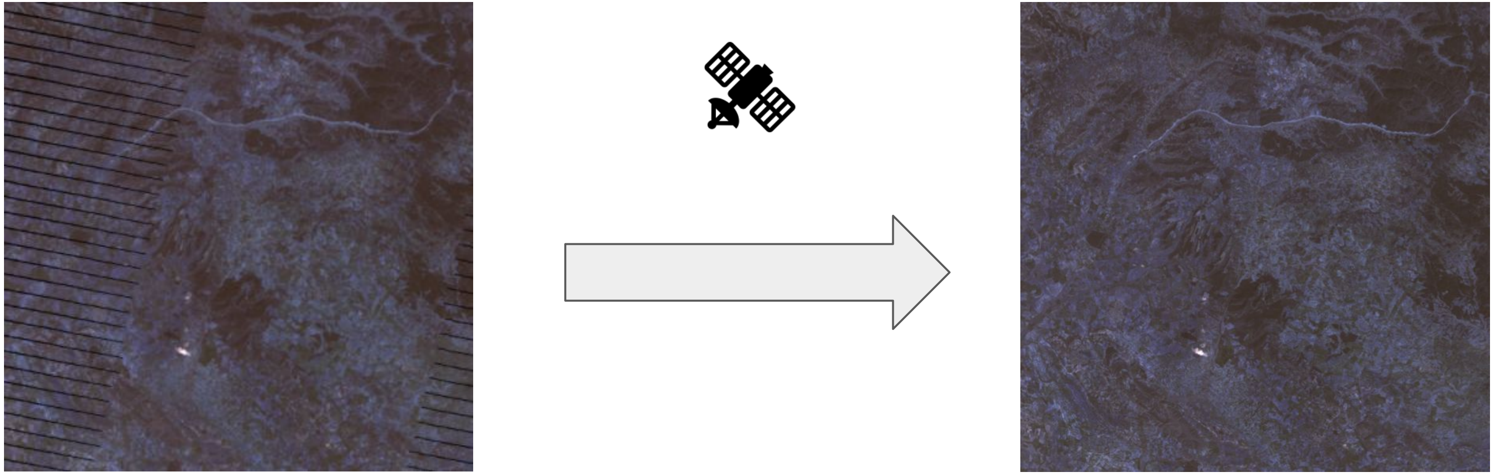
Convolutional Neural Processes for Inpainting Satellite Images

Alexander Pondaven, Hamzah Hashim
Imperial College London



Team

Alexander Pondaven*

MEng Electronic and Information Engineering

Märt Bakler*

MSc Artificial Intelligence

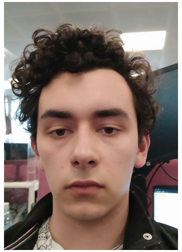
Donghu Guo

MSc Environmental Data Science and Machine Learning

Hamzah Hashim

MSci Theoretical Physics

Martin Ignatov



MSci Physics

Harrison Zhu

PhD Modern Statistics and Statistical Machine Learning

*equal contribution

Our Contributions

- ☑ Repair LANDSAT 7 imagery with **Convolutional Neural Processes**
- ☑ State-of-the-art inpainting performance on in-distribution and **especially** out-of-distribution (OOD) inpainting
- ☑ Strong performance with synthetic **downstream regression** tasks

Satellite Imagery: LANDSAT 7

- LANDSAT 7 - images collected by NASA/USGS via the LANDSAT programme
- High-resolution (30m) images publicly available (**massive, terabytes**!)
- Scanline corrector (SLC) failure on 31st May 2003 → missing values at scanlines 😟

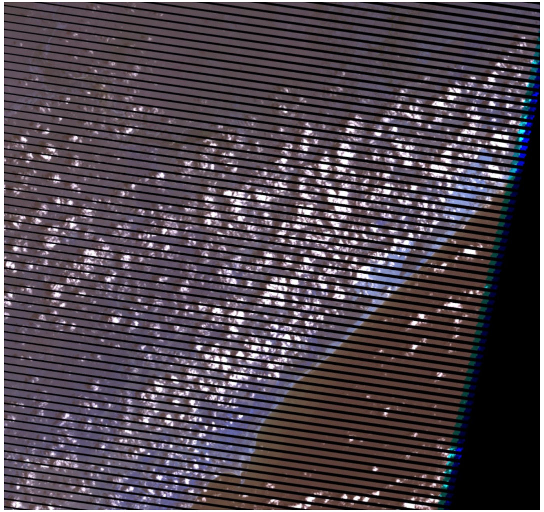
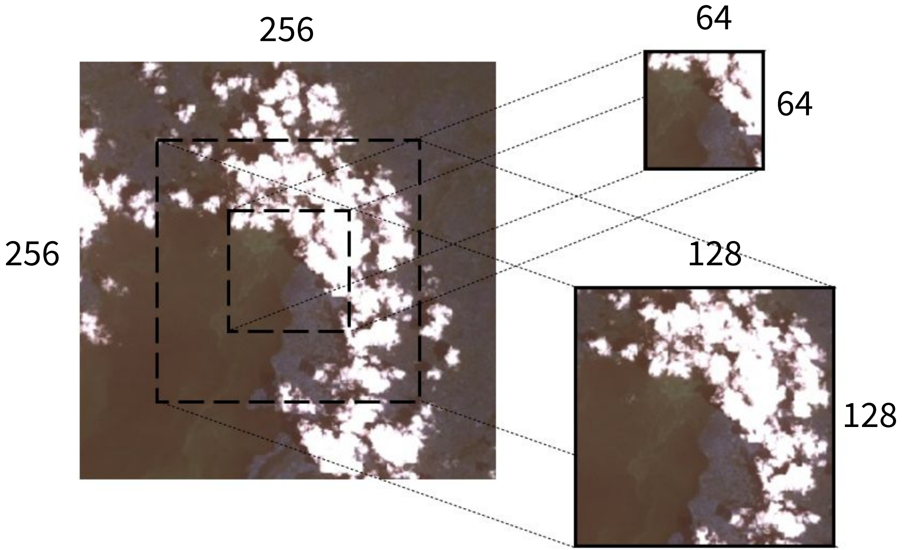


Figure 1: Snapshot in Kenya. Taken on 3rd January, 2005, after the SLC failure

Data from Google Earth Engine

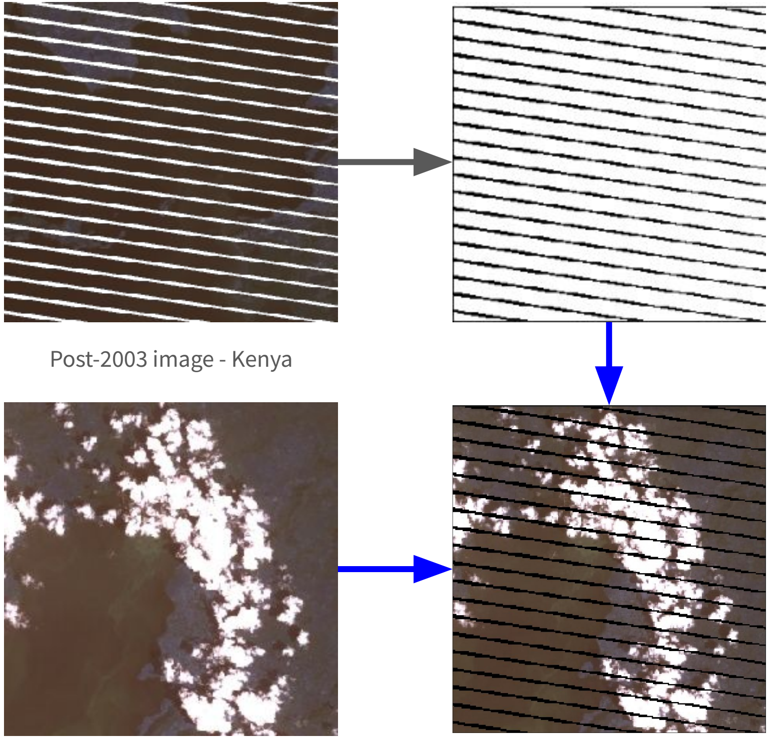
- LANDSAT 7 Satellite images extracted using Google Earth Engine API (Gorelick et al. 2017)
- RGB channels/bands
- 256x256 images downloaded
- Cropped to 128x128 and 64x64 for training



- In-distribution country
  - Kenya
- Out-of-distribution countries
  - UK
  - Brazil
  - Nepal
  - Norway

Data Processing for Training

- Post-2003 images used to extract scanline bitmasks
- Pre-2003 uncorrupted images used for training



Post-2003 image - Kenya

Pre-2003 image - Kenya

Extract scanline from post-2003 images

Apply scanline mask to pre-2003 images for training

Baselines: Previous Attempts

Classical approaches:

- Interpolation & PDEs (Bertalmio et al. 2001; Richard and Chang 2001; Telea 2004) - deterministic
- Official LANDSAT 7 inpainting (Scaramuzza & Barsi 2005)- linear regression via clean and corrupt image matching

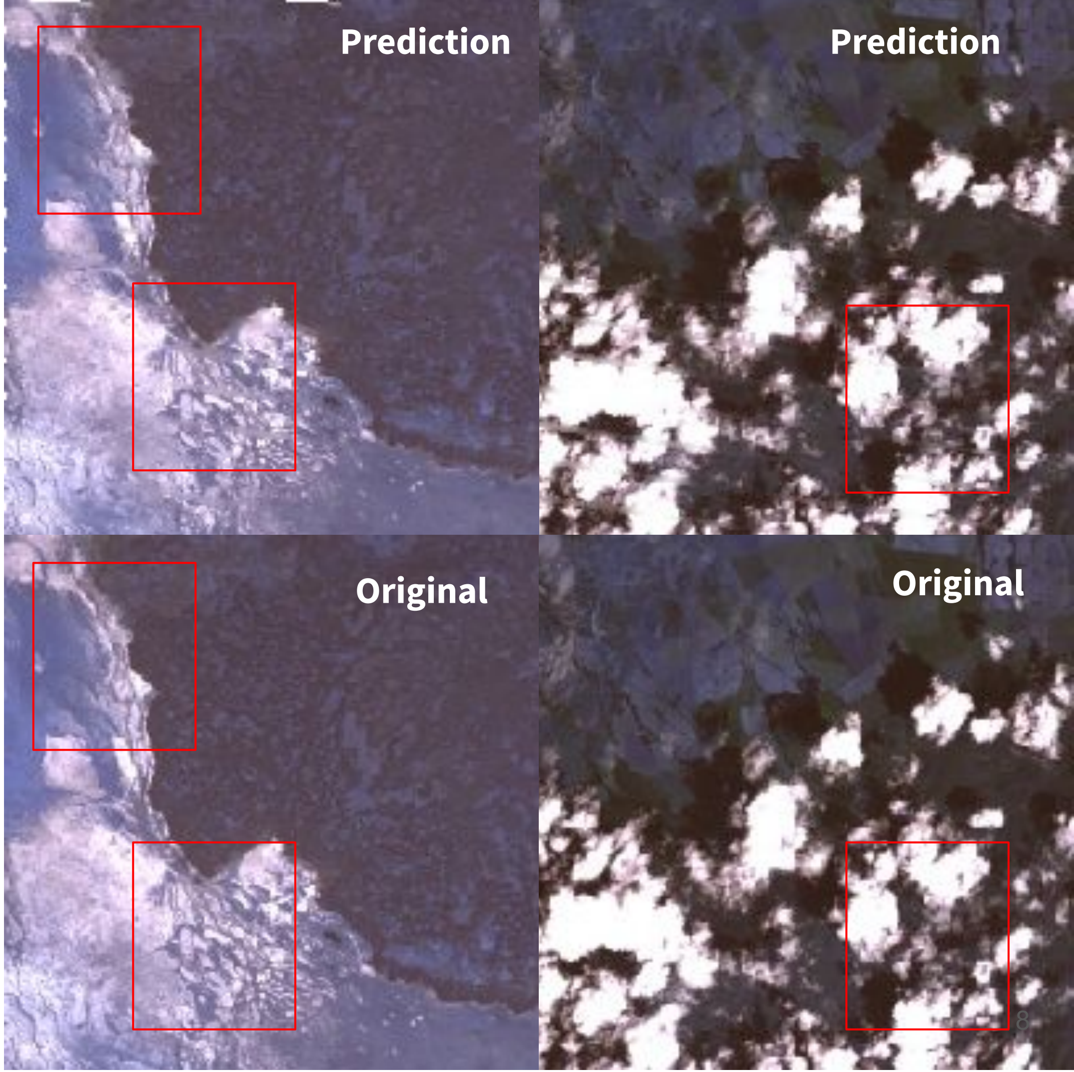
Deep learning:

- **U-Net** (Ronneberger et al. 2015)
- GANs (Pathak et al. 2016)
- Partial Convolutions (**PartialConv**; Liu et al. 2018)
- HI-VAE (Nazabal et al. 2020)
- Recently: Convolutional Neural Processes (**ConvNPs**; Foong et al. 2020; Markou et al. 2022), denoising diffusion probabilistic models (Lugmayr et al. 2022)

Bertalmio et al. (2001); Richard & Chang (2001); Telea (2004); Scaramuzza & Barsi (2005); Ronneberger et al. (2015) Pathak et al. (2016); Liu et al. (2018); Nazabal et al. (2020); Foong et al. (2020); Lugmayr et al. (2022); Markou et al. (2022)

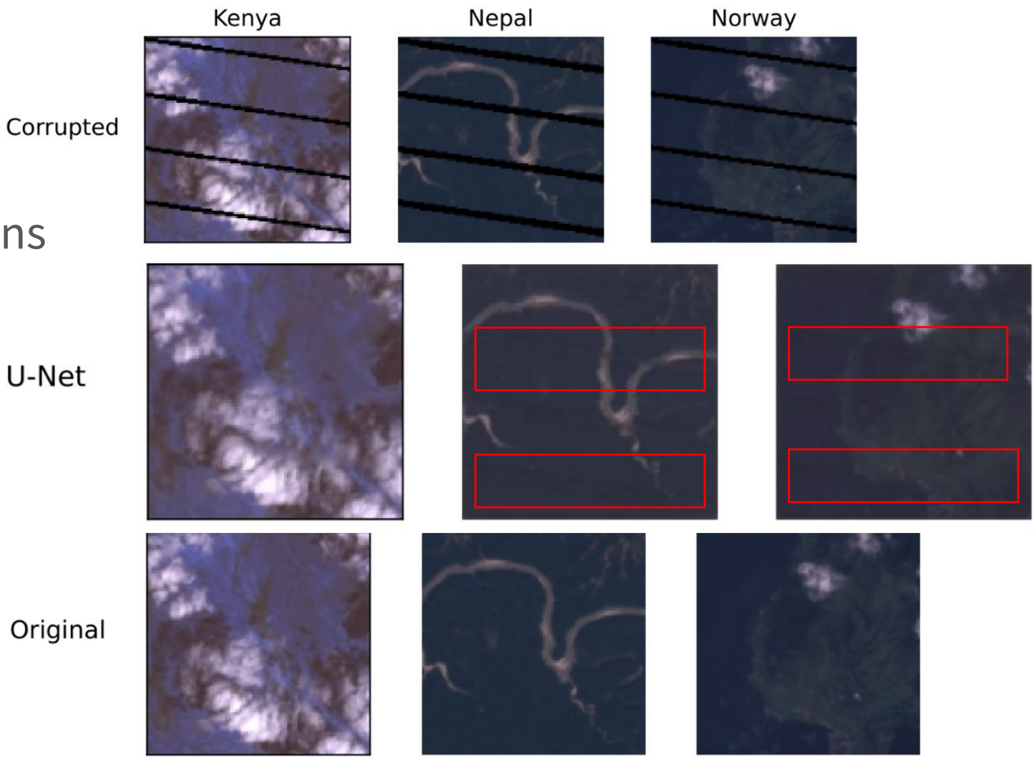
Navier-Stokes (NS)

- Bad at borders between different colors (clouds - land, sea - land)
- Scanlines generally noticeable



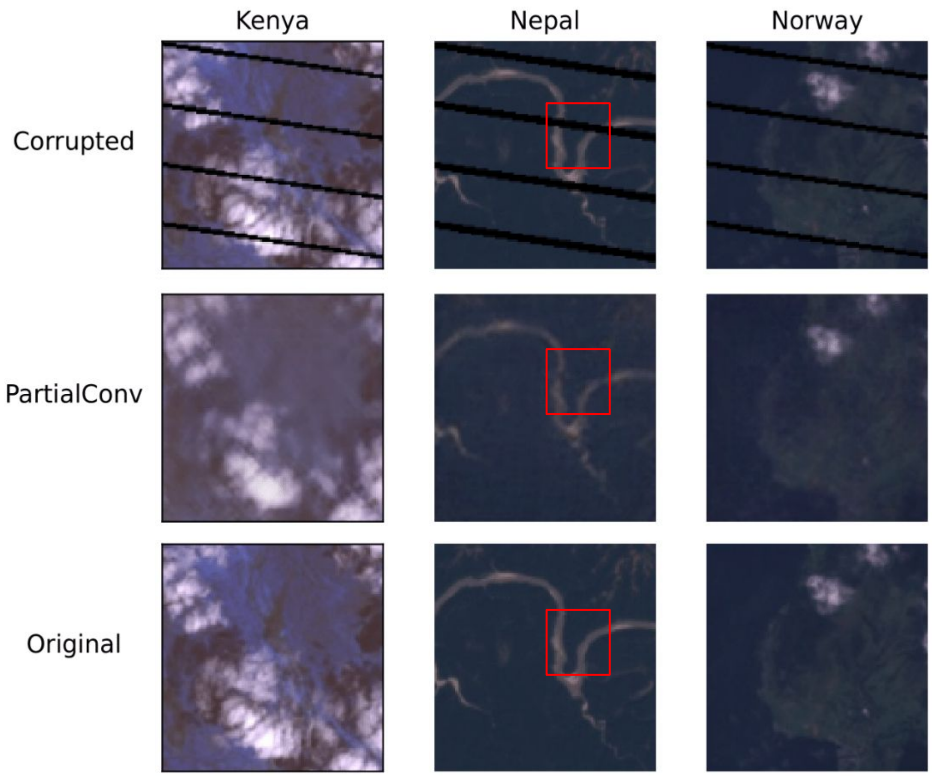
U-Net

- Learns global function
- In-distribution Kenya does well
- Poor out-of-distribution predictions



Partial Convolutions (PartialConv)

- U-Net-like architecture
- Partial convolutional - mask-aware
- Blurry in general and scanlines also generally visible



Baselines: Comparison

Navier-Stokes

- ☑ Fast
- ☒ No information sharing between images

U-Net

- ☑ Expressive and works quite well for a lot of problems
- ☒ OOD requires large datasets and data augmentation

PartialConv

- ☑ Convolution takes into account of masks/missing pixels
- ☒ Requires large datasets and long training times

Supervised Learning

- Single dataset (context)

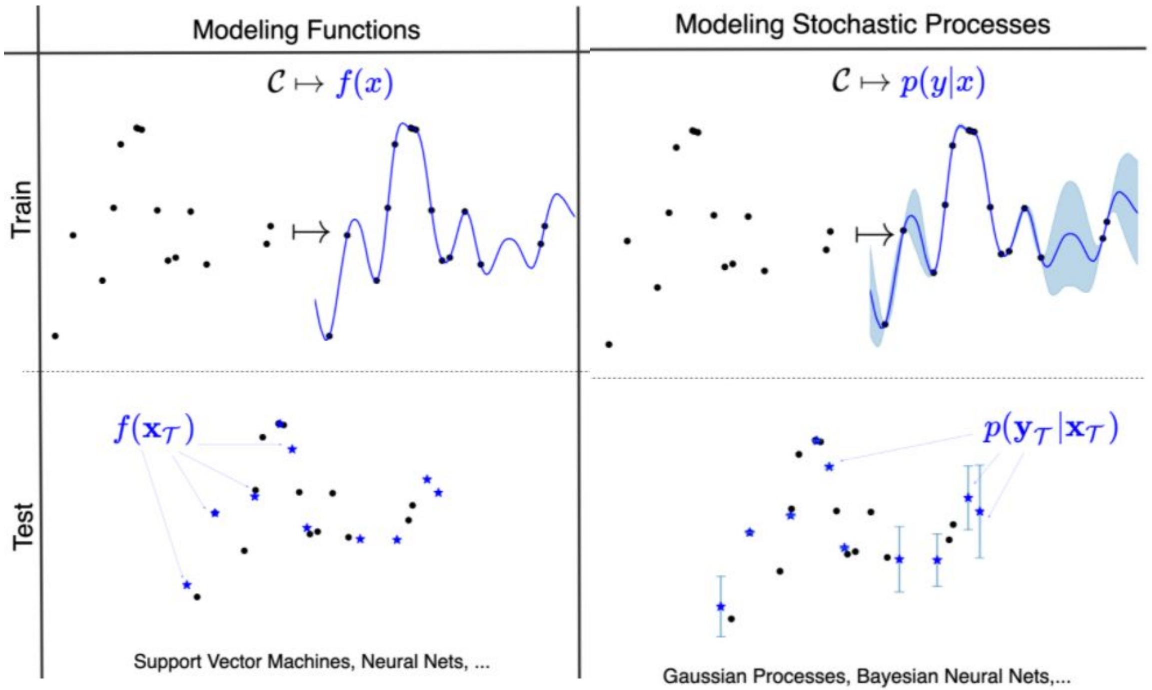
  $$\mathcal{C} := \{(x^{(c)}, y^{(c)})\}_{c=1}^{C}$$

- Learns predictor

  $$f(x)$$

- Predict target points

  $$f(\mathbf{x}_{\mathcal{T}})$$



Further reading: https://yanndubs.github.io/Neural-Process-Family

Meta Learning

- "Learning to learn" - Adapt to new supervised tasks

- Collection of datasets/tasks (Meta-dataset)

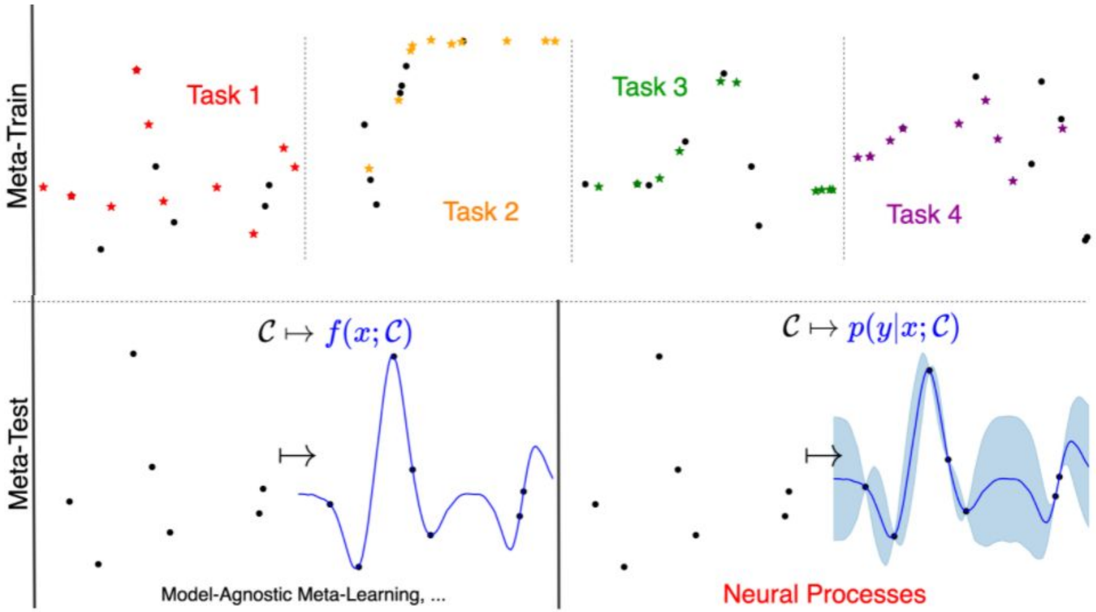
$$\mathcal{M} = \{\mathcal{D}_i\}_{i=1}^{N_{\text{tasks}}}$$

- Learns mapping

$$\mathcal{C} \mapsto f(x; \mathcal{C})$$

- Adapt predictor to new context set

$$f(x; \mathcal{C})$$

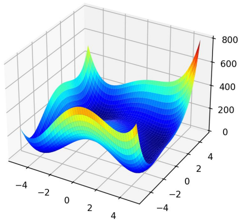


Further reading: https://yanndubs.github.io/Neural-Process-Family

Satellite inpainting ➡ Meta-Learning problem

$x \in \mathbb{R}^2$ Pixel location on grid

$y \in \mathbb{R}^3$ RGB pixel value



Task is 2D function

Context set $(x_C, y_C) := \{x_i, y_i\}_{i=1}^{N_C}$

Target Set $(x_T, y_T) := \{\bar{x}_i, \bar{y}_i\}_{i=1}^{N_T}$

Task $D := \{C, T\}$ where $C = \{x_C, y_C\}$

$T = \{x_T, y_T\}$

Supervised approach

- Learn global function f_θ that predicts $y_T \approx f_\theta(x_T)$
- Implicitly distinguish between different tasks

$$f_\theta(x_{C_m}, y_{C_m}, x_{T_m}) \approx f_{\theta_m}(x_{T_m})$$

Meta-learning approach

- Objective function $\mathbb{E}_{m\sim\mathcal{M}}[\mathcal{L}(\underbrace{D_\eta(E_\xi(x_{C_m}, y_{C_m}))(x_{T_m})}_{f_{\theta_m}(x_{T_m})}, y_T)]$

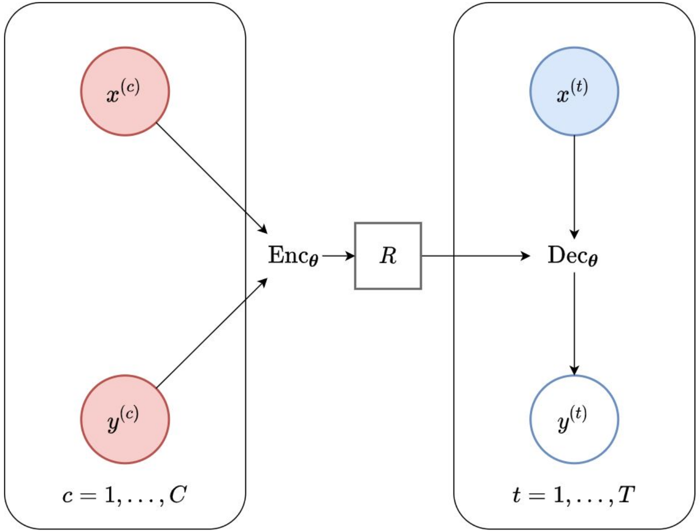
$f_{\theta_m}(x_{T_m})$ where $\theta = (\eta, \xi)$

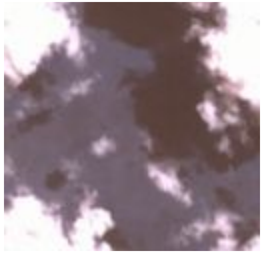
- E_ξ encodes context (x_C, y_C) to task-specific representation
- D_η decodes representation and target location to output
- $\mathcal{L}$ is the loss

Neural Processes for Inpainting

- Satellite images are different regression problems
  - Different location and time
- Small dataset for each task

Context points are non-scanline pixels





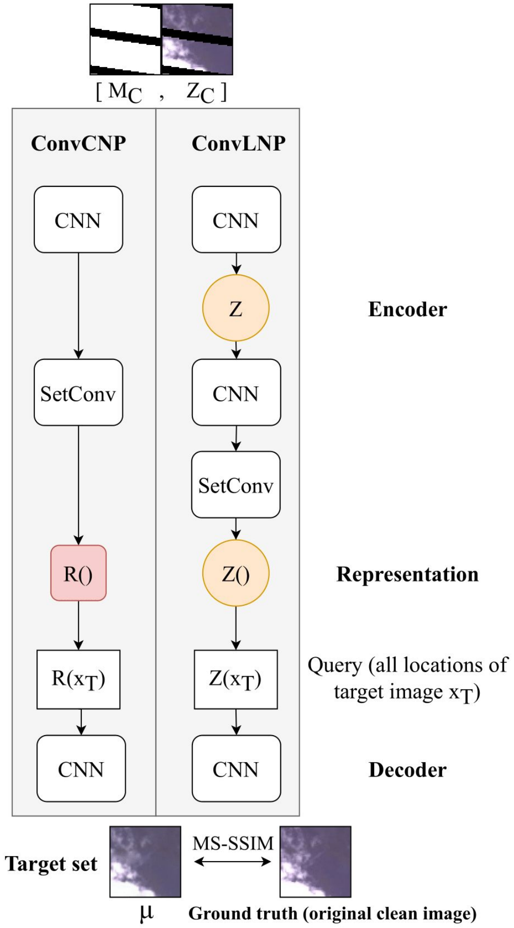
Target points are entire image (for continuity)

Convolutional Neural Processes

- Translational equivariance

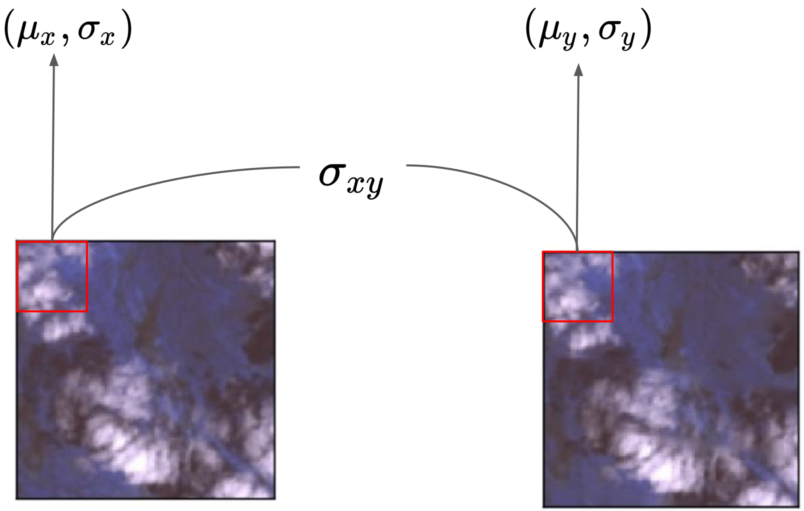
- Convolutional Conditional Neural Processes
- Convolutional Latent Neural Processes

- Trained using Maximum Likelihood
- Multi-Scale Structural Similarity (MS-SSIM) Loss (Wang et al. 2003) generates sharper images



Multi-Scale Structural Similarity (Wang et al. 2003)

- In practice, calculated on **windows** between 2 images - convolution with Gaussian kernel
- Then average SSIM over windows
- Spatial structure-aware



$$l(\mathbf{x}, \mathbf{y}) = \frac{2\,\mu_x\,\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1},$$

$$c(\mathbf{x}, \mathbf{y}) = \frac{2\,\sigma_x\,\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2},$$

$$s(\mathbf{x}, \mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x\,\sigma_y + C_3},$$

Structural Similarity (SSIM):

$$\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = [l(\mathbf{x}, \mathbf{y})]^{\alpha} \cdot [c(\mathbf{x}, \mathbf{y})]^{\beta} \cdot [s(\mathbf{x}, \mathbf{y})]^{\gamma}$$

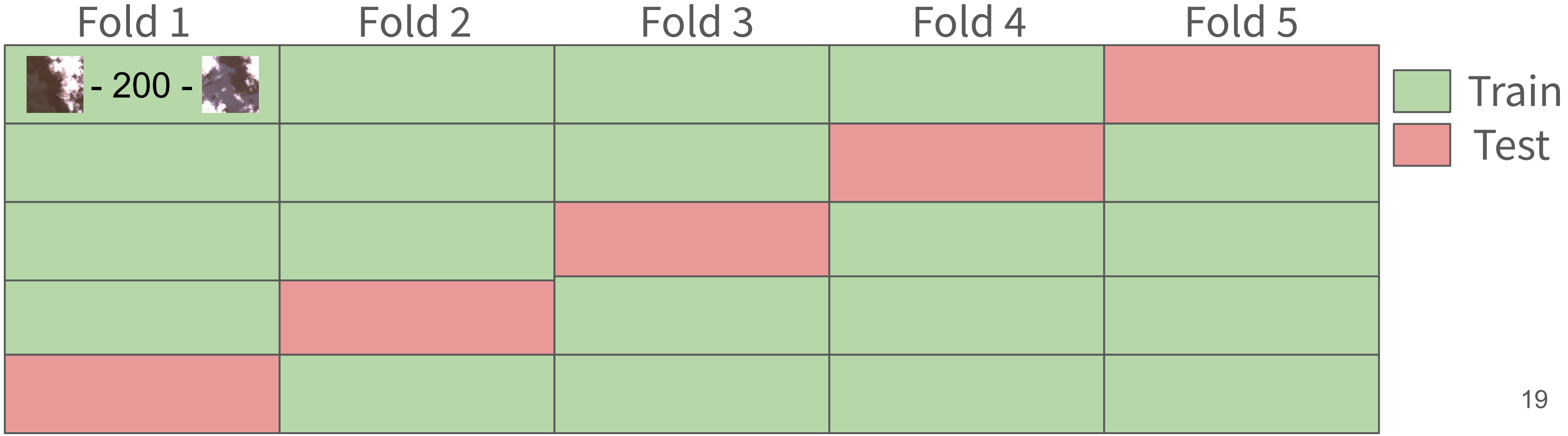
Multi-Scale Structural Similarity (MS-SSIM):

$$\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = [l_M(\mathbf{x}, \mathbf{y})]^{\alpha_M} \cdot \prod_{j=1}^{M} [c_j(\mathbf{x}, \mathbf{y})]^{\beta_j} [s_j(\mathbf{x}, \mathbf{y})]^{\gamma_j}$$

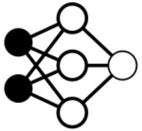
From https://github.com/VainF/pytorch-msssim; See https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e for a deeper explanation

Experiments: Data Collection and Training

- NP models from Github implementation by Yann Dubois (Dubois et al. 2020).
- Models trained on Kenya
- Kenya model used for inference on all countries
- Each country has dataset of 1000 images
- 5-fold cross validation with 80:20 split



Experiment 1: Setup



ConvCNP

- 10-layer ResNet encoder
- 128 channel representation
- 4-layer MLP in decoder

- 400 epochs
- Batch size 8
- Learning rate 1e-4
- Exponential decay by factor 5

ConvLNP 64x64

- 8-layer ResNet encoder
- Latent samples:
  - 16 for training
  - 32 for inference

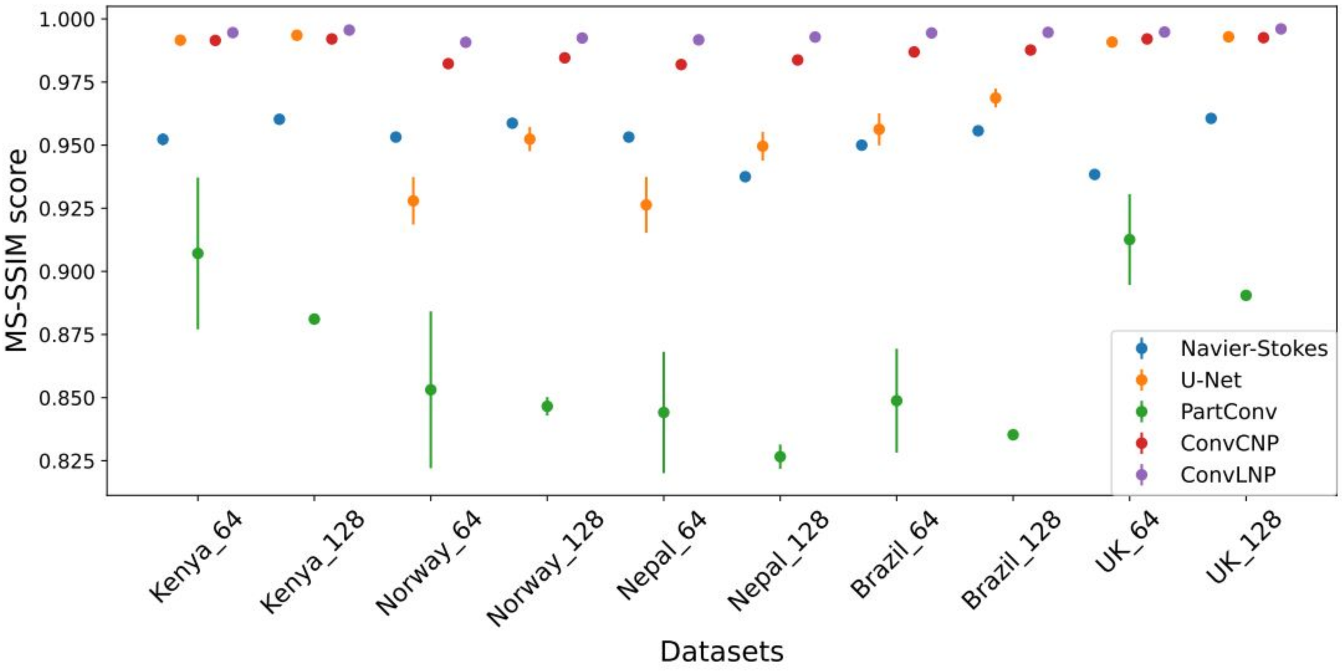
ConvLNP 128x128

- 8-layer ResNet encoder
- Latent samples:
  - 4 for training
  - 8 for inference

- 200 epochs
- Batch size 4
- Learning rate 5e-4

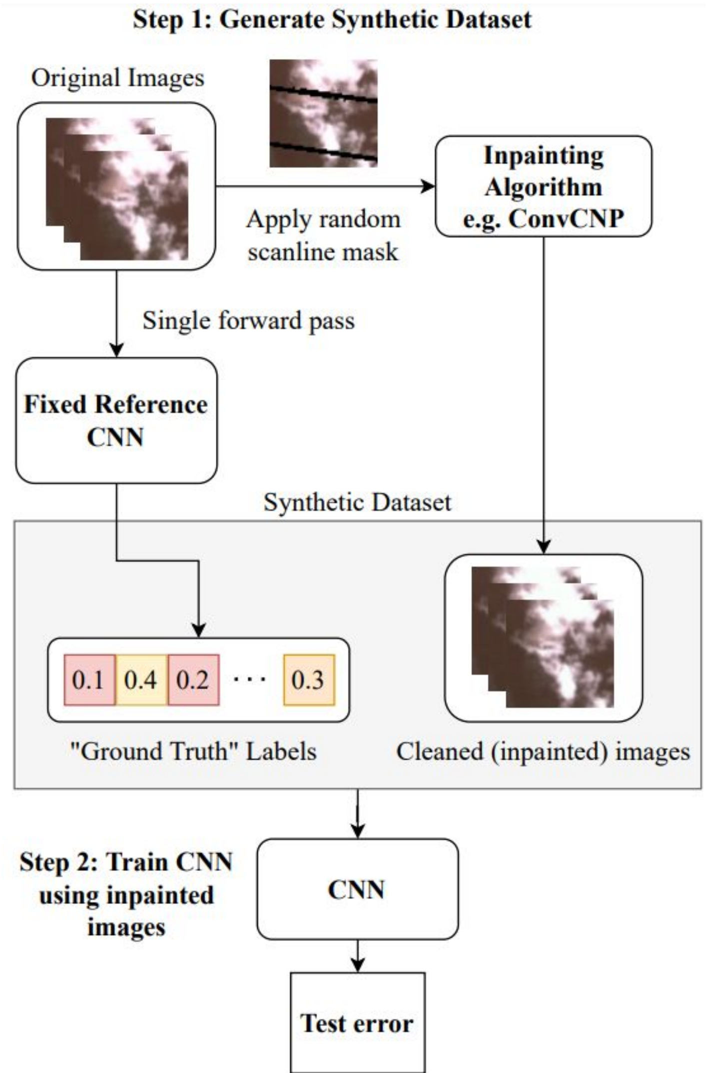
Inpainting results

- ConvLNP
- ConvCNP
- U-Net
  - In-distribution
  - Out-of-distribution
- Navier-Stokes
- PartialConv

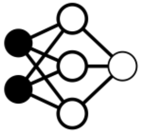


Experiment 2: Synthetic Downstream Task

- Performance of inpainted results on downstream regression task
- Only 64x64 images
- Clean image and corrupted image (with scanline) also used for downstream task as reference



Downstream task setup

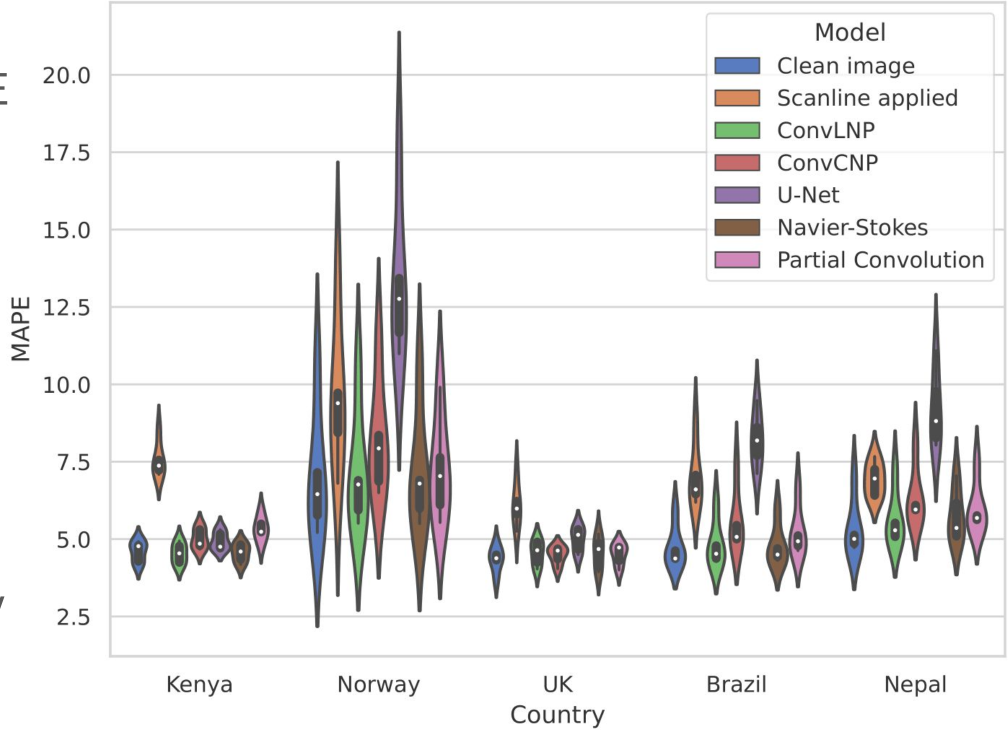


- CNN
  - 2 convolutional layers
  - Kernel size 3
  - Final fully connected layer
- MSE loss

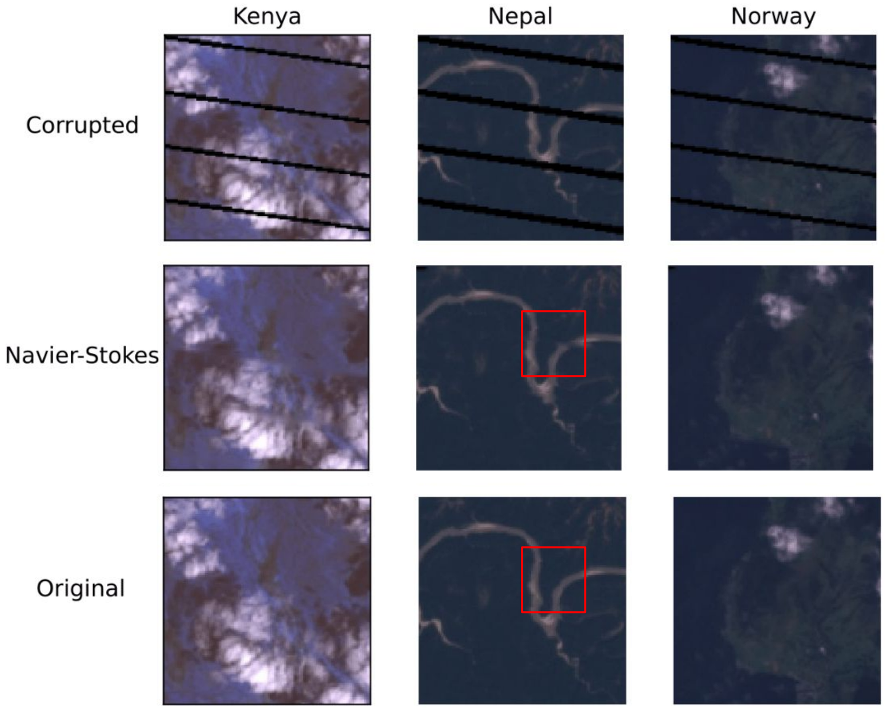
- 300 epochs
- Batch size 8
- Learning rate 1e-3 with reduction on plateau
- Early stopping with patience 8 epochs
- 5-fold cross validation

Downstream Task Results

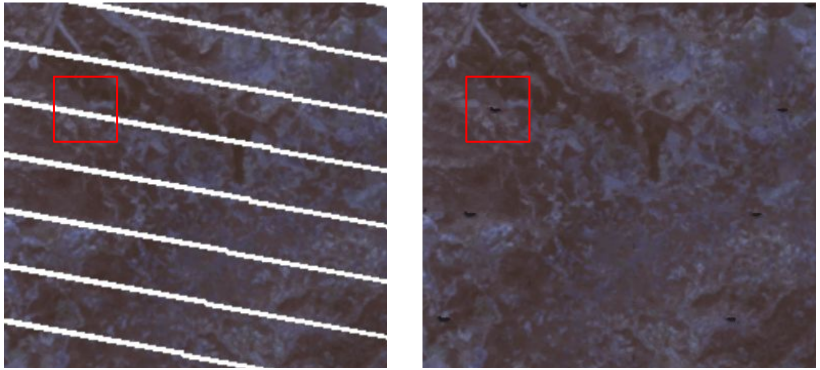
- Violin plot shows variation in MAPE over 5 folds of cross-validation
- ConvLNP performs best
- U-Net performs badly out-of-distribution
- Navier-Stokes
  - Only scanline changes
- Norway is a difficult task
- Not a good measure of PartialConv performance



Navier-Stokes Results



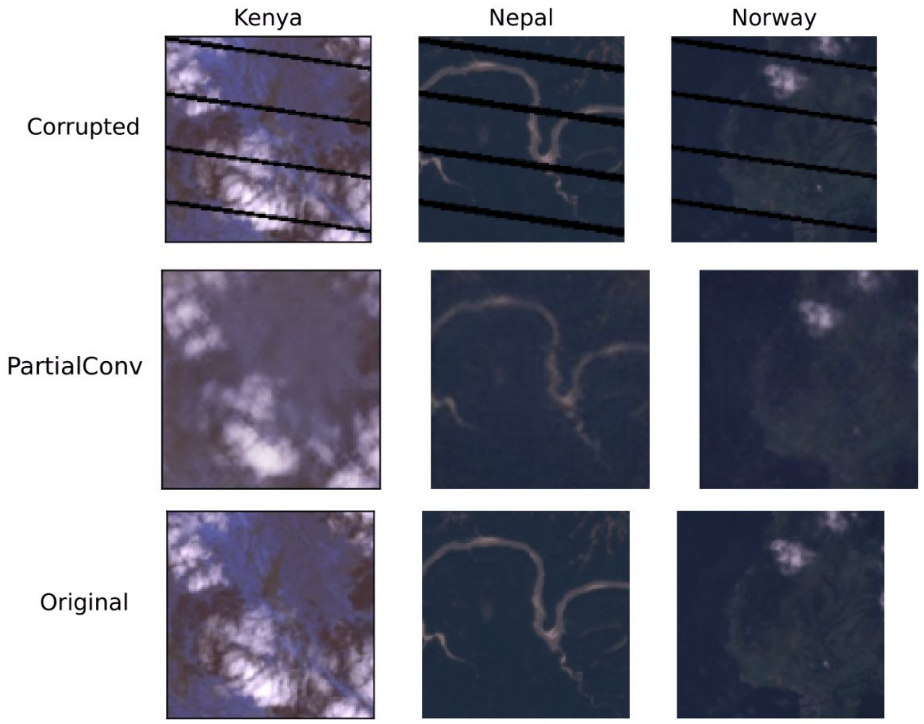
Inpainting patches of larger image

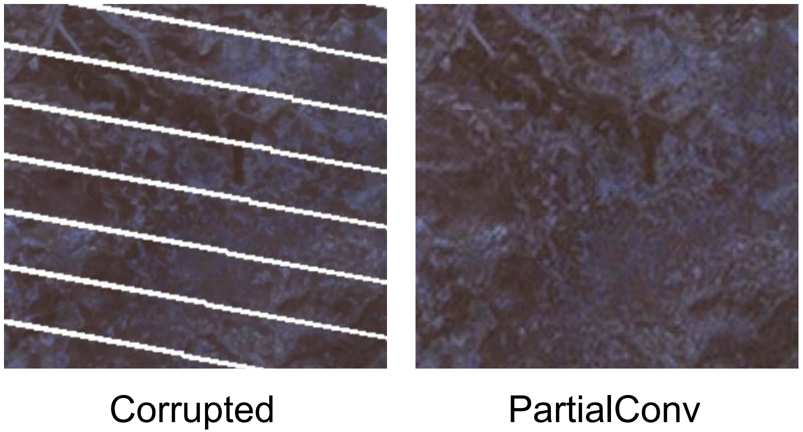


Corrupted

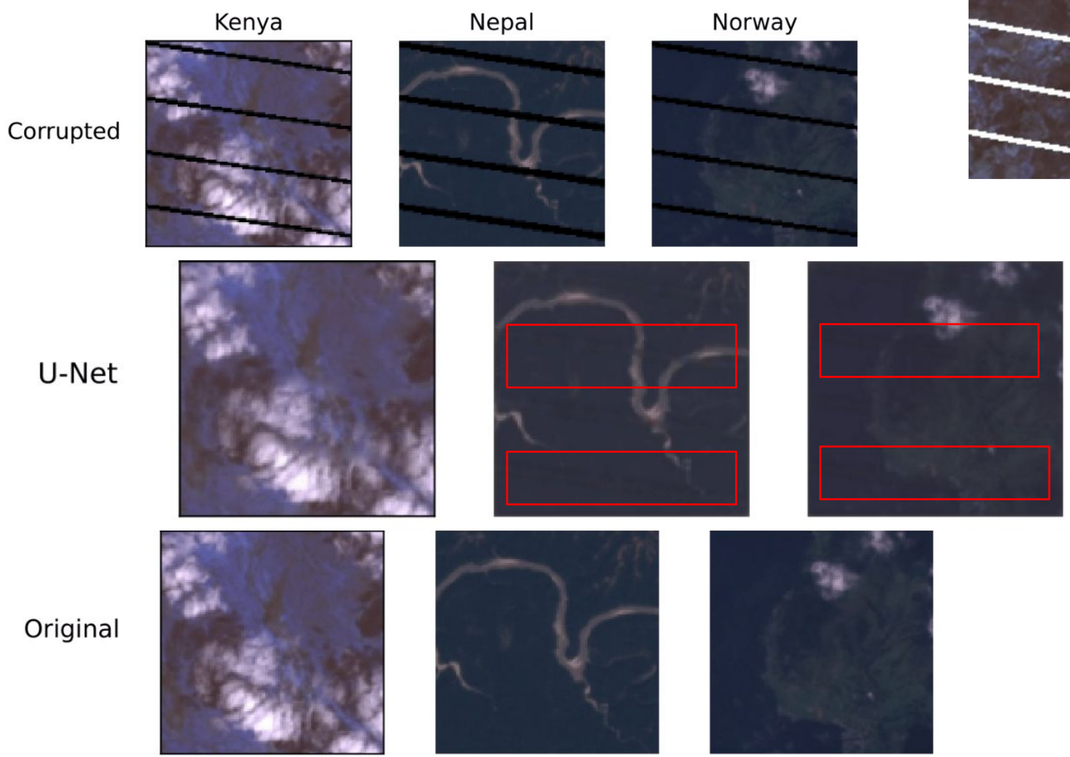
Navier-Stokes Inpainted

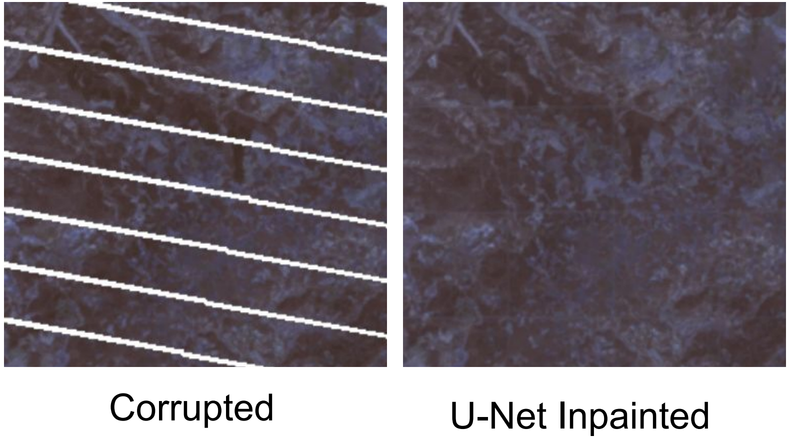
PartialConv Results



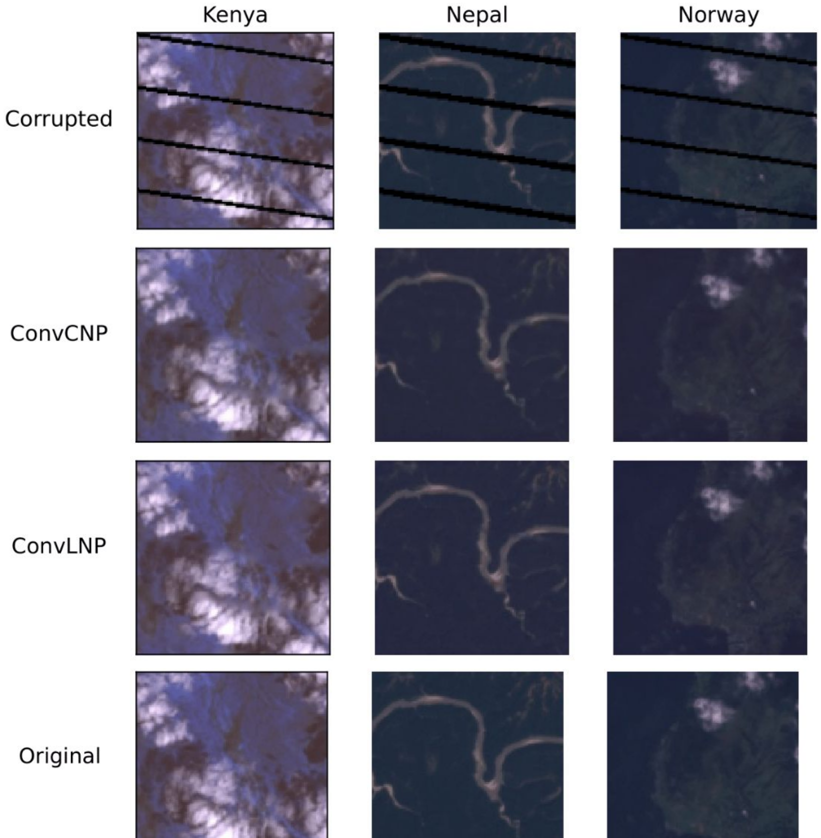


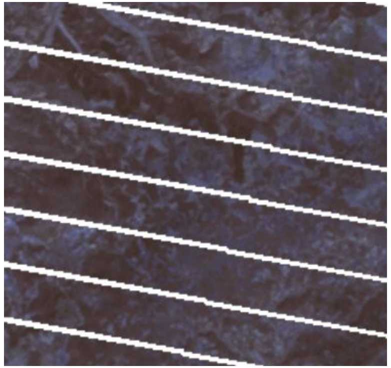
U-Net Results



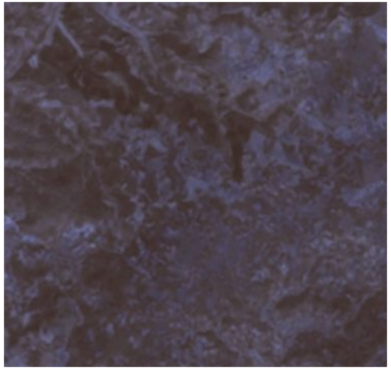


ConvNP Inpainted Results

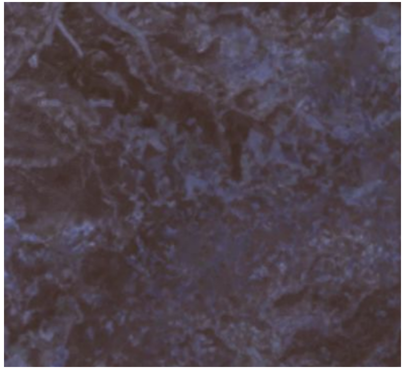




Corrupted



ConvCNP
Inpainted



ConvLNP
Inpainted

Conclusion and Discussion

- ☑ ConvNPs successful at inpainting in-distribution and out-of-distribution
  - ☑ Take advantage of different spatiotemporal structure of satellite images
  - ☑ Global inpainter for LANDSAT 7 by only training small subset of locations
- Bigger scanlines
- Cloud removal
- More interesting downstream tasks

Potential Downstream Applications

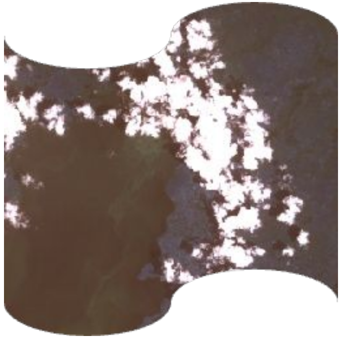
🌎 Housing/Macroeconomic Mapping:

- **Inputs:** Imputed Landsat 7 maps
- **Model:** CNN/Transformers/GNN
- **Outputs:** Housing inequality index, or potentially multivariate outputs

🦟 Malaria Prevalence Mapping:

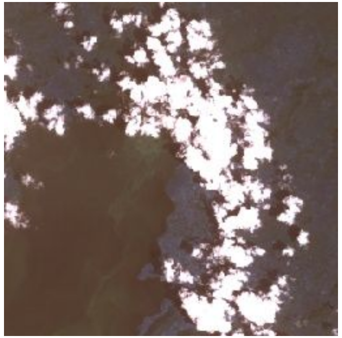
- **Inputs:** Pixels of Landsat 7 maps inside regions of interest
- **Model:** DeepSets, Set Transformer, Gaussian processes over distributions
- **Outputs:** Malaria cases

$\{x_i\}_{i=1}^{143}$



or

$x \in R^{64\times 64}$



Irregular/set/distribution data → DeepSets/Set Transformer/Gaussian processes over distributions

$f(x)$

Image data → CNN/Transformers/GNN

Housing Inequality Index or Malaria cases

Thank you!

Our paper: https://arxiv.org/pdf/2205.12407.pdf

Any questions? 

References

- Bertalmio et al. (2001). Navier-Stokes, Fluid Dynamics and Image and Video Inpainting. CVPR
- Dubois et al. (2020). Neural Process Family. https://yanndubs.github.io/Neural-Process-Family
- Foong et al. (2020). Meta-Learning Stationary Stochastic Process Prediction with convolutional Neural Processes. NeurIPS
- Garnelo et al. (2018). Conditional neural processes. ICML
- Garnelo et al. (2018). Neural Processes. *ICML Workshop on Theoretical Foundations and Applications of Deep Generative Models*
- Gordon et al. (2020). Convolutional conditional neural processes. *ICLR*
- Gorelick et al. (2017). Google earth engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*
- Liu et al. (2018). Image inpainting for irregular holes using partial convolutions. ECCV
- Lugmayr et al. (2022). Repaint: Inpainting using denoising diffusion probabilistic models
- Markou et al. (2022). Practical conditional neural processes via tractable dependent predictions. *ICLR*
- Nazabal et al. (2020). Handling incomplete heterogeneous data using vaes. *Pattern Recognition*
- Richard and Chang (2001). Fast digit image inpainting. VIIP
- Ronneberger et al. (2015). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*
- Scaramuzza and Barsi(2005). Landsat 7 scan line corrector-off gap-filled product development
- Telea (2004). An image inpainting technique based on the fast marching method. *Journal of graphics tools*
- USGS. Landsat 7 Courtesy of the U.S. Geological Survey.
- Wang et al. (2003). Multiscale structural similarity for image quality assessment. *Asilomar Conference on Signals, Systems & Computers*