predictive modelling, we propose an ensemble learning approach for spatial prediction tasks using augmented *SpaceGAN* samples as training data for an ensemble of base models. We show that this approach outperforms existing methods in ensemble learning and spatial data augmentation. We further comment on training behaviour and convergence, highlighting how the MIE metric differs–and sometimes even behaves antithetically–from existing measures such as $RMSE$. Our experiments over different application domains and spatial scales further highlight the generalisability of the method. Overall, *SpaceGAN* challenges the state-of-the-art of resampling spatial data.

In developing *SpaceGAN*, we seek to further the agenda of spatial representations in deep learning. As many real-world applications of deep learning algorithms deal with geospatial data, tools tailored to these tasks are needed–and currently sparse [148]. Nevertheless, the potential applications of *SpaceGAN* go beyond the geospatial and other 2-dimensional data domains. The use of neighbourhood structures as well as the Moran's I metric allow for the handling of data distributed in n-dimensional space. This would enable applications on more complex earth systems data such as 3-dimensional spatial data (e.g. elevation maps) or even spatio-temporal data. Some of the limitations of *SpaceGAN* include its reliance of a predefined spatial neighborhood. We illustrate the sensitivity of the method to this factor in our discussion on differences between the **California Housing 15/ 50** dataset. *SpaceGAN* also does not yet explicitly use spatial embeddings such as the Moran's I during model training. We seek to pursue this angle, e.g. through the implementation of auxilliary loss functions or multi-task training procedures, in the following chapters of this dissertation.

Chapter 4

Local Moran's I for Auxiliary Task Learning

4.1 Introduction

In this Chapter, we expand the ideas from Chapter 3 and, rather than integrating geospatial domain expertise implicitly through model selection, explicitly incorporate the local Moran's I metric into neural networks through auxiliary task learning. Practically, this is facilitated through a multi-task learning process, where the Moran's I at each data point is learned as an auxiliary task, sharing the model parameters with the main task and hence nudging the model to learn both the original data and its spatial autoregressive structure in parallel. To also account for longer-distance spatial relations, we propose a novel multi-resolution local Moran's I by gradually coarsening the input data. By providing a learner with prior knowledge on the autoregressive nature of the data we improve performance on a broad range of spatial modelling tasks.

The main contributions of this chapter can be summarised as follows:

- We propose a novel, flexible multi-resolution expansion of the Moran's I measure of local spatial autocorrelation.
- We use the traditional and multi-resolution local Moran's I as embeddings to be learned in an auxiliary learning procedure to capture both short- and long-distance spatial effects.
- We provide a practical framework to adapt our method to arbitrary neural network architectures and different supervised (predictive) and unsupervised (generative) spatial learning tasks.
- For the purpose of balancing the losses of multiple tasks in a generative modelling setting, we develop a novel auxiliary task GAN loss based on uncertainty weights [32].

We evaluate our approach, which we refer to as **SXL** (**s**patially e**x**plicit **l**earning), on both generative and predictive spatial modelling tasks, providing empirical evidence for consistent and robust performance gains across multiple synthetic and real-world experimental settings.

The remainder of this chapter is structured as follows: In section 4.2 we outline our methodology, including a brief introduction to the spatial embeddings used throughout. In section 4.3 we provide our experimental findings based on a range of synthetic and real-world tasks. In section 4.4 we discuss our results and the road ahead for geospatial machine learning. Lastly, we summarise the chapter in section 4.5.

4.2 Methodology

4.2.1 Multi-resolution Local Moran's I

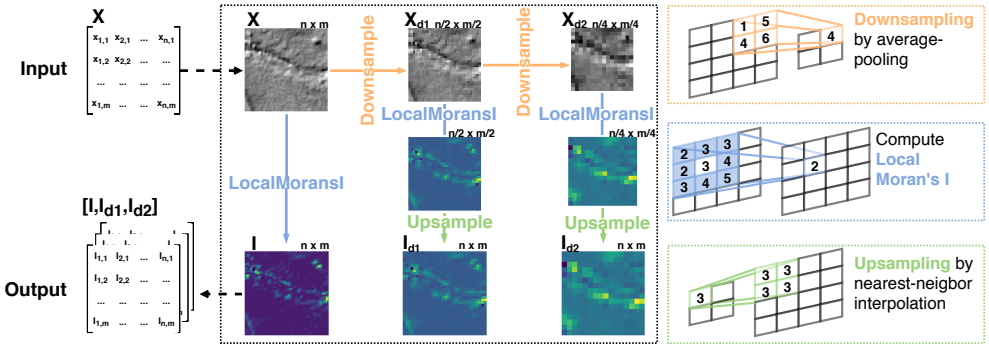


Figure 4.1: Illustration of the multi-resolution local Moran's I calculation with an example input at three different resolutions: Original input size $(n \times m)$, downsampled by factor 2 $(n/2 \times m/2)$ and downsampled by factor 4 $(n/4 \times m/4)$.

As discussed extensively in the previous chapters, working with geospatial data requires a careful assessment of and accounting for potential autoregressive effects–an intuition which neural networks traditionally do not provide. One of the most prominent measures, capturing spatial autocorrelation at the point-level, is the local Moran's I metric [2]. Local Moran's I measures the direction and extent of similarity between each observation and its local spatial neighbourhood. As such, it provides an indication for both local spatial clusters and spatial outliers.

In this chapter, we will solely work with spatial data that can be represented on a regular grid (e.g., images). Formally, let X be a 2-d spatial matrix

$$X^{n \times m} = \begin{bmatrix} x_{1,1} & \dots & x_{1,m} \\ \dots & \dots & \dots \\ x_{n,1} & \dots & x_{n,m} \end{bmatrix} \tag{4.1}$$

In vector notation, $\mathbf{x} = vec(X)$ consists of $n_x = nm$ real-valued observations

x_i, referenced by an index set $N_x = \{1, 2, ..., n_x\}$. We define the *spatial neighbourhood* of observation i to be $\mathcal{N}_{x_i} = \{j \in N_x : w_{i,j} > 0\}$. Here, $w_{i,j}$ corresponds to a binary spatial weight matrix, indicating whether any observation j is a neighbour of i. For continuous data, the creation of this matrix requires computing a pair-wise distance matrix or kd-tree, however in our case using discrete $n \times m$ matrices, this problem is trivial. Throughout this chapter, we use queen contiguity (i.e. all adjacent grid cells, including diagonals, are neighbours), but the approach generalises to arbitrary neighbourhood definitions. We revisit the computation of the local Moran's I statistic, I_i, of observation x_i with the mean over all observations $\bar{x}$ as:

$$I_i = (n_x - 1)\frac{x_i - \bar{x}}{\sum_{j=1, j\neq i}^{n_x}(x_j - \bar{x})^2} \sum_{j=1, j\neq i}^{n_x} w_{i,j}(x_j - \bar{x}) \tag{4.2}$$

Combining all local Moran's I values gives the matrix $I^{n\times m}$, of the same size as $X^{n\times m}$. I_i can take positive or negative values: a positive value suggests that a data point is similar to its neighbours, which could indicate latent cluster structure. A negative value suggests that the data point is distinctly different from neighbouring data points, which could indicate a changepoint or edge. While the Local Moran's I statistic is closely correlated to its input, their relationship can take different forms, depending on the complexity of the inputs' spatial structure.

One of the main limitations of the local Moran's I metric is its restriction to represent local spatial dependence only at the scale provided by the pre-defined neighbourhood matrix w. Thus spatial dependencies at other scales (neighborhoods) than the one provided can be lost. We also require some prior knowledge or intuition on how the most appropriate neighborhood definition to be used. This scale sensitivity of the local Moran's I is known as a common challenge in applications [45, 125, 204]. Nevertheless, to our knowledge, no alternative metric accounting for this issue exists. Here, we propose a novel, multi-resolution representation of the local Moran's I by increasingly coarsening the input data for the Moran's I computation and then upsampling the output back to the original data shape. The coarsening step here is analogous to a 2-d average pooling operation, so that our coarsened input is given as:

$$\begin{aligned} X_d^{n/a\times m/a}(i,j) = \text{mean}\{X^{n\times m}(ai+k, aj+l)\}, \\ \text{for } 0 \leq k < a \text{ and } 0 \leq l < a, \end{aligned} \tag{4.3}$$

where a gives the kernel size and the subscript d indicates *downsampling*. The coarsened spatial matrix X_d corresponds to the vector $\mathbf{x}^{(d)}$ of length

$n_{x^{(d)}}$. The coarsened local Moran's I, $I(\mathbf{x}^{(d)})$, is then computed according to Equation (2), using the spatial weight matrix $w^{x^{(d)}}$, corresponding to the new size $n/a \times m/a$ of the downsampled input. In the last step, the coarsened Moran's I is upsampled again to the original input size $n \times m$ using nearest-neighbour interpolation. This whole process can be repeated several times to compute the local Moran's I at increasingly coarse resolutions. The local Moran's I values at different resolutions can then be stacked on top of one another, much like a multi-channel image (e.g. RGB image). As such, tensors provide an ideal data structure for our metric. We illustrate this with an example in Figure 4.1.

4.2.2 Auxiliary Learning of Spatial Autoregressive Structures

Auxiliary task learning shares the benefits of multi-task learning: auxiliary tasks hint at specific patterns in the data for the model to focus attention on. Further, they introduce a representation bias, whereas the model prefers latent representations of the data that work for both primary and auxiliary tasks, thus helping with generalisation. Lastly, auxiliary tasks can work as regularisers by introducing inductive bias and decreasing the risk of overfitting the model.

Here, we want to use the local Moran's I embedding and our newly introduced multi-resolution Moran's I as auxiliary tasks. The main motivation for any auxiliary tasks is "relatedness" to the primary task: spatial theory characterises a spatial pattern as a reflection of underlying spatial processes. Accordingly, Chou [29] concludes that "[...] the capability of generalising and quantifying spatial patterns is a prerequisite to understanding the complicated processes governing the distribution of spatial phenomena."–explicitly mentioning the power of the Moran's I metric to capture these effects. This statement can be translated directly into a learning algorithm, where the learning of a spatial pattern is constrained by a simultaneous learning of the underlying spatial process.

Together with the well documented success of spatial auxiliary tasks in computer vision, this makes auxiliary tasks based on the local Moran's I well motivated by both spatial theory and machine learning research. Recent research further highlights the importance of learning at multiple resolutions to support a comprehensive understanding of spatial processes [138, 165, 193]. Lastly, the local Moran's I (and the multi-resolution variant) can be constructed for any numerical input, and can thus be seamlessly integrated in a broad range of learning settings and with arbitrary neural network architectures. This requires adding a new prediction head for each auxiliary task to the model. A neural network with one primary and three auxiliary tasks is comprised of

a set of shared neural network layers (used for all tasks) and four different prediction heads–task-specific neural network layers that predict the outcomes of each task (this is also pictured in Figure 4.2).

With our experiments, we focus on two distinct settings: generative spatial modelling using GANs [64], and predictive spatial modelling in the form of spatial interpolation. To outline the application of our proposed auxiliary task approach, we introduce the GAN example in detail–the predictive modelling formulation follows from this straightforward. To briefly recap, GANs are a family of generative models comprised of two neural networks, a Generator **G** that produces fake data and a Discriminator **D** that seeks to distinguish between real and fake data. These two networks are agents in a two-player game, where **G** learns to produce synthetic data samples that are faithful to the true data generating process, and **D** learns to separate real from fake samples, thus pushing **G** to produce increasingly realistic synthetic data. We also need to briefly recap the GAN loss functions here, as we modify it later on in this chapter. It is given as given as:

$$\begin{aligned}\min_G \max_D \mathcal{L}_{GAN}(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\big[\log D(\mathbf{x})\big] + \\ \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}\big[\log(1-D(G(\mathbf{z})))\big],\end{aligned} \tag{4.4}$$

consisting of the Discriminator and Generator losses

$$\mathcal{L}^{(D)}_{GAN} = \max_D[\log(D(\mathbf{x})) + \log(1-D(G(\mathbf{z})))], \tag{4.5}$$

$$\mathcal{L}^{(G)}_{GAN} = \min_G[\log(D(\mathbf{x})) + \log(1-D(G(\mathbf{z})))]. \tag{4.6}$$

Our auxiliary task approach augments the Discriminator with a loss based on the Moran's I embeddings of the real and the fake data:

$$\mathcal{L}^{(D)}_{AT} = \max_D[\log(D(I(\mathbf{x}))) + \log(1-D(I(G(\mathbf{z}))))], \tag{4.7}$$

so that the composite loss for N auxiliary tasks (single- or multi-resolution) is given as:

$$\begin{aligned}\min_G \max_D \mathcal{L}_{MRES-MAT}(D,G) = \mathcal{L}_{GAN}(D,G) + \\ \lambda(\mathcal{L}^{(D)}_{AT_1} + \cdots + \mathcal{L}^{(D)}_{AT_N}).\end{aligned} \tag{4.8}$$

Both loss functions use a customary weight hyper-parameter λ for the auxiliary losses. Alternatively, we could fit separate weights for each auxiliary task. The approaches are further illustrated in Figure 4.2. Integrating the auxiliary tasks into predictive models for spatial interpolation is more straightforward: We simply let a regressor f predict the (multi-resolution) local Moran's I of

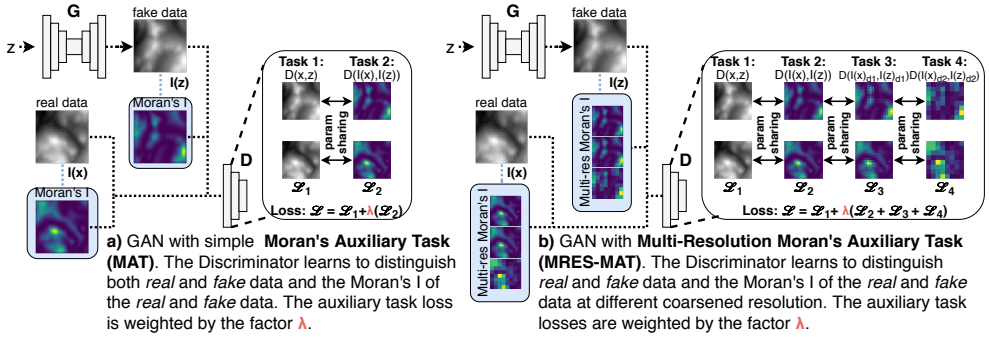


Figure 4.2: GAN with spatially explicit auxiliary tasks, using the Moran's I (**a**) and multi-resolution Moran's I (**b**) embeddings in the Discriminator. The discriminator architectures comprise the shared and task-specific layers of the auxiliary learning framework.

the output, $I(y) \sim f(I(x))$, simultaneously with the main task $y \sim f(x)$.

4.2.3 Loss Balancing Using Task Uncertainty

In addition to the hard loss weight λ outlined above, we also test a setting for automatically learning the loss weight of main and auxiliary tasks. For this, we follow the approach conceptualised by Cipolla et al. [32] and use each task's homoskedastic uncertainty to inform the loss weight. To formalise this, Cipolla et al. [32] define a probabilistic multi-task regression problem with N tasks and likelihood

$$p(\mathbf{y}_1, \dots, \mathbf{y}_N | f(\mathbf{x})) = p(\mathbf{y}_1 | f(\mathbf{x})) \dots p(\mathbf{y}_N | f(\mathbf{x})), \tag{4.9}$$

Here, $\mathbf{y}_1, \dots, \mathbf{y}_N$ are the main and auxiliary model outputs and $\mathbf{x}$ is the model input. Using the maximum likelihood method, the minimisation objective of the multi-task regression is $\min \mathcal{L}(\sigma_1, \dots, \sigma_N)$:

$$\begin{aligned} &= -\log p(\mathbf{y}_1, ..., \mathbf{y}_N | f(\mathbf{x})) \\ &= \frac{1}{2\sigma_1^2}\mathcal{L}_1 + \dots + \frac{1}{2\sigma_N^2}\mathcal{L}_N + \sum_{i=1}^{N} \log \sigma_i, \end{aligned} \tag{4.10}$$

Here, $\sigma_1, \dots, \sigma_N$ are the model noise parameters. Minimising this objective can be interpreted as learning the relative weight of each task's contribution to the composite loss. The noise is kept from decreasing infinitely by the last term of the loss, which serves as a regulariser. This loss constitutes the objective we use for our predictive modelling task. However, no approach for uncertainty weighted auxiliary task GANs exists. We hence propose an extension to the auxiliary GAN loss outlined in the previous section: Instead of using a fixed auxiliary task weight λ, we augment both main and auxiliary discriminator losses with uncertainty weights, so that

$$\min_G \max_D \mathcal{L}_{MRES-MAT}(D,G) = \mathcal{L}_{GAN}^{(G)} + \\ (\frac{1}{2\sigma_1^2}\mathcal{L}_{AT_1}^{(D)} + \cdots + \frac{1}{2\sigma_N^2}\mathcal{L}_{AT_N}^{(D)} + \sum_{i=1}^{N} \log \sigma_i). \tag{4.11}$$

This constitutes the first adaption of the uncertainty task balancing principles to the multi-task GAN family. In the following experiments, we report results from both fixed weight parameters λ and uncertainty weights. Throughout rigorous experiments, we find weights of $\lambda = [0.1, 0.01]$ to be particularly helpful and hence use these throughout our experiments. We use auxiliary learning settings with hard-parameter sharing, where the top layers of the respective models are task-specific. For the GAN experiment, just the last layer is task-specific, and for the interpolation experiment the last two layers are task specific.

Having outlined our approach for GANs in detail, the adaptation of **SXL** to predictive spatial models is trivial: Just as with the GAN discriminator, a second prediction head is added to the predictive model (e.g. a neural network regressor for spatial interpolation), aiming to predict the (multi-resolution) local Moran's I embedding of the output in parallel to the main task. The losses of both tasks are then combined the same way as for the GAN discriminator, using either a hard weighting parameter or uncertainty weights, as outlined in Equation 4.10.

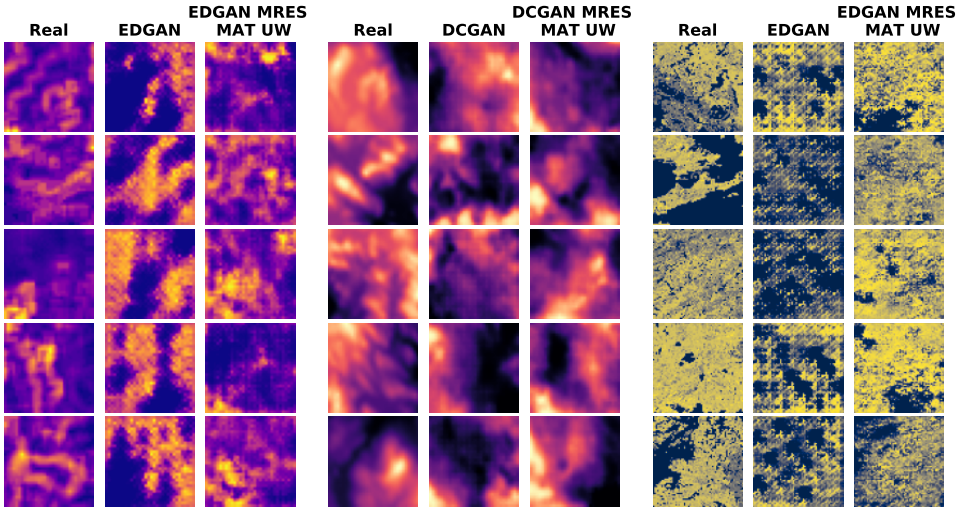
4.3 Experiments

4.3.1 *Experiment 1:* Generative Spatial Modelling

Model	Toy	PetrelGrid	DEM	TreeCanopy
GAN [64]	0.0934	0.4106	**0.1120**	0.1138
GAN-MAT UW	0.1077	0.4860	0.1814	0.1132
GAN-MRES-MAT UW	**0.0917**	**0.4014**	0.1180	**0.1038**
DCGAN [144]	0.1534	0.2993	0.0591	0.0654
DCGAN-MAT UW	0.2319	0.3049	**0.0591**	0.1009
DCGAN-MRES-MAT UW	**0.0938**	**0.2793**	0.0612	**0.0635**
EDGAN [209]	0.0269	**0.2909**	0.0499	0.0322
EDGAN-MAT UW	0.0276	0.3061	0.0481	0.0316
EDGAN-MRES-MAT UW	**0.0241**	0.2971	**0.0469**	**0.0314**

Table 4.1: Test MMD scores (lower is better) of different GAN configurations. We compare synthetic samples from these generators to held-out test data to compute the scores. Shown are models trained with uncertainty weighted auxiliary tasks.

Selection & Evaluation: In our first experiment, we want to examine whether our proposed method can improve the learning of spatial data gener-



(a) *PetrelGrid*: Our *MRES MAT* model produces synthetic data with smoother edges compared to the baseline. We can further see that *MRES MAT* is less prone to producing checkerboard artifacts.

(b) *DEM*: Again, we can observe that our *MRES MAT* model reduces checkerboard artifacts, though the effect is smaller and harder to spot in this example.

(c) *TreeCanopy*: We observe that the baseline model exhibits mode collapse, where a certain pattern is reproduced in every generated sample. This effect is mitigated by our *MRES MAT* model.

Figure 4.3: Example images highlighting the positive effects of *MAT* / *MRES MAT* on different GAN architectures, across different datasets.

Model	**Toy**	**PetrelGrid**	**DEM**	**TreeCanopy**
GAN [64]	0.0934	0.4106	0.1120	0.1138
GAN-MAT $\lambda = 0.1$	0.0994	0.4319	0.1028	0.1140
GAN-MAT $\lambda = 0.01$	0.1125	**0.3965**	0.1034	**0.0972**
GAN-MRES-MAT $\lambda = 0.1$	**0.0922**	0.4394	0.1086	0.1221
GAN-MRES-MAT $\lambda = 0.01$	0.1133	0.3989	**0.0983**	0.1026
DCGAN [7]	0.1534	0.2993	0.0591	0.0654
DCGAN-MAT $\lambda = 0.1$	0.2360	0.2858	**0.0569**	0.0692
DCGAN-MAT $\lambda = 0.01$	**0.1494**	0.2977	0.0578	**0.0596**
DCGAN-MRES-MAT $\lambda = 0.1$	0.2147	**0.2828**	0.0590	0.0602
DCGAN-MRES-MAT $\lambda = 0.01$	0.2154	0.2868	0.0576	0.0640
EDGAN [209]	0.0269	0.2909	0.0499	0.0322
EDGAN-MAT $\lambda = 0.1$	0.0289	0.2973	0.0460	**0.0316**
EDGAN-MAT $\lambda = 0.01$	0.0269	0.3012	0.0467	0.0319
EDGAN-MRES-MAT $\lambda = 0.1$	0.0253	**0.2676**	0.0482	0.0317
EDGAN-MRES-MAT $\lambda = 0.01$	**0.0247**	0.2898	**0.0438**	0.0321

Table 4.2: Test MMD scores (lower is better) of different GAN architectures. We compare synthetic samples from these generators to held-out test data to compute the scores. Shown are models trained with hard auxiliary task weights λ.

ating processes. We generate synthetic data from several types of GANs, with and without including the Moran's I auxiliary tasks, and compare how faithful

the generated samples are compared to the true distribution of samples. To assess model quality, we use the Maximum Mean Discrepancy (MMD) metric [21], a distance measure between distributions based on mean embeddings of the features. For data distributions P and Q, the MMD is defined as $MMD(P,Q) = ||\mu_P - \mu_Q||_{\mathbb{R}^d}$. The empirical MMD for random variables x_i and y_i of length n is given as

$$\begin{aligned} \widehat{MMD}^2 = \frac{1}{n(n-1)} \sum_{i \neq j} k(x_i, x_j) + \\ \frac{1}{n(n-1)} \sum_{i \neq j} k(y_i, y_j) - \frac{2}{n^2} \sum_{i,j} k(x_i, y_j), \end{aligned} \tag{4.12}$$

where $k : \mathcal{X} \times \mathcal{X}$ represents a positive-definite kernel—in our case a radial basis function (RBF) kernel. The more similar the data distributions P and Q are, the closer the MMD metric gets to 0. A lower MMD score between samples of real and synthetic data thus indicates higher quality of the synthetic samples. As GAN training is notoriously difficult and prone to mode collapse and other issues, we opt for the following training and selection procedure to ensure our findings are robust: For each architecture and training strategy combination (e.g. EDGAN **MRES MAT UW**) we train ten cycles of GANs. For each cycle, we save the generator which optimises the MMD metric on a held-out validation set (separate from the training data and from the test data used for evaluation), rather than choosing the final model after all training iterations. We then choose the one of these ten generators that optimises the MMD metric on the validation set, and finally evaluate the MMD score of that generator on a separate, held-out test set. In short, this training process allows each architecture and training strategy combination ten cycles to train the best possible generator (as measured on the validation set), which is then evaluated on the test set and compared to all other combinations of architecture and training strategy.

Data: As our task is different from the experiments presented in Chapter 3, we select four new datasets for our experiments: *(1)* A toy dataset of 7000 32×32 tiles with a Gaussian peak and a Gaussian dip, where the position of the dip mirrors the position of the peak. *(2)* The *PetrelGrid* seabed relief dataset [111], processed into a grid of 195 32×32 tiles. *(3)* Digital Elevation Model (*DEM*) data of the area surrounding Lake Sunapee (NH, USA), as found in the `elevatr` *R* package [1], processed into a grid of 1156 32×32 tiles. *(4) Tree canopy* data of the University of Maryland's "Global Tree Change" project [67], processed into a grid of 1800 64×64 tiles. These datasets are chosen to represent a range of different geospatial patterns occurring in real-world physical environments and relate to important modelling challenges in the

[1] See: `https://github.com/jhollist/elevatr`

earth sciences, ecology, or geography. For more information on the data used, including summary statistics, please refer to Appendix B.

Benchmark Models: The modularity of our proposed auxiliary task learning method allows us to test it on a range of different GAN models. We chose the original GAN implementation, denoted here as *GAN* [64], consisting of a Generator with four hidden linear layers, supported by Leaky ReLU and $1d$ BatchNorm layers. The Discriminator has two hidden linear layers supported by Leaky ReLU layers and one linear task-specific layer. *DCGAN* [144] consists of a Generator with a linear initialization layer, followed by three hidden (de-)convolutional layers, supported by ReLU and $2d$ BatchNorm layers. The Discriminator contains two convolutional layers supported by Leaky ReLU and $2d$ BatchNorm layers, followed by one task-specific convolutional layer with a final linear transformation.

Lastly, Encoder-Decoder GAN (*EDGAN*),recently proposed by Zhu et al. [209] and explicitly designed for geospatial applications, consists of an Encoder-Decoder Generator, where the Encoder contains three convolutional layers, supported by Leaky ReLU and $2d$ BatchNorm layers and the Decoder contains three (de-)convolutional layers supported by ReLU layers. The Discriminator has five hidden convolutional layers suppoerted by Leaky ReLU and $2d$ BatchNorm layers, followed by a last, task-specific convolutional layer.

All models are optimised using the same, traditional GAN objective lined out in the previous section. We test all benchmark models with the single- and multi-resolution Moran's I auxiliary task (**MAT** / **MRES-MAT**) as well as with fixed (λ) and uncertainty (**UW**) based task weights. For more details on the experimental setup, including hyperparameters, please refer to Appendix B.

Model Training: All models are trained using the binary cross entropy criterion to compute losses. Optimization through backpropagation is conducted using the Adam algorithm with a learning rate of 0.001 and β values of $[0.5, 0.999]$. Experiments with the *Toy* dataset run for 40 epochs, with the *PetrelGrid* dataset for 500 epochs, with the *DEM* dataset for 100 epochs and with the *TreeCanopy* dataset for 100 epochs. For more details on training, please again refer to Appendix B.

Findings: Table 4.1 and 4.2 show the MMD scores of generators selected according to the strategy outlined in the *Evaluation & Selection* paragraph. Table 4.1 highlights results from the uncertainty weighting strategy, and Table 4.3 from the hard loss weights λ. We can see that for both strategies the auxiliary task settings improve performance for most experiments, agnostic of the underlying GAN architecture, by usually 3-10%. We believe the auxiliary tasks support the learning process in two ways: (1) GAN Discriminators are known to exhibit spare capacity (i.e., they can be too powerful), which can

cause them to over-specialise, leading to worse generalisation performance [68]—thus adding a second, closely correlated task should not pose a problem but might help prevent over-specialisation in the bottom Discriminator layers. (2) The losses stemming from the auxiliary tasks have a regularising effect throughout training, further preventing Discriminator over-specialisation.

This leads to several beneficial effects, some even visually apparent, as highlighted in Figure 4.3. Figure 4.3a shows how generators trained with **MRES MAT** appear to be better at smoothing (but not over=smoothing) spatial artifacts—residual, noisy spatial patterns and hard edges introduced by the model—compared to the same GAN backbone trained without our auxiliary task. Figure 4.3c shows two EDGANs, one trained with and one trained without **MRES-MAT**. We observe that the model trained without the auxiliary task exhibits "mode collapse", a phenomenon common with GANs where a Generator always produces the same image or some parts of the image are always the same, while the Generator including the auxiliary task does not exhibit this behavior. It is important to note that these examples are not cherry-picked but represent a pattern we can observe throughout all our experiments: the auxiliary tasks consistently improved model performance.

However, the optimal setting for applying the auxiliary task appears to vary, both in terms of the task weighting strategy and using the simple **MAT** versus the **MRES-MAT**. For example, while for the Vanilla GAN architecture, single-resolution **MAT** with hard task weights seems superior, both DCGAN and EDGAN appear to benefit especially from **MRES-MAT** with uncertainty weights. We also observe cases where adding the auxiliary tasks with a particular weighting strategy massively increases the MMD score (e.g. Toy dataset, GAN-**MAT UW**). This can happen when the auxiliary task "overpowers" the main task, causing the Generator to produce synthetic Moran's I embeddings. This further justifies the generator selection strategy employed throughout our experiments. Overall, our GAN experiments highlight that while our auxiliary learning framework SXL generally improves performance, the best configuration (with respect to **MAT** versus **MRES MAT** and weighting strategy) appears to be application and data dependent.

Task weighting: Tables 1 and 2 show that models employing our **MAT** / **MRES MAT** auxiliary tasks can reliably produce generators that outperform naively trained models. Uncertainty weight models with **MRES MAT** represent the "winning" Generator in 9 of 12 cases. For the hard loss weights, we are always able to find a combination of λ and **MAT** / **MRES MAT** that outperforms naive training. We thus conclude that while training strategies appear to be highly data and model dependent, one can find a performance-increasing setting in almost all cases. Here, the **MRES MAT UW** strategy seems to be the safest bet, as it does not require further, manual weight parameter

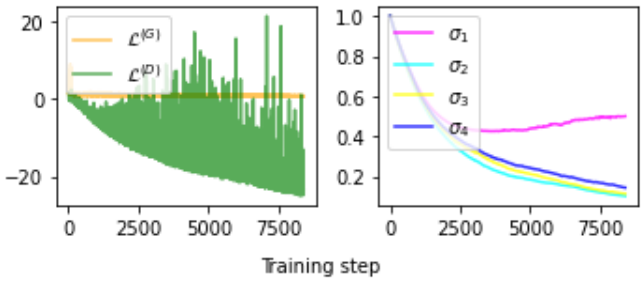


Figure 4.4: Training progression of generator and discriminator losses $(\mathcal{L}^{(G)}, \mathcal{L}^{(D)})$, main task uncertainty σ_1 and auxiliary task uncertainties $\sigma_2, \sigma_3, \sigma_4$ throughout an example training cycle of EDGAN **MRES MAT UW** using *DEM* training data. The x-axis shows the respective loss and task uncertainty values, the y-axis shows training steps. We can see that the main task uncertainty (pink line) reaches a local minimum at around 5000 training steps, increase thereafter and stabilize at a slightly higher level, indicating that the discriminator has converged to a best possible solution.

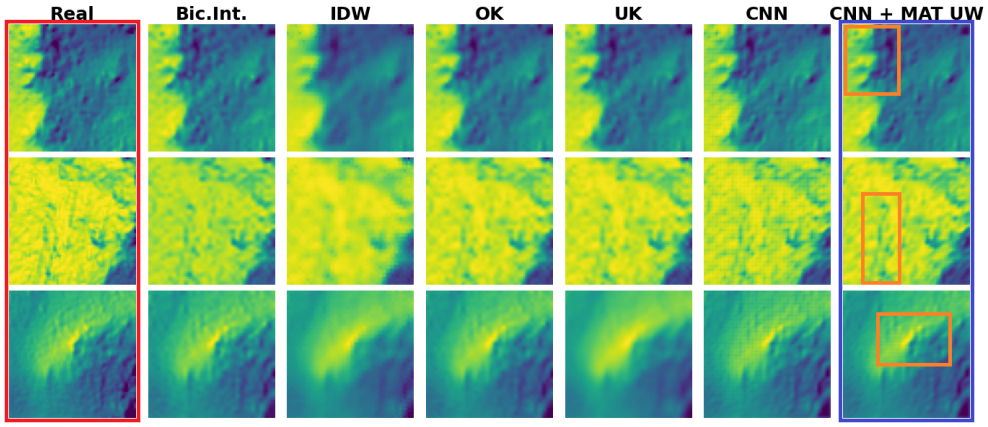


Figure 4.5: Interpolation results on samples from the test set, across the different benchmark models, presenting our CNN + **MAT UW** model. The orange boxes highlight areas where the improvement over the benchmark models becomes visually apparent.

tweaking. Figure 4.4 shows the losses and learned task uncertainties in an example training cycle of an EDGAN **MRES MAT UW** model using the *DEM* dataset. More details on all our data, model architectures and training settings can be found in the Appendix.

4.3.2 *Experiment 2:* Predictive Spatial Modelling

Evaluation: In the second experiment, we tackle spatial interpolation, that is, obtaining high-resolution spatial data from a low-resolution input. Spatial interpolation is widely used in real world applications, for example with meteorological measurements [180], air quality assessment [197] or mobile sensing [132]. It is a regression task and can be evaluated using the residual mean

squared error (RMSE) between real and predicted high-resolution output. As such, this task is comparable to image super-resolution, a popular task in computer vision. Nevertheless, spatial interpolation is particularly focused on reconstructing the spatial patterns of the output. We again train 10 models per strategy and compare their performance when no model selection is used (final model used for prediction on test set) and when model selection on a validation set is applied, saving the 10 best models, one from each run. Again, this task is different to the experiments in Chapter 3 and requires a new dataset.

Data: As a common use case in geography and ecology, we use hillshades of DEM data from the National Ecological Observatory Network (NEOS) [2] for the interpolation task. The data is pre-processed into a grid of 1674 64×64 tiles (output) and a low-res 32×32 version (input) by removing every second row and column of the image matrix. For more information on the data used, including summary statistics, please refer to AppendixB.

Benchmark Models: Spatial statistics provides a range of tools to tackle interpolation problems. Commonly used methods we focus on here are: (1) Bicubic interpolation (*BicInt*), commonly used for interpolating 2-d regular grids, (2) Inverse Distance Weighting (IDW) [157], a weighted rolling-average approach, (3) Ordinary Kriging (OK) [33], spatial interpolation closely related to Gaussian Process regression and (4) Universal Kriging (UK) [162], a generalisation of OK assuming a polynomial trend model. We compare these established methods to a simple CNN implementation with two hidden layers (5). Again, the modularity of `SXL` allows us to simply plug-in our **MAT** by having the CNN interpolate the spatial pattern and its Moran's I embedding from low to high resolution, making the last layers task-specific. In this setting, we do not use the **MRES MAT**, as our experiments show that further coarsening the already-reduced image is counterproductive. The CNN model main tasks are optimised using MSE loss, while the auxiliary tasks use $\ell 1$ loss. All CNN models consist of three convolutional layers, supported by ReLU and $2d$ BatchNorm layers. When applying the auxiliary tasks to the model, the last two convolutional layers are made task-specific. For more details on the experimental setup, please refer to Appendix B.

Model Training: All models are trained using the mean squared error (MSE) criterion to compute losses. Optimization through backpropagation is conducted using the Adam algorithm with a learning rate of 0.001 and β values of $[0.5, 0.999]$, running for 150 epochs. For more details, please refer to Appendix B.

Findings: The results of our experiments are presented in Table 4.3 and Figure 4.5. Here, the orange boxes indicate areas in which our improvements become visible at close inspection. We can again see a positive effect of

[2] See: `https://www.neonscience.org/da-viz-neon-lidar-co13flood-R`

Model / Task	RMSE $32 \rightarrow 64$ (no model selection)	(model selection)
BicInt	0.0667	–
IDW	0.0693	–
OK	0.0801	–
UK	0.0796	–
CNN	0.0678($\pm$0.0128)	0.0503($\pm$0.0008)
CNN + MAT UW	**0.0649**($\pm$**0.0119**)	**0.0496**($\pm$**0.0006**)
CNN + MAT $\lambda = 0.1$	0.0665($\pm$0.0118)	0.0516($\pm$0.0018)
CNN + MAT $\lambda = 0.01$	0.0666($\pm$0.0178)	0.0532($\pm$0.0033)

Table 4.3: Model RMSE scores and their standard deviation on held-out test data for the $32 \rightarrow 64$ interpolation task. The CNN scores are obtained by averaging over 10 runs each, once taking the final trained model (no model selection) and once selecting the best model according to the validation set (model selection).

the **MAT** on the performance of the CNN model—outperforming all other benchmarks. If no model selection is deployed, both hard loss weights and uncertainty weights produce models that outperform the naively trained CNN. **MAT UW** models provide the best average performance increase, of around 5%. If model selection is used, the **MAT UW** strategy outperforms the naive CNN by about 1.5%. Both of these performance increases are statistically significant, according to a paired t-test of the mean prediction scores. While our improvements are small, they still are noteworthy as none of the underlying neural network architectures have been changed. Rather, the improvements can be attributed solely to the introduction of the auxiliary task, confirming its usefulness over two different experiments.

Task weighting: Finding the optimal task weighting strategy appears much less tricky as for generative modelling, as the **MAT UW** strategy prevails in all interpolation experiments, whether model selection is applied or not. We can thus conclude that the **MAT UW** both consistently and significantly improves training. More details on all our data, model architectures and training settings can be found in Appendix B.

4.4 Discussion

We now want to contextualise this chapter with respect to the literature on geographic-explicit machine learning and the broader scope of this dissertation. As geographic context has proven to be of relevance in many machine learning applications [5, 12, 31, 195], core concepts from GIS and geography have gradually attracted more attention in the machine learning community. While methodologies for geospatial data and problems have previously used traditional

machine learning approaches like tree-based models [81] or ensemble learners [82], focus has recently switched more towards neural networks: for example, Mai et al. [121] and Yin et al. [198] propose a vector embedding to capture spatial context, inspired by word embeddings such as *Word2Vec* [126]. This is extensively discussed in section 2.2.2.

But while these approaches focus more on learning of representations for geospatial data, the techniques introduced in this chapter aim to integrate geospatial domain expertise directly into the learning process on a downstream tasks. Examples for such geographically explicit neural network models include the following: Zammit-Mangion et al. [202] propose to model complex spatial covariance patterns through injective warping functions, learned by a deep net. And while there exist approaches for capturing spatial autocorrelation in neural networks [206], the Moran's I metric has previously not been used explicitly for learning. More generally, geographic metrics describing spatial phenomena have barely been used in neural network models–which we believe is a missed opportunity.

In this chapter, we use auxiliary learning, an approach using multi-task learning where we are only interested in the performance of a primary task. Originally conceptualised by Suddarth and Kergosien [164], the authors propose to provide learners with secondary tasks related to the main task. This can help to improve training speed and performance of the primary task. In order for this approach to work, the auxiliary task has to provide meaningful information to the learner that synergises with the main objective. Examples of auxiliary task learning can be found deep reinforcement learning, where this approach is especially popular [48]. Steering a wheel can be improved using auxiliary tasks related to image segmentation and optical flow estimation [73]. These auxiliary tasks are already related to spatial perception, which is common when working with image data: recent work has highlighted the applicability of pixel control tasks [77] or depth estimation [130].

Auxiliary tasks have also been successfully deployed for generative modelling with GANs: Auxiliary Classifier GAN (AC-GAN) [133] extends the original GAN loss function by an auxiliary classifier to improve the fidelity of generated images. However, this approach comes with complications. As recent work by Gong et al. [62] notes, AC-GANs can lead to perfect separability–where the GAN discriminator is easily winning the two-player game against the generator, preventing efficient and balanced learning. Another use of spatial semantics in an auxiliary task GAN setting is proposed by Wang and Gupta [188]: the authors propose to generate surface normal maps, a 3D representation of the 2D image, and use these in an auxiliary task. In an explicit geospatial setting, auxiliary tasks have been used in semantic visual localisation [155] and semantic segmentation [27], in addition to dedicated spatial and spatio-temporal multi-

task frameworks [192]. Nevertheless, to the best of our knowledge, measures of spatial autocorrelation such as Moran's I have never previously been used in any kind of multi-task learning setting, be it for generative or predictive spatial modelling.

This chapter builds on the findings of chapter 3, which showed us that the Moran's I metric can be a useful heuristic for selecting geospatial neural network models. In this chapter, we take this idea a step further and directly integrate the metric into neural network models, following the intuition of auxiliary task learning. Again, we can show performance improvements on different geospatial modelling tasks. Chapters 3 and 4 combined highlight the applicability of an established tool in the GIS, Moran's I, for improving neural network models.

4.5 Summary

In this chapter, we introduce **SXL**, an auxiliary-task learning framework for geospatial data using the local Moran's I metric. Our experiments give some insight into the way the auxiliary learning mechanism works, allowing us to compare the method to related ideas in machine learning: First, as mentioned before, we believe the auxiliary tasks to have a regularising effect on the learning process, preventing models from overfitting on the primary task by forcing them to follow "spatial rules". Second, we believe the **MRES MAT** shares the intuition of moment matching, as we seek to simultaneously minimise the loss of one function at several coarsened resolutions. Third, the **MRES MAT** also shares the same goal as recent developments in visual self-attention: moving beyond the short distance spatial learning of convolutional layers and accounting for longer distance spatial effects. We also make some empirical observations with respect to the most important design choices of our models: (1) The use of **MAT** vs. **MRES MAT** appears to depend on both the input data and the model architecture used, with **MRES MAT** prevailing in most cases. (2) The optimal auxiliary task weighting strategy varies across generative and predictive modelling experiments, but in most cases uncertainty weights appear to have the edge. As **SXL** shows performance improvements over many different datasets and two different tasks (generative modeling and spatial interpolation), our experiments further highlights its generalisability to different tasks and application domains united by spatial dependence in the data.

With **SXL** we propose the use of single- and multi-resolution measures of local spatial autocorrelation for improving the learning of geospatial processes. We introduce a novel, flexible multi-resolution version of the local Moran's I statistic using coarsened inputs. We demonstrate its integration as an

auxiliary task into generative and predictive neural network models, using both hard (static) and task uncertainty (automatically learned) loss weights. We empirically show robust, consistent and significant performance gains of up to 10% for generative spatial modelling and up to 5% for predictive spatial modelling when using this strategy. We comment on the importance of the exact configuration of the auxiliary tasks, especially choosing single- versus multi-resolution auxiliary tasks and the weighting strategy for auxiliary losses. This sensitivity to the best possible setting can be seen as one of the limitations of **SXL**. Beyond, **SXL** is restricted to only one measure of spatial dependence, the local Moran's I metric. For certain applications or tasks, different metrics, such as the extensions of Moran's I presented in 2.1.4, might be better suited. In the scope of this dissertation, this study is further evidence of the importance of integrating domain expertise from GIS into neural network methods for geospatial data.

Chapter 5

Geographic Embedding Learning for Graph Neural Networks

5.1 Introduction

In this chapter, we move away from traditional neural networks, CNNs and GANs towards a new class of models. While these approaches do not have an intuition (or a limited one) to account for spatial dynamics in continuous geographic data (e.g., latitude longitude coordinates), graph neural networks (GNNs) can operate on spatial graphs obtained from such data. The recent years have seen many applications leveraging GNNs for modeling tasks in the geographic domain, such as inferring properties of a point-of-interest [210] or predicting the speed of traffic at a certain location [28].

Nonetheless, as we show in this chapter, GNNs are not necessarily sufficient for modeling complex spatial effects: spatial context can be different at each location, which may be reflected in the relationship with its spatial neighbourhood. The study of spatial context and dependencies has attracted increasing attention in the machine learning community, for example in work on spatial context embedding [121, 198].

In this chapter, we seek to merge these streams of research. We propose the positional encoder graph neural network (**PE-GNN**), a flexible approach for better encoding spatial context into GNN-based predictive models. **PE-GNN** is highly modular and can work with any GNN backbone. **PE-GNN** contains a positional encoder (PE) [121, 177], which learns a contextual embedding for point coordinates throughout training. The embedding returned by PE is then concatenated with other node features to provide the training data for the GNN operator. **PE-GNN** further predicts the Moran's I metric of spatial autocorrelation of the output as an auxiliary task in parallel to the main

objective, expanding the approach proposed in chapter 4 to continuous spatial coordinates. Lastly, we train **PE-GNN** by constructing a novel training graph, based on k-nearest-neighbourhood, from a randomly sampled batch of points at each training step. This forces PE to learn generalisable features, as the same point coordinate might have different spatial context (neighbours) at different training steps. Similarly, this training approach leads us to compute a "shuffled" Moran's I, implicitly nudging the model to learn a general representation of spatial autocorrelation which works across varying neighbour sets. Over a range of spatial regression tasks, we show that **PE-GNN** improves performance of different GNN backbones in most settings, except when very little training data is available.

The contributions of our study are as follows:

- We propose **PE-GNN**, a novel GNN architecture including a positional encoder learning spatial context embeddings for each point coordinate to improve predictions.
- We propose a novel way of training the positional encoder, distinct from Mai et al. [121]. We use the output of PE concatenated with other node features to predict an outcome variable. PE learns through backpropagation on the main regression loss. Training PE thus takes into account not only the eventual variable of interest, but also further contextual information at the current location–and its relation to other points.
- We expand the Moran's I auxiliary task learning framework proposed in chapter 4 for continuous spatial data.
- Our training strategy involves the creation of a new training graph at each training step from the current, random point batch. This enables learning of a more generalisable PE embedding and allows computation of a "shuffled" Moran's I, which accounts for different neighbours at different training steps, thus overcoming the well-known scale sensitivity of Moran's I.
- To the best of our knowledge, **PE-GNN** is the first GNN based approach that is competitive with Gaussian Processes on pure spatial interpolation tasks, i.e., predicting a (continuous) output based solely on spatial coordinates, as well as substantially improving GNN performance on both spatial regression and spatial interpolation tasks.

The remainder of this chapter is structured as follows: Section 5.2 introduces our methodology, including a detailed description of GNNs and neural network-based geographic location embeddings. Section 5.3 presents our experiments on

a range of real-world datasets, highlighting the benefits of **PE-GNN**. Section 5.4 discusses our work with respect to existing literature and the broader scope of this dissertation. Section 5.5 summarises the chapter.

5.2 Methodology

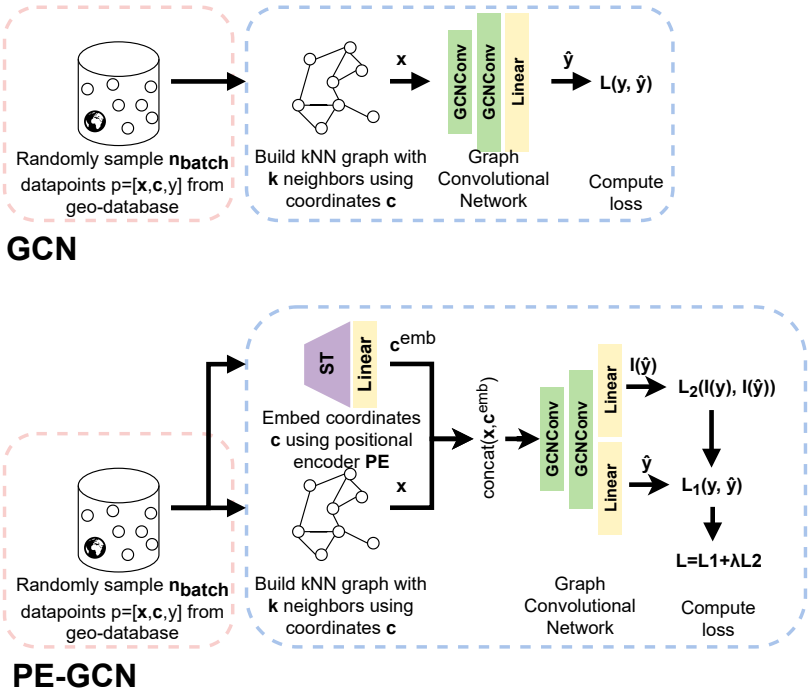


Figure 5.1: **PE-GCN** compared to the **GCN** baseline: **PE-GCN** contains a (1) positional encoder network, learning a spatial context embedding throughout training which is concatenated with node-level features and (2) an auxiliary learner, predicting the spatial autocorrelation of the outcome variable simultaneously to the main regression task.

5.2.1 Graph Neural Networks with Geographic Data

We elaborate on our method using Graph Convolutional Networks (GCNs) as an example backbone for our novel PE-GNN approach. Let us first define a datapoint $p_i = \{y_i, \mathbf{x_i}, \mathbf{c_i}\}$, where y_i is a continuous target variable (scalar), $\mathbf{x_i}$ is a vector of predictive features and $\mathbf{c_i}$ is a vector of point coordinates, mapping the datapoint into geographic space (e.g., latitude and longitude). We use the great-circle distance $d_{ij} = haversin(\mathbf{c_i}, \mathbf{c_j})$ between point coordinates to create a graph of all points $P = \{p_1, \ldots, p_n\}$ in the set. This can be done either by using a k-nearest-neighbour approach or by drawing a boundary radius around each coordinate to define each point's neighbourhood and create an adjacency matrix $\mathbf{A}$. The graph $G = (V, E)$ consists of a set of vertices (or nodes) $V = \{v_1, \ldots, v_n\}$ and a set of edges $E = \{e_1, \ldots, e_m\}$ as assigned

by $\mathbf{A}$. Each vertex $i \in V$ has respective node features $\mathbf{x_i}$ and target variable y_i. While the adjacency matrix $\mathbf{A}$ usually comes as a binary matrix, one can account for different distances between nodes and use point distances d_{ij} or kernel transformations thereof [6] to construct $\mathbf{A}$. Now, let $\mathbf{D}$ define the degree matrix and $\mathbf{I}$ the identity matrix of our graph G. We can then define the normalised adjacency matrix $\bar{\mathbf{A}}$ as:

$$\bar{\mathbf{A}} = \mathbf{D}^{-1/2}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-1/2} \tag{5.1}$$

As proposed by Kipf and Welling [90], a GCN layer can now be defined as

$$\mathbf{H}^{(l)} = \sigma(\bar{\mathbf{A}}\mathbf{H}^{(l-1)}\mathbf{\Theta}^{(l)}), l = 1, \ldots, L \tag{5.2}$$

where σ describes an activation function (e.g., ReLU) and $\mathbf{\Theta}^{(l)}$ is a weight matrix parameterising GCN layer l. The input for the first GCN layer $\mathbf{H^{(0)}}$ is given by the feature matrix $\mathbf{X}$ containing all node feature vectors $\mathbf{x_1}, \ldots, \mathbf{x_n}$. The assembled GCN predicts the output $\hat{\mathbf{y}} = GCN(\mathbf{X}, \Theta_{GCN})$ parameterised by Θ_{GCN}.

5.2.2 Context-aware Spatial Coordinate Embeddings

In the traditional GCN setting, the only intuition for spatial context stems from connections between nodes which allow for graph convolutions, akin to pixel convolutions with image data. This restricts the capacity of the GCN to capture spatial patterns in several ways:

- While defining good neighbourhood structures can be crucial for GCN performance, this often comes down to somewhat arbitrary choices like selecting the k nearest neighbours of each node. Without prior knowledge on the underlying data, the process of setting the right neighbourhood parameters can be difficult and require extensive testing. Furthermore, a single value of k might not be best for all nodes: different locations might be more or less dependent on their neighbours.

- Assuming that no underlying graph connecting point locations is known, one would typically construct a graph using straight-line distance between pairs of points. In many real world settings (e.g., points-of-interest along a road network) this assumption is unrealistic and may lead to poorly defined neighbourhoods.

- GCNs contain no intrinsic tool to transform point coordinates into a different (latent) space that might be more informative for representing the spatial structure, with respect to the particular problem the GCN is trying to solve.

As such, GCNs struggle with tasks that explicitly require learning of complex spatial dependencies. For example, we show in our experiments that simple GCNs are not able to solve simple spatial interpolation tasks, i.e., predicting a continuous outcome variable from the point coordinates only. We propose a novel approach to overcome these difficulties, by devising a new position encoder module, and learning a flexible spatial context encoding for each geographic coordinate, motivated by recent advances in transformers and spatial representation learning.

Given a batch of datapoints, we create the spatial coordinate matrix $\mathbf{C}$ from individual point coordinates $\mathbf{c_1}, \ldots, \mathbf{c_n}$ and define a positional encoder $PE(\mathbf{C}) = NN(ST(\mathbf{C}, \sigma_{min}, \sigma_{max}), \Theta_{PE})$, consisting of a sinusoidal transform ST and a fully-connected neural network $NN(\Theta_{PE})$, parameterised by Θ_{PE}. Following the intuition of transformers [177] for geographic coordinates [121], the sinusoidal transform is a concatenation of scale-sensitive sinusoidal functions at different frequencies, so that

$$\begin{aligned} &ST(\mathbf{C}, \sigma_{min}, \sigma_{max}) = \\ &[ST_0(\mathbf{C}, \sigma_{min}, \sigma_{max}); \ldots; ST_{S-1}(\mathbf{C}, \sigma_{min}, \sigma_{max})] \end{aligned} \tag{5.3}$$

with S being the total number of grid scales and σ_{min} and σ_{max} setting the minimum and maximum grid scale (comparable to the lengthscale parameter of a kernel). The scale-specific encoder $ST_s(\mathbf{C}, \sigma_{min}, \sigma_{max}) = [ST_{s,1}(\mathbf{C}, \sigma_{min}, \sigma_{max}); ST_{s,2}(\mathbf{C}, \sigma_{min}, \sigma_{max})]$ processes the spatial dimensions v (e.g., latitude and longitude) of $\mathbf{C}$ separately, so that

$$\begin{aligned} &ST_{s,v}(\mathbf{C}, \sigma_{\mathbf{min}}, \sigma_{\mathbf{max}}) = \\ &\left[\cos\left(\frac{\mathbf{x}^{[v]}}{\sigma_{min} g^{s/(S-1)}}\right); \sin\left(\frac{\mathbf{x}^{[v]}}{\sigma_{min} g^{s/(S-1)}}\right)\right] \\ &\forall s \in \{0, \ldots, S-1\}, \forall v \in \{1, 2\}, \end{aligned} \tag{5.4}$$

where $g = \frac{\sigma_{max}}{\sigma_{min}}$. The output from ST is then fed through the fully connected neural network $NN(\Theta_{PE})$ to transform it into the desired vector space shape, creating the coordinate embedding matrix $\mathbf{C}_{emb}$.

5.2.3 Auxiliary Learning of Spatial Autocorrelation

A further particularity of geographic data is that it often exhibits spatial autocorrelation: observations are related, in some shape or form, to their geographic neighbours. We have already successfully integrated measures of spatial autocorrelation into neural networks for discrete spatial data (images) in chapter 4. Here, we want to expand this approach to continuous spatial data.

Spatial autocorrelation can be measured using the Moran's I metric of

local spatial autocorrelation [2]. Moran's I captures localised homogeneity and outliers and thus functions as a detector of both spatial clustering and spatial change patterns. Revisiting the formulation, let the Moran's I measure of spatial autocorrelation for our outcome variable y_i be defined as:

$$I_i = (n-1)\frac{(y_i - \bar{y}_i)}{\sum_{j=1,j\neq i}^{n}(y_j - \bar{y}_j)^2} \sum_{j=1,j\neq i}^{n} a_{i,j}(y_j - \bar{y}_j), \tag{5.5}$$

where $a_{i,j} \in \mathbf{A}$ denotes adjacency of observations i and j.

As proposed in chapter 4, predicting the Moran's I metric of the output can be used as auxiliary task during training. Auxiliary task learning [164] is a special case of multi-task learning, where one learning algorithm tackles two or more tasks at once. In auxiliary task learning, we are only interested in the predictions of one task; however, adding additional, auxiliary tasks to the learner might improve performance on the primary problem: the auxiliary task can add context to the learning problem that can help solve the main problem. This approach is commonly used, for example in reinforcement learning [48] or computer vision [73, 77].

Translated to our GCN setting, we seek to predict the outcome $\mathbf{Y}$ and its local Moran's I metric $I(\mathbf{Y})$ using the same network, so that $[\hat{\mathbf{Y}}, I(\hat{\mathbf{Y}})] = GCN(\mathbf{X})$. As we note in chapter 4, the local Moran's I metric is scale-sensitive and, due to its restriction to local neighbourhoods, can miss out on longer-distance spatial effects [45, 125]. But while in chapter 4 we propose to compute the Moran's I at different resolutions, the GCN setting allows for a different, novel approach to overcome this issue.

Training of GCNs is often conducted using minibatches of points at each training step. Rather than constructing the graph of training points a-priori, we opt for a procedure where in each training step, n_{batch} points are sampled from the training data as batch B. A graph with corresponding adjacency matrix $\mathbf{A}_B$ is constructed for the batch, along with the Moran's I metric of the outcome variable $I(\mathbf{Y}_B)$. This approach brings a unique advantage: When training with (randomly shuffled) batches, points may have different neighbours in different training iterations. The Moran's I for point i can thus change throughout iterations, reflecting a differing set of more distant or closer neighbours. This also naturally helps to overcome Moran's I scale sensitivity without the need to compute the metric at different resolutions. Altogether, we refer to this altered Moran's I as "shuffled Moran's I".

5.2.4 Positional Encoder Graph Neural Network (PE-GNN)

We now assemble the different modules of our method and introduce the Positional Encoder Graph Neural Network (**PE-GNN**). The modular approach allows for a seamless integration of the positional encoder and the spatial autocorrelation auxiliary task with any graph neural network architecture. Again, taking GCN as the backbone architecture, **PE-GCN** is constructed as follows:

Assuming a batch B of randomly sampled points $p_1, \ldots, p_{n_{batch}} \in B$, a spatial graph is constructed from point coordinates $\mathbf{c}_1, \ldots, \mathbf{c}_{n_{batch}}$ using k-nearest-neighbourhood, resulting in adjacency matrix $\mathbf{A}_B$. The point coordinates are then subsequently fed through the positional encoder $PE(\Theta_{PE})$, consisting of the sinusoidal transform ST and a single fully-connected layer with sigmoid activation, embedding the $2d$ coordinates in a customisable latent space, returning vector embeddings $\mathbf{c}_1^{emb}, \ldots, \mathbf{c}_{n_{batch}}^{emb} = \mathbf{C}_B^{emb}$. The neural network allows for explicit learning of spatial context, reflected in the vector embedding. We then concatenate the positional encoder output with the node features, to create the input for the first layer of our GCN as:

$$\mathbf{H}^{(0)} = concat(\mathbf{X}_B, \mathbf{C}_B^{emb}) \tag{5.6}$$

The subsequent layers follow according to Equation 5.2.

To integrate the Moran's I auxiliary task, we compute the metric $I(\mathbf{Y}_B)$ for our outcome variable $\mathbf{Y}_B$ at the beginning of each training step according to Equation 5.5, using spatial weights from $\mathbf{A}_B$. We then duplicate the last layer of the GCN to predict both $\mathbf{Y}_B$ and $I(\mathbf{Y}_B)$:

$$\begin{aligned} \hat{\mathbf{Y}}_B &= \mathbf{H}^{(l)} \\ I(\hat{\mathbf{Y}}_B) &= \mathbf{H}_{aux}^{(l)} \end{aligned} \tag{5.7}$$

The loss of **PE-GCN** can be computed with any regression criterion, for example mean squared error (MSE):

$$\mathcal{L} = MSE(\hat{\mathbf{Y}}_B, \mathbf{Y}_B) + \lambda MSE(I(\hat{\mathbf{Y}}_B), I(\mathbf{Y}_B)) \tag{5.8}$$

where λ denotes the auxiliary task weight. The final model is denoted as $M_{\Theta_{PE}, \Theta_{GCN}}$. Our approach is outlined in Figure 5.1. Algorithm 3 describes a training cycle.

PE-GNN, with any GNN backbone, helps to tackle the particular challenges of geographic patterns, as outlined in the previous sections:

- While our approach still includes the somewhat arbitrary choice of k-nearest neighbours to define the spatial graph, the proposed positional encoder network is not bound by this restriction, as it does not operate on

Algorithm 3: PE-GCN Training

Parameters: M, hyper-parameter
for *number of training steps (tsteps)* **do**
  Sample minibatch B of n_{batch} points with features $\mathbf{X}_B$, coordinates $\mathbf{C}_B$ and outcome $\mathbf{Y}_B$;
  Construct a spatial graph with adjacency matrix $\mathbf{A}_B$ from coordinates $\mathbf{C}_B$ using k-nearest neighbours;
  Using spatial adjacency $\mathbf{A}_B$, compute Moran's I of output as $I(\mathbf{Y}_B)$;
  Predict outcome $[\hat{\mathbf{Y}}_B, I(\hat{\mathbf{Y}}_B)] = M_{\Theta_{PE},\Theta_{GCN}}(\mathbf{X}_B, \mathbf{C}_B)$;
  Compute loss $\mathcal{L}(\mathbf{Y}_B, I(\mathbf{Y}_B), \hat{\mathbf{Y}}_B, I(\hat{\mathbf{Y}}_B), \lambda)$;
  Update the parameters $\Theta_{GCN}, \Theta_{PE}$ of model M using stochastic gradient descent;
end
return M

the graph. This enables a separate learning of context-aware embeddings for each coordinate, accounting for neighbours at any potential distance within the batch.

- While the spatial graph used still relies on straight-line distances between points, the positional encoder embeds latitude and longitude values in a high-dimensional latent space. These high-dimensional coordinates are able to reflect spatial complexities much more flexibly and, added as node features, can communicate these throughout the learning process.

- Batched **PE-GNN** training is not conducted on a single graph, but a new graph consisting of randomly sampled training points at each iteration. As such, at different iterations, focus is put on the relationships between different clusters of points. This helps our method to generalise better, rather than just memorising neighbourhood structures.

- The differing training batches also help us to compute a "shuffled" version of the Moran's I metric, capturing autocorrelation at the same location for different (closer or more distant) neighbourhoods.

5.3 Experiments

5.3.1 Data

We evaluate **PE-GNN** and baseline competitors on four real-world geographic datasets of different spatial resolutions (regional, continental and global). As our task is similar to the regression experiments from Chapter 3, we include one of the datasets used there (California Housing) here. Nonetheless, we also

chose some new datasets to highlight the applicability of **PE-GNN** across different geospatial applications:

California Housing[1]*:* This dataset contains the prices of over $20,000$ California houses from the 1990 U.S. census and was introduced by Kelley Pace and Barry [86]. The regression task at hand is to predict house prices y using features $\mathbf{x}$ (e.g., house age, number of bedrooms) and location $\mathbf{c}$. California housing is a standard dataset for assessment of spatial autocorrelation.

Election[2]*:* This dataset contains the election results of over $3,000$ counties in the United States and was proposed by Jia and Benson [80]. The regression task here is to predict election outcomes y using socio-demographic and economic features (e.g., median income, education) $\mathbf{x}$ and county locations $\mathbf{c}$.

Air temperature[3]*:* The air temperature dataset [72] contains the coordinates of $3,000$ weather stations around the globe. For this regression task we seek to predict mean temperatures y from a single node feature x, mean precipitation, and location c.

3d Road[4]*:* The 3d road dataset [85] provides 3-dimensional spatial coordinates (latitude, longitude, and altitude) of the road network in Jutland, Denmark. The dataset comprises over $430,000$ points and can be used for a regression task where altitude y is predicted using latitude and longitude coordinates $\mathbf{c}$. Note that this dataset does not contain any node features $\mathbf{x}$, so it is only used for the spatial interpolation task.

5.3.2 Baselines

We compare **PE-GNN** with four different graph neural network backbones: The original GCN formulation, introduced by Kipf and Welling [90] and outlined in the Methods section, graph attention mechanisms (GAT) [178] and GraphSAGE [66]. Lastly, we use Kriging Convolutional Networks (KCN) [6]. KCN differs from GCN primarily in two ways: it transforms the distance-weighted adjacency matrix $\mathbf{A}$ using a Gaussian kernel and adds the outcome variable and features of neighbouring points to the features of each node. To avoid a data leak, test set points can only access neighbours from the training set to extract these features. We compare the naive version of all these approaches to the same four backbone architectures augmented with our **PE-GNN** modules. We expect that the naive GCN, GAT and GraphSAGE

[1] Access via sklearn: `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_california_housing.html`

[2] Access via Github: `https://github.com/JunwenBai/correlation-gnn`; county locations can be accessed via the US census: `https://www2.census.gov/geo/docs/maps-data/data/gazetteer/2020_Gazetteer/`; a convenient downloader is contained with our code

[3] Access via Figshare: `https://springernature.figshare.com/collections/A_global_dataset_of_air_temperature_derived_from_satellite_remote_sensing_and_weather_stations/4081802`

[4] Access via UCI: `https://archive.ics.uci.edu/ml/datasets`

approaches are less capable of learning spatial dependencies in the data as they are restricted to capturing effects at the pre-defined neighborhood scale given by the input graph. **PE-GNN** on the other hand has the capacity to learn location-specific context embeddings which, in theory, should allow it to spatial effects across the whole dataset and hence to improve performance. Beyond GNN-based approaches, we also compare **PE-GNN** to the most popular method for modeling continuous spatial data: Gaussian processes. For all approaches, we compare a range of different training settings and hyperparameters, as discussed below.

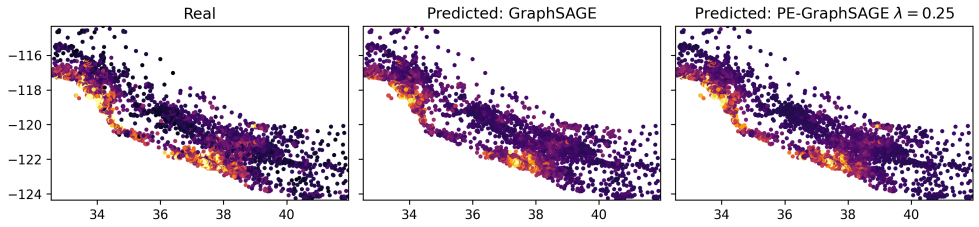
5.3.3 Experimental Setup

To allow for a fair comparison between the different GNN approaches, we equip all models with the same architecture, consisting of two GCN / GAT / GraphSAGE layers with ReLU activation and dropout, followed by a final linear layer as regression head. The KCN model also uses GCN layers, following the author specifications.For the auxiliary task setting, the final fully-connected layer is duplicated and used task-specific. We test four different auxiliary task weights $\lambda = \{0, 0.25.0.5, 0.75\}$, where $\lambda = 0$ implies no auxiliary task. The positional encoder contains the sinusoidal transforms, followed by three fully-connected layers with Tanh activation functions (except the last layer) to project the output into the desired vector space. Spatial graphs are constructed assuming $k = 5$ nearest neighbours, following results from previous work [6, 80] and our own rigorous testing. We include a sensitivity analysis of the k parameter and different batch sizes in our results section. Training for the GNN models is conducted using PyTorch [139] and PyTorch Geometric [46]. We use the Adam algorithm to optimise our models [89] and the mean squared error (MSE) as loss function. Gaussian process models (exact and approximate) are trained using GPyTorch [55]. Due to the size of the dataset, we only provide an approximate GP result for 3d Road. All training is conducted on single CPUs via Google Colab.

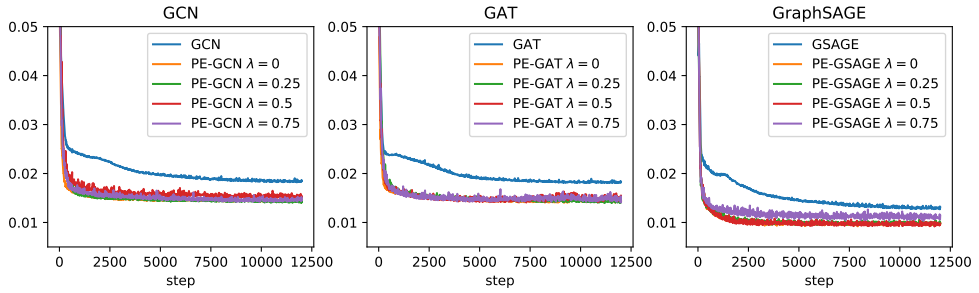
5.3.4 Results

We test our methods on two tasks: *Spatial Interpolation*, predicting outcomes from spatial coordinates alone, and *Spatial Regression*, where additional node features are available. The results of our experiments are shown in Table 5.1 and 5.2. We evaluate both models using the mean squared error (MSE) and mean absolute error (MAE) metrics. Figure 5.2 exemplary shows predictions as well as test error curves for the California Housing dataset. We also show prediction confidence intervals in figure 5.3

For the *spatial interpolation* task, we observe that the **PE-GNN** approaches



(a) A scatterplot of real values and predictions using GraphSAGE and PE-GraphSAGE. X- and y-axis represent latitude and longitude locations of the observations respectively.



(b) Test error curves of GCN, GAT and GraphSAGE based models, measured by the MSE metric on the y-axis.

Figure 5.2: Visualising predictive performance on the California Housing dataset.

Model	**Cali. Housing**		**Election**		**Air Temp.**		**3d Road**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
GCN [90]	0.0558	0.1874	0.0034	0.0249	0.0225	0.1175	0.0169	0.1029
PE-GCN $\lambda = 0$	0.0161	**0.0868**	0.0032	0.0241	0.0040	0.0432	**0.0031**	**0.0396**
PE-GCN $\lambda = 0.25$	**0.0155**	0.0882	0.0032	**0.0236**	0.0037	0.0417	0.0032	0.0416
PE-GCN $\lambda = 0.5$	0.0156	0.0885	**0.0031**	0.0241	**0.0036**	**0.0401**	0.0033	0.0421
PE-GCN $\lambda = 0.75$	0.0160	0.0907	**0.0031**	0.0240	0.0040	0.0429	0.0033	0.0424
GAT [178]	0.0558	0.1877	0.0034	0.0249	0.0226	0.1165	0.0178	0.0998
PE-GAT $\lambda = 0$	**0.0159**	0.0918	**0.0032**	**0.0234**	**0.0039**	0.0429	0.0060	0.0537
PE-GAT $\lambda = 0.25$	0.0161	**0.0867**	**0.0032**	0.0235	0.0040	**0.0417**	**0.0058**	**0.0530**
PE-GAT $\lambda = 0.5$	0.0162	0.0897	**0.0032**	0.0238	0.0045	0.0465	0.0061	0.0548
PE-GAT $\lambda = 0.75$	0.0162	0.0873	**0.0032**	0.0237	0.0041	0.0429	0.0062	0.0562
GraphSAGE [66]	0.0558	0.1874	0.0034	0.0249	0.0274	0.1326	0.0180	0.0998
PE-GraphSAGE $\lambda = 0$	0.0157	0.0896	**0.0032**	**0.0237**	0.0039	0.0428	0.0060	**0.0534**
PE-GraphSAGE $\lambda = 0.25$	**0.0097**	0.0664	**0.0032**	0.0242	0.0040	0.0418	0.0059	**0.0534**
PE-GraphSAGE $\lambda = 0.5$	0.0100	0.0682	0.0033	0.0239	0.0043	0.0461	0.0060	0.0536
PE-GraphSAGE $\lambda = 0.75$	0.0100	**0.0661**	**0.0032**	0.0241	**0.0036**	**0.0399**	**0.0058**	0.0541
KCN [6]	0.0292	0.1405	0.0367	0.1875	0.0143	0.0927	0.0081	0.0758
PE-KCN $\lambda = 0$	0.0288	0.1274	0.0598	0.2387	0.0648	0.2385	**0.0025**	**0.0310**
PE-KCN $\lambda = 0.25$	0.0324	0.1380	0.0172	0.1246	**0.0059**	**0.0593**	0.0037	0.0474
PE-KCN $\lambda = 0.5$	**0.0237**	**0.1117**	0.0072	0.0714	0.0077	0.0664	0.0077	0.0642
PE-KCN $\lambda = 0.75$	0.0260	0.1194	**0.0063**	**0.0681**	0.0122	0.0852	0.0110	0.0755
Approximate GP	0.0353	0.1382	0.0031	0.0348	0.0481	0.0498	0.0080	0.0657
Exact GP	0.0132	0.0736	0.0022	0.0253	0.0084	0.0458	-	-

Table 5.1: *Spatial Interpolation:* Test MSE and MAE scores from four different datasets, using four different GNN backbones with and without our proposed architecture.

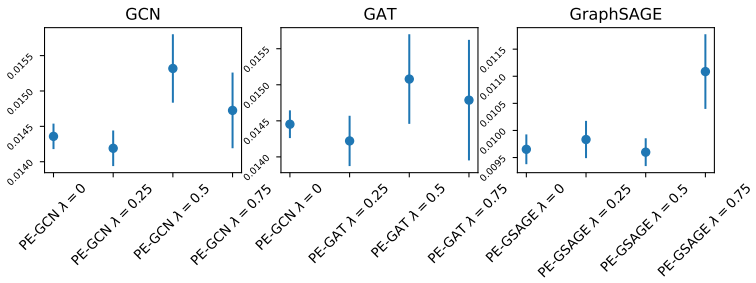
consistently and vastly improve performance for all four backbone architectures across the California Housing, Air Temperature and 3d Road datasets and by a small margin for the Election dataset.

Model	Cali. Housing		Election		Air Temp.	
	MSE	MAE	MSE	MAE	MSE	MAE
GCN	0.0185	0.1006	**0.0025**	**0.0211**	0.0225	0.1175
PE-GCN $\lambda = 0$	**0.0143**	**0.0814**	0.0026	0.0213	0.0040	0.0432
PE-GCN $\lambda = 0.25$	**0.0143**	0.0816	0.0026	0.0213	0.0037	0.0417
PE-GCN $\lambda = 0.5$	**0.0143**	0.0828	0.0027	0.0217	**0.0036**	**0.0401**
PE-GCN $\lambda = 0.75$	0.0147	0.0815	0.0027	0.0219	0.0040	0.0429
GAT	0.0183	0.0969	**0.0024**	**0.0211**	0.0226	0.1165
PE-GAT $\lambda = 0$	0.0144	0.0836	0.0028	0.0218	**0.0039**	0.0429
PE-GAT $\lambda = 0.25$	**0.0141**	**0.0817**	0.0028	0.0219	0.0040	**0.0417**
PE-GAT $\lambda = 0.5$	0.0155	0.0851	0.0030	0.0225	0.0045	0.0465
PE-GAT $\lambda = 0.75$	0.0145	0.0824	0.0029	0.0223	0.0041	0.0429
G.SAGE	0.0131	0.0798	**0.0007**	**0.0127**	0.0219	0.1153
PE-G.SAGE $\lambda = 0$	0.0099	0.0667	0.0011	0.0154	0.0037	0.0422
PE-G.SAGE $\lambda = 0.25$	**0.0098**	**0.0648**	0.0010	0.0152	**0.0029**	**0.0381**
PE-G.SAGE $\lambda = 0.5$	**0.0098**	0.0679	0.0012	0.0157	0.0037	0.0445
PE-G.SAGE $\lambda = 0.75$	0.0114	0.0766	0.0012	0.0152	0.0038	0.0459
KCN	0.0292	0.1405	0.0367	0.1875	0.0143	0.0927
PE-KCN $\lambda = 0$	0.0288	0.1274	0.0598	0.2387	0.0648	0.2385
PE-KCN $\lambda = 0.25$	0.0324	0.1380	0.0172	0.1246	**0.0059**	**0.0593**
PE-KCN $\lambda = 0.5$	**0.0237**	**0.1117**	0.0072	0.0714	0.0077	0.0664
PE-KCN $\lambda = 0.75$	0.0260	0.1194	**0.0063**	**0.0681**	0.0122	0.0852
Approximate GP	0.0195	0.1008	0.0050	0.0371	0.0481	0.0498
Exact GP	0.0036	0.0375	0.0006	0.0139	0.0084	0.0458

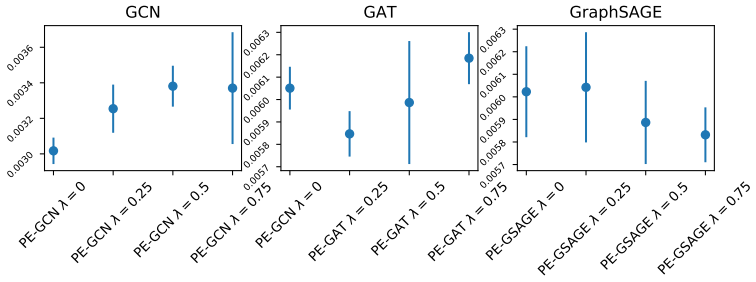
Table 5.2: *Spatial Regression:* Test MSE and MAE scores from three different datasets, using four different GNN backbones with and without our proposed architecture.

For the *spatial regression* task, we observe that the **PE-GNN** approaches consistently and substantially improve performance for all four backbone architectures on the California Housing and Air Temperature datasets. Performance remains unchanged or decreases by very small margins in the Election dataset, except for the KCN backbone which benefits tremendously from the **PE-GNN** approach, particularly with auxiliary tasks.

Our results are interesting in several ways: Generally, **PE-GNN** substantially improves over baselines in regression and interpolation settings. Most of the improvement can be attributed to the positional encoder, however the auxiliary task learning also has substantial beneficial effects in some settings, especially for the KCN models. The best setting for the task weight hyperparameter λ seems to heavily depend on the data, which confirms findings from chapter 4. To our knowledge, **PE-GNN** is the first GNN-based learning approach that can compete with Gaussian Processes on simple spatial interpolation baselines. For KCN models, which include neighbour information as additional node features, we observe a proneness to overfitting. As the authors of KCN mention, this effect diminishes in large enough data domains [6]. For example, KCNs are the best performing method on the 3d road dataset–by far



(a) California Housing.



(b) 3d Road.

Figure 5.3: MSE bar plots of mean performance and 2σ confidence intervals obtained from 10 different training checkpoints.

our largest experimental dataset. We observe that in cases when KCN learns well, **PE-KCN** can still improve its performance, as highlighted by the 3d road experiments. The KCN experiments also highlight the strongest effects of the Moran's I auxiliary tasks: In cases when KCN overfits (Election, Cali. Housing datasets), **PE-KCN** without auxiliary task ($\lambda = 0$) is not sufficient to overcome the problem. However, adding the auxiliary task ($\lambda = [0.25, 0.5, 0.75]$) can mitigate most of the overfitting issue. This directly confirms a theory of [93] on the beneficial effects of auxiliary learning of spatial autocorrelation. We also confirm the finding from Chapter 4, that the ideal weighting of the auxiliary tasks is highly data dependent and no one-size-fits-all solution exists. Throughout our experiments, **PE-GNN** provides a powerful and flexible framework for learning with geographic coordinates. The positional encoder allows for the learning of spatial context features to inform the downstream task, while the Moran's I auxiliary task helps to overcome potential overfitting issues.

Lastly, Figure 5.4 highlights some results from our sensitivity analyses with the k and n_{batch} (batch size) parameters. After rigorous testing, we opt for $k = 5$-NN approach to create the spatial graph and compute the shuffled Moran's I across all models. We chose $n_{batch} = 2048$ for California Housing and 3d Road datasets and $n_{batch} = 1024$ for the election dataset.

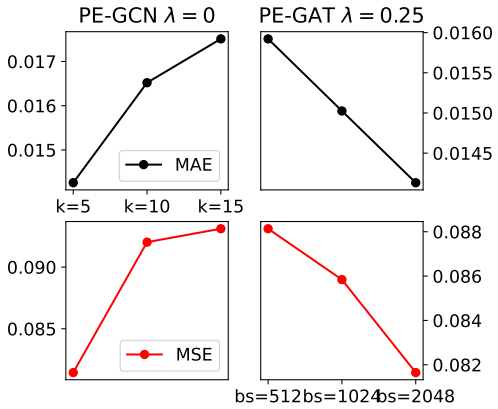


Figure 5.4: Predictive performance, measured by MAE and MSE, of PE-GCN and PE-GAT models trained on the California Housing dataset, using different values of k for constructing nearest-neighbour graphs and different batch sizes (bs).

5.4 Discussion

In this chapter we consider the problem of spatial regression modeling. Recalling the background chapter (specifically section 2.1.5), this poses a distinct challenge, as standard regression models (such as OLS) fail to address the spatial nature of the data, which can result in spatially correlated residuals. To address this, spatial lag models [4] add a spatial lag term to the regression equation that is proportional to the dependent variable values of nearby observations, assigned by a weight matrix. Likewise, kernel regression takes a weighted average of nearby points when predicting the dependent variable. Nevertheless, the most popular off-the-shelf methods for modeling continuous spatial data are based on Gaussian processes [37].

With the emergence of machine learning, there has been a rise of research on applications of neural network models for spatial modeling tasks. More specifically, GNNs are often used for these tasks with the spatial data represented graphically. Particularly, they offer flexibility and scalability advantages over traditional spatial modeling approaches. Specific GNN architectures including Graph Convolutional Networks [90], Graph Attention Networks [178] and GraphSAGE [66] are powerful methods for inference and representation learning with spatial data.

Recently, GNN approaches tailored to the specific complexities of geospatial data have been developed. The authors of Kriging Convolutional Networks [6] propose using GNNs to perform a modified kriging task. Hamilton et al. [66] apply GNNs for a spatio-temporal Kriging task, recovering data from unsampled nodes on an input graph. We look to extend this line of research by providing stronger, explicit capacities for GNNs to learn spatial structures.

Additionally, our proposed method is highly modular and can be combined with any GNN backbone.

This chapter also addresses the problem of embedding geospatial context. Through many decades of research on spatial patterns, a myriad of measures, metrics, and statistics have been developed to cover a broad range of spatial interactions. All of these measures seek to transform spatial locations, with optional associated features, into some meaningful embedding, for example, a theoretical distribution of the locations or a measure of spatial association.

A set of point locations in $2d$-space may be distributed randomly, or may follow some underlying spatial pattern. Spatial point processes are some of the oldest methods of analysing spatial data and are a useful tool when point locations depend on one another, e.g., in the case of epidemiological or ecological data. Point processes are particularly popular for modeling geographic data, for example Cox processes [1] and Hawkes processes [176]. Empirical observations can be embedded using these point processes, for example, to make predictions on future spatial spread. If we now assume that point locations also come with a continuous feature value, the range of potential spatial effects and associated metrics is vastly expanded. This includes metrics like the Moran's I [2], which is used here, or expansions, like local spatial heteroskedasticity [137] and local spatial dispersion [189]. Measures of spatial autocorrelation have already been shown to be useful for improving neural network models through auxiliary task learning [93], model selection [95] and localised representation learning [51].

Beyond these traditional metrics, recent years have seen the emergence of neural network based embeddings for geographic information. Wang et al. [182] use kernel embeddings to learn social media user locations. Fu et al. [51] devise an approach using local point-of-interest (POI) information to learn region embeddings and integrate similarities between neighbouring regions to learn mobile check-ins. Yin et al. [198] develop GPS2Vec, an embedding approach for latitude-longitude coordinates, based on a grid cell encoding and spatial context (e.g., tweets and images). Mai et al. [121] developed Space2Vec, another latitude-longitude embedding without requiring further context like tweets or POIs. Space2Vec transforms the input coordinates using sinusoidal functions and then reprojects them into a desired output space using linear layers. All of these studies highlight the potential of spatial coordinate embeddings (with or without spatial context enrichment) for improving predictive models for geographic data. Our findings in this chapter confirm the value of such approaches.

5.5 Summary

With **PE-GNN**, we introduce a flexible, modular new GNN-based learning framework for geographic data. **PE-GNN** leverages recent findings in embedding spatial context into neural networks to improve predictive models. Our empirical findings confirm a strong performance, with **PE-GNN** approaches consistently and substantially outperforming their naive GNN baselines. Our experiments explore datasets from different domains (from weather data to house prices) and at different spatial scales, from regional (e.g. California) to global coverage. Altogether, our findings imply that **PE-GNN** generalises well across application domains, spatial scales and underlying GNN operators. This chapter also highlights once more how domain expertise from applied academic disciplines like geography can help improve machine learning models for the particular data that is common in these disciplines. Nonetheless, **PE-GNN** also comes with some limitations. Most importantly, **PE-GNN** still relies on the definition of a spatial graph, which comes with many assumptions (e.g. regarding the distance measure) on the spatial structure of the data. This can be particularly challenging in applications where we lack domain expertise to construct these distances, and ad-hoc methods like Euclidean distance are only poor approximations of the true distances (e.g. when considering the road network). Lastly, **PE-GNN** is currently restricted to the processing of latitude longitude coordinate pairs and has not yet been tested on other coordinate systems or mapping schemes.

With respect to the broader scope of this dissertation, this chapter discusses the use of both *functional embeddings* (like the Moran's I) and *parametric neural-network embeddings* (like Space2Vec) for improving GNN models in the geospatial domain. Our experiments show that learning contextual features from geographic coordinates allows GNNs to better incorporate information on spatial dependencies, leading to better predictions.

Chapter 6

Autoregressive Embedding Loss for Spatio-temporal GANs

6.1 Introduction

In this chapter, we expand the scope of our work from the spatial to the spatio-temporal domain. Here, deep learning has found particularly successful applications in the video domain, for example for trajectory forecasting, video super-resolution or object tracking. Nevertheless, data observed over (discrete) space and time can take many more shapes than just RGB videos: many of the systems and processes governing our planet, from ocean streams to the spread of viruses, exhibit complex spatio-temporal dynamics. As we have discussed previously, current deep learning approaches often struggle to account for these [148]. As such, there are many calls for more concerted research efforts aiming to improve the capacity of deep neural networks for modelling earth systems data. Recently, the emergence of physics-informed deep learning, discussed in section 2.2.3, has reinforced the integration of physical constraints as a research domain [87, 146, 187, 203].

In this chapter, we propose a novel GAN tailored to the challenges of spatio-temporal complexities. We first devise a novel measure of spatio-temporal association—SPATE—expanding on the Moran's I measure of spatial auto-correlation. SPATE uses the deviance of an observation from its space-time expectation, and compares it to neighbouring observations to identify regions of (relative) change and regions of (relative) homogeneity over time. We propose three different approaches to calculate the space-time expectations, coming with varying assumptions and advantages for different applications.

We then encode a SPATE-based embedding into COT-GAN [194] to formulate a new GAN framework, named SPATE-GAN. The motivation of choosing

COT-GAN as the base model is that its principle of respecting temporal dependencies in sequential modelling is in line with our intuition for SPATE; see details in section 6.2. Lastly, we test our approach on a range of different datasets. Specifically, we select data characterised by complex spatio-temporal patterns such as fluid dynamics [179, 187], disease spread [24] or global surface temperatures [160]. We observe that SPATE-GAN outperforms baseline models. This finding is particularly interesting as we do not change the architecture of the existing COT-GAN backbone, implying that our performance gains can be solely attributed to our novel SPATE-based embedding loss.

To summarise, the contributions of this chapter are as follows:

- We introduce SPATE, a new measure of spatio-temporal association, by expanding the intuition of the Moran's I metric into the temporal dimension.
- We introduce SPATE-GAN, a novel GAN for complex spatio-temporal data using SPATE to construct an embedding loss "nudging" the model to focus on the learning of autoregressive structures. This is facilitated through an embedding loss function based on the SPATE metric. To harmonise this approach with the logic of causal optimal transport (COT) that underlies COT-GAN training, we devise an instance of SPATE that respects the sequentially of the input data, i.e. that is only calculated using past time steps.
- We test SPATE-GAN against baseline GANs designed for image/video generation on datasets representing fluid dynamics, disease spread and global surface temperature. We show performance gains of SPATE-GAN over the baseline models.

The remainder of this chapter is structured as follows: Section 6.2 introduces a novel measure for spatio-temporal autocorrelation, SPATE. We then integrate this metric into the loss function of video GANs to reinforce the learning of spatio-temporal dependencies throughout training. Section 6.3 then showcases experimental findings on a range of datasets with high real-world relevance. Section 6.4 discusses this chapter, relevant literature and the contributions to this dissertation. We then summarise the chapter in section 6.5.

6.2 Methodology

6.2.1 SPATE: Spatio-temporal Association

To build our new metric, let us start from the local Moran's I metric. For a static, discrete spatial pattern (e.g. a grid of pixels forming an image) consisting

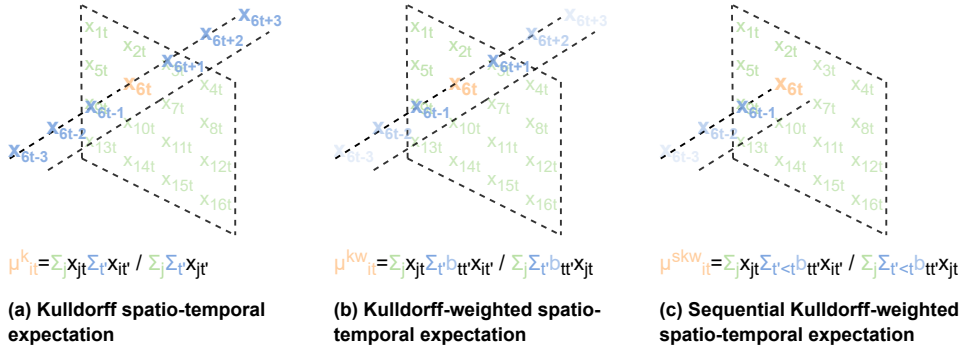


Figure 6.1: Illustrating the three proposed options to obtain *spatio-temporal expectations* μ_{it} used in the computation of SPATE for single-channel data.

of continuous values $x \in \mathbb{R}^n$ where $n \in \mathbb{N}$ is the dimensionality of x, x_i for $i \in 1, \ldots, n$ represents the i'th pixel value on a regular grid.

The local Moran's I statistic I_i is computed as follows:

$$I_i = (n-1)\frac{z_i}{\sum_{j=1, j\neq i}^{n} z_j^2} \sum_{j=1, j\neq i}^{n} w_{i,j} z_j \tag{6.1}$$

,

where $z_i = x_i - \bar{x}$ is the deviance of observation x_i from the global mean $\bar{x}$, n_i is the number of spatial neighbours of pixel x_i, j indexes neighbours of x_i for $j \in \{1, ..., n_i\}$ and $j \neq i$, and $w_{i,j}$ is a binary spatial weight matrix, indicating spatial neighbourhood of observations i and j. I_i can be interpreted as a measure of similarity to neighbouring pixels: positive values imply homogeneous clusters, while negative values suggest outliers, change patterns or edges.

Now, let us assume that we observe a sequence of spatial patterns over time t: $x = (x_1, ..., x_T) \in \mathbb{R}^{n \times T}$ where n is the dimensionality of x_t at each time t and T is the total time steps of the sequence. Of course, a naive adoption of the approach above is simply to ignore the time component of a sequence and compute the local Moran's I values I_{it} around pixel i using mean values $\bar{x}_t$ at each time t. Unfortunately, this approach would strictly separate spatial and temporal effects. In fact, the much more realistic assumption is that space and time are not separable, but do in fact interact and form joint patterns. For this reason, we expand the concept of Moran's I for spatio-temporal expectations.

First, we follow the intuition outlined in Kulldorff et al. [107] and define expected values of $\mu_{it}(x)$. We refer to this approach as Kulldorff spatio-temporal expectation ("k"):

$$\mu_{it}^{(k)} = \frac{\sum_j x_{jt} \sum_{t'} x_{it'}}{\sum_j \sum_{t'} x_{jt'}}. \tag{6.2}$$

$\mu_{it}^{(k)}$ in Equation 6.2 involves using all spatial units (pixels) available at

time step t and across all time steps at a single spatial unit (pixel position) i. This computation of the *spatio-temporal expectations* assumes independence of space and time, and thus the residual $z_{it} = x_{it} - \mu_{it}^{(k)}$ can be thought of as a local measure of space-time interaction at pixel i and time t. Moreover, this formulation of μ_{it} makes two critical assumptions: (1) Different time steps are equally important, irrespective of how distant they are from the current time step. (2) At each time step, we assume availability of the whole time series (i.e. looking into the future is possible). We can modify the computation of μ_{it} by imposing alternative assumptions.

First, assuming that distant time steps have less significant impact on the current time step, we can integrate temporal weights into the computation, and apply decreasingly lower weights to more distant time steps. For example, one can consider an exponential kernel:

$$\mu_{it}^{(kw)} = \frac{\sum_j x_{jt} \sum_{t'} b_{tt'} x_{it'}}{\sum_j \sum_{t'} b_{tt'} x_{jt'}}, \tag{6.3}$$

where $b_{tt'} = \exp(-|t - t'|/l)$ and l is the lengthscale of the exponential kernel. We refer to this approach as Kulldorff-weighted spatio-temporal expectation ("*kw*").

Second, we can restrict the computation of μ_{it} at time step t to only account for time steps $< t$, so that the expectation at each time step is independent of future observations. Thus, the computation respects the generating logic of sequential data. We refer to this last approach as Kulldorff-sequential-weighted spatio-temporal expectation ("*ksw*"):

$$\mu_{it}^{(ksw)} = \frac{\sum_j x_{jt} \sum_{t'<t} b_{tt'} x_{it'}}{\sum_j \sum_{t'<t} b_{tt'} x_{jt'}}. \tag{6.4}$$

Note that for the first time step $t = 0$, we cannot access past time steps to calculate spatio-temporal expectations $\mu_{i0}^{(ksw)}$.

We can now simply plug in our new spatio-temporal expectations into the Moran's I metric at time t by replacing spatial only expectations with a spatio-temporal expectation of our choice. As such, we define our novel measure of **spa**tio-**te**mporal association (SPATE),

$$S_{it}(x, w) = (n_i - 1)\frac{z_{it}}{\sum_{j=1}^{n_x} z_{jt}^2} \sum_{j=1, j\neq i}^{n_x} w_{i,j} z_{jt} \tag{6.5}$$

where $z_{it} = x_{it} - \mu_{it}$ and μ_{it} can be any option from $\mu_{it}^{(k)}$, $\mu_{it}^{(kw)}$ and $\mu_{it}^{(ksw)}$. When using $\mu_{it}^{(ksw)}$, SPATE does not return values for $t = 0$, as no previous time steps are available to calculate spatio-temporal expectations. See Fig 6.1 for an illustration of the three proposed options.

SPATE measures spatio-temporal autocorrelation at the input resolution. Its behavior can be closely related to that of the Moran's I metric. While Moran's I evaluates the deviance z_i between each pixel and the spatial expectation, SPATE does so by using the spatio-temporal expectation z_{it}. Like Moran's I, SPATE acts as a detector of spatio-temporal clusters and change patterns. Like Moran's I, SPATE identifies positive and negative space-time autocorrelation, i.e. homogeneous areas of similar behavior and outliers that behave differently from their immediate neighbourhood. The difference between Moran's I and SPATE is that the later explicitly captures space-time interactions. For example, if pixel x_i and all other data points (not just its neighbours) are increased at a given time step t, Moran's I at time t (for all points) will be high, but SPATE will not be. SPATE of x_{it} will be high if *(a)* pixel x_{it} is high compared to its expectation at the same time t, *(b)* its neighbours are high compared to their expectations, while *(c)* its non-neighbours are not particularly high compared to their expectations.

In the kw and ksw settings, the lengthscale parameter l governs whether the metric captures longer or shorter term temporal patterns. For example, if pixel x_{it} and its neighbours increase slowly over time, that change will only cause SPATE to be high for larger lengthscales l, while smaller l values imply that current values are compared to those that are close in time. As such, the lengthscale determines what changes are considered "slow" (incorporated into the mean, not detected as space-time interaction) and "fast" (current values are different from the mean, detected as space-time interaction). The ksw setting further allows for scenarios where we might wish to compute the metric based on previous time-steps alone, i.e. to preserve sequential logic. The differences between the Moran's I and SPATE, in its different configurations, are also highlighted in Figure 6.2.

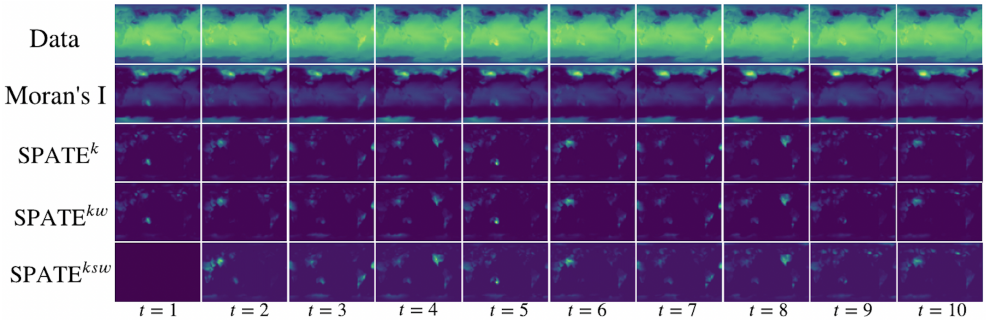


Figure 6.2: The SPATE metric in its different forms computed for an example from the LGCP datasets. While SPATE^k and SPATE^{kw} are visually undistinguishable, SPATE^{ksw} changes as increasingly more past time steps become available, converging to SPATE^{kw} at time T. We can also observe how the Moran's I metric remains static in time, while all versions of SPATE behave dynamically in space and time.

6.2.2 COT-GAN

Built upon the theory of Causal Optimal Transport (COT), COT-GAN was introduced by Xu et al. [194] as an adversarial algorithm to train implicit generative models for sequential learning. COT can be considered as a maximisation over the classical (Kantorovich) optimal transport (OT) with a temporal causality constraint, which restricts the transporting of mass on the arrival sequence at any time t to depend on the starting sequence only up to time t. This motivated us to design the spatio-temporal expectation μ_{it}^{ksw} in order to respect the nature of sequential data that are generated in an autoregressive manner.

In sequential generation, we are interested in learning a model that produces $y = (y_1, ..., y_T) \in \mathbb{R}^{n \times T}$ to mimic $x = (x_1, ..., x_T) \in \mathbb{R}^{n \times T}$. Given two probability measures μ, ν defined on $\mathbb{R}^{n \times T}$, and a cost function $c : \mathbb{R}^{n \times T} \times \mathbb{R}^{n \times T} \to \mathbb{R}$, the causal optimal transport of μ into ν is formulated as:

$$\mathcal{W}_c^{\mathcal{K}}(\mu, \nu) := \inf_{\pi \in \Pi^{\mathcal{K}}(\mu,\nu)} \mathbb{E}^{\pi}[c(x, y)], \tag{6.6}$$

where $\Pi^{\mathcal{K}}(\mu, \nu)$ is the set of probability measures π on $\mathbb{R}^{n \times T} \times \mathbb{R}^{n \times T}$ with marginals μ, ν, which also satisfy the constraint:

$$\pi(dy_t | dx_{1:T}) = \pi(dy_t | dx_{1:t}) \quad \text{for all } \ t = 1, \cdots, T-1. \tag{6.7}$$

Such plans in $\Pi^{\mathcal{K}}(\mu, \nu)$ are called *causal transport plans*. Here $c(x, y)$ is a cost function that measures the loss incurred by transporting a unit of mass from x to y. $\mathcal{W}_c^{\mathcal{K}}(\mu, \nu)$ is thus the minimal total cost for moving the mass μ to ν in a causal way.

In comparison, the (classic) OT is defined on $\mathbb{R}^n$ and differs from COT by searching an optimal plan that leads to the least cost in a less restricted space $\Pi(\mu, \nu)$ which contains all transport plans:

$$\mathcal{W}_c(\mu, \nu) := \inf_{\pi \in \Pi(\mu,\nu)} \mathbb{E}^{\pi}[c(x, y)]. \tag{6.8}$$

COT-GAN constructed an adversarial objective by reformulating the COT Equation 6.6 as below:

$$\mathcal{W}_c^{\mathcal{K}}(\mu, \nu) = \sup_{c^{\mathcal{K}} \in \mathcal{C}^{\mathcal{K}}(\mu,c)} \inf_{\pi \in \Pi(\mu,\nu)} \{\mathbb{E}^{\pi}[c^{\mathcal{K}}(x, y)]\}, \tag{6.9}$$

where $\mathcal{C}^{\mathcal{K}}(\mu, c)$ is a special family of costs that encode the causality constraint, see Appendix C for details. Note that the inner optimisation problem is equivalent to OT in Equation 6.8 with a specific class of cost function, i.e., $\mathcal{W}_{c^{\mathcal{K}}}(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \{\mathbb{E}^{\pi}[c^{\mathcal{K}}(x, y)]\}$.

In the implementation of COT-GAN, learning is done via stochastic gradient descent (SGD) on mini-batches. Given a distribution ζ on some latent space $\mathcal{Z}$, the generator g_θ is a function that maps a latent variable $z \sim \zeta$ to the generated sequence y in the path space. Given a mini-batch of size m from training data $\{x^d_{1:T}\}_{i=1}^m$ and that from generated samples $\{y^d_{1:T}\}_{i=1}^m$, we define the empirical measures for the mini-batches as:

$$\hat{\mu} := \frac{1}{m}\sum_{d=1}^{m} \delta_{x^d_{1:T}}, \quad \hat{\nu}_\theta := \frac{1}{m}\sum_{d=1}^{m} \delta_{y^d_{1:T}}, \tag{6.10}$$

where $\hat{\nu}_\theta$ incorporates the parameterisation of g_θ.

Thus, the duality of COT given by Equation 6.9 between the empirical measures of two mini-batches can be written as:

$$\mathcal{W}_c^{\mathcal{K}}(\hat{\mu}, \hat{\nu}_\theta) = \sup_{c^{\mathcal{K}} \in \mathcal{C}^{\mathcal{K}}(\hat{\mu}, c)} \mathcal{W}_{c^{\mathcal{K}}}(\hat{\mu}, \hat{\nu}_\theta). \tag{6.11}$$

Computing Equation 6.11 involves solving the optimisation problem of classic OT. COT-GAN opted for the Sinkhorn algorithm, see [35, 57], and a modified version of Sinkhorn divergence for an approximation. Here we have the entropic-regularised OT defined as:

$$\mathcal{W}_{c,\varepsilon}(\mu, \nu) := \inf_{\pi \in \Pi(\mu,\nu)} \{\mathbb{E}^\pi[c(x, y)] - \varepsilon H(\pi)\}, \quad \varepsilon > 0, \tag{6.12}$$

where $H(\pi)$ is the Shannon entropy of π.

Whilst the Sinkhorn divergence was also computed between two mini-batches, the authors of COT-GAN noticed the bias was better reduced by the mixed Sinkhorn divergence,

$$\begin{aligned}\widehat{\mathcal{W}}^{\text{mix}}_{c,\varepsilon}(\hat{\mu}, \hat{\nu}, \hat{\mu}', \hat{\nu}') := \mathcal{W}_{c,\varepsilon}(\hat{\mu}, \hat{\nu}) + \mathcal{W}_{c,\varepsilon}(\hat{\mu}', \hat{\nu}') \\ - \mathcal{W}_{c,\varepsilon}(\hat{\mu}, \hat{\mu}') - \mathcal{W}_{c,\varepsilon}(\hat{\nu}, \hat{\nu}'),\end{aligned} \tag{6.13}$$

where $\hat{\mu}$ and $\hat{\mu}'$ represent the empirical measures corresponding to two different mini-batches of the training data, and $\hat{\nu}$ and $\hat{\nu}'$ are the ones to the generated samples.

Finally, we have the following objective function for COT-GAN:

$$\inf_\theta \sup_\varphi \left\{ \widehat{\mathcal{W}}^{\text{mix}}_{c^{\mathcal{K}}_\varphi,\varepsilon}(\hat{\mu}, \hat{\nu}_\theta, \hat{\mu}', \hat{\nu}'_\theta) - \lambda[p_{\mathbf{M}_{\varphi_2}}(\hat{\mu}) + p_{\mathbf{M}_{\varphi_2}}(\hat{\mu}')] \right\}, \tag{6.14}$$

where the role of discriminator is played by the cost function $c^{\mathcal{K}}$ parameterised by φ, and $p_{\mathbf{M}_{\varphi_2}}$ is the martingale penalisation required for the formulation of the new class of costs $\mathcal{C}_{\varphi}^{\mathcal{K}}$, and φ_2 is a subset of the discriminator parameters φ, see details in Appendix C. Similar to common GAN training scheme, the discriminator learns the worst-case cost $c_{\varphi}^{\mathcal{K}}$ by maximising the objective over φ, and the generator is updated by minimising it over θ.

To summarize the intuition of COT-GAN, let us picture the following example: We want to generate global weather patterns based on training data representing global surface temperatures over several days. A traditional Wasserstein GAN would compute the loss by computing the Wasserstein distance between samples of real and generated data, agnostic of the sequential nature of the data. COT-GAN on the other hand actively incorporates sequentiality: In the COT-GAN loss, the probability mass moved to the target sequence at time t can only depend on the source sequence up to time t. This formulation is beneficial for the learning of highly time-dependent data generating processes, such as global surface temperatures. Weather patterns are characterized by a myriad of time-dependent phenomena, such as high- and low-pressure systems or extreme events such as hurricanes or tropical cyclones. As such, COT-GAN provides an improved framework for modeling such data.

6.2.3 SPATE-GAN

In SPATE-GAN, we integrate our newly devised spatio-temporal metric into the COT-GAN objective function. We compute the embedding for each x_{it}^d and y_{it}^d in minibatches $\{x_{1:T}^d\}_{d=1}^m$ and $\{y_{1:T}^d\}_{d=1}^m$ by:

$$\hat{x}_{it}^d = S_{it}(x_{1:T}^d, w) \quad \text{and} \quad \hat{y}_{it}^d = S_{it}(y_{1:T}^d, w), \tag{6.15}$$

where the binary spatial weight matrix w is pre-defined.

The corresponding embeddings are then concatenated with the training data and generated samples on the channel dimension. We define the empirical measures for the concatenated sequences as:

$$\widehat{\mu}^e := \frac{1}{m}\sum_{d=1}^{m} \delta_{\text{concat}(x_{1:T}^d, \hat{x}_{1:T}^d))}, \tag{6.16}$$

$$\widehat{\nu}_{\theta}^e := \frac{1}{m}\sum_{d=1}^{m} \delta_{\text{concat}(y_{1:T}^d, \hat{y}_{1:T}^d)}, \tag{6.17}$$

where concat$(.,.)$ is an operator that concatenates inputs along the channel dimension.

We thus arrive at the objective function for SPATE-GAN:

$$\inf_{\theta}\sup_{\varphi}\Big\{\widehat{\mathcal{W}}^{\text{mix}}_{c^{\mathcal{K}}_{\varphi},\varepsilon}(\widehat{\mu}^{e},\widehat{\nu}^{e}_{\theta},\widehat{\mu}^{e'},\widehat{\nu}^{e'}_{\theta}) - \lambda\big[p_{\mathbf{M}_{\varphi_2}}(\widehat{\mu}^{e}) + p_{\mathbf{M}_{\varphi_2}}(\widehat{\mu}^{e'})\big]\Big\}. \tag{6.18}$$

We maximise the objective function over φ to search for a worst-case distance between the two empirical measures, and minimise it over θ to learn a distribution that is as close as possible to the real distribution. The algorithm is summarised in Algorithm 4. Its time complexity scales as $\mathcal{O}((J+2n)LTm^2)$ in each iteration where J is the output dimension of the discriminator (see Appendix C for details), and L is the number of Sinkhorn iterations (see [35, 57] for details).

In the experiment section, we will compare SPATE-GAN with three different expectations $\mu_{it}^{(k)}$, $\mu_{it}^{(kw)}$ and $\mu_{it}^{(ksw)}$ in the computation of SPATE. Hence, we denote the corresponding models as SPATE-GAN^{k}, SPATE-GAN^{kw}, and SPATE-GAN^{ksw}, respectively.

Last, we emphasise that, although all three embeddings consider the space-time interactions in a certain way, the non-anticipative assumption of $\mu_{it}^{(ksw)}$ is consistent with the generating process of the type of data we are investigating. As the causality constraint in COT-GAN also restricts the search of transport plans to those that satisfy non-anticipative transporting of mass, SPATE-GAN^{ksw} is a model that fully respects temporal causality in learning, whilst SPATE-GAN^{k} and SPATE-GAN^{kw} also combine information from the future.

6.3 Experiments

6.3.1 Data

To empirically evaluate SPATE-GAN, we use three datasets characterised by different spatio-temporal complexities.

Extreme Weather (EW): This dataset, introduced by Racah et al. [143], was originally proposed for detecting extreme weather events from a range of climate variables (e.g. zonal winds, radiation). Each of these climate variables is observed four times a day for a 128×192 pixel representation of the whole earth. We chose to model surface temperature as it comes with several interesting spatio-temporal characteristics: It exhibits both static (e.g. continent outlines) and dynamic patterns as well as abnormal patterns (e.g. in the presence of tropical cyclones or atmospheric rivers). Furthermore, simulating climate data is an important potential downstream application of deep generative models. Our final dataset of surface temperatures includes 146 instances of 128×192

Algorithm 4: training SPATE-GAN by SGD

Data: $\{x^d_{1:T}\}_{d=1}^n$ (input data), ζ (latent distribution)
Parameters: θ_0, φ_0 (parameter initializations), m (batch size), ε (regularization parameter), α (learning rate), λ (martingale penalty coefficient)
Initialize: $\theta \leftarrow \theta_0$, $\varphi \leftarrow \varphi_0$
for $b = 1, 2, \ldots$ **do**
  Sample $\{x^d_{1:T}\}_{d=1}^m$ from real data;
  Sample $\{z^d_{1:T}\}_{d=1}^m$ from ζ;
  Generate sequences from latent: $(y^d_{1:T}) \leftarrow g_\theta(z^d_{1:T})$;
  Compute the embeddings: $\hat{x}^d_{it} = S_{it}(x^d_{1:T}, w), \quad \hat{y}^i_{it} = S_{it}(y^i_{1:T}, w)$;
  Concatenated the data with embeddings:
  $\mathrm{concat}(x^d_{1:T}, \hat{x}^d_{1:T}), \mathrm{concat}(y^d_{1:T}, \hat{y}^d_{1:T})$;
  Update discriminator parameter:
  $\varphi \leftarrow \varphi + \alpha\nabla_\varphi(\widehat{\mathcal{W}}^{\mathrm{mix}}_{c^K_\varphi,\varepsilon}(\widehat{\mu}^e, \widehat{\nu}^e_\theta, \widehat{\mu}^{e'}, \widehat{\nu}^{e'}_\theta) - \lambda[p_{\mathbf{M}_{\varphi_2}}(\widehat{\mu}^e) + p_{\mathbf{M}_{\varphi_2}}(\widehat{\mu}^{e'})])$;
  Sample $\{z^d_{1:T}\}_{d=1}^m$ from ζ;
  Generate sequences from latent: $(y^d_{1:T}) \leftarrow g_\theta(z^d_{1:T})$;
  Compute the embeddings: $\hat{x}^d_{it} = S_{it}(x^d_{1:T}, w), \quad \hat{y}^d_{it} = S_{it}(y^d_{1:T}, w)$;
  Concatenated the data with embeddings:
  $\mathrm{concat}(x^d_{1:T}, \hat{x}^d_{1:T}), \mathrm{concat}(y^d_{1:T}, \hat{y}^d_{1:T})$;
  Update generator parameter:
  $\theta \leftarrow \theta - \alpha\nabla_\theta\left(\widehat{\mathcal{W}}^{\mathrm{mix}}_{c^K_\varphi,\varepsilon}(\widehat{\mu}^e, \widehat{\nu}^e_\theta, \widehat{\mu}^{e'}, \widehat{\nu}^{e'}_\theta)\right)$;
end

pixel frames over 10 time-steps.

LGCP: This dataset represents the intensities (number of events in a grid cell) of a log-Gaussian Cox process (LGCP), a continuous spatio-temporal point process. LGCPs are a popular class of models for simulating contagious spatio-temporal patterns and have various applications, for example in epidemiology. We simulate 300 different LGCP intensities on a 64×64 grid over 10 time steps using the R package *LGCP* [168].

Turbulent Flows (TF): This dataset, proposed by Wang et al. [187], simulates velocity fields according to the Navier-Stokes equation. This is a class of partial differential equations describing the motion of fluids. Fluid dynamics and simulation is another potential application of deep generative models. Following the approach of Wang et al. [187], we divide the data into 7000 instances of 64×64 pixel frames over 7 time-steps. Please note that we only use the first velocity field, so that all our utilised datasets are single-channel.

6.3.2 Baselines and Evaluation Metrics

We use COT-GAN [194] and GAN proposed by [57], which we name as SinkGAN, as base models. We augment both models with our new embedding loss, using SPATE with k, kw and ksw configurations. We refer to all models using a COT-GAN backbone in combination with our new embedding loss as SPATE-GAN. We further denote the SinkGAN models corresponding to three SPATE settings as SinkGAN^{k}, SinkGAN^{kw} and SinkGAN^{ksw}. To compare our approach to a non-time-sensitive embedding, we also deploy models using the Moran's I metric using the same embedding loss procedure, denoted as COT-GAN^{M} and SinkGAN^{M}.

To compare our GAN output to real data samples, we use three different metrics: Earth Mover Distance (EMD), Maximum Mean Discrepancy (MMD) [21] and a classifier two-sample test based on a k-nearest-neighbour (KNN) classifier with $k = 1$ [118]. All these measures are general purpose GAN metrics. While GAN metrics specialised on video data exist, they rely on extracting features from models pre-trained on three-channel RGB video data. As we are working with single-channel, non-image data, these methods are not applicable in our case.

To compute our three metrics, let us first assume that we have a set of real data samples ($\mathcal{P}$) and synthetic data samples ($\mathcal{S}$). EMD is defined as:

$$EMD(\mathcal{P}, \mathcal{S}) = \min_{\phi:\mathcal{P}\to\mathcal{S}} \sum_{p\in\mathcal{P}} \|p - \phi(p)\| \tag{6.19}$$

where $\phi : \mathcal{P} \to \mathcal{S}$ is a bijection. MMD is defined as:

$$\begin{aligned} \widehat{MMD}^2(\mathcal{P}, \mathcal{S}) = \frac{1}{n(n-1)} \sum k(p,p) + \\ \frac{1}{n(n-1)} \sum k(s,s) - \frac{2}{n^2} \sum k(p,s) \end{aligned} \tag{6.20}$$

where k denotes a positive-definite kernel (e.g. RBF kernel) and n is the number of (real or synthetic) samples.

Lastly, to compute the KNN score, we first split our real and synthetic samples $\mathcal{P}$ and $\mathcal{S}$ into training and test datasets $\mathcal{D}_{tr}$ and $\mathcal{D}_{te}$ so that $\mathcal{D} = \mathcal{D}_{tr} \cup \mathcal{D}_{te}$. We train the KNN classifier $f : \mathcal{X}_{tr} \to [0, 1]$ using training data. The accuracy of the trained classifier is then obtained using test samples $\mathcal{D}_{te}$ and given as:

$$\hat{t} = \frac{1}{n_{te}} \sum_{(z_i,l_i)\in\mathcal{D}_{te}} \mathbb{I}\left[\left(f(z_i) > \frac{1}{2}\right) = l_i\right] \tag{6.21}$$

where $f(z_i)$ estimates the conditional probability distribution $p(l = 1|z_i)$. A classifier accuracy approaching random chance (50%) indicates better synthetic data. As suggested by Lopez-Paz and Oquab [118], we use a 1-NN classifier to

obtain the score.

6.3.3 Experimental Setting

We compare SPATE-GAN to a range of baseline configurations. We use the same GAN architecture for all these settings to ensure comparability. Our GAN generators feed the noise input through two LSTM layers with batch normalization to obtain time-dependent features. These are then mapped into the desired shape for deconvolutional operations using a fully-connected layer with batch normalization and a leaky ReLU activation. Lastly, 4 deconvolutional layers map the output into video frames, all also with batch normalization and leaky ReLU activations, except the last layer. Our discriminators initially feed video input through three convolutional layers with batch normalization and leaky ReLU activations. The outputs from the convolutional operations are then reshaped and fed through two LSTM layers, the first of which is followed by batch normalization, to create the final discriminator outputs.

We set the following hyperparameters throughout model training: Hyperparameter settings are as follows: the Sinkhorn regulariser is set to $\epsilon = 0.8$, with $L = 100$ Sinkhorn iterations. The lengthscale is set to $l = 20$ and the martingale penalty to $\lambda = 1.5$. We use the Adam optimizer with learning rate 0.0001, beta values of $\beta_1 = 0.5$ and $\beta_2 = 0.9$. All our models are implemented in PyTorch and trained for $60,000$ iterations. Our experiments are conducted on a single Geforce 1080Ti or RTX 3090 GPU. Further training details can be found in Appendix C.

6.3.4 Results

Results from our experiments are shown in Table 6.1. Visual comparisons between real and generated data from the different models are shown in Figures 6.3,6.4, and 6.5. For larger figures including results from all tested model configurations, please see the Appendix C. Through all experiments we can observe that SPATE-GAN^{ksw} consistently outperforms the competing approaches, achieving the best scores across all datasets and evaluation metrics.

This finding is interesting as the ksw setting theoretically looses information over the k and kw approaches, which both have access to future time steps when calculating the SPATE metric. Nevertheless, this result underlines the strong synergies between SPATE^{ksw} and the COT-GAN backbone: The metric is calculated in sequential fashion and thus respects the same causality constraints that restrict COT-GAN. As such, the outcome, while noteworthy, is not surprising.

This result is strengthened by a comparison with the SinkGAN-based approaches: SinkGAN does not follow the same restrictions and, as we observe,

Table 6.1: Evaluations for LGCP, EW and TF datasets. Lower values in EMD and MMD indicate better sample quality, while values close to 0.5 are more desirable for KNN.

LGCP	EMD	MMD	KNN
SinkGAN	12.46 (0.02)	0.38 (0.001)	0.14 (0.001)
SinkGAN^{M}	12.46 (0.02)	0.38 (0.001)	0.14 (0.001)
SinkGAN^{k}	12.65 (0.03)	0.38 (0.001)	0.15 (0.001)
SinkGAN^{kw}	10.60 (0.01)	0.63 (0.008)	0.30 (0.002)
SinkGAN^{ksw}	13.33 (0.01)	0.36 (0.001)	0.38 (0.003)
COT-GAN	12.38 (0.02)	**0.30 (0.001)**	0.20 (0.004)
COT-GAN^{M}	12.38 (0.02)	**0.30 (0.001)**	0.20 (0.004)
SPATE-GAN^{k}	11.56 (0.02)	0.32 (0.01)	0.31 (0.01)
SPATE-GAN^{kw}	10.92 (0.03)	0.64 (0.035)	0.15 (0.006)
SPATE-GAN^{ksw}	**10.47 (0.02)**	**0.30 (0.001)**	**0.39 (0.01)**
Extreme Weather			
SinkGAN	29.40 (0.05)	0.49 (0.001)	0.41 (0.004)
SinkGAN^{M}	29.27 (0.05)	0.72 (0.002)	0.22 (0.01)
SinkGAN^{k}	32.57 (0.03)	0.81 (0.001)	0.16 (0.004)
SinkGAN^{kw}	32.78 (0.05)	0.81 (0.001)	0.18 (0.004)
SinkGAN^{ksw}	30.00 (0.04)	0.50 (0.001)	0.41 (0.004)
COT-GAN	26.66 (0.09)	0.43 (0.002)	**0.42 (0.002)**
COT-GAN^{M}	36.42 (0.14)	0.65 (0.002)	0.09 (0.01)
SPATE-GAN^{k}	33.58 (0.07)	0.73 (0.002)	0.15 (0.01)
SPATE-GAN^{kw}	33.36 (0.09)	0.72 (0.002)	0.13 (0.003)
SPATE-GAN^{ksw}	**26.24 (0.07)**	**0.42 (0.002)**	**0.42 (0.002)**
Turbulent Flows			
SinkGAN	26.52 (0.007)	1.23 (0.001)	0.15 (0.001)
SinkGAN^{M}	28.02 (0.005)	1.22 (0.0002)	0.01 (0.002)
SinkGAN^{k}	28.14 (0.002)	1.32 (0.002)	0.08 (0.001)
SinkGAN^{kw}	30.98 (0.001)	1.50 (0.001)	0.03 (0.001)
SinkGAN^{ksw}	25.47 (0.008)	1.24 (0.0002)	0.13 (0.002)
COT-GAN	27.03 (0.01)	1.22 (0.001)	**0.16 (0.002)**
COT-GAN^{M}	24.93 (0.01)	1.19 (0.001)	0.09 (0.002)
SPATE-GAN^{k}	25.70 (0.02)	1.21 (0.001)	0.12 (0.003)
SPATE-GAN^{kw}	24.30 (0.002)	1.42 (0.001)	0.13 (0.004)
SPATE-GAN^{ksw}	**22.98 (0.01)**	**1.16 (0.001)**	**0.16 (0.002)**

is not improved as consistently by the SPATE-based embedding losses. In fact, in some cases the naive SinkGAN performs better than its derivatives using SPATE or Moran's I based embedding losses.

We also observe that throughout all settings, models using Moran's I perform similarly to their naive counterparts. This confirms that in fact, simply using measures of spatial autocorrelation computed over a sequence is not sufficient for capturing complex spatio-temporal effects. On the contrary, the other two SPATE settings, k and kw, both appear to have beneficial effects

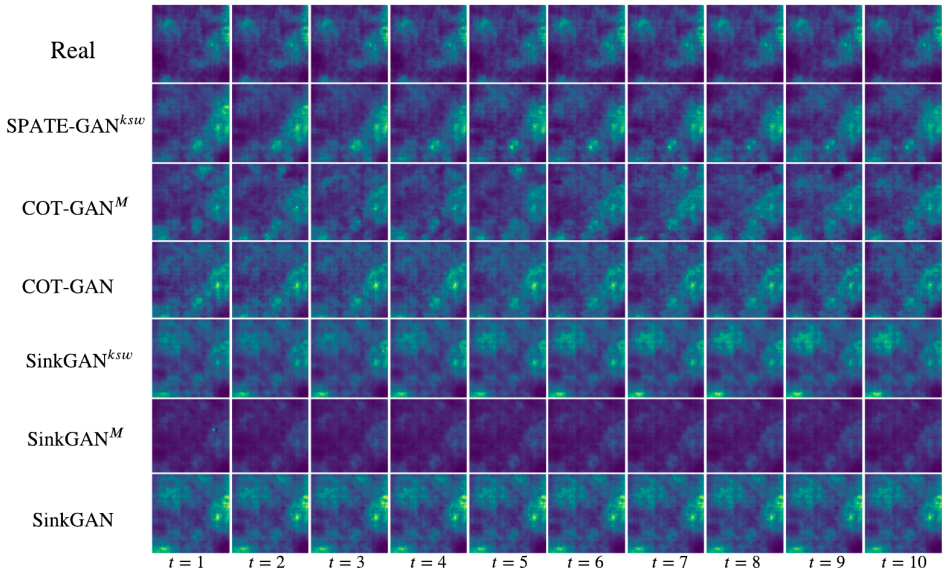


Figure 6.3: Selected samples of the LGCP dataset.

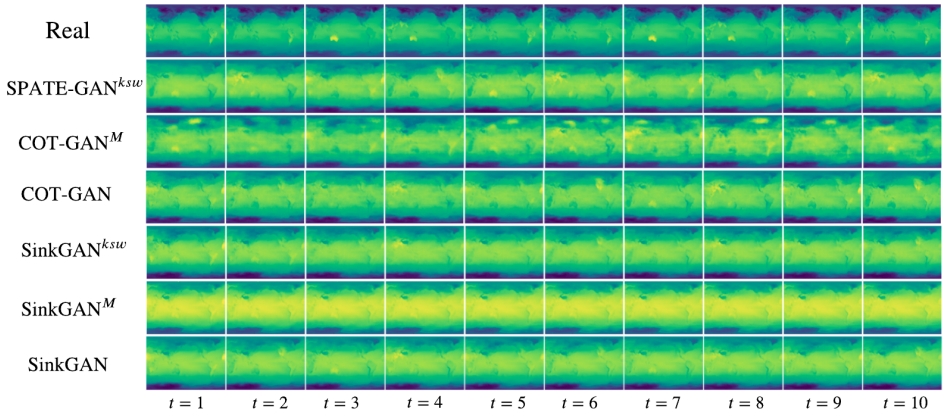


Figure 6.4: Selected samples of the Extreme Weather dataset.

and improve performance.

In summary, our results highlight how COT-GAN combined with a non-anticipative measure of space-time association can improve the modelling of complex spatio-temporal patterns. This finding represents another step on the way towards deep learning methods specialised on the dynamics driving many systems on our planet.

Furthermore, we provide an investigation on the impact of the lengthscale parameter l in the spatio-temporal expectations for $l \in \{1, 10, 20, 30, 50\}$. As shown in Figure 6.6, $l = 20$ leads to better EMD and KNN results whilst all MMD scores remain unchanged. For the results presented in this paper, we set $l = 20$ in all our experiments.

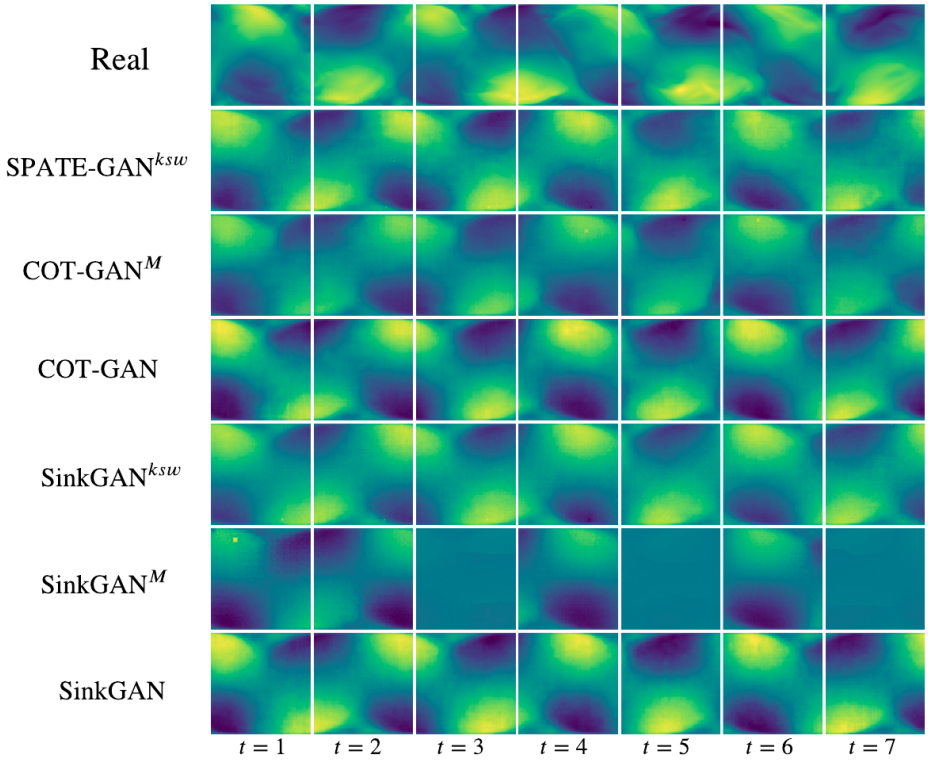


Figure 6.5: Selected samples of the Turbulent Flows dataset.

6.4 Discussion

We now want to discuss this chapter with respect to existing literature, as well as the broader scope of this dissertation.

6.4.1 Autocorrelation Metrics for Spatio-temporal Phenomena

Analysing autoregressive patterns in spatial and spatio-temporal data has a long tradition in different academic domains (e.g. GIS, ecology) which over time developed diverse measures to describe these phenomena. Applications of the Moran's I metric [2] range from identifying rare earth contamination [201]

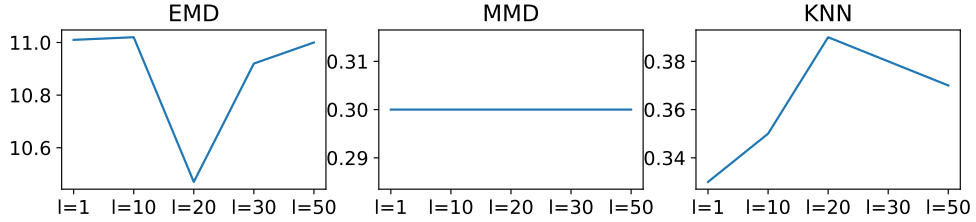


Figure 6.6: Evaluations of SPATE-GAN^{ksw} (left: EMD, middle: MMD, and right: KNN) on LGCP dataset given lengthscale $l \in \{1, 10, 20, 30, 50\}$.

to analysing land cover change patterns [36]. Moran's I has also seen some expansions into the spatio-temporal domain. Matthews et al. [124] use the metric iteratively to model disease spread over time. Lee and Li [108] and Gao et al. [53] propose novel spatio-temporal expansions of the Moran's I metric, returning static outputs at a purely spatial resolution. Siino et al. [158] design an extended Moran's I for spatio-temporal point processes. However, to the best of our knowledge, neither the Moran's I nor its spatio-temporal extensions have been applied to discrete spatio-temporal video data. It is evident that metrics of spatio-temporal autocorrelation can provide meaningful embeddings of complex data, capturing underlying patterns throughout a range of different application domains.

6.4.2 Deep Learning & GANs for Spatial and Spatio-temporal Data

Deep learning describes a powerful family of methods capable of dealing with the highly complex and non-linear nature of many real world spatial and spatio-temporal patterns [5, 12, 31, 58, 195]. Paradigms like physics-informed deep learning aim to devise methods which integrate (geo)physical constraints explicitly into neural network models [187]. There is also an increasing number of studies tackling specific challenges associated with geographic data: Mai et al. [121] and Yin et al. [198] propose context-aware vector embeddings for geographic coordinates. All these studies highlight the benefits of explicitly encoding spatial context into neural networks to improve performance.

Narrowing down on the GAN context specifically, we find that spatio-temporal applications have mostly focused on video data [88, 174, 194]. Beyond this, GANs have been used for conditional density estimation of traffic [205], trajectory prediction [76] or extreme weather event simulation [99]. Nevertheless, to the best of our knowledge, metrics capturing spatio-temporal autocorrelation have never been integrated into GANs. As previous studies highlight the value on encoding spatial context, this chapter seeks to provide a first-principle approach of integrating metrics of spatio-temporal autocorrelation into GANs for modelling of complex spatio-temporal patterns.

6.4.3 Embedding Loss Functions

As SPATE-GAN integrates spatio-temporal metrics into COT-GAN as embeddings, here we briefly revisit previous work and insights into embedding losses. Embedding losses describe approaches where the loss function of a model is computed on an embedding of the data. This can have desirable outcomes, such as training stability or focusing on specific patterns in the data. Embedding losses have become popular in computer vision over the last years: Ghafoorian

et al. [61] use embedding losses to improve GAN-based lane detection. Filntisis et al. [47] use visual-semantic embedding losses to improve predictions of bodily expressed emotions. Wang et al. [186] introduce CLIFFNet, using hierarchical embeddings of depth maps for molecular depth estimation. Bailer et al. [11] introduce a threshold loss to improve optical flow estimation. It is clear that embedding losses have shown great potential for particularly challenging visual problems, especially those involving complex spatio-temporal dynamics. Lastly, after discussion model selection (see chapter 3), auxiliary task learning (see chapter 4) and feature learning (see chapter 5), embedding losses are yet another approach for integrating geospatial embeddings into neural network models.

6.5 Summary

Recent studies have called for more research into improving deep learning models for spatio-temporal earth systems data [148]. Other academic domains have dealt with these data for many decades and have developed methods for capturing specific spatial and spatio-temporal effects. Inspired by their approaches, we devise SPATE, a measure of spatio-temporal association capable of detecting emerging space-time clusters and homogeneous areas in the data. We then develop a novel that embeds loss for video GANs using SPATE as a means of reinforcing the learning of these patterns-of-interest. Our new generative modelling approach, SPATE-GAN, shows performance increases on a range of different datasets emulating the real-world complexities of spatio-temporal dynamics. We highlight this throughout experiments on three different datasets, representing real-world spatio-temporal phenomena: spatio-temporal point processes, broadly used to model disease spread, global weather patterns represented by surface temperatures and a turbulent flows simulation. SPATE-GAN shows improved performance on all of these datasets, which speaks to its generalisability across different application domains unified by their spatio-temporal dynamics. As such, this study highlights how domain expertise from applied academic areas can help to motivate methodological advances in machine learning. SPATE-GAN's limitations include its increased computational cost, particularly for large spatial or temporal dimensions, stemming from the computation of the SPATE embedding. While we find the lengthscale parameter l to consistently perform optimally at $l = 20$, this might be different for other dataset and would need to be tested and adjusted accordingly.

Revisiting the objectives of this dissertation, in this chapter we have proposed a novel functional embedding for discrete spatio-temporal patterns, SPATE. We then use this metric to construct an embedding loss for a spatio-

temporal GAN. After looking into model selection, auxiliary task learning and feature learning in the previous chapters, this chapter completes our study of four different options for integrating geographic domain expertise into neural network models.

Chapter 7

Conclusions and Future Work

In this dissertation, we have proposed different approaches for integrating geospatial domain expertise into neural network models. We have tested these approaches in extensive experiments on a range of generative and predictive spatial modelling tasks. In the final chapter, we will first summarise our contributions in section 7.1. In section 7.2 we will discuss our findings with respect to the geospatial machine learning challenges that we highlighted in the introduction, section 1.1.2. We will then discuss potential applications of our work, focusing on the urban analytics space, in section 7.3. Lastly, we will elaborate on future research directions in section 7.4 and provide concluding remarks in section 7.5.

7.1 Summary of Contributions

This dissertation advances research on the modelling of geospatial data by merging ideas from the GIS and machine learning. Recalling Chapter 2.3, our studies rely on two building blocks that are essential to understanding the contributions of this dissertation:

- **Geospatial context embeddings:** All technical chapters make use of geospatial context embeddings. These are transforms of geospatial data that seek to capture information on underlying spatial patterns. As discussed in chapter 2.3, we distinguish between *functional embeddings*, (e.g. the Moran's I metric) and *parametric neural network embeddings* (e.g. the positional encoder), introduced in chapter 5.
- **Mechanisms for integrating domain expertise into neural networks:** We identify four mechanisms for integrating geospatial domain expertise into neural network models. These approaches, discussed extensively in chapter 2.2.3, are *model selection*, *auxiliary task learning*, *feature learning* and *embedding loss functions*.

In each of our four technical chapters, we combine these building blocks to devise geographically explicit neural network models. The contributions of each chapter are listed below:

- Chapter3 proposes to use the Moran's I metric for model selection.
  - We propose a novel conditional GAN, SpaceGAN, conditioned on the features of each observations spatial neighborhood. This approach allows the model to learn spatial dependencies and patterns contained within the data.
  - We devise a novel metric, MIE, based on the local Moran's I measure to evaluate ability of GAN generated data to reproduce spatial autocorrelation present in the real data. We deploy this metric for model selection, selecting the best generator out of a set of generators.
  - We propose the use of SpaceGAN with MIE model selection in an ensemble learning setting where synthetic, SpaceGAN-generated data is used to train an ensemble of base learners.
- Chapter4 introduces an auxiliary task learning framework using the Moran's I metric.
  - We propose a multi-resolution extension of the local Moran's I metric, computed through repeated downsampling of the input to compute the Moran's I at coarsened resolutions, before upsampling again to the original resolution.
  - We propose to use single- and multi-resolution local Moran's I in an auxiliary task learning setting. This "nudges" neural network models to capture spatial autocorrelation as part of the learning process. We refer to this framework as SXL.
  - We provide an uncertainty weighting scheme for auxiliary task GANs, building on research by Cipolla et al. [32], which allows parameter-free learning of loss weights.
- Chapter5 presents a feature learning approach for graph neural networks using neural network-based geographic coordinate embeddings.
  - We devise a novel GNN method for geographic coordinates. Our approach concatenates node features with the vector embeddings of a context-aware positional encoder which is learnt throughout the training process.
  - Training is conducted on a subset of observations (minibatches) at each training step. The same point might can thus have a different

set of neighbours at different training steps. This training approach makes the training more robust and improves generalisation.

- Chapter6 expands the Moran's I metric to spatio-temporal data and proposes a new embedding loss function for GANs.
  - We propose a novel measure of spatio-temporal autocorrelation: SPATE. The metric builds on the Moran's I metric, but expands its intuition from purely spatial to spatio-temporal expectations.
  - We propose an embedding loss function based on SPATE to train video GANs. We augment a state of the art video GAN, COT-GAN [194] with an instance of our SPATE metric that respects the sequential nature of the data and is computed only using past time steps.

All chapters include experiments and application studies that highlight how each respective approach improves over naive, non-geographically-explicit baselines. Altogether, the experiments showcase how the integration of geospatial domain expertise in the form of contextual embeddings can improve the performance of neural network models on a range of relevant real-world tasks.

7.2 Discussion of Impacts

We now want to revisit the challenges of geospatial machine learning, posed in the seminal review study by Reichstein et al. [148] and highlighted in chapter 1.1.2. Here, we will discuss how this dissertation proposes avenues for tackling the outlined problems.

Reichstein et al. [148] view the integration of **physical constraints** as one of the key challenges for improving neural networks in the geospatial domain. While this dissertation is not concerned with the integration of hard constraints (e.g. geophysical laws), the use of metrics measuring known spatial phenomena such as autocorrelation can be seen as integrating "soft" constraints. In many geospatial settings, such as modelling rainfall or predicting housing prices, we know about spatial autocorrelation in the data a priori. We can exploit this knowledge by explicitly integrating it into the models. We propose such approaches for the Moran's I metric in chapter 4 and for our novel SPATE metric in chapter 6: Both approaches can be seen as enforcing a level of spatial or spatio-temporal coherence in the learning process. Nonetheless, this work more touches on this issue rather than addressing it directly. Specifically, throughout the thesis we focus on measures of spatio-temporal autocorrelation, which is only one of a myriad of effects in dynamics systems.

Another challenge noted by Reichstein et al. [148] is the **complexity and scale-sensitivity** of geospatial data. Off-the-shelf neural network models can struggle to capture complex and non-linear spatial effects over different spatial scales. We tackle this problem in the chapters 4 and 5. In chapter 4, we devise a novel multi-resolution version of the local Moran's I metric of spatial autocorrelation, to overcome its known scale sensitivity. In chapter 5, we propose the use of a neural network-based coordinate embedding that learns geographic context throughout the training process. Specifically, our embedding can also learn to adapt to different spatial scales and neighbourhood structures; thus, the static nature of the GNN backbone, which assumes a predefined (spatial) graph, is improved. While our work addresses crucial aspects of the complexity and scale-sensitivity challenge, our work is limited in scope. As mentioned in the previous paragraph, when it comes to the complexity of the data, we focus mostly on measurable dynamics in the form of spatial and spatio-temporal autocorrelation. Other complexities, though they might be implicitly learnt in approaches such as PE-GNN in chapter 5, are not addressed here. And while we do address the different scales of spatial effects throughout the thesis, all our experiments are conducted with single-resolution training data. To further advances neural network methods for geospatial data, approaches learning with and interactions between inputs at multiple resolutions will be required.

Lastly, the third challenge we address in this dissertation is that of **label scarcity**. Reichstein et al. [148] note that geospatial applications can often lack sufficient training labels for training generalisable models. And while the authors see this challenge more present with Earth observation data, it also has some relevance to urban applications. For example, in some urban settings, training labels are sensitive and can only be obtained in small numbers, so as to protect citizens privacy (e.g. crime data, personal location data). Collecting labels can also be expensive, particularly in low-resource settings (e.g. within countries in the Global South). To overcome this, synthetic data samples from simulators might offer valuable alternatives. In chapters 3, 4 and 6 we introduce methods for generating high-quality, synthetic spatial data using GANs. Particularly, our experiments in chapter 3 show the potential of such approaches for improving predictive downstream tasks in scarce data settings. A limiting factor is the high specialization of all these approaches. Chapters 3, 4 and 6 all represent different application settings: tabular data with geographic coordinates, data on a discrete regular grid (i.e. single-channel image data) and spatio-temporal data, respectively. They all require different, specialized modeling frameworks and there is no one-size-fits-all solution. To enable a more broad adaption of these models, more work focused on harmonizing spatial and spatio-temporal modeling approaches in a generalisable framework needs

to be conducted.

The other two challenges posed by Reichstein et al. [148] include **interpretability** and **computational demand**. While we do not center this in our work, some of our experiments highlight how geographically-explicit neural network models can accelerate training convergence. Specifically, enforcing learning of spatial patterns in models, be that through auxiliary tasks or embedding losses, allows us to better comprehend learning processes and make assumptions on model outputs. It is also important to mention here that as this thesis is concerned with neural networks, there is very limited room for the interpretation of their inner workings, as neural networks are inherently black-box models. Lastly, the geographic embeddings, functional or parametric, that are used in this thesis can have non-trivial computational requirements. Particularly the computational cost of the SPATE metric introduced in chapter 6 scales poorly into large spatial and temporal dimensions. Especially in big-data settings, such as are common in for example Earth observation, this can be a limiting factor.

7.3 Applications

Beyond our contributions to methodological advances in geospatial machine learning, there are many avenues for potentially impactfull applications of this dissertation. Here, we want to discuss some of these applications in detail:

7.3.1 Operations and Deployment Optimisation for Urban Services

As discussed in our introduction chapter 1.1.1, cities produce large amounts of geospatial data on a daily basis. At the same time, they are becoming more and more crowded as the global trend of urbanisation accelerates. Urban services, from transportation systems to energy grids, are charged with adapting to these increasing demand levels. Together, urban big data and geospatial machine learning approaches can help to tackle these challenges. A great example is the case of shared urban mobility systems such as bike sharing or electric vehicle (EV) sharing. Previous research has highlighted the importance spatial and spatio-temporal effects in mobility demand [131, 190]. While GNNs have proven to be powerful tools for optimising shared mobility systems to meet demand, research has also shown that integrating geographic context (e.g. on urban structures and local amenities) is crucial to improving these models [119, 120]. Here, learnable geographic context embeddings, such as the one proposed in chapter 5, offer great potential for improving GNN backbones.

7.3.2 Digital Twins of Cities and our Planet

The term "digital twins" was first coined by Shafto et al. [156] and describes a digital counterpart to some physical entity, which allows for the modelling and simulation of different conditions and settings. Digital twins have become particularly popular for studying cities [16] and Earth systems, such as oceans [13]. Keeping with the example of cities, a digital twin might be used to simulate the effects of a new policy (e.g. a ban on cars in the city center) or new infrastructure (e.g. the opening of a new subway line) on the rest of the urban system. This example reveals the key challenge faced by the digital twins framework: it aims to represent a hyper-complex, multi-layered process in digital form. Of course, we are not at a point where digital twins can hope to emulate all the intricate details of urban systems. However, as Batty [16] notes, "some models are closer to the real thing than others, with the whole panoply of models ranging from thought experiments which are entirely conceptual to closely tailored digital representations that attempt to mirror as many features of the real system as possible". Thus, Accurately representing spatial and spatio-temporal structures is key to developing powerful digital twins. This also corresponds to one of the key geospatial deep learning challenges identified by Reichstein et al. [148]. Batty [16] further notes that another challenge for digital twins is to "scale to the level of all the physical assets in the city", which entails capturing urban dynamics at different spatial scales. The technical chapters of this dissertation address these challenges in parts by devising methods to enforce the learning of spatial effects in neural network models (chapters 4, 5 and 6) and to learn at multiple spatial scales (chapters 4 and 5).

7.3.3 Synthetic Samples of Sensitive Geospatial Data

Geospatial data can be sensitive in nature. For instance, in city-level data, this can includes the locations of sexual crimes [97], which need to be kept private in order to protect survivors. Beyond cities, many ecological data are sensitive. For example, geo-tagged social media posts can reveal the locations of endangered species to poachers [18]. However, in both of these cases, the data can also be important for developing targeted policy interventions. Generative modelling of geospatial data offers an avenue for publishing synthetic data that are representative of an underlying sensitive real-world dataset, as highlighted in existing research [172, 199]. While this dissertation does not address such applications or provide privacy guarantees directly, we extensively discuss generative models (in chapters 3, 4 and 6), which might be used as backbones for such approaches.

7.3.4 Modelling Earth Systems and Climate Dynamics

Climate change is an essential challenge humanity faces today. To mitigate and adapt to climate change, we must build accurate models of geophysical systems (e.g. atmospheric or ocean currents) and climate dynamics. A recent review paper highlights the potential of machine learning techniques for tackling these challenges [153]. This includes models that are capable of emulating complex geographic patterns. For example, deep learning applications have recently been used to simulate fluid dynamics [187] and extreme weather events [99, 143]. Most chapters in this dissertation are relevant to such applications, even when they don't tackle this issue directly or at the scale of current real-world systems. In particular, chapter 6 explicitly uses example datasets representing global surface temperatures and turbulent flows for spatio-temporal modelling. Altogether, this this thesis serves as a primer and motivation for the further integration of geospatial-domain expertise in neural network models of our planet's dynamics.

7.3.5 Spatial and Spatio-temporal Epidemiology and Public Health

As the COVID-19 pandemic continues to hold our planet in a tight grab, the importance of accurately modelling disease spread over space and time has become apparent. Deep learning approaches are growing in popularity for such tasks, due to their flexibility and scalability [10, 185]. The increasing availability of other geospatial data sources (e.g. geo-tagged social media posts) can further improve such approaches, but requires models that are able to capture spatial and spatio-temporal interactions at different scales [207]. Epidemiological processes are often assessed using point processes, which are powerful tools to forecast disease spread. In chapter 6, we show that spatio-temporal GANs can emulate such processes and, once trained, produce high quality synthetic data without the need for computationally expensive numerical simulation. And while our experiments are conducted on simulated data and on a small scale, they nonetheless can be used as stepping stones and motivation for applications and further methodological improvements in the public health domain.

7.4 Future Work

In this section, we discuss promising avenues for future work, building on the findings of this dissertation.

7.4.1 General-purpose, Universal, Geographic Context Embeddings

Our work in chapter 5 touches on the idea of general-purpose geographic context embeddings. These describe the idea of a pretrained vector embedding (akin to approaches like *word2vec* [126] in NLP) that captures all available geographic context at a given location. For example, if we seek to predict the price of a house, this embedding would include information about the physical (e.g. distance to shops and road connectivity) and non-physical structure (e.g. local social media sentiment) of the vicinity. Such embeddings fuse data inputs from various sources and can be pre-trained in an unsupervised learning setting. First approaches in this direction exist already [198]; however, they currently offer limited flexibility and only represent selected data inputs. Future work might expand this research using the free-floating point location encoder proposed in chapter 5 and merging it with suitable methods for encoding multi-modal geospatial context. Multi-modal learning describes machine learning techniques working with different data modes (e.g. text and video data) simultaneously. These jointly learnt representations can often be more powerful than single-modal baselines [9].

7.4.2 Privacy-preserving Geospatial Machine Learning

As discussed in section 7.3.3, synthetic data offers a solution to a range of settings involving sensitive geospatial information [34]. While our work has contributed mechanisms for generating high-quality geospatial data, we have not integrated privacy guarantees into these models. There is a broad range of literature on machine learning approaches with global and local differential privacy [172, 199]; however, few such studies exist for explicitly geospatial domains. A potential avenue for future research is to merge our work on geospatial GANs with these approaches, to create classes of generative models that are able to learn the data generating process behind sensitive, real-world data. Such models could then be used to safely publish synthetic data, maintaining the properties of the underlying sensitive data.

7.4.3 Federated Geospatial Machine Learning

Federated learning [84] describes the decentralised training of learning algorithms on a set of edge devices. Naturally, these devices are distributed over space. While this can lead to challenges with respect to distributed computing, geospatially-explicit approaches can help to improve such modalities through encoding geo-locations throughout modelling [161]. Nonetheless, more research into this area is needed. Specifically, the effects of different spatial distances and resolutions should be further studied. This dissertation may help to inform and

improve geospatial federated learning approaches, and open up new research directions.

7.4.4 Machine Learning for Multi-layered and Multi-resolution Geospatial Data

Geographic data are often available at more than one spatial scale. While we propose metrics such as the multi-resolution Moran's I (see chapter 4), all experiments in this dissertation are conducted on single-scale geospatial data. The joint modelling of multiple resolutions is an important direction for future research. Within cities, data might be available at the location level (e.g. point-of-interest locations) and areal level (e.g. median income in a census tract) simultaneously. A downstream model that employs both location- and areal-level data must include an appropriate mechanism for fusing geospatial data at different resolutions. It also needs to reflect inherent interdependencies between resolutions. Research into scale-sensitive, adaptive models is of high importance to many urban modelling problems and directly relates to one of the key challenges of geospatial data, as outlined in chapter 2.1.2.

7.4.5 Learning Neighbourhoods and Areal Units

As discussed in chapter 2.1.1, many spatial metrics and models require a definition of spatial "neighbourhood". These are often chosen arbitrarily and can be ill-informed or suffer from scaling issues [45, 125, 204]. Rather than predefining spatial neighbourhoods based on domain-expertise (or a lack thereof), future machine learning approaches may seek to learn local neighbourhood structures throughout training; this enables finding the best possible spatial representation of the data. For example, GNN models may construct their input graph on the fly, where the radius around nodes defining spatial neighbourhood is a learnable parameter.

7.5 Final Remarks

In this dissertation, we have merged ideas from the GIS and machine learning communities to build geographically-explicit neural network models. We have documented the implementations of our methods and the associated experiments in publicly available repositories that contain our code and data. Specifically, the code for chapter 3 can be found at `https://github.com/konstantinklemmer/spacegan`, the code for chapter 4 at `https://github.com/konstantinklemmer/sxl`, the code for chapter 5 at `https://github.com/konstantinklemmer/pe-gnn` and the code for chapter 6 at `https://github.com/konstantinklemmer/spate-gan`.

Appendix A

Appendix for Chapter 3

A.I Experimental Data

Here we provide a more elaborate description of the datasets used for evaluating *Experiment 1* and *Experiment 2*.

Toy 1: We create a synthetic dataset of $n = 400$ observations. Following the notation $\mathbf{d} = (\mathbf{x}, y, \mathbf{c})$, we first set the spatial resolution, i.e. the spatial coordinates $\mathbf{c} = (c^{(1)}, c^{(2)})$:

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_{1,1} = (2.5, 2.5) & \mathbf{c}_{1,2} = (7.5, 2.5) & ... & \mathbf{c}_{1,400} = (97.5, 2.5) \\ \mathbf{c}_{2,1} = (2.5, 7.5) & ... & ... & ... \\ ... & ... & ... & ... \\ \mathbf{c}_{400,1} = (2.5, 97.5) & ... & ... & \mathbf{c}_{400,400} = (97.5, 97.5) \end{bmatrix} \tag{A.1}$$

We then add an independent feature x as a random draw from a Gaussian distribution with mean $\mu = 1$ and standard deviation $\sigma = 0$:

$$x \sim Normal(\mu, \sigma) \tag{A.2}$$

Now, we create the target variable y as a function of spatial coordinates $\mathbf{c}$ and the random noise $\mathbf{x}$ as follows:

$$y = \sin x + (c^{(1)} - c^{(2)})^2 \tag{A.3}$$

Toy 2: We create a synthetic dataset of $n = 841$ observations. We again

start by setting the spatial resolution, i.e. the spatial coordinates $\mathbf{c} = (c^{(1)}, c^{(2)})$:

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_{1,1} = (1.75, 1.75) & \mathbf{c}_{1,2} = (5.25, 1.75) & ... & \mathbf{c}_{1,841} = (99.75, 1.75) \\ \mathbf{c}_{2,1} = (1.75, 5.25) & ... & ... & ... \\ ... & ... & ... & ... \\ \mathbf{c}_{841,1} = (1.75, 99.75) & ... & ... & \mathbf{c}_{841,841} = (99.75, 99.75) \end{bmatrix} \tag{A.4}$$

We again add an independent variable x as a random draw from a Gaussian distribution with mean $\mu = 1$ and standard deviation $\sigma = 0$:

$$x \sim Normal(\mu, \sigma) \tag{A.5}$$

Lastly, we create the target variable y as a more complex function of spatial coordinates $\mathbf{c}$ and the random noise x as follows:

$$y = \sin(c^{(1)} + c^{(2)}) * 2\pi + \lfloor z \rfloor * 0.1c^{(1)} \tag{A.6}$$

where $z \sim U(1.75, 99.75) * 0.01$.

California Housing: This real world dataset, introduced by [86], is widely popular for analyzing spatial patterns and accessible via *Kaggle*[1] (it is also integrated into *sklearn*[2]). It contains $n = 20,640$ observations.

We can break the dataset down into the familiar notation $\mathbf{d} = (\mathbf{x}, y, \mathbf{c})$ as follows:

$$\mathbf{c} = (longitude, latitude) \tag{A.7}$$

$$\mathbf{x} = (housing\_median\_age, total\_rooms, total\_bedrooms, population, households, median\_income) \tag{A.8}$$

$$y = (median\_house\_value) \tag{A.9}$$

Infant Mortality: This is another standard dataset for spatial analysis, introduced by [122]. It is relatively small and comprised of infant mortality data from Auckland 1991. The $\mathbf{d} = (\mathbf{x}, y, \mathbf{c})$ follows:

$$\mathbf{c} = (longitude, latitude) \tag{A.10}$$

$$\mathbf{x} = (pop\_u5) \tag{A.11}$$

$$y = (deaths\_u5) \tag{A.12}$$

Election: This dataset, introduced by [86], provides district-level election results from the 1980 U.S. election. The regression task is to predict voter

[1] See: `https://www.kaggle.com/camnugent/california-housing-prices`

[2] See: `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_california_housing.html`

turnout using socio-economic features. The $\mathbf{d} = (\mathbf{x}, y, \mathbf{c})$ follows:

$$\mathbf{c} = (longitude, latitude) \tag{A.13}$$

$$\mathbf{x} = (pc\_college, pc\_home, pc\_income) \tag{A.14}$$

$$y = (pc\_turnout) \tag{A.15}$$

A.II Experimental Setting

The tables below provide details on architecture and configuration of the neural networks used in *SpaceGAN* during our experiments. Note that the kernel size parameter for **Toy 1** and **Toy 2** corresponds to the queen neighbourhood (for discrete spatial data) outlined in 3.1 and is the same neighbourhood that is used for spatial conditioning and spatial cross validation (see Appendix E). The kernel size for **California Housing 15** and **California Housing 50** corresponds to same kNN-neighbourhood (with $k = 50$) that is used for spatial conditioning and spatial cross-validation.

Table A.1: Dataset-specific configurations of the *SpaceGAN* architecture for the experiments.

Parameter	**Values**
Architecture	1D-CNN
Number of hidden layers	1
Training steps	20000
Batch Size	100
Optimizer	Stochastic Gradient Descent
Optimizer Parameters	learning rate = 0.01
Noise prior $p_{\mathbf{z}}(\mathbf{z})$	$N(0, 1)$
Snapshot frequency ($snap$)	500
Number of samples for evaluation	$C = 500$
Input features scaling function	Z-score (standardization)
Target scaling function	Z-score (standardization)

Table A.2: Overview of the general *SpaceGAN* architecture and its hyperparameters.

Parameter	**Toy 1**	**Toy 2**	**California 15**	**California 50**
(G, D) filters ($\lvert\mathcal{N}_i\rvert$)	(50, 50)	(100, 100)	(100, 100)	(200, 200)
(G, D) kernel size	(8, 8)	(8, 8)	(15, 15)	(50, 50)
(G, D) hidden layer function	(relu, tanh)	(relu, tanh)	(relu, tanh)	(relu, tanh)
(G, D) output layer function	(linear, sigmoid)	(linear, sigmoid)	(linear, sigmoid)	(linear, sigmoid)
Noise dimension $dim(\mathbf{z})$	8	8	15	15

A.III Neighbourhood conditioning

As outlined in the methodological section, we believe the choice of neighbourhood weights $w_{i,j}$ is of important for *SpaceGAN's* performance. While we test different configurations in our experiments, the reported results always apply the same neighbours to *SpaceGAN* conditioning, as are used for the spatial-cross validation process. For example, this implies that in **Toy 1** experiments,

we use queen neighbourhood (see Figure 3.1) spatial cross-validation as well as conditioning. This means that each datapoint is conditioned on the feature vectors of its queen neighbours.

We also provide results for the effect that different neighbourhood definitions in the same dataset have, using the **California Housing** dataset. Here, we once condition on 15-NN, once on 50-NN: Our findings show that the 50-NN conditioned *SpaceGAN* performs better, both in Experiment 1 and in Experiment 2. While the neighbourhood definition never "breaks" a model training, we will conduct more, extensive experiments for future work.

A.IV Spatial Cross-Validation

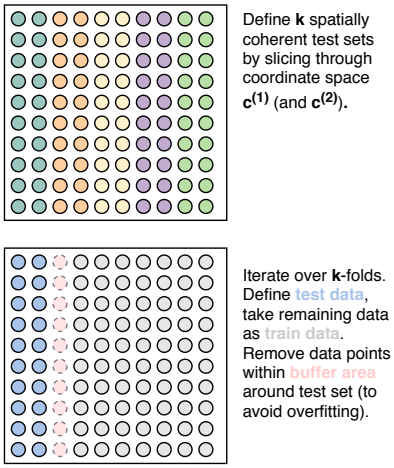


Figure A.1: Illustration of the spatial k-fold cross validation process.

In this section, we provide a graphical illustration of the spatial k-folds cross validation process. This is pictured in Figure A.1.

Appendix B

Appendix for Chapter 4

B.I Data Description

Dataset	n	Min	q_1	$\tilde{x}$	$\bar{x}$	q_3	Max	s	IQR	#NA
PetrelGrid	199680	0	3	7	11.2	15	153	12.2	12	0
DEM (Gen.Mod.)	1183743	76.2	246.9	327.5	331.0	406.6	886.4	117.2	159.7	1
TreeCanopy	10240000	0	0	31	33.8	67	100	34.1	67	0
DEM (Sp.Int)	6856704	0	126	165	157.1	186	254	47.6	60	0

Table B.1: Descriptive statistics of the four real world datasets used for the generative modeling and spatial interpolation experiments.

Due to a lack of geospatial benchmark datasets within the machine learning community, we run our experiments using one toy dataset and three datasets from real-world geospatial applications. All data is chosen to represent different spatial patterns and to be closely related to important applications in fields such as climate science or geology.

Toy: The Toy dataset consists of 32×32 matrices with values coming from a function creating a Gaussian peak at a random location, which is mirrored diagonally by a Gaussian dip. This function is given as:

$$\begin{aligned} f(\mathbf{c_X}, \mathbf{c_Y}, s) = 0.75 \exp(-((9\mathbf{c_X} - a)^2 + (9\mathbf{c_Y} - b)^2)/s) \\ -(0.75 \exp(-((9\mathbf{c_X} - d)^2 + (9\mathbf{c_Y} - e)^2)/s)) \end{aligned} \tag{B.1}$$

where $\mathbf{c_X}$ and $\mathbf{c_Y}$ are the spatial coordinates mapping the values to the 32×32 matrix (so in our cases, integers in the range $[0, 31]$, s is a positive constant determining the size of the Gaussian peak and dip (we use $s = 7$), a and b are random draws from integers in the range $[0, 10]$, determining the location of the Gaussian peak and $d = 10 - a$ and $e = 10 - b$ are the location of the Gaussian dip, mirroring the peak diagonally.

PetrelGrid: The PetrelGrid dataset [111] is composed of geo-referenced seabed relief data. It can be accessed via *R* here: `https://rdrr.io/cran/spm/man/petrel.grid.html`

DEM (Generative Modeling): We use two different digital elevation model (DEM) based datasets, one for the generative modeling experiments and one for the spatial interpolation experiments. The DEM for generative modeling is chosen as it is rather small, enabling us to assess how our proposed method deals with data scarcity. An applicably small DEM dataset, providing a DEM of the area surrounding Lake Sunapee, NH, USA can be found as part of the `elevatr` *R* package; accessible via: `https://rdrr.io/cran/elevatr/man/lake.html`.

TreeCanopy: This dataset contains data on global forest coverage. We use tree canopy, which describes canopy closure for all vegetation taller than 5m in height. The data comes from the University of Maryland's "Global Forest Change" project [67], documenting the global loss of forests in the light of climate change and forest exploitation. Specifically, we use data within the geographic area 50-60N / 100-110W; an area lying in continental Canada and representing a broad range of forest coverage types. The data can be accessed via: `http://earthenginepartners.appspot.com/science-2013-global-forest/download_v1.6.html`.

DEM (Spatial Interpolation): The second, larger DEM dataset used for the spatial interpolation experiments is part of a LiDAR data collection conducted by the National Ecological Observatory Network (NEOS). Specifically, we use DEM hillshades from the NEOS training exercise outlined here: `https://www.neonscience.org/da-viz-neon-lidar-co13flood-R`. Hillshades are used to visualize terrain as shaded reliefs, where shades depend on a (synthetic) light source (e.g. the sun shining at a modelled angle).

All our data is processed into regular grids of either size 32×32 or 64×64. For the exact processing of each of the datasets, please refer to our code.

B.II Experimental Setting, Model Architectures and Compute

Generative Spatial Modeling

Setup: Our main experimental findings, the MMD scores displayed in Table 4.1 and Table 4.2 (main body), are obtained from training generative models on 60% of the data, holding out 20% of the data for validation and model selection, and 20% for computing the displayed test scores. This setting is used for all four experimental datasets. For each dataset, model architecture and auxiliary task setting, we train 10 GANs with different random initializations, in each cycle saving the best generator according to tests on validation data. We then choose the best out of the 10 trained generators (again according to the validation score) to compute test scores.

Model Architecture and Optimization: Here we briefly describe the model architectures of the different generative models used in the experiments working with 32×32 inputs (the models for the 64×64 input are adapted to fit the larger input). For the implementation of these models, please refer to our code.

The Vanilla **GAN** architecture used consists of a Generator with four hidden linear layers, supported by Leaky ReLU and $1d$ BatchNorm layers. The Discriminator has two hidden linear layers supported by Leaky ReLU layers and one linear task-specific layer.

The **DCGAN** architecture used consists of a Generator with a linear initialization layer, followed by three hidden (de-)convolutional layers, supported by ReLU and $2d$ BatchNorm layers. The Discriminator contains two convolutinal layers supported by Leaky ReLU and $2d$ BatchNorm layers, followed by one task-specific convolutional layer with a final linear transformation. For more information on DCGAN, please refer to the original publication [144].

The **EDGAN** architecture used consists of an Encoder-Decoder Generator, where the Encoder contains three convolutional layers, supported by Leaky ReLU and $2d$ BatchNorm layers and the Decoder contains three (de-)convolutional layers supported by ReLU layers. The Discriminator has five hidden convolutional layers suppoerted by Leaky ReLU and $2d$ BatchNorm layers, followed by a last, task-specific convolutional layer. For more information on the EDGAN architecture, please refer to [209], the study which motivated the use of this benchmark.

Model Training: All models are trained using the binary cross entropy criterion to compute losses. Optimization through backpropagation is conducted using the Adam algorithm with a learning rate of 0.001 and β values of $[0.5, 0.999]$. Experiments with the *Toy* dataset run for 40 epochs, with the *PetrelGrid* dataset for 500 epochs, with the *DEM* dataset for 100 epochs and with the *TreeCanopy* dataset for 100 epochs. All training is conducted on GPUs provided via *Google Colab*, which includes *Tesla K80, Tesla T4 and Tesla P100* GPUs. The model training times do not exceed 30 minutes at the longest.

Evaluation: To evaluate our models, we generate synthetic data from the different types of GANs and compare how faithful the generated samples are compared to true samples. To assess model quality, we use the Maximum Mean Discrepancy (MMD) metric [21], a distance measure between distributions based on mean embeddings of the features. For distributions P and Q, the MMD is defined as $MMD(P, Q) = ||\mu_P - \mu_Q||_{\mathbb{R}^d}$. The empirical MMD for random variables x_i and y_i of length n is given as

$$\widehat{MMD}^2 = \frac{1}{n(n-1)} \sum_{i \neq j} k(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i \neq j} k(y_i, y_j) - \frac{2}{n^2} \sum_{i,j} k(x_i, y_j), \tag{B.2}$$

where $k : \mathcal{X} \times \mathcal{X}$ represents a positive-definite kernel—in our case a radial basis function (RBF) kernel. The more similar the data distributions P and Q are, the closer the MMD metric gets to 0.

Predictive Spatial Modeling

Setup: Our main experimental findings for the spatial interpolation experiments, the RMSE scores displayed in Table 3, are obtained from training the CNN models on 60% of the data, selecting the best model using 20% and finally computing the scores on held-out 20% held-out test data. This is done ten times and the test scores are then averaged. Note that the non-neural network based benchmark models (bicubic interpolation, IDW, and kriging) do not require training; rather inference is made directly on the testing samples.

Model Architecture and Optimization: We use a simple CNN for the predictive modeling experiments. It consists of three convolutional layers, supported by ReLU and $2d$ BatchNorm layers. When applying the auxiliary tasks to the model, the last two convolutional layers are made task-specific. Please refer to our code for the exact implementation of the models.

Model Training: All models are trained using the mean squared error (MSE) criterion to compute losses. Optimization through backpropagation is conducted using the Adam algorithm with a learning rate of 0.001 and β values of $[0.5, 0.999]$, running for 150 epochs. All training is conducted on GPUs provided via *Google Colab*, which includes *Tesla K80, Tesla T4 and Tesla P100* GPUs. The individual model training times do not exceed 15 minutes at the longest.

Evaluation: The final evaluation scores on held-out test data are computed as the root mean squared error (RMSE) between real values y_i and predicted values $\hat{y}_i$ of length n:

$$RMSE(y_i, \hat{y}_i) = \sqrt{\frac{1}{n} \sum (y_i - \hat{y}_i)^2} \tag{B.3}$$

Appendix C

Appendix for Chapter 6

C.I Details for COT-GAN

The family of cost functions $\mathcal{C}^{\mathcal{K}}(\mu, c)$ is given by

$$\mathcal{C}^{\mathcal{K}}(\mu, c) := \Bigg\{ c(x, y) + \sum_{j=1}^{J} \sum_{t=1}^{T-1} h_t^j(y) \Delta_{t+1} M^j(x) : \\ J \in \mathbb{N}, (h^j, M^j) \in \mathcal{H}(\mu) \Bigg\},$$

where $\Delta_{t+1} M(x) := M_{t+1}(x_{1:t+1}) - M_t(x_{1:t})$ and $\mathcal{H}(\mu)$ is a set of functions depicting causality:

$$\mathcal{H}(\mu) := \{(h, M) : h = (h_t)_{t=1}^{T-1},\ h_t \in \mathcal{C}_b(\mathbb{R}^{n \times t}), \\ M = (M_t)_{t=1}^{T} \in \mathcal{M}(\mu), M_t \in \mathcal{C}_b(\mathbb{R}^{n \times t})\},$$

with $\mathcal{M}(\mu)$ being the set of martingales on $\mathbb{R}^{n \times T}$ w.r.t. the canonical filtration and the measure μ, and $\mathcal{C}_b(\mathbb{R}^{n \times t})$ the space of continuous, bounded functions on $\mathbb{R}^{n \times t}$.

Moreover, in the implementation of COT-GAN, the dimensionality of the sets of $\mathbf{h} := (h^j)_{j=1}^J$ and $\mathbf{M} := (M^j)_{j=1}^J$ is bounded by a fixed $J \in \mathbb{N}$. The discriminator in COT-GAN is formulated by parameterizing $\mathbf{h}_{\varphi_1}$ and $\mathbf{M}_{\varphi_2}$ in the cost function $c^{\mathcal{K}}$ as two separate neural networks that respect causality,

$$c_{\varphi}^{\mathcal{K}}(x, y) = c(x, y) + \sum_{j=1}^{J} \sum_{t=1}^{T-1} h_{\varphi_1, t}^j(y) \Delta_{t+1} M_{\varphi_2}^j(x), \tag{C.1}$$

where $\varphi := (\varphi_1, \varphi_2)$ and J corresponds to the output dimensionality of the two networks. Thus, we update the parameters based upon the loss between the empirical distributions of two mini-batches.

Given a mini-batch of size m from training data $\{x_{1:T}^d\}_{i=1}^m$ we define the

Table C.1: Generator architecture.

Generator	Configuration
Input	$z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
0	LSTM(state size = 64), BN
1	LSTM(state size = 128), BN
2	Dense(8*8*256), BN, LeakyReLU
3	reshape to 4D array of shape (m, 8, 8, 256)
4	DCONV(N256, K5, S1, P=SAME), BN, LeakyReLU
5	DCONV(N128, K5, S2, P=SAME), BN, LeakyReLU
6	DCONV(N64, K5, S2, P=SAME), BN, LeakyReLU
7	DCONV(N1, K5, S2, P=SAME)

Table C.2: Discriminator architecture.

Discriminator	Configuration
Input	
0	CONV(N64, K5, S2, P=SAME), BN, LeakyReLU
1	CONV(N128, K5, S2, P=SAME), BN, LeakyReLU
2	CONV(N256, K5, S2, P=SAME), BN, LeakyReLU
3	reshape to 3D array of shape (m, T, -1)
4	LSTM(state size = 256), BN
5	LSTM(state size = 64)

empirical measure for the mini-batch as

$$\hat{\mu} := \frac{1}{m} \sum_{d=1}^{m} \delta_{x^d_{1:T}}.$$

As the last piece of the puzzle, Xu et al. [194] enforced $\mathbf{M}$ to be close to a martingale by a regularization term to penalize deviations from being a martingale on the level of mini-batches.

$$p_{\mathbf{M}}(\widehat{\mu}) := \frac{1}{mT} \sum_{j=1}^{J} \sum_{t=1}^{T-1} \left| \sum_{d=1}^{m} \frac{M^j_{t+1}(x^d_{1:t+1}) - M^j_t(x^d_{1:t})}{\sqrt{\mathrm{Var}[M^j] + \eta}} \right|,$$

where $\mathrm{Var}[M]$ is the empirical variance of M over time and batch, and $\eta > 0$ is a small constant.

C.II Training details

We used a smaller size of model with the same network architectures as COT-GAN to train all three datasets. The architectures for generator and discriminator are given in Tables C.1 and C.2.

Hyperparameter settings are as follows: the Sinkhorn regularizer $\epsilon = 0.8$, Sinkhorn iteration $L = 100$, the lengthscale $l = 20$ and martingale penalty $\lambda = 1.5$. We used Adam optimizer with learning rate 0.0001, $\beta_1 = 0.5$ and $\beta_2 = 0.9$. All models are trained for $60,000$ iterations.

C.III Extended figures

In this section, we provide more additional and larger figures for our experiments.

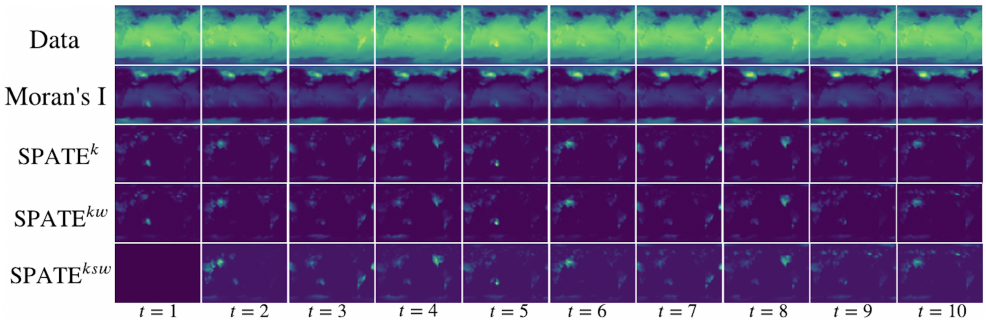


Figure C.1: Larger version of Figure 6.2 for the purpose of visual comparison.

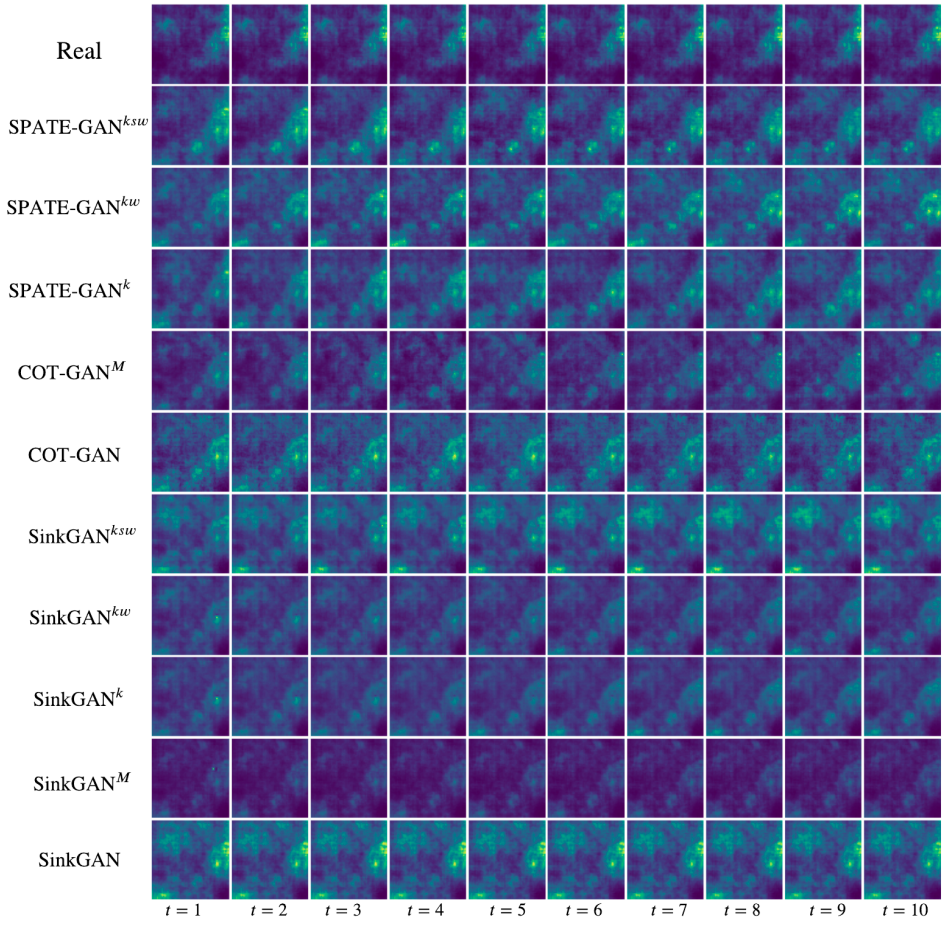


Figure C.2: More selected samples for the log-Gaussian Cox process (LGCP) dataset.

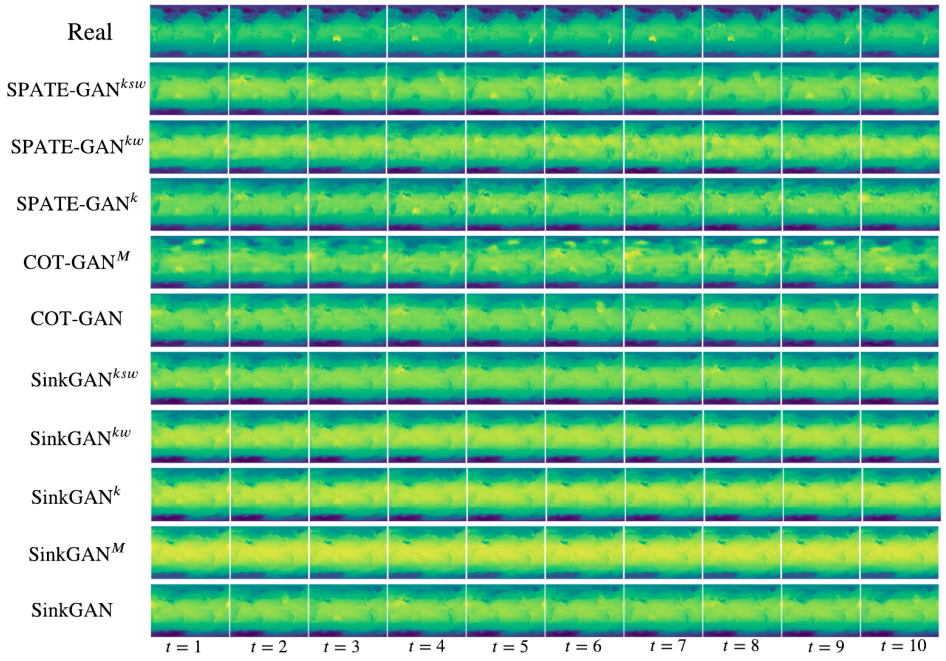


Figure C.3: More selected samples for the extreme weather (EW) dataset.

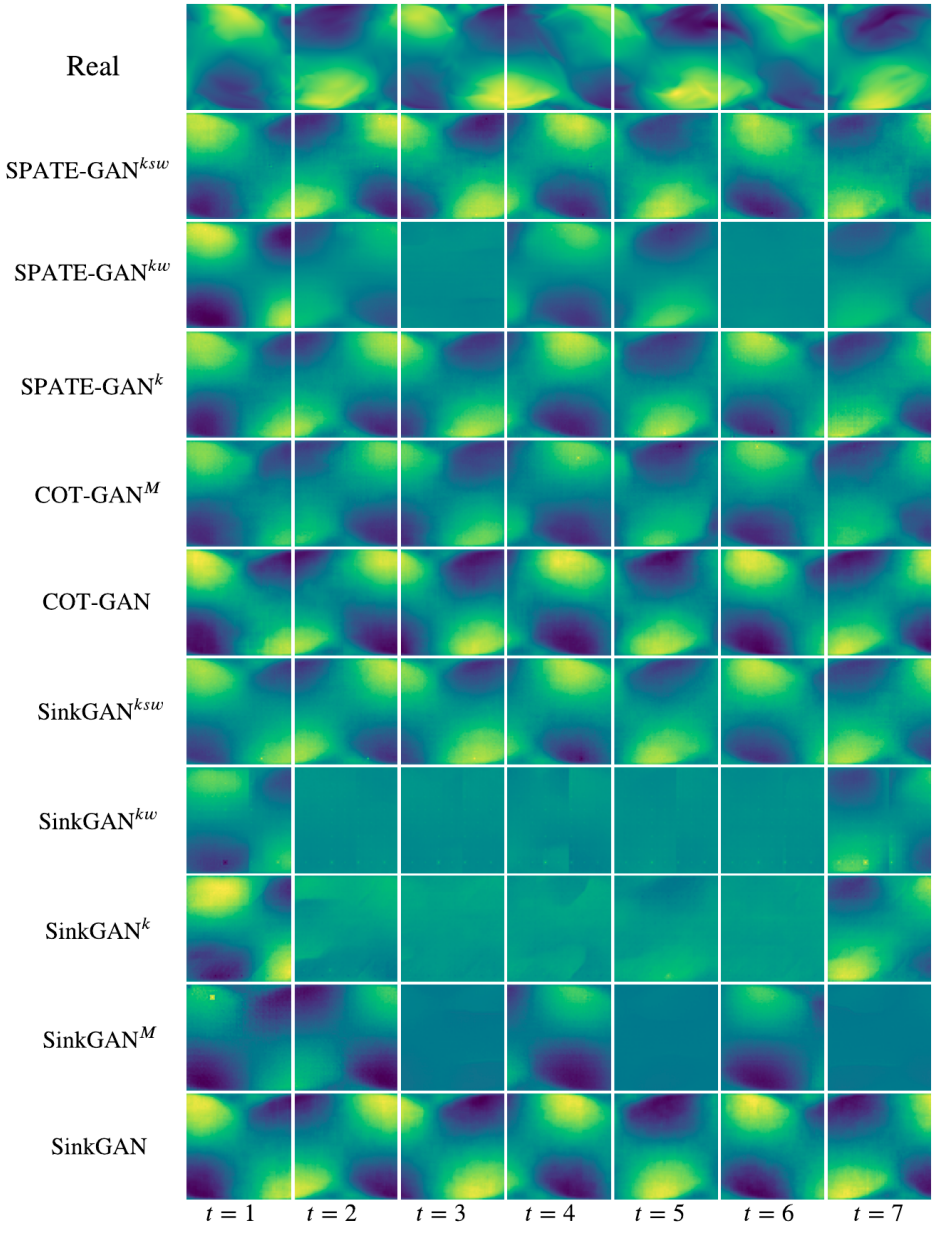


Figure C.4: More selected samples for the turbulent flow (TF) dataset.

Bibliography

[1] Virginia Aglietti, Edwin V Bonilla, Theodoros Damoulas, and Sally Cripps. Structured variational inference in continuous cox process models. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL `https://arxiv.org/abs/1906.03161`.

[2] Luc Anselin. Local Indicators of Spatial Association—LISA. *Geographical Analysis*, 27(2):93–115, sep 1995. ISSN 15384632. doi: 10.1111/j.1538-4632.1995.tb00338.x. URL `http://doi.wiley.com/10.1111/j.1538-4632.1995.tb00338.x`.

[3] Luc Anselin. A Local Indicator of Multivariate Spatial Association: Extending Geary's c. *Geographical Analysis*, 51(2):133–150, apr 2019. ISSN 15384632. doi: 10.1111/gean.12164. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/gean.12164`.

[4] Luc Anselin et al. Spatial econometrics. *A companion to theoretical econometrics*, 310330, 2001.

[5] Oisin Mac Aodha, Elijah Cole, and Pietro Perona. Presence-only geographical priors for fine-grained image classification. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. ISBN 9781728148038. doi: 10.1109/ICCV.2019.00969.

[6] Gabriel Appleby, Linfeng Liu, and Li Ping Liu. Kriging convolutional networks. In *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, volume 34, pages 3187–3194. AAAI press, apr 2020. ISBN 9781577358350. doi: 10.1609/aaai.v34i04.5716. URL `www.aaai.org`.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *34th International Conference on Machine Learning, ICML 2017*, 2017. ISBN 9781510855144.

[8] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[9] Yuki M. Asano, Mandela Patrick, Christian Rupprecht, and Andrea Vedaldi. Labelling unlabelled videos from scratch with multi-modal self-supervision. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[10] Carolyn Augusta, Rob Deardon, and Graham Taylor. Deep learning for supervised classification of spatial epidemics. *Spatial and Spatio-temporal Epidemiology*, 29:187–198, jun 2019. ISSN 18775853. doi: 10.1016/j.sste.2018.08.002.

[11] Christian Bailer, Kiran Varanasi, and Didier Stricker. CNN-based patch matching for optical flow with thresholded hinge embedding loss. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 2710–2719, 2017. ISBN 9781538604571. doi: 10.1109/CVPR.2017.290.

[12] Han Bao, Xun Zhou, Yingxue Zhang, Yanhua Li, and Yiqun Xie. COVID-GAN: Estimating Human Mobility Responses to COVID-19 Pandemic through Spatio-Temporal Conditional Generative Adversarial Networks. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, pages 273–282, New York, NY, USA, nov 2020. Association for Computing Machinery. ISBN 9781450380195. doi: 10.1145/3397536.3422261. URL https://dl.acm.org/doi/10.1145/3397536.3422261.

[13] Alexander Barbie, Niklas Pech, Wilhelm Hasselbring, Sascha Flogel, Frank Wenzhofer, Michael Walter, Elena Shchekinova, Marc Busse, Matthias Turk, Michael Hofbauer, and Stefan Sommer. Developing an Underwater Network of Ocean Observation Systems with Digital Twin Prototypes - A Field Report from the Baltic Sea. *IEEE Internet Computing*, 2021. ISSN 19410131. doi: 10.1109/MIC.2021.3065245.

[14] Sarah Barns. Smart cities and urban data platforms: Designing interfaces for smart governance. *City, Culture and Society*, 12:5–12, mar 2018. ISSN 18779174. doi: 10.1016/j.ccs.2017.09.006.

[15] Michael Batty. Big data, smart cities and city planning. *Dialogues in Human Geography*, 3(3):274–279, nov 2013. ISSN 20438214. doi: 10.1177/2043820613513390. URL http://www.ncbi.nlm.nih.gov/pubmed/29472982http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5808818.

[16] Michael Batty. Digital twins, sep 2018. ISSN 23998091. URL http://journals.sagepub.com/doi/10.1177/2399808318796416.

[17] Michael Batty. Urban analytics defined, mar 2019. ISSN 23998091. URL http://journals.sagepub.com/doi/10.1177/2399808319839494.

[18] Tanya Y. Berger-Wolf, Daniel I. Rubenstein, Charles V. Stewart, Jason A. Holmberg, Jason Parham, Sreejith Menon, Jonathan Crall, Jon Van Oast, Emre Kiciman, and Lucas Joppa. Wildbook: Crowdsourcing, computer vision, and data science for conservation. oct 2017. URL http://arxiv.org/abs/1710.08880.

[19] Christopher M Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[20] Roger S. Bivand, Edzer Pebesma, and Virgilio Gómez-Rubio. *Applied Spatial Data Analysis with R: Second Edition*. Springer New York, jan 2013. ISBN 9781461476184. doi: 10.1007/978-1-4614-7618-4.

[21] Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans Peter Kriegel, Bernhard Schölkopf, and Alex J. Smola. Integrating structured biological data by Kernel Maximum Mean Discrepancy. In *Bioinformatics*, 2006. doi: 10.1093/bioinformatics/btl242.

[22] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. GAN Augmentation: Augmenting Training Data using Generative Adversarial Networks. *arXiv Preprint*, 2018. URL https://arxiv.org/abs/1810.10863http://arxiv.org/abs/1810.10863.

[23] Alexander Brenning. Spatial cross-validation and bootstrap for the assessment of prediction rules in remote sensing: The R package sperrorest. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 5372–5375. IEEE, jul 2012. ISBN 978-1-4673-1159-5. doi: 10.1109/IGARSS.2012.6352393. URL http://ieeexplore.ieee.org/document/6352393/.

[24] Anders Brix and Peter J. Diggle. Spatiotemporal prediction for log-Gaussian Cox processes. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 63(4):823–841, nov 2001. ISSN 13697412. doi: 10.1111/1467-9868.00315. URL https://onlinelibrary.wiley.com/doi/10.1111/1467-9868.00315.

[25] Chris Brunsdon, A. Stewart Fotheringham, and Martin E. Charlton. Geographically weighted regression: a method for exploring spatial non-stationarity. *Geographical Analysis*, 28(4):281–298, sep 1996. ISSN 00167363. doi: 10.1111/j.1538-4632.1996.tb00936.x. URL `https://onlinelibrary.wiley.com/doi/10.1111/j.1538-4632.1996.tb00936.x`.

[26] Caitlin Dempsey. Where is the Phrase "80% of Data is Geographic" From? - GIS Lounge, 2012. URL `https://www.gislounge.com/80-percent-data-is-geographic/`.

[27] Bodhiswatta Chatterjee and Charalambos Poullis. Semantic Segmentation from Remote Sensor Data and the Exploitation of Latent Learning for Classification of Auxiliary Tasks. dec 2019. URL `http://arxiv.org/abs/1912.09216`.

[28] Cen Chen, Kenli Li, Sin G. Teo, Xiaofeng Zou, Kang Wang, Jie Wang, and Zeng Zeng. Gated residual recurrent graph neural networks for traffic prediction. In *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, volume 33, pages 485–492. AAAI Press, jul 2019. ISBN 9781577358091. doi: 10.1609/aaai.v33i01.3301485. URL `www.aaai.org`.

[29] Yue Hong Chou. Spatial pattern and spatial autocorrelation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 988, pages 365–376. Springer Verlag, 1995. ISBN 3540603921. doi: 10.1007/3-540-60392-1_24.

[30] Hafedh Chourabi, Taewoo Nam, Shawn Walker, J. Ramon Gil-Garcia, Sehl Mellouli, Karine Nahon, Theresa A. Pardo, and Hans Jochen Scholl. Understanding smart cities: An integrative framework. In *Proceedings of the Annual Hawaii International Conference on System Sciences*, pages 2289–2297. IEEE Computer Society, 2012. ISBN 9780769545257. doi: 10.1109/HICSS.2012.615.

[31] Grace Chu, Brian Potetz, Weijun Wang, Andrew Howard, Yang Song, Fernando Brucher, Thomas Leung, Hartwig Adam, and Google Research. Geo-Aware Networks for Fine-Grained Recognition. In *International Conference on Computer Vision (ICCV)*, pages 0–0, 2019. URL `https://github.com/visipedia/fg`.

[32] Roberto Cipolla, Yarin Gal, and Alex Kendall. Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics.

In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00781.

[33] Noel Cressie. Spatial prediction and ordinary kriging. *Mathematical Geology*, 1988. ISSN 08828121. doi: 10.1007/BF00892986.

[34] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. Privacy-Preserving Synthetic Location Data in the Real World. In *ACM International Conference Proceeding Series*, pages 23–33, New York, NY, USA, aug 2021. Association for Computing Machinery. ISBN 9781450384254. doi: 10.1145/3469830.3470893. URL https://dl.acm.org/doi/10.1145/3469830.3470893.

[35] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013.

[36] Monidipa Das and Soumya K. Ghosh. Measuring Moran's I in a cost-efficient manner to describe a land-cover change pattern in large-scale remote sensing imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(6):2631–2639, jun 2017. ISSN 21511535. doi: 10.1109/JSTARS.2017.2660766.

[37] Abhirup Datta, Sudipto Banerjee, Andrew O. Finley, and Alan E. Gelfand. Hierarchical Nearest-Neighbor Gaussian Process Models for Large Geostatistical Datasets. *Journal of the American Statistical Association*, 111(514):800–812, apr 2016. ISSN 1537274X. doi: 10.1080/01621459.2015.1044091. URL https://www.tandfonline.com/doi/full/10.1080/01621459.2015.1044091.

[38] Molly Margaret Davies and Mark J. Van Der Laan. Optimal Spatial Prediction Using Ensemble Machine Learning. *International Journal of Biostatistics*, 12(1):179–201, may 2016. ISSN 15574679. doi: 10.1515/ijb-2014-0060. URL http://www.degruyter.com/view/j/ijb.2016.12.issue-1/ijb-2014-0060/ijb-2014-0060.xml.

[39] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. URL http://papers.nips.cc/paper/6081-convolutional-neural-networks-on-graphs-with-fast-localized-spectral-filteringhttp://arxiv.org/abs/1606.09375.

[40] Peter J. Diggle. *Statistical analysis of spatial and spatio-temporal point patterns, third edition*. CRC Press, jan 2013. ISBN 9781466560246. doi:

10.1201/b15326. URL https://www.taylorfrancis.com/books/mono/10.1201/b15326/statistical-analysis-spatial-spatio-temporal-point-patterns-peter-diggle.

[41] Peter J. Diggle, Paula Moraga, Barry Rowlingson, and Benjamin M. Taylor. Spatial and spatio-temporal log-gaussian cox processes: Extending the geostatistical paradigm. *Statistical Science*, 28(4): 542–563, nov 2013. ISSN 08834237. doi: 10.1214/13-STS441. URL https://projecteuclid.org/journals/statistical-science/volume-28/issue-4/Spatial-and-Spatio-Temporal-Log-Gaussian-Cox-Processes--Extending/10.1214/13-STS441.full.

[42] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021.

[43] Bradley Efron and Trevor Hastie. *Computer age statistical inference*, volume 5. Cambridge University Press, 2016.

[44] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A Point Set Generation Network for 3D Object Reconstruction From a Single Image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 605–613, 2017. URL http://openaccess.thecvf.com/content{_}cvpr{_}2017/html/Fan{_}A{_}Point{_}Set{_}CVPR{_}2017{_}paper.html.

[45] Yongjiu Feng, Lijuan Chen, and Xinjun Chen. The impact of spatial scale on local Moran's I clustering of annual fishing effort for Dosidicus gigas offshore Peru. *Journal of Oceanology and Limnology*, 37(1):330–343, jan 2019. ISSN 25233521. doi: 10.1007/s00343-019-7316-9.

[46] Matthias Fey and Jan Eric Lenssen. Fast Graph Representation Learning with PyTorch Geometric. mar 2019. ISSN 2331-8422. URL http://arxiv.org/abs/1903.02428.

[47] Panagiotis Paraskevas Filntisis, Niki Efthymiou, Gerasimos Potamianos, and Petros Maragos. Emotion Understanding in Videos Through Body, Context, and Visual-Semantic Embedding Loss. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12535 LNCS, pages 747–755. Springer Science and Business Media Deutschland GmbH, aug

2020. ISBN 9783030664145. doi: 10.1007/978-3-030-66415-2_52. URL https://doi.org/10.1007/978-3-030-66415-2{_}52.

[48] Yannis Flet-Berliac and Philippe Preux. MERL: Multi-Head Reinforcement Learning. In *NeurIPS 2019 - Deep Reinforcement Learning Workshop*, sep 2019. URL http://arxiv.org/abs/1909.11939.

[49] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using GAN for improved liver lesion classification. In *Proceedings - International Symposium on Biomedical Imaging*, volume 2018-April, pages 289–293. IEEE, apr 2018. ISBN 9781538636367. doi: 10.1109/ISBI.2018.8363576. URL https://ieeexplore.ieee.org/document/8363576/.

[50] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:, 2001.

[51] Yanjie Fu, Pengyang Wang, Jiadi Du, Le Wu, and Xiaolin Li. Efficient region embedding with multi-view spatial networks: A perspective of locality-constrained spatial autocorrelations. In *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, volume 33, pages 906–913. AAAI Press, jul 2019. ISBN 9781577358091. doi: 10.1609/aaai.v33i01.3301906. URL www.aaai.org.

[52] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, apr 1980. ISSN 03401200. doi: 10.1007/BF00344251. URL https://pubmed.ncbi.nlm.nih.gov/7370364/.

[53] Yong Gao, Jing Cheng, Haohan Meng, and Yu Liu. Measuring spatio-temporal autocorrelation in time series data of collective human mobility. *Geo-Spatial Information Science*, 22(3):166–173, jul 2019. ISSN 10095020. doi: 10.1080/10095020.2019.1643609. URL https://www.tandfonline.com/doi/full/10.1080/10095020.2019.1643609.

[54] Pilar García-Soidán, Raquel Menezes, and Óscar Rubiños. Bootstrap approaches for spatial data. *Stochastic Environmental Research and Risk Assessment*, 28(5):1207–1219, oct 2014. ISSN 14363259. doi: 10.1007/s00477-013-0808-9.

[55] Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. URL http://papers.nips.cc/paper/7985-gpytorch-blackbox-matrix-matrix-gaussian-process-inference-with-gpu-accelerationhttp://arxiv.org/abs/1809.11165.

[56] C. E. Gehlke and Katherine Biehl. Certain Effects of Grouping upon the Size of the Correlation Coefficient in Census Tract Material. *Journal of the American Statistical Association*, 29(185A):169–170, mar 1934. ISSN 0162-1459. doi: 10.1080/01621459.1934.10506247. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1934.10506247.

[57] Aude Genevay, Gabriel Peyre, and Marco Cuturi. Learning generative models with sinkhorn divergences. In *AISTATS*, 2018.

[58] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, volume 33, pages 3656–3663. AAAI Press, jul 2019. ISBN 9781577358091. doi: 10.1609/aaai.v33i01.33013656. URL www.aaai.org.

[59] Sebastian Gerke, Karsten Müller, and Ralf Schäfer. Soccer Jersey Number Recognition Using Convolutional Neural Networks. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2016-Febru, pages 734–741. IEEE, dec 2016. ISBN 9781467383905. doi: 10.1109/ICCVW.2015.100. URL http://ieeexplore.ieee.org/document/7406449/.

[60] Arthur Getis and J. K. Ord. The Analysis of Spatial Association by Use of Distance Statistics. *Geographical Analysis*, 24(3):189–206, sep 1992. ISSN 15384632. doi: 10.1111/j.1538-4632.1992.tb00261.x. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1538-4632.1992.tb00261.x.

[61] Mohsen Ghafoorian, Cedric Nugteren, Nóra Baka, Olaf Booij, and Michael Hofmann. EL-GAN: Embedding loss driven generative adversarial networks for lane detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11129 LNCS, pages 256–272, 2019. ISBN 9783030110086. doi: 10.1007/978-3-030-11009-3_15.

[62] Mingming Gong, Yanwu Xu, Chunyuan Li, Kun Zhang, and Kayhan Batmanghelich. Twin Auxiliary Classifiers GAN. jul 2019. ISSN 1049-5258. URL http://arxiv.org/abs/1907.02690.

[63] Michael F. Goodchild. Challenges in Spatial Analysis. In *The SAGE Handbook of Spatial Analysis*, pages 465–479. SAGE Publications, Ltd, oct 2012. doi: 10.4135/9780857020130.n24.

[64] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2672–2680, 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.

[65] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[66] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 2017-Decem, pages 1025–1035. Neural information processing systems foundation, jun 2017. URL http://arxiv.org/abs/1706.02216.

[67] M. C. Hansen, P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R.G. Townshend. High-resolution global maps of 21st-century forest cover change. *Science*, 342(6160):850–853, nov 2013. ISSN 10959203. doi: 10.1126/science.1244693.

[68] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. MD-GAN: Multi-discriminator generative adversarial networks for distributed datasets. In *Proceedings - 2019 IEEE 33rd International Parallel and Distributed Processing Symposium, IPDPS 2019*, 2019. ISBN 9781728112466. doi: 10.1109/IPDPS.2019.00095.

[69] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep Convolutional Networks on Graph-Structured Data. In *Advances in Neural Information Processing Systems (NeurIPS)*, jun 2015. URL http://arxiv.org/abs/1506.05163.

[70] Robert J. Hijmans, Jacob van Etten, Matteo Mattiuzzi, Michael Sumner, Jonathan A Greenberg, Oscar Perpinan Lamigueiro, Andrew Bevan,

Etienne B Racine, and Ashton Shortridge. Geographic Data Analysis and Modeling: Package "raster". *R CRAN Project*, 2.3-40:1–134, jan 2015. URL https://cran.r-project.org/package=rasterhttp://cran.r-project.org/web/packages/raster/.

[71] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, nov 1997. ISSN 08997667. doi: 10.1162/neco.1997.9.8.1735.

[72] Josh Hooker, Gregory Duveiller, and Alessandro Cescatti. Data descriptor: A global dataset of air temperature derived from satellite remote sensing and weather stations. *Scientific Data*, 5(1):1–11, nov 2018. ISSN 20524463. doi: 10.1038/sdata.2018.246. URL https://www.nature.com/articles/sdata2018246.

[73] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning to Steer by Mimicking Features from Heterogeneous Auxiliary Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01): 8433–8440, jul 2019. ISSN 2159-5399. doi: 10.1609/aaai.v33i01.33018433.

[74] Yipeng Hu, Eli Gibson, Li Lin Lee, Weidi Xie, Dean C. Barratt, Tom Vercauteren, and J. Alison Noble. Freehand ultrasound image simulation with spatially-conditioned generative adversarial networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10555 LNCS, pages 105–115. Springer, Cham, 2017. ISBN 9783319675633. doi: 10.1007/978-3-319-67564-0_11. URL http://link.springer.com/10.1007/978-3-319-67564-0{_}11.

[75] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot Ensembles: Train 1, get M for free. In *International Conference on Learning Representations (ICLR)*, mar 2017. URL http://arxiv.org/abs/1704.00109.

[76] Lei Huang, Jihui Zhuang, Xiaoming Cheng, Riming Xu, and Hongjie Ma. STI-GAN: Multimodal Pedestrian Trajectory Prediction Using Spatiotemporal Interactions and a Generative Adversarial Network. *IEEE Access*, 9: 50846–50856, 2021. ISSN 21693536. doi: 10.1109/ACCESS.2021.3069134.

[77] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations (ICLR)*, nov 2017. URL https://youtu.be/Uz-zGYrYEjA.

[78] Krzysztof Janowicz, Song Gao, Grant McKenzie, Yingjie Hu, and Budhendra Bhaduri. GeoAI: spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. *International Journal of Geographical Information Science*, 34(4):625–636, apr 2020. ISSN 13623087. doi: 10.1080/13658816.2019.1684500.

[79] Neal Jean, Sherrie Wang, Anshul Samar, George Azzari, David Lobell, and Stefano Ermon. Tile2Vec: Unsupervised representation learning for spatially distributed data. In *AAAI Conference on Artificial Intelligence*, may 2019. URL http://arxiv.org/abs/1805.02855.

[80] Junteng Jia and Austion R. Benson. Residual Correlation in Graph Neural Network Regression. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 588–598, New York, NY, USA, aug 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403101. URL https://dl.acm.org/doi/10.1145/3394486.3403101.

[81] Zhe Jiang, Shashi Shekhar, Pradeep Mohan, Joseph Knight, and Jennifer Corcoran. Learning spatial decision tree for geographical classification: A summary of results. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, pages 390–393, New York, New York, USA, 2012. ACM Press. ISBN 9781450316910. doi: 10.1145/2424321.2424372. URL http://dl.acm.org/citation.cfm?doid=2424321.2424372.

[82] Zhe Jiang, Yan Li, Shashi Shekhar, Lian Rampi, and Joseph Knight. Spatial Ensemble Learning for Heterogeneous Geographic Data with Class Ambiguity. In *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–10, New York, New York, USA, 2017. ACM Press. ISBN 9781450354905. doi: 10.1145/3139958.3140044. URL http://dl.acm.org/citation.cfm?doid=3139958.3140044.

[83] Peter A. Johnson, Renee Sieber, Teresa Scassa, Monica Stephens, and Pamela Robinson. The Cost(s) of Geospatial Open Data. *Transactions in GIS*, 21(3):434–445, jun 2017. ISSN 14679671. doi: 10.1111/tgis.12283. URL https://onlinelibrary.wiley.com/doi/10.1111/tgis.12283.

[84] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawit, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser,

Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. 2021.

[85] Manohar Kaul, Bin Yang, and Christian S. Jensen. Building accurate 3D spatial networks to enable next generation intelligent transportation systems. In *Proceedings - IEEE International Conference on Mobile Data Management*, 2013. doi: 10.1109/MDM.2013.24.

[86] R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, may 2003. ISSN 01677152. doi: 10.1016/s0167-7152(96)00140-x. URL `https://www.sciencedirect.com/science/article/pii/S016771529600140X`.

[87] Jungeun Kim, Kookjin Lee, Dongeun Lee, Sheo Yon Jin, and Noseong Park. DPM: A Novel Training Method for Physics-Informed Neural Networks in Extrapolation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(9):8146–8154, may 2020. ISSN 2374-3468. URL `www.aaai.orghttp://arxiv.org/abs/2012.02681`.

[88] Soo Ye Kim, Jihyong Oh, and Munchurl Kim. JSI-GAN: GAN-based joint super-resolution and inverse tone-mapping with pixel-wise task-specific filters for UHD HDR video. In *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, volume 34, pages 11287–11295. AAAI press, apr 2020. ISBN 9781577358350. doi: 10.1609/aaai.v34i07.6789. URL `www.aaai.org`.

[89] Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[90] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, sep 2017. URL `http://arxiv.org/abs/1609.02907`.

[91] Rob Kitchin. The real-time city? Big data and smart urbanism. *GeoJournal*, 79(1):1–14, feb 2014. ISSN 0343-2521. doi: 10.1007/s10708-013-

9516-8. URL http://link.springer.com/10.1007/s10708-013-9516-8.

[92] K. Klemmer, S. Wagner, C. Willing, and T. Brandt. Explaining spatio-temporal dynamics in carsharing: A case study of Amsterdam. In *AMCIS 2016: Surfing the IT Innovation Wave - 22nd Americas Conference on Information Systems*, 2016.

[93] Konstantin Klemmer and Daniel B. Neill. Auxiliary-task learning for geographic data with autoregressive embeddings. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, 2021.

[94] Konstantin Klemmer, Tobias Brandt, and Stephen Jarvis. Isolating the effect of cycling on local business environments in London. *PLOS ONE*, 13(12):e0209090, dec 2018. ISSN 1932-6203. doi: 10.1371/journal.pone.0209090. URL http://dx.plos.org/10.1371/journal.pone.0209090.

[95] Konstantin Klemmer, Adriano Koshiyama, and Sebastian Flennerhag. Augmenting correlation structures in spatial data using deep generative models. *arXiv:1905.09796*, 2019. URL http://arxiv.org/abs/1905.09796.

[96] Konstantin Klemmer, Godwin Yeboah, João Porto de Albuquerque, and Stephen A Jarvis. Population Mapping in Informal Settlements with High-Resolution Satellite Imagery and Equitable Ground-Truth. In *ML-IRL Workshop, ICLR'20*, sep 2020. URL http://arxiv.org/abs/2009.08410.

[97] Konstantin Klemmer, Daniel B. Neill, and Stephen A. Jarvis. Understanding spatial patterns in rape reporting delays. *Royal Society Open Science*, 8(2), feb 2021. ISSN 20545703. doi: 10.1098/rsos.201795. URL https://doi.org/10.1098/rsos.201795.

[98] Konstantin Klemmer, Nathan Safir, and Daniel B Neill. Positional Encoder Graph Neural Networks for Geographic Data. *arXiv:2111.10144*, 2021. URL https://arxiv.org/abs/2111.10144.

[99] Konstantin Klemmer, Sudipan Saha, Matthias Kahl, Tianlin Xu, and Xiao Xiang Zhu. Generative modeling of spatio-temporal weather patterns with extreme event conditioning. In *AI: Modeling Oceans and Climate Change (AIMOCC 2021) Workshop, ICLR 2021*, apr 2021. URL http://arxiv.org/abs/2104.12469.

[100] Konstantin Klemmer, Sudipan Saha, Matthias Kahl, Tianlin Xu, and Xiao Xiang Zhu. Generative modeling of spatio-temporal weather patterns

with extreme event conditioning. *ICLR'21 Workshop AI: Modeling Oceans and Climate Change (AIMOCC)*, apr 2021. URL http://arxiv.org/abs/2104.12469.

[101] Konstantin Klemmer, Tianlin Xu, Beatrice Acciaio, and Daniel B. Neill. SPATE-GAN: Improved Generative Modeling of Dynamic Spatio-Temporal Patterns with an Autoregressive Embedding Loss. In *AAAI 2022 - 36th AAAI Conference on Artificial Intelligence*, 2022.

[102] A. B. Knudsen and R. Slooff. Vector-borne disease problems in rapid urbanization: New approaches to vector control. *Bulletin of the World Health Organization*, 70(1):1–6, 1992. ISSN 00439686. URL /pmc/articles/PMC2393336/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC2393336/.

[103] Adriano Koshiyama, Nick Firoozye, and Philip Treleaven. Generative Adversarial Networks for Financial Trading Strategies Fine-Tuning and Combination. *arXiv Preprint*, jan 2019. URL http://arxiv.org/abs/1901.01751.

[104] Mark A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, feb 1991. ISSN 15475905. doi: 10.1002/aic.690370209. URL https://onlinelibrary.wiley.com/doi/10.1002/aic.690370209.

[105] Martin Kulldorff. A spatial scan statistic. *Communications in Statistics - Theory and Methods*, 26(6):1481–1496, 1997. ISSN 03610926. doi: 10.1080/03610929708831995. URL https://www.tandfonline.com/action/journalInformation?journalCode=lsta20.

[106] Martin Kulldorff. Tests of spatial randomness adjusted for an inhomogeneity: A general framework, sep 2006. ISSN 01621459. URL https://www.tandfonline.com/action/journalInformation?journalCode=uasa20.

[107] Martin Kulldorff, Richard Heffernan, Jessica Hartman, Renato Assunção, and Farzad Mostashari. A space-time permutation scan statistic for disease outbreak detection. *PLoS Medicine*, 2(3):0216–0224, feb 2005. ISSN 15491277. doi: 10.1371/journal.pmed.0020059. URL https://dx.plos.org/10.1371/journal.pmed.0020059.

[108] Jay Lee and Shengwen Li. Extending Moran's Index for Measuring Spatiotemporal Clustering of Geographic Events. *Geographical Analysis*, 49(1):36–57, jan 2017. ISSN 00167363. doi: 10.1111/gean.12106. URL https://onlinelibrary.wiley.com/doi/10.1111/gean.12106.

[109] Yaguo Lei, Feng Jia, Jing Lin, Saibo Xing, and Steven X. Ding. An Intelligent Fault Diagnosis Method Using Unsupervised Feature Learning Towards Mechanical Big Data. *IEEE Transactions on Industrial Electronics*, 63(5):3137–3147, may 2016. ISSN 02780046. doi: 10.1109/TIE.2016.2519325.

[110] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point Cloud GAN. *arXiv Preprint*, oct 2018. URL http://arxiv.org/abs/1810.05795.

[111] J. Li. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. In *Proceedings - 20th International Congress on Modelling and Simulation, MODSIM 2013*, 2013. ISBN 9780987214331. doi: 10.36334/modsim.2013.a9.li.

[112] Mike Li, Elija Perrier, and Chang Xu. Deep Hierarchical Graph Convolution for Election Prediction from Geospatial Census Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):647–654, jul 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.3301647. URL https://aaai.org/ojs/index.php/AAAI/article/view/3841.

[113] Yu Li, Dongbo Min, Minh N. Do, and Jiangbo Lu. Fast guided global interpolation for depth and motion. In *European Conference on Computer Vision (ECCV)*, volume 9907 LNCS, pages 717–733. Springer, Cham, 2016. ISBN 9783319464862. doi: 10.1007/978-3-319-46487-9_44. URL http://link.springer.com/10.1007/978-3-319-46487-9{_}44.

[114] Chieh Hubert Lin, Chia-Che Chang, Yu-Sheng Chen, Da-Cheng Juan, Wei Wei, and Hwann-Tzong Chen. COCO-GAN: Generation by Parts via Conditional Coordinating. mar 2019. URL http://arxiv.org/abs/1904.00284.

[115] Daoyu Lin, Kun Fu, Yang Wang, Guangluan Xu, and Xian Sun. MARTA GANs: Unsupervised Representation Learning for Remote Sensing Image Classification. *IEEE Geoscience and Remote Sensing Letters*, 14(11): 2092–2096, nov 2017. ISSN 1545598X. doi: 10.1109/LGRS.2017.2752750. URL http://ieeexplore.ieee.org/document/8059820/.

[116] Drew Linsley, Junkyung Kim, Vijay Veerabadran, and Thomas Serre. Learning long-range spatial dependencies with horizontal gated-recurrent units. In *Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates Inc., 2018. URL https://dl.acm.org/citation.cfm?id=3326958http://arxiv.org/abs/1805.08315.

[117] Paul A Longley, Mike Goodchild, David J Maguire, and David W Rhind. *Geographic Information Systems and Science*. Wiley Publishing, 3rd edition, 2010. ISBN 0470721448.

[118] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2019.

[119] Man Luo, Bowen Du, Konstantin Klemmer, Hongming Zhu, Hakan Ferhatosmanoglu, and Hongkai Wen. D3P: Data-driven Demand Prediction for Fast Expanding Electric Vehicle Sharing Systems. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(1):1–21, mar 2020. ISSN 24749567. doi: 10.1145/3381005. URL https://dl.acm.org/doi/10.1145/3381005.

[120] Man Luo, Bowen Du, Konstantin Klemmer, Hongming Zhu, and Hongkai Wen. Deployment Optimization for Shared e-Mobility Systems With Multi-Agent Deep Neural Search. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–12, 2021. ISSN 1524-9050. doi: 10.1109/tits.2021.3125745. URL https://ieeexplore.ieee.org/document/9616397/.

[121] Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells. In *International Conference on Learning Representations (ICLR)*, feb 2020. URL http://arxiv.org/abs/2003.00824.

[122] Roger J. Marshall. Mapping disease and mortality rates using empirical Bayes estimators. *Journal of the Royal Statistical Society. Series C, Applied statistics*, 40(2):283–294, jun 1991. ISSN 00359254. doi: 10.2307/2347593. URL https://www.jstor.org/stable/10.2307/2347593?origin=crossref.

[123] G. Matheron. Principles of geostatistics. *Economic Geology*, 58(8): 1246–1266, dec 1963. ISSN 03610128. doi: 10.2113/gsecongeo.58.8.1246.

[124] Jennifer L. Matthews, Norou Diawara, and Lance A. Waller. Quantifying Spatio-Temporal Characteristics via Moran's Statistics. In *STEAM-H: Science, Technology, Engineering, Agriculture, Mathematics and Health*, pages 163–177. Springer Nature, 2019. doi: 10.1007/978-3-030-11431-2_9. URL https://doi.org/10.1007/978-3-030-11431-2{_}9.

[125] Yan Meng, Chao Lin, Weihong Cui, and Jian Yao. Scale selection based on Moran's i for segmentation of high resolution remotely sensed images. In *International Geoscience and Remote Sensing Symposium (IGARSS)*,

pages 4895–4898. Institute of Electrical and Electronics Engineers Inc., nov 2014. ISBN 9781479957750. doi: 10.1109/IGARSS.2014.6947592.

[126] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013.

[127] Andrew Miller, Luke Bornn, Ryan Adams, and Kirk Goldsberry. Factorized Point Process Intensities: A Spatial Analysis of Professional Basketball. In *International Conference on Machine Learning (ICML)*, 2014. URL `http://www.jmlr.org/proceedings/papers/v32/miller14.pdfhttp://arxiv.org/abs/1401.0942`.

[128] Mehdi Mirza and Simon Osindero. Conditional generative adversarial networks. *Manuscript: https://arxiv. org/abs/1709.02023*, 2014.

[129] P. A. Moran. Notes on continuous stochastic phenomena. *Biometrika*, 37(1-2):17–23, jun 1950. ISSN 00063444. doi: 10.1093/biomet/37.1-2.17. URL `https://www.jstor.org/stable/2332142?origin=crossref`.

[130] Taylor Mordan, Gilles Henaff, Nicolas Thome, and Matthieu Cord. Revisiting multi-task learning with rock: A deep residual auxiliary block for visual detection. In *Advances in Neural Information Processing Systems*, 2018.

[131] Fernando Munoz-Mendez, Ke Han, Konstantin Klemmer, and Stephen Jarvis. Community Structures, Interactions and Dynamics in London's Bicycle Sharing Network. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers - UbiComp '18*, pages 1015–1023, New York, New York, USA, 2018. ACM Press. ISBN 9781450359665. doi: 10.1145/3267305.3274156. URL `http://dl.acm.org/citation.cfm?doid=3267305.3274156`.

[132] Silvia Nittel, J. C. Whittier, and Qinghan Liang. Real-time spatial interpolation of continuous phenomena using mobile sensor data streams. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, pages 530–533, New York, New York, USA, 2012. ACM Press. ISBN 9781450316910. doi: 10.1145/2424321.2424407. URL `http://dl.acm.org/citation.cfm?doid=2424321.2424407`.

[133] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *34th International*

Conference on Machine Learning, ICML 2017, volume 6, pages 4043–4055, 2017. ISBN 9781510855144. URL http://proceedings.mlr.press/v70/odena17a.html.

[134] S. Openshaw. The modifiable areal unit problem. *CATMOG (Concepts & Techniques in Modern Geography)*, 38:60–69, 1983. URL https://ci.nii.ac.jp/naid/10003011548.

[135] Guy H Orcutt, Harold W Watts, and John B Edwards. Data Aggregation and Information Loss. *The American Economic Review*, 58(4):773–787, 1968. ISSN 00028282. URL http://www.jstor.org/stable/1815532.

[136] J. K. Ord and Arthur Getis. Local Spatial Autocorrelation Statistics: Distributional Issues and an Application. *Geographical Analysis*, 27(4):286–306, sep 1995. ISSN 15384632. doi: 10.1111/j.1538-4632.1995.tb00912.x. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1538-4632.1995.tb00912.x.

[137] J. Keith Ord and Arthur Getis. Local spatial heteroscedasticity (LOSH). *Annals of Regional Science*, 48(2):529–539, apr 2012. ISSN 05701864. doi: 10.1007/s00168-011-0492-y. URL https://link.springer.com/article/10.1007/s00168-011-0492-y.

[138] Jung Yeon Park, Kenneth Theo Carr, Stephan Zheng, Yisong Yue, and Rose Yu. Multiresolution Tensor Learning for Efficient and Interpretable Spatial Analysis. feb 2020. URL http://arxiv.org/abs/2002.05578.

[139] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[140] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(Oct): 2825–2830, 2012. ISSN ISSN 1533-7928. URL http://www.jmlr.org/papers/v12/pedregosa11a.htmlhttp://arxiv.org/abs/1201.0490.

[141] Jonne Pohjankukka, Tapio Pahikkala, Paavo Nevalainen, and Jukka Heikkonen. Estimating the prediction performance of spatial models via spatial k-fold cross validation. *International Journal of Geographical Information Science*, 31(10):2001–2019, oct 2017. ISSN 13623087. doi: 10.1080/13658816.2017.1346255. URL `https://www.tandfonline.com/doi/full/10.1080/13658816.2017.1346255`.

[142] Harrison Quick, Scott H. Holan, Christopher K. Wikle, and Jerome P. Reiter. Bayesian marked point process modeling for generating fully synthetic public use data with point-referenced geography. *Spatial Statistics*, 14:439–451, nov 2015. ISSN 22116753. doi: 10.1016/j.spasta.2015.07.008. URL `https://www.sciencedirect.com/science/article/pii/S2211675315000718`.

[143] Evan Racah, Christopher Beckham, Tegan Maharaj, Samira Ebrahimi Kahou, Prabhat, and Christopher Pal. ExtremeWeather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. In *Advances in Neural Information Processing Systems*, volume 2017-Decem, pages 3403–3414, 2017. URL `https://github.com/eracah/hur-detect`.

[144] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[145] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, feb 2019. ISSN 10902716. doi: 10.1016/j.jcp.2018.10.045.

[146] Chengping Rao, Hao Sun, and Yang Liu. Physics-informed deep learning for incompressible laminar flows. *Theoretical and Applied Mechanics Letters*, 10(3):207–212, mar 2020. ISSN 20950349. doi: 10.1016/j.taml.2020.01.039.

[147] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, dec 2018. doi: 10.7551/mitpress/3206.001.0001.

[148] Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and Prabhat. Deep learning and process understanding for data-driven Earth system science. *Nature*, 566

(7743):195–204, feb 2019. ISSN 14764687. doi: 10.1038/s41586-019-0912-1. URL http://www.nature.com/articles/s41586-019-0912-1.

[149] B. D. Ripley. The second-order analysis of stationary point processes. *Journal of Applied Probability*, 13(2):255–266, jun 1976. ISSN 0021-9002. doi: 10.2307/3212829. URL https://doi.org/10.2307/3212829.

[150] Amal Ben Rjab and Sehl Mellouli. Smart Cities in the Era of Artificial Intelligence and Internet of Things: Literature Review from 1990 to 2017. In *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, page 10, New York, NY, USA, 2018. ACM. ISBN 9781450365260. URL https://doi.org/10.1145/3209281.3209380.

[151] David R. Roberts, Volker Bahn, Simone Ciuti, Mark S. Boyce, Jane Elith, Gurutzeta Guillera-Arroita, Severin Hauenstein, José J. Lahoz-Monfort, Boris Schröder, Wilfried Thuiller, David I. Warton, Brendan A. Wintle, Florian Hartig, and Carsten F. Dormann. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure, aug 2017. ISSN 16000587. URL https://onlinelibrary.wiley.com/doi/10.1111/ecog.02881.

[152] W. S. Robinson. Ecological correlations and the behavior of individuals. *International Journal of Epidemiology*, 38(2):337–341, apr 2009. ISSN 03005771. doi: 10.1093/ije/dyn357. URL https://academic.oup.com/ije/article-lookup/doi/10.1093/ije/dyn357.

[153] David Rolnick, Priya L. Donti, Lynn H. Kaack, Kelly Kochanski, Alexandre Lacoste, Kris Sankaran, Andrew Slavin Ross, Nikola Milojevic-Dupont, Natasha Jaques, Anna Waldman-Brown, Alexandra Luccioni, Tegan Maharaj, Evan D. Sherwin, S. Karthik Mukkavilli, Konrad P. Kording, Carla Gomes, Andrew Y. Ng, Demis Hassabis, John C. Platt, Felix Creutzig, Jennifer Chayes, and Yoshua Bengio. Tackling Climate Change with Machine Learning. jun 2019. ISSN 2331-8422. URL http://arxiv.org/abs/1906.05433.

[154] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[155] Johannes L. Schonberger, Marc Pollefeys, Andreas Geiger, and Torsten Sattler. Semantic Visual Localization. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00721.

[156] Mike Shafto, Mike Conroy, Rich Doyle, Ed Glaessgen, Chris Kemp, Jacqueline LeMoigne, and Lui Wang. Modeling, simulation, information technology & processing roadmap. *National Aeronautics and Space Administration*, 32:1–38, 2012.

[157] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference, ACM 1968*, 1968.

[158] Marianna Siino, Francisco J. Rodríguez-Cortés, Jorge Mateu, and Giada Adelfio. Testing for local structure in spatiotemporal point pattern data. In *Environmetrics*, volume 29, page e2463. John Wiley and Sons Ltd, aug 2018. doi: 10.1002/env.2463. URL `https://onlinelibrary.wiley.com/doi/10.1002/env.2463`.

[159] B. W. Silverman. *Density estimation: For statistics and data analysis*. CRC Press, jan 2018. ISBN 9781351456173. doi: 10.1201/9781315140919. URL `https://www.taylorfrancis.com/books/mono/10.1201/9781315140919/density-estimation-statistics-data-analysis-silverman`.

[160] Panagiotis Sismanidis, Benjamin Bechtel, Iphigenia Keramitsoglou, and Chris T. Kiranoudis. Mapping the Spatiotemporal Dynamics of Europe's Land Surface Temperatures. *IEEE Geoscience and Remote Sensing Letters*, 15(2):202–206, feb 2018. ISSN 15580571. doi: 10.1109/LGRS.2017.2779829.

[161] Michael R. Sprague, Amir Jalalirad, Marco Scavuzzo, Catalin Capota, Moritz Neun, Lyman Do, and Michael Kopp. Asynchronous federated learning for geospatial applications. In *Communications in Computer and Information Science*, volume 967, pages 21–28. Springer Verlag, sep 2019. ISBN 9783030148799. doi: 10.1007/978-3-030-14880-5_2. URL `http://www.here.com`.

[162] A. Stein and L. C. A. Corsten. Universal Kriging and Cokriging as a Regression Procedure. *Biometrics*, 1991. ISSN 0006341X. doi: 10.2307/2532147.

[163] A. Stewart Fotheringham and Peter A. Rogerson. GIS and spatial analytical problems. *International Journal of Geographical Information Systems*, 7(1):3–19, 1993. ISSN 02693798. doi: 10.1080/02693799308901936. URL `https://www.tandfonline.com/doi/abs/10.1080/02693799308901936`.

[164] S. C. Suddarth and Y. L. Kergosien. Rule-injection hints as a means of improving network performance and learning time. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 412 LNCS, pages 120–129. Springer Verlag, 1990. ISBN 9783540522553. doi: 10.1007/3-540-52255-7_33.

[165] Yusuke Tanaka, Tomoharu Iwata, Toshiyuki Tanaka, Takeshi Kurashima, Maya Okawa, and Hiroyuki Toda. Refining coarse-grained spatial data using auxiliary spatial data sets with various granularities. In *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, 2019. ISBN 9781577358091. doi: 10.1609/aaai.v33i01.33015091.

[166] Jovan Tanevski, Ljupčo Todorovski, Yannis Kalaidzidis, and Sašo Džeroski. Domain-specific model selection for structural identification of the Rab5-Rab7 dynamics in endocytosis. *BMC Systems Biology*, 9(1):31, jun 2015. ISSN 17520509. doi: 10.1186/s12918-015-0175-x. URL `https://bmcsystbiol.biomedcentral.com/articles/10.1186/s12918-015-0175-x`.

[167] Kevin Tang, Manohar Paluri, Li Fei-Fei, Rob Fergus, and Lubomir Bourdev. Improving image classification with location context. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pages 1008–1016, 2015. ISBN 9781467383912. doi: 10.1109/ICCV.2015.121.

[168] Benjamin M. Taylor, Tilman M. Davies, Barry S. Rowlingson, and Peter J. Diggle. Bayesian inference and data augmentation schemes for spatial, spatiotemporal and multivariate log-gaussian cox processes in R. *Journal of Statistical Software*, 63(7):1–48, jan 2015. ISSN 15487660. doi: 10.18637/jss.v063.i07. URL `https://www.jstatsoft.org/index.php/jss/article/view/v063i07/v63i07.pdfhttps://www.jstatsoft.org/index.php/jss/article/view/v063i07`.

[169] Luke Taylor and Geoff Nitschke. Improving Deep Learning with Generic Data Augmentation. In *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, pages 1542–1547. IEEE, nov 2019. ISBN 9781538692769. doi: 10.1109/SSCI.2018.8628742. URL `https://ieeexplore.ieee.org/document/8628742/`.

[170] Mahyat Shafapour Tehrany, Simon Jones, Farzin Shabani, Francisco Martínez-Álvarez, and Dieu Tien Bui. A novel ensemble modeling

approach for the spatial prediction of tropical forest fire susceptibility using LogitBoost machine learning classifier and multi-source geospatial data. *Theoretical and Applied Climatology*, pages 1–17, sep 2018. ISSN 14344483. doi: 10.1007/s00704-018-2628-9. URL http://link.springer.com/10.1007/s00704-018-2628-9.

[171] W. R. Tobler. A Computer Movie Simulating Urban Growth in the Detroit Region. *Economic Geography*, 46:234, jun 1970. ISSN 00130095. doi: 10.2307/143141. URL https://www.jstor.org/stable/143141?origin=crossref.

[172] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *IEEE CVPR*, June 2019.

[173] Yiting Tsai, Susan A. Baldwin, and Bhushan Gopaluni. Identifying indicator species in ecological habitats using Deep Optimal Feature Learning. *PLoS ONE*, 16(9 September):e0256782, sep 2021. ISSN 19326203. doi: 10.1371/journal.pone.0256782. URL https://dx.plos.org/10.1371/journal.pone.0256782.

[174] Sergey Tulyakov, Ming Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing Motion and Content for Video Generation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1526–1535, 2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00165. URL https://github.com/sergeytulyakov/mocogan.

[175] United Nations. The World 's Cities in 2018. Technical report, dec 2018. URL https://www.un-ilibrary.org/content/books/9789210476102https://www.un.org/en/events/citiesday/assets/pdf/the{_}worlds{_}cities{_}in{_}2018{_}data{_}booklet.pdf.

[176] Juliette T.H. Unwin, Isobel Routledge, Seth Flaxman, Marian Andrei Rizoiu, Shengjie Lai, Justin Cohen, Daniel J. Weiss, Swapnil Mishra, and Samir Bhatt. Using hawkes processes to model imported and local malaria cases in near-elimination settings. *PLoS Computational Biology*, 17(4):e1008830, apr 2021. ISSN 15537358. doi: 10.1371/JOURNAL.PCBI.1008830. URL https://dx.plos.org/10.1371/journal.pcbi.1008830.

[177] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Pro-*

cessing Systems, volume 2017-Decem, pages 5999–6009, 2017. URL https://research.google/pubs/pub46201/.

[178] Petar Veličković, Arantxa Casanova, Pietro Liò, Guillem Cucurull, Adriana Romero, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, oct 2018. URL https://arxiv.org/abs/1710.10903v3.

[179] James M Wallace. Space-time correlations in turbulent flow: A review. *Theoretical and Applied Mechanics Letters*, 4(2):022003, 2014. ISSN 20950349. doi: 10.1063/2.1402203.

[180] Bingsheng Wang and Jinjun Xiong. Novel geospatial interpolation analytics for general meteorological measurements. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014. ISBN 9781450329569. doi: 10.1145/2623330.2623367.

[181] Chaoyue Wang, Chang Xu, Xin Yao, and Dacheng Tao. Evolutionary generative adversarial networks. *arXiv preprint arXiv:1803.00657*, 2018.

[182] Fengjiao Wang, Chun Ta Lu, Yongzhi Qu, and Philip S. Yu. Collective geographical embedding for geolocating social network users. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10234 LNAI, pages 599–611. Springer Verlag, 2017. ISBN 9783319574530. doi: 10.1007/978-3-319-57454-7_47.

[183] Jin Feng Wang, A. Stein, Bin Bo Gao, and Yong Ge. A review of spatial sampling, dec 2012. ISSN 22116753.

[184] Ke Alexander Wang, Geoff Pleiss, Jacob R Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact Gaussian Processes on a Million Data Points. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14648–14659, 2019. URL http://arxiv.org/abs/1903.08114.

[185] Lijing Wang, Jiangzhuo Chen, and Madhav Marathe. TDEFSI: Theory-guided Deep Learning-based Epidemic Forecasting with Synthetic Information. *ACM Transactions on Spatial Algorithms and Systems*, 6 (3):1–39, may 2020. ISSN 23740361. doi: 10.1145/3380971. URL https://dl.acm.org/doi/10.1145/3380971.

[186] Lijun Wang, Jianming Zhang, Yifan Wang, Huchuan Lu, and Xiang Ruan. CLIFFNet for Monocular Depth Estimation with Hierarchical

Embedding Loss. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12350 LNCS, pages 316–331. Springer Science and Business Media Deutschland GmbH, aug 2020. ISBN 9783030585570. doi: 10.1007/978-3-030-58558-7_19. URL https://doi.org/10.1007/978-3-030-58558-7{_}19.

[187] Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards Physics-informed Deep Learning for Turbulent Flow Prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020. ISBN 9781450379984. doi: 10.1145/3394486.3403198.

[188] Xiaolong Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9908 LNCS, pages 318–335. Springer Verlag, mar 2016. ISBN 9783319464923. doi: 10.1007/978-3-319-46493-0_20. URL http://arxiv.org/abs/1603.05631.

[189] Rene Westerholt, Bernd Resch, Franz Benjamin Mocnik, and Dirk Hoffmeister. A statistical test on the local effects of spatially structured variance. *International Journal of Geographical Information Science*, 32(3):571–600, mar 2018. ISSN 13623087. doi: 10.1080/13658816.2017.1402914. URL https://www.tandfonline.com/doi/abs/10.1080/13658816.2017.1402914.

[190] Christoph Willing, Konstantin Klemmer, Tobias Brandt, and Dirk Neumann. Moving in time and space – Location intelligence for car-sharing decision support. *Decision Support Systems*, 99:75–85, jul 2017. ISSN 01679236. doi: 10.1016/j.dss.2017.05.005. URL http://linkinghub.elsevier.com/retrieve/pii/S0167923617300830.

[191] Shuai Xiao, Mehrdad Farajtabar, Xiaojing Ye, Junchi Yan, Le Song, and Hongyuan Zha. Wasserstein Learning of Deep Generative Point Process Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3247–3257, 2017. URL http://papers.nips.cc/paper/6917-wasserstein-learning-of-deep-generative-point-process-models.

[192] Jianpeng Xu, Pang Ning Tan, Lifeng Luo, and Jiayu Zhou. GSpartan: A geospatio-temporal multi-task learning framework for multi-location prediction. In *16th SIAM International Conference on Data Mining 2016, SDM 2016*, 2016. ISBN 9781510828117. doi: 10.1137/1.9781611974348.74.

[193] Jianpeng Xu, Xi Liu, Tyler Wilson, Pang Ning Tan, Pouyan Hatami, and Lifeng Luo. Muscat: Multi-scale spatio-temporal learning with application to climate modeling. In *IJCAI International Joint Conference on Artificial Intelligence*, 2018. ISBN 9780999241127. doi: 10.24963/ijcai.2018/404.

[194] Tianlin Xu, Li K. Wenliang, Michael Munn, and Beatrice Acciaio. COT-GAN: Generating sequential data via causal optimal transport. In *Advances in Neural Information Processing Systems*, volume 2020-Decem. Neural information processing systems foundation, jun 2020. URL `http://arxiv.org/abs/2006.08571`.

[195] Bo Yan, Gengchen Mai, Krzysztof Janowicz, and Song Gao. From ITDL to Place2Vec – Reasoning About Place Type Similarity and Relatedness by Learning Embeddings From Augmented Spatial Contexts. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, 2017. ISBN 9781450354905. doi: 10.1145/3139958.3140054.

[196] Chaowei Yang, Manzhu Yu, Fei Hu, Yongyao Jiang, and Yun Li. Utilizing Cloud Computing to address big geospatial data challenges. *Computers, Environment and Urban Systems*, 61:120–128, jan 2017. ISSN 01989715. doi: 10.1016/j.compenvurbsys.2016.10.010.

[197] Xiuwen Yi, Junbo Zhang, Zhaoyuan Wang, Tianrui Li, and Yu Zheng. Deep distributed fusion network for air quality prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018. ISBN 9781450355520. doi: 10.1145/3219819.3219822.

[198] Yifang Yin, Zhenguang Liu, Ying Zhang, Sheng Wang, Rajiv Ratn Shah, and Roger Zimmermann. GPS2Vec: Towards generating worldwide GPS embeddings. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, pages 416–419, New York, NY, USA, nov 2019. Association for Computing Machinery. ISBN 9781450369091. doi: 10.1145/3347146.3359067. URL `https://dl.acm.org/doi/10.1145/3347146.3359067`.

[199] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *ICLR*, 2019. URL `https://openreview.net/forum?id=S1zk9iRqF7`.

[200] Binbin Yu. Ecological effects of new-type urbanization in China. *Renewable and Sustainable Energy Reviews*, 135:110239, jan 2021. ISSN 18790690. doi: 10.1016/j.rser.2020.110239.

[201] Yumin Yuan, Mark Cave, and Chaosheng Zhang. Using Local Moran's I to identify contamination hotspots of rare earth elements in urban soils of London. *Applied Geochemistry*, 88:167–178, jan 2018. ISSN 18729134. doi: 10.1016/j.apgeochem.2017.07.011.

[202] Andrew Zammit-Mangion, Tin Lok James Ng, Quan Vu, and Maurizio Filippone. Deep Compositional Spatial Models. jun 2019. URL `http://arxiv.org/abs/1906.02840`.

[203] Jincheng Zhang and Xiaowei Zhao. Spatiotemporal wind field prediction based on physics-informed deep learning and LIDAR measurements. *Applied Energy*, 288:116641, apr 2021. ISSN 03062619. doi: 10.1016/j.apenergy.2021.116641.

[204] Na Zhang and Hongyan Zhang. Scale variance analysis coupled with Moran's I scalogram to identify hierarchy and characteristic scale. *International Journal of Geographical Information Science*, 25(9):1525–1543, sep 2011. ISSN 13658816. doi: 10.1080/13658816.2010.532134.

[205] Yingxue Zhang, Yanhua Li, Xun Zhou, Xiangnan Kong, and Jun Luo. Curb-GAN: Conditional Urban Traffic Estimation through Spatio-Temporal Generative Adversarial Networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 20, pages 842–852, New York, NY, USA, aug 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403127. URL `https://dl.acm.org/doi/10.1145/3394486.3403127`.

[206] Yunchao Zhang, Yanjie Fu, Pengyang Wang, Xiaolin Li, and Yu Zheng. Unifying inter-region autocorrelation and intra-region structures for spatial embedding via collective adversarial learning. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019. ISBN 9781450362016. doi: 10.1145/3292500.3330972.

[207] Liang Zhao, Jiangzhuo Chen, Feng Chen, Wei Wang, Chang Tien Lu, and Naren Ramakrishnan. SimNest: Social media nested epidemic simulation via online semi-supervised deep learning. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, volume 2016-Janua, pages 639–648. Institute of Electrical and Electronics Engineers Inc., jan 2016. ISBN 9781467395038. doi: 10.1109/ICDM.2015.39.

[208] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499. IEEE, jun

2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00472. URL https://ieeexplore.ieee.org/document/8578570/.

[209] Di Zhu, Ximeng Cheng, Fan Zhang, Xin Yao, Yong Gao, and Yu Liu. Spatial interpolation using conditional generative adversarial neural networks. *International Journal of Geographical Information Science*, pages 1–24, apr 2019. ISSN 1365-8816. doi: 10.1080/1365881YYxxxxxxxx. URL https://www.tandfonline.com/doi/full/10.1080/13658816.2019.1599122.

[210] Di Zhu, Fan Zhang, Shengyin Wang, Yaoli Wang, Ximeng Cheng, Zhou Huang, and Yu Liu. Understanding Place Characteristics in Geographic Contexts through Graph Convolutional Neural Networks. *Annals of the American Association of Geographers*, 110(2):408–420, mar 2020. ISSN 24694460. doi: 10.1080/24694452.2019.1694403. URL https://www.tandfonline.com/doi/full/10.1080/24694452.2019.1694403.