

Auxiliary-task learning for geographic data with autoregressive embeddings

Konstantin Klemmer

University of Warwick & New York University
UK
k.klemmer@warwick.ac.uk

Daniel B. Neill

New York University
USA
daniel.neill@nyu.edu

ABSTRACT

Machine learning is gaining popularity in a broad range of areas working with geographic data, such as ecology or atmospheric sciences. Here, data often exhibit spatial effects, which can be difficult to learn for neural networks. In this study, we propose SXL, a method for embedding information on the autoregressive nature of spatial data directly into the learning process using auxiliary tasks. We utilize the local Moran's I, a popular measure of local spatial autocorrelation, to "nudge" the model to learn the direction and magnitude of local spatial effects, complementing the learning of the primary task. We further introduce a novel expansion of Moran's I to multiple resolutions, thus capturing spatial interactions over longer and shorter distances simultaneously. The novel multi-resolution Moran's I can be constructed easily and as a multi-dimensional tensor offers seamless integration into existing machine learning frameworks. Throughout a range of experiments using real-world data, we highlight how our method consistently improves the training of neural networks in unsupervised and supervised learning tasks. In generative spatial modeling experiments, we propose a novel loss for auxiliary task GANs utilizing task uncertainty weights. Our proposed method outperforms domain-specific spatial interpolation benchmarks, highlighting its potential for downstream applications. This study bridges expertise from geographic information science and machine learning, showing how this integration of disciplines can help to address domain-specific challenges. The code for our experiments is available via <https://drive.google.com/file/d/1ShOBV7RifMdS9LYsySOM084KL5aD-wkm/view?usp=sharing>.

CCS CONCEPTS

- Computing methodologies → Neural networks; Image representations;
- Applied computing → Earth and atmospheric sciences.

KEYWORDS

Auxiliary Task Learning, Spatial Autocorrelation, GAN, Spatial Interpolation, GIS

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SIGSPATIAL '21, November 2–5, 2021, Beijing, China

© 2021 Association for Computing Machinery.

ACM ISBN XXX-X-XXXX-XXXX-X/XX/XX...\$15.00

<https://doi.org/XX.XXXX/XXXXXX.XXXXXXX>

ACM Reference Format:

Konstantin Klemmer and Daniel B. Neill. 2021. Auxiliary-task learning for geographic data with autoregressive embeddings. In *29th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, November 2–5, 2021, Beijing, China*. ACM, New York, NY, USA, 12 pages. <https://doi.org/XX.XXXX/XXXXXX.XXXXXXX>

1 INTRODUCTION

When monitoring the physical environment, from crop yields to air pollution, the gathered data are often geographic in nature and follow some spatial process: data values depend on their spatial locations. This violates a key assumption of many statistical learning frameworks, that data are independent and identically distributed (*iid*). Even models which are explicitly designed to account for these dependencies, such as kernel methods, may struggle with over- and under-smoothing of spatial patterns or with distinguishing stationary from non-stationary spatial effects. The complexities of geographic data concern researchers in many academic fields and are a key focus of the geographic information sciences (GIS). The GIS community has a long tradition of analyzing spatial phenomena, developing metrics to measure spatial effects and deploying models to account for spatial dependencies [1, 12, 29]. With the growing popularity of scalable machine learning methods, particularly deep neural networks, applications of these approaches to geospatial data domains have become more and more common. Nevertheless, these applications have only rarely inspired methodological innovation [22] in neural networks. Recently, [39] specifically pointed to a lack of deep learning methods tailored to geospatial and spatio-temporal data in the context of earth system science. In light of this call-to-action, we propose SXL, a novel neural network method for geospatial data domains, that is inspired by domain expertise from GIS and explicitly learns spatial dependencies contained within the data. This is facilitated through a multi-task learning process, where a spatial embedding capturing local autoregressive effects 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 embedding in parallel. We integrate one of the most prominent and widely used metrics in GIS into the model: the Moran's I measure of local spatial autocorrelation [2]. 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 seek to improve performance on a broad range of spatial modeling tasks.

Our main contributions can be summarized as follows: (1) We propose a novel, flexible multi-resolution expansion of the Moran's I measure of local spatial autocorrelation. 2) 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. (3) We provide a practical framework to adapt our method to arbitrary neural network architectures and different supervised (predictive) and unsupervised (generative) spatial learning tasks. (4) For the purpose of balancing the losses of multiple tasks in a generative modeling setting, we develop a novel auxiliary task GAN loss based on uncertainty weights [10]. (5) We evaluate SXL on both generative and predictive spatial modeling tasks, providing empirical evidence for consistent and robust performance gains across multiple synthetic and real-world experimental settings.

2 RELATED WORK

2.1 Machine learning with geospatial data

Geography has a long tradition of modeling and empirical analysis. Ideas from GIS and spatial statistics have inspired popular approaches in modern machine learning, from Gaussian Processes (GPs), pioneered by the development of Kriging in the 1960s [31], to spatial scan statistics [26], widely used for event and pattern detection. Nevertheless, since the emergence of the era of deep neural networks, the relationship between the GIS and machine learning communities have been mostly defined through applications of existing neural network models to geographic data. Only rarely have concepts and ideas from GIS and spatial statistics motivated methodological advancements in neural networks—whereas application areas such as vision, bioinformatics, and computational linguistics have strongly influenced the deep learning state-of-the-art we know today. Nevertheless, some recent advances in machine learning have been exceptionally useful to the GIS community. Kernel methods such as GPs have seen huge progress towards overcoming computational bottlenecks stemming 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 [15, 46]. In neural networks, the emergence of graph-based networks and particularly graph convolutions have allowed for the modeling of asymmetric and non-Euclidean spatial relationships [28], and the emergence of physics-informed deep learning has reinforced the need for neural networks to model complex spatio-temporal patterns [47].

Nevertheless, as geographic context has proven to be of relevance in many machine learning applications [3, 5, 9, 51], core concepts from GIS and geography have gradually attracted more attention in the ML community. While methodologies for geospatial data and problems have previously utilized traditional machine learning approaches like tree-based models [24] or ensemble learners [23], focus has recently switched more towards neural networks: for example, [30] and [53] propose a vector embedding to capture spatial context, inspired by word embeddings such as *Word2Vec* [33]. [54] propose to model complex spatial covariance patterns through injective warping functions, learned by a deep net. While there exist approaches for capturing spatial autocorrelation in neural networks [56], the Moran's I metric has previously only been used once in an explicit machine learning setting: [25] propose to use it as a heuristic for early stopping in generative adversarial network (GAN) training, but not 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.

2.2 Learning with auxiliary tasks

In this paper, we use auxiliary learning, an approach using multi-task learning to improve performance on a primary task, originally conceptualized by [43]. The authors propose to give learners “hints” related to the original task throughout training in order 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 in the data, highlighted by the auxiliary objective. It also implies that the auxiliary task has to provide some meaningful embedding of the primary task. Auxiliary learning has been particularly successful in deep reinforcement learning and is widely utilized [14]: for example, learning to steer a wheel can be improved using auxiliary tasks related to image segmentation and optical flow estimation [20]. 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 [21] or depth estimation [34]. Auxiliary tasks have also been successfully deployed for generative modeling with GANs: Auxiliary Classifier GAN (AC-GAN) [36] 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 [16] 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 [48]: 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 localization [40] and semantic segmentation [7], in addition to dedicated spatial and spatio-temporal multi-task frameworks [50]. 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 modeling.

3 METHODOLOGY

3.1 Multi-resolution local Moran's I

As outlined above, 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.

Formally, let X be a 2-d spatial matrix (image)

$$X^{n \times m} = \begin{bmatrix} x_{1,1} & \dots & x_{1,m} \\ \dots & \dots & \dots \\ x_{n,1} & \dots & x_{n,m} \end{bmatrix} \quad (1)$$

and the vector $\mathbf{x} = \text{vec}(X)$ consists of $n_x = nm$ real-valued observations x_i , referenced by an index set $N_x = \{1, 2, \dots, n_x\}$. We

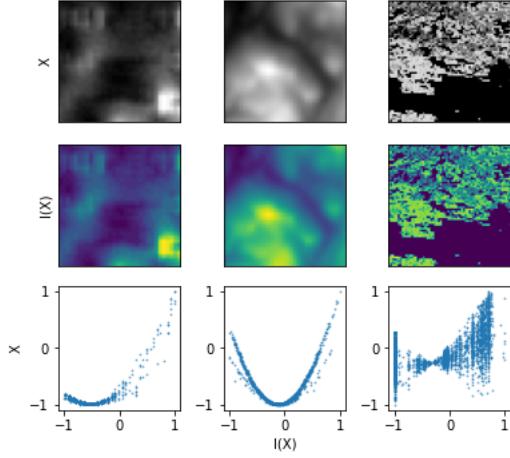


Figure 1: Example of the different shapes the relationship between X and $I(X)$ can take using seabed relief, digital elevation map (DEM) and tree canopy data (from left to right). The figure also highlights how the local Moran's I can work both, as a measure of spatial outliers (first column) and homogenous spatial clusters (third column).

define the *spatial neighbourhood* of observation i to be $\mathcal{N}_{x_i} = \{j \in \mathcal{N}_x : w_{i,j} > 0\}$. Here, $w_{i,j}$ corresponds to a binary spatial weight matrix, indicating whether any observation j is a neighbor 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 study, we utilize queen contiguity (i.e. all adjacent grid cells, including diagonals, are neighbors), but the approach generalizes to arbitrary neighborhood definitions. We can compute the local Moran's I statistic, I_i , of observation x_i with the mean over all observations \bar{x} as:

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

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 neighbors, which could indicate latent cluster structure. A negative value suggests that the data point is distinctly different from neighboring 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. Figure 1 provides examples of different geospatial data X and their respective $I(X)$ representations.

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 immediate neighbors as defined by the neighborhood matrix w . Thus longer-range spatial dependencies can be lost. This scale sensitivity of the local Moran's I is known as a common challenge in applications [13, 32, 55]. 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:

$$X_d^{n/a \times m/a}(i, j) = \text{mean}\{X^{n \times m}(ai + k, aj + l)\}, \quad (3)$$

for $0 \leq k < a$ and $0 \leq l < a$,

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-neighbor 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 2.

3.2 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 generalization. Lastly, auxiliary tasks can work as regularizers 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 characterizes a spatial pattern as a reflection of underlying spatial processes. Accordingly, [8] concludes that “[...] the capability of generalizing 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 (see *Related Work*), 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 [37, 44, 49]. 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.

With our experiments, we focus on two distinct settings: generative spatial modeling using GANs [17], and predictive spatial

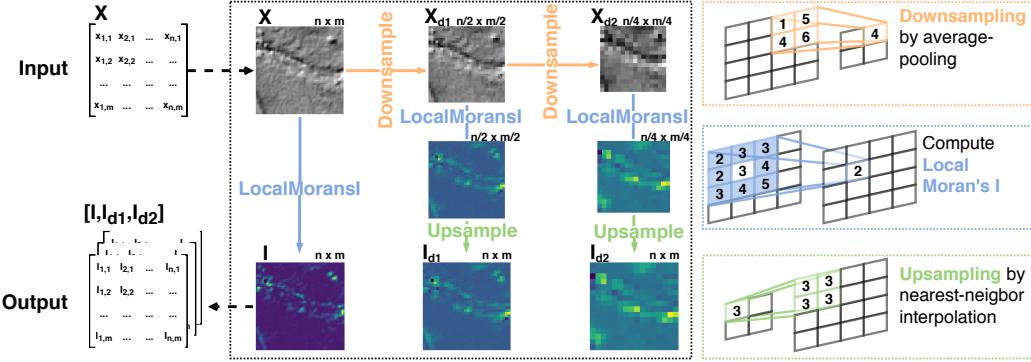


Figure 2: Multi-resolution local Moran's I calculation with an example input at three different resolutions: Original input size ($n \times m$), downsampled by factor 2 ($n/2 \times m/2$) and downsampled by factor 4 ($n/4 \times m/4$).

modeling 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 modeling formulation follows from this straightforward. GANs are a family of generative models comprised of two neural networks, a Generator \mathbf{G} that produces fake data and a Discriminator \mathbf{D} that seeks to distinguish between real and fake data. These two networks are agents in a two-player game, where \mathbf{G} learns to produce synthetic data samples that are faithful to the true data generating process, and \mathbf{D} learns to separate real from fake samples, thus pushing \mathbf{G} to produce increasingly realistic synthetic data. The standard GAN loss function is thus given as:

$$\min_G \max_D \mathcal{L}_{GAN}(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))], \quad (4)$$

consisting of the Discriminator and Generator losses

$$\mathcal{L}_{GAN}^{(D)} = \max_D [\log(D(\mathbf{x})) + \log(1 - D(G(\mathbf{z})))], \quad (5)$$

$$\mathcal{L}_{GAN}^{(G)} = \min_G [\log(D(\mathbf{x})) + \log(1 - D(G(\mathbf{z})))]. \quad (6)$$

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

$$\mathcal{L}_{AT}^{(D)} = \max_D [\log(D(I(\mathbf{x}))) + \log(1 - D(I(G(\mathbf{z})))], \quad (7)$$

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

$$\min_G \max_D \mathcal{L}_{MRES-MAT}(D, G) = \mathcal{L}_{GAN}(D, G) + \lambda (\mathcal{L}_{AT_1}^{(D)} + \dots + \mathcal{L}_{AT_N}^{(D)}). \quad (8)$$

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

3.3 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 conceptualized by [10] and utilize each task's homoskedastic uncertainty to inform the loss weight. To formalize this, [10] define a probabilistic multi-task regression problem with N tasks and likelihood

$$p(y_1, \dots, y_N | f(\mathbf{x})) = p(y_1 | f(\mathbf{x})) \dots p(y_N | f(\mathbf{x})), \quad (9)$$

where y_1, \dots, y_N are the main and auxiliary model outputs and \mathbf{x} is the model input. Using the maximum likelihood method, the minimization objective of the multi-task regression is $\min \mathcal{L}(\sigma_1, \dots, \sigma_N)$:

$$\begin{aligned} &= -\log p(y_1, \dots, y_N | f(\mathbf{x})) \\ &= \frac{1}{2\sigma_1^2} \mathcal{L}_1 + \dots + \frac{1}{2\sigma_N^2} \mathcal{L}_N + \sum_{i=1}^N \log \sigma_i, \end{aligned} \quad (10)$$

where $\sigma_1, \dots, \sigma_N$ are the model noise parameters. Minimizing 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 regularizer. This loss constitutes the objective we use for our predictive modeling 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

$$\begin{aligned} \min_G \max_D \mathcal{L}_{MRES-MAT}(D, G) &= \mathcal{L}_{GAN}^{(G)} + \\ &(\frac{1}{2\sigma_1^2} \mathcal{L}_{AT_1}^{(D)} + \dots + \frac{1}{2\sigma_N^2} \mathcal{L}_{AT_N}^{(D)} + \sum_{i=1}^N \log \sigma_i). \end{aligned} \quad (11)$$

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 utilize these throughout our experiments. We use auxiliary learning settings with hard-parameter sharing, where the top layers of the

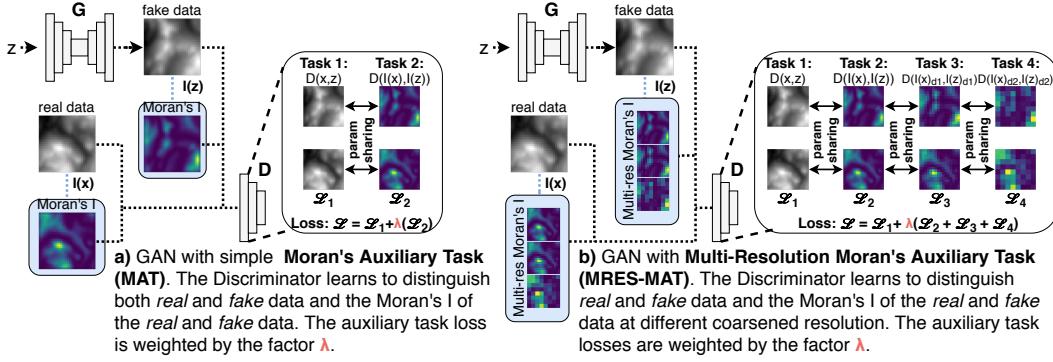


Figure 3: GAN with spatially explicit auxiliary tasks, using the Moran's I (a) and multi-resolution Moran's I (b) embeddings in the Discriminator.

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 parallelly 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 10*.

4 EXPERIMENTS

4.1 Experiment 1: Generative spatial modeling

Selection & Evaluation: In our first experiment, we want to examine whether our proposed method can improve the learning of spatial data generating 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 [6], 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

$$\widehat{MMD}^2 = \frac{1}{n(n-1)} \sum_{i \neq j} k(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i \neq j} k(y_i, y_j) - \frac{2}{n^2} \sum_{i,j} k(x_i, y_j), \quad (12)$$

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 optimizes 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 optimizes 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: We select four 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 [27], 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 package¹, processed into a grid of 1156 32×32 tiles. (4) *Tree canopy* data of the University of Maryland's "Global Tree Change" project [18], 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 modeling challenges in the earth sciences, ecology, or geography. For more information on the data used, including summary statistics, please refer to Appendix A.

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* [17], the *DCGAN* [38] and lastly an Encoder-Decoder GAN (*EDGAN*) architecture recently proposed by [57] and explicitly designed for geospatial applications. All models are optimized 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

¹See: <https://github.com/jhollist/elevatr>

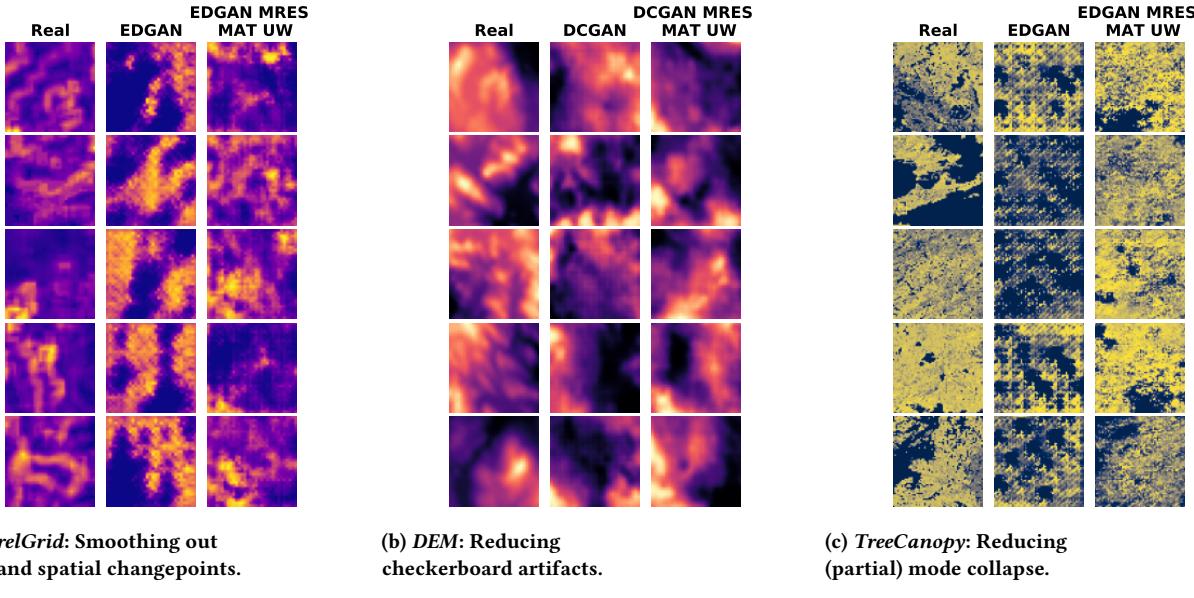


Figure 4: Example images highlighting the positive effects of *MAT* / *MRES MAT* on different GAN architectures, across different datasets.

Model	Toy	PetrelGrid	DEM	TreeCanopy
GAN [17]	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 [38]	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 [57]	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 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.

the experimental setup, including hyperparameters and detailed model architectures, please refer to Appendix B.I.

Findings: Table 1 and 2 show the MMD scores of generators selected according to the strategy outlined in the *Evaluation & Selection* paragraph. Table 1 highlights results from the uncertainty weighting strategy, and Table 2 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-specialize, leading to worse generalization performance [19]—thus adding a second, closely correlated task should not pose a problem but might help prevent over-specialization in the bottom Discriminator layers. (2)

The losses stemming from the auxiliary tasks have a regularizing effect throughout training, further preventing Discriminator over-specialization.

This leads to several beneficial effects, some even visually apparent, as highlighted in Figure 4. Figure 4a 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 4c 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

Model	Toy	PetrelGrid	DEM	TreeCanopy
GAN [17]	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 [4]	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 [57]	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 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 .

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.

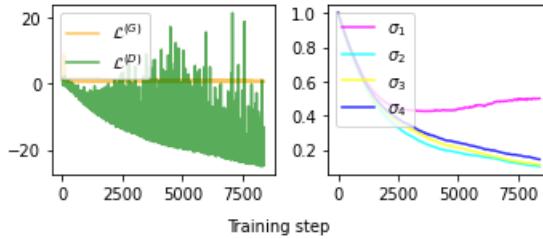


Figure 5: Training progression of generator and discriminator losses ($\mathcal{L}^{(G)}$, $\mathcal{L}^{(D)}$), main task uncertainty σ_1 and auxiliary task uncertainties $\sigma_2, \sigma_3, \sigma_4$ throughout an example training cycle of EDGAN MRES MAT UW using DEM training data.

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 tweaking. Figure 5 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.

4.2 Experiment 2: Predictive spatial modeling

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 [45], air quality assessment [52] or mobile sensing [35]. 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.

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 Appendix A.

²See: <https://www.neonscience.org/da-viz-neon-lidar-co13flood-R>

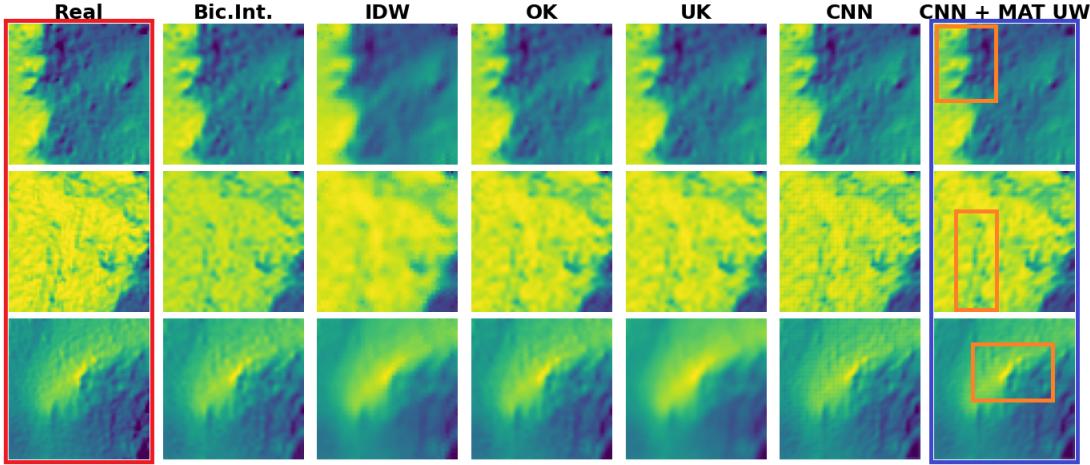


Figure 6: Interpolation results on samples from the test set, across the different benchmark models, presenting our CNN + MAT UW model. The orange boxes highlight areas where the improvement over the benchmark models becomes visually apparent.

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*) [41], a weighted rolling-average approach, (3) Ordinary Kriging (*OK*) [11], spatial interpolation closely related to Gaussian Process regression and (4) Universal Kriging (*UK*) [42], a generalization 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 ℓ_1 loss. For more details on the experimental setup, please refer to Appendix B.II.

Findings: The results of our experiments are presented in Table 3 and Figure 6. We can again see a positive effect of 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 utilized, 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.

Task weighting: Finding the optimal task weighting strategy appears much less tricky as for generative modeling, 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 the Appendix.

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

Table 3: Model RMSE scores and their standard deviation on held-out test data for the 32 → 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).

5 CONCLUSION

In general, 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 regularizing 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 minimize 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 modeling experiments, but in most cases uncertainty weights appear to have the edge.

In summary, 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 modeling and up to 5% for predictive spatial modeling 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. We see this study further evidence of the importance of integrating domain expertise from GIS into neural network methods for geospatial data, opening a broad range of further research directions: In future work, we plan to expand this idea beyond discrete, regularly gridded data to continuous spatial processes and to assess the applicability of other common spatial metrics for a more comprehensive learning of complex geospatial patterns.

ACKNOWLEDGMENTS

This work is supported in part by the UK Engineering and Physical Sciences Research Council under Grant No. EP/L016400/1.

REFERENCES

- [1] Luc Anselin. 1988. *Spatial econometrics : methods and models*. Kluwer Academic Publishers, Dordrecht ;Boston. 284 pages. <https://doi.org/10.1007/978-94-015-7799-1>
- [2] Luc Anselin. 1995. Local Indicators of Spatial Association—LISA. *Geographical Analysis* 27, 2 (sep 1995), 93–115. <https://doi.org/10.1111/j.1538-4632.1995.tb00338.x> arXiv:1011.1669
- [3] Oisin Mac Aodha, Elijah Cole, and Pietro Perona. 2019. Presence-only geographical priors for fine-grained image classification. In *Proceedings of the IEEE International Conference on Computer Vision*. <https://doi.org/10.1109/ICCV.2019.00969> arXiv:1906.05272
- [4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *34th International Conference on Machine Learning, ICML 2017*.
- [5] Han Bao, Xun Zhou, Yingxue Zhang, Yanhua Li, and Yiqun Xie. 2020. 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*. Association for Computing Machinery, New York, NY, USA, 273–282. <https://doi.org/10.1145/3397536.3422261>
- [6] Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans Peter Kriegel, Bernhard Schölkopf, and Alex J. Smola. 2006. Integrating structured biological data by Kernel Maximum Mean Discrepancy. In *Bioinformatics*. <https://doi.org/10.1093/bioinformatics/btl242>
- [7] Bodhiswatta Chatterjee and Charalambos Poullis. 2019. Semantic Segmentation from Remote Sensor Data and the Exploitation of Latent Learning for Classification of Auxiliary Tasks. (dec 2019). arXiv:1912.09216 <http://arxiv.org/abs/1912.09216>
- [8] Yue Hong Chou. 1995. Spatial pattern and spatial autocorrelation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 988. Springer Verlag, 365–376. https://doi.org/10.1007/3-540-60392-1_24
- [9] Grace Chu, Brian Potetz, Weijun Wang, Andrew Howard, Yang Song, Fernando Brucher, Thomas Leung, Hartwig Adam, and Google Research. 2019. Geo-Aware Networks for Fine-Grained Recognition. In *International Conference on Computer Vision (ICCV)*. 0–0. <https://github.com/visipedia/fg>
- [10] Roberto Cipolla, Yarin Gal, and Alex Kendall. 2018. 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*. <https://doi.org/10.1109/CVPR.2018.00781> arXiv:1705.07115
- [11] Noel Cressie. 1988. Spatial prediction and ordinary kriging. *Mathematical Geology* (1988). <https://doi.org/10.1007/BF00892986>
- [12] Noel A. C. Cressie. 1991. *Statistics for spatial data*. John Wiley & Sons.
- [13] Yongjiu Feng, Lijuan Chen, and Xinjun Chen. 2019. 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 (jan 2019), 330–343. <https://doi.org/10.1007/s00343-019-7316-9>
- [14] Yannis Fleit-Berliac and Philippe Preux. 2019. MERL: Multi-Head Reinforcement Learning. (sep 2019). arXiv:1909.11939 <http://arxiv.org/abs/1909.11939>
- [15] Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. 2018. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. In *Advances in Neural Information Processing Systems (NeurIPS)*. <https://papers.nips.cc/paper/7985-gpytorch-blackbox-matrix-matrix-gaussian-process-inference-with-gpu-acceleration> <http://arxiv.org/abs/1809.11165>
- [16] Mingming Gong, Yanwu Xu, Chunyuan Li, Kun Zhang, and Kayhan Batmanghelich. 2019. Twin Auxiliary Classifiers GAN. (jul 2019). arXiv:1907.02690 <http://arxiv.org/abs/1907.02690>
- [17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2672–2680. <http://papers.nips.cc/paper/5423-generative-adversarial-nets>
- [18] 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. Townshend. 2013. High-resolution global maps of 21st-century forest cover change. *Science* 342, 6160 (nov 2013), 850–853. <https://doi.org/10.1126/science.1244693>
- [19] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. 2019. MD-GAN: Multi-discriminator generative adversarial networks for distributed datasets. In *Proceedings - 2019 IEEE 33rd International Parallel and Distributed Processing Symposium, IPDPS 2019*. <https://doi.org/10.1109/IPDPS.2019.00095> arXiv:1811.03850
- [20] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Changz Loy. 2019. Learning to Steer by Mimicking Features from Heterogeneous Auxiliary Networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (jul 2019), 8433–8440. <https://doi.org/10.1609/aaai.v33i01.33018433> arXiv:1811.02759
- [21] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. 2017. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations (ICLR)*. arXiv:1611.05397 <https://youtu.be/Uz-zGYrYEjA>
- [22] Krzysztof Janowicz, Song Gao, Grant McKenzie, Yingjie Hu, and Budhendra Bhaduri. 2020. GeoAI: spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. *International Journal of Geographical Information Science* 34, 4 (apr 2020), 625–636. <https://doi.org/10.1080/13658816.2019.1684500>
- [23] Zhe Jiang, Yan Li, Shashi Shekhar, Lian Rambi, and Joseph Knight. 2017. Spatial Ensemble Learning for Heterogeneous Geographic Data with Class Ambiguity. In *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM Press, New York, New York, USA, 1–10. <https://doi.org/10.1145/3139958.3140044>
- [24] Zhe Jiang, Shashi Shekhar, Pradeep Mohan, Joseph Knight, and Jennifer Corcoran. 2012. 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*. ACM Press, New York, New York, USA, 390–393. <https://doi.org/10.1145/2424321.2424372>
- [25] Konstantin Klemmer, Adriano Koshiyama, and Sebastian Flennherag. 2019. Augmenting correlation structures in spatial data using deep generative models. arXiv:1905.09796 (2019). arXiv:1905.09796 <http://arxiv.org/abs/1905.09796>
- [26] Martin Kulldorff. 1997. A spatial scan statistic. *Communications in Statistics - Theory and Methods* 26, 6 (1997), 1481–1496. <https://doi.org/10.1080/03610929708831995>
- [27] J. Li. 2013. 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*. <https://doi.org/10.36334/modsim.2013.a9.li>
- [28] Mike Li, Elja Perrier, and Chang Xu. 2019. Deep Hierarchical Graph Convolution for Election Prediction from Geospatial Census Data. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (jul 2019), 647–654. <https://doi.org/10.1609/aaai.v33i01.3301647>
- [29] Jeremy W. Lichstein, Theodore R. Simons, Susan A. Shriner, and Kathleen E. Franzreb. 2002. Spatial autocorrelation and autoregressive models in ecology. *Ecological Monographs* 72, 3 (aug 2002), 445–463. [https://doi.org/10.1890/0012-9615\(2002\)072\[0445:SAAAMI\]2.0.CO;2](https://doi.org/10.1890/0012-9615(2002)072[0445:SAAAMI]2.0.CO;2)
- [30] Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. 2020. Multi-Scale Representation Learning for Spatial Feature Distributions

using Grid Cells. In *International Conference on Learning Representations (ICLR)*. arXiv:2003.00824 <http://arxiv.org/abs/2003.00824>

[31] G. Matheron. 1963. Principles of geostatistics. *Economic Geology* 58, 8 (dec 1963), 1246–1266. <https://doi.org/10.2113/gsecongeo.58.8.1246>

[32] Yan Meng, Chao Lin, Weihong Cui, and Jian Yao. 2014. Scale selection based on Moran's I for segmentation of high resolution remotely sensed images. In *International Geoscience and Remote Sensing Symposium (IGARSS)*. Institute of Electrical and Electronics Engineers Inc., 4895–4898. <https://doi.org/10.1109/IGARSS.2014.6947592>

[33] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*. arXiv:1301.3781

[34] Taylor Mordan, Gilles Henaff, Nicolas Thome, and Matthieu Cord. 2018. Revisiting multi-task learning with rock: A deep residual auxiliary block for visual detection. In *Advances in Neural Information Processing Systems*.

[35] Silvia Nittel, J. C. Whittier, and Qinghan Liang. 2012. 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*. ACM Press, New York, New York, USA, 530–533. <https://doi.org/10.1145/2424321.2424407>

[36] Augustus Odena, Christopher Olah, and Jonathon Shlens. 2017. Conditional image synthesis with auxiliary classifier gans. In *34th International Conference on Machine Learning, ICML 2017*, Vol. 6. 4043–4055. arXiv:1610.09585 <http://proceedings.mlr.press/v70/odena17a.html>

[37] Jung Yeon Park, Kenneth Theo Carr, Stephan Zheng, Yisong Yue, and Rose Yu. 2020. Multiresolution Tensor Learning for Efficient and Interpretable Spatial Analysis. (feb 2020). arXiv:2002.05578 <http://arxiv.org/abs/2002.05578>

[38] Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. arXiv:1511.06434

[39] Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and Prabhat. 2019. Deep learning and process understanding for data-driven Earth system science. *Nature* 566, 7743 (feb 2019), 195–204. <https://doi.org/10.1038/s41586-019-0912-1>

[40] Johannes L. Schonberger, Marc Pollefeys, Andreas Geiger, and Torsten Sattler. 2018. Semantic Visual Localization. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2018.00721> arXiv:1712.05773

[41] Donald Shepard. 1968. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference, ACM 1968*.

[42] A. Stein and L. C. A. Corsten. 1991. Universal Kriging and Cokriging as a Regression Procedure. *Biometrics* (1991). <https://doi.org/10.2307/2532147>

[43] S. C. Sudartha and Y. L. Kergosien. 1990. 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)*, Vol. 412 LNCS. Springer Verlag, 120–129. https://doi.org/10.1007/3-540-52255-7_33

[44] Yusuke Tanaka, Tomoharu Iwata, Toshiyuki Tanaka, Takeshi Kurashima, Maya Okawa, and Hiroyuki Toda. 2019. 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*. <https://doi.org/10.1609/aaai.v33i01.33015091> arXiv:1809.07952

[45] Bingsheng Wang and Jinjun Xiong. 2014. Novel geospatial interpolation analytics for general meteorological measurements. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. <https://doi.org/10.1145/262330.2623367>

[46] Ke Alexander Wang, Geoff Pleiss, Jacob R Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. 2019. Exact Gaussian Processes on a Million Data Points. In *Advances in Neural Information Processing Systems (NeurIPS)*, 14648–14659. arXiv:1903.08114 <http://arxiv.org/abs/1903.08114>

[47] Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. 2020. Towards Physics-informed Deep Learning for Turbulent Flow Prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. <https://doi.org/10.1145/3394486.3403198> arXiv:1911.08655

[48] Xiaolong Wang and Abhinav Gupta. 2016. 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)*, Vol. 9908 LNCS. Springer Verlag, 318–335. https://doi.org/10.1007/978-3-319-46493-0_20 arXiv:1603.05631

[49] Jianpeng Xu, Xi Liu, Tyler Wilson, Pang Ning Tan, Pouyan Hatami, and Lifeng Luo. 2018. Muscat: Multi-scale spatio-temporal learning with application to climate modeling. In *IJCAI International Joint Conference on Artificial Intelligence*. <https://doi.org/10.24963/ijcai.2018/404>

[50] Jianpeng Xu, Pang Ning Tan, Lifeng Luo, and Jiayu Zhou. 2016. GSpartan: A geospatio-temporal multi-task learning framework for multi-location prediction. In *16th SIAM International Conference on Data Mining 2016, SDM 2016*. <https://doi.org/10.1137/1.9781611974348.74>

[51] Bo Yan, Gengchen Mai, Krzysztof Janowicz, and Song Gao. 2017. 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*. <https://doi.org/10.1145/3139958.3140054>

[52] Xiwen Yi, Junbo Zhang, Zhao yuan Wang, Tianrui Li, and Yu Zheng. 2018. Deep distributed fusion network for air quality prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. <https://doi.org/10.1145/3219819.3219822>

[53] Yifang Yin, Zhenguang Liu, Ying Zhang, Sheng Wang, Rajiv Ratn Shah, and Roger Zimmermann. 2019. GPS2Vec: Towards generating worldwide GPS embeddings. In *SIGSPATIAL: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*. Association for Computing Machinery, New York, NY, USA, 416–419. <https://doi.org/10.1145/3347146.3359067>

[54] Andrew Zammit-Mangion, Tin Lok James Ng, Quan Vu, and Maurizio Filippone. 2019. Deep Compositional Spatial Models. (jun 2019). arXiv:1906.02840 <http://arxiv.org/abs/1906.02840>

[55] Na Zhang and Hongyan Zhang. 2011. Scale variance analysis coupled with Moran's I scalogram to identify hierarchy and characteristic scale. *International Journal of Geographical Information Science* 25, 9 (sep 2011), 1525–1543. <https://doi.org/10.1080/13658816.2010.532134>

[56] Yunchao Zhang, Yanjie Fu, Pengyang Wang, Xiaolin Li, and Yu Zheng. 2019. 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*. <https://doi.org/10.1145/3292500.3330972>

[57] Di Zhu, Ximeng Cheng, Fan Zhang, Xin Yao, Yong Gao, and Yu Liu. 2019. Spatial interpolation using conditional generative adversarial neural networks. *International Journal of Geographical Information Science* (apr 2019), 1–24. <https://doi.org/10.1080/1365881YYxxxxxxxx>

APPENDIX

A: Data Description

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:

$$\begin{aligned} f(cx, cy, s) = & 0.75 \exp(-((9cx - a)^2 + (9cy - b)^2)/s) \\ & - (0.75 \exp(-((9cx - d)^2 + (9cy - e)^2)/s)) \end{aligned} \quad (13)$$

where cx and cy 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 [27] 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 [18], 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: Experimental Setting, Model Architectures and Compute

B.I: Generative Spatial Modeling

Setup: Our main experimental findings, the MMD scores displayed in Table 1 and Table 2 (main paper), 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 convolutional 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 [38].

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 supported 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 [57], 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 back-propagation 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

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

[6], 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

$$\widehat{MMD}^2 = \frac{1}{n(n-1)} \sum_{i \neq j} k(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i \neq j} k(y_i, y_j) - \frac{2}{n^2} \sum_{i,j} k(x_i, y_j), \quad (14)$$

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.

B.II: 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 :

$$RMSE(y_i, \hat{y}_i) = \sqrt{\frac{1}{n} \sum (y_i - \hat{y}_i)^2} \quad (15)$$

C: Code

All our experiments are implemented using *PyTorch*³. The code for our experiments can be found here: <https://drive.google.com/file/d/1ShOBV7RifMdS9LYsySOM084KL5aD-wkm/view?usp=sharing>.

³<https://pytorch.org/>