A Generative Deep Learning Approach to Stochastic Downscaling of Precipitation Forecasts

Lucy Harris & Andrew McRae, AOPP, University of Oxford
Mat Chantry, ECMWF
Peter Dueben, ECMWF
Tim Palmer, AOPP
Problem and motivation

Global forecast models produce good large-scale predictions, but output cannot be used directly for small spatial scales (<1km). Precipitation is particularly variable over small lengthscales (c.f. pressure, wind, …)

Current workflows include further postprocessing, running LAMs, etc.

Post-processing downscaling of precipitation forecasts is necessary to assess the impact of extreme rainfall situations.

Can ML models help? What are the limits of postprocessing global model output?
Stochastic Super-Resolution for Downscaling **Time-Evolving** Atmospheric Fields with a GAN
Leinonen et al. 2020
Increasing the accuracy and resolution of precipitation forecasts using deep generative models

Price & Rasp 2022
NIMROD

1km C-band radar-based rainfall, adjusted with gauge measurements. Regridded to 0.01°, accumulated to 1-hour. Domain: 49.5 - 59 latitude, -7.5 - 2 longitude.

IFS

-9km forecast output (lead-times 6-18 hours) Regridded to 0.1°, 1-hour. Fields: Total precipitation, convective precipitation, CAPE, u-700hPa, v-700hPa, TOA incident solar radiation, Total column cloud liquid water, Total.
inputs

Lo-res IFS fields:  
(based on ecPoint approach)
- Total precipitation  
- Convective precipitation  
- Surface pressure  
- TOA incident solar radiation  
- Convective available potential energy  
- Total column cloud liquid water  
- Total column water vapour  
- u700 and v700

Hi-res ‘constant’ inputs:
- Orography  
- Land-sea mask
convolutional NN with residual blocks
Wasserstein GAN with gradient penalty

**trainable parameters**
gen – 0.83 million (64 filters)
3.2 million (128 filters)
disc – 64.1 million (512 filters)

---

**Diagram**
- **Noise**: $20 \times 20 \times N$
- **9 IFS fields**: $20 \times 20 \times 9$
- **2 constant fields**: $200 \times 200 \times 2$
- **Generator**: upscaler and concatenates
- **Fake Image**: $200 \times 200 \times 1$
- **Discriminator**: classifies each image
- **Score Fake**
- **Score Real**
- **Loss Function**: passed to optimizer
Convolutional layer:

- A convolution is a linear operation: the multiplication of a set of weights with the input.
- Multiplication (dot product) is performed between an array of input data and a 2D array of weights called a filter.
- The filter is smaller than the input and is applied systematically to each overlapping patch of the input data.
- This allows the filter to detect features across the entire image.
- Many filters used at once, e.g., 64, 128, 256.
NN architecture

Residual block:

- skip-connection blocks
- learns residual functions with reference to the layer inputs
- $x \rightarrow x + F(x)$
- $F(x)$ based on two 3x3 convolutions
Content Loss term

- Used in Deepmind nowcasting paper
- *They borrowed it from earlier ‘DVD-GAN’ work, where it was crucial*

Idea: train generator on

\[(\text{discriminator loss}) + \text{weight} \times \text{MSE(ensemble mean, truth)}\]
Results
Extreme situation (Storm Ciara)

NB. change of colorbar from 0.1-30mm
Extreme situation
(16:00-17:00 UTC 31/07/19)

NB. change of colorbar from 0.1-30mm
Quantitative metrics (256 full hourly images)

<table>
<thead>
<tr>
<th>Model</th>
<th>Evaluation Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CRPS (mm/hr)</td>
</tr>
<tr>
<td></td>
<td>pixelwise</td>
</tr>
<tr>
<td>GAN</td>
<td>0.0883</td>
</tr>
<tr>
<td>VAE-GAN</td>
<td>0.0914</td>
</tr>
<tr>
<td>ecPoint no-corr</td>
<td>0.0905</td>
</tr>
<tr>
<td>ecPoint part-corr</td>
<td>0.0905</td>
</tr>
<tr>
<td>RainFARM</td>
<td>0.1331</td>
</tr>
<tr>
<td>Lanczos</td>
<td>0.1412</td>
</tr>
<tr>
<td>Deterministic CNN</td>
<td>0.1347</td>
</tr>
</tbody>
</table>

ecPoint = ecPoint approach, calibrated on training dataset for this problem
- no-corr, part-corr = naive methods of generating full images from ecPoint pixel data
RainFARM = stochastic method by Rebora et al. (2006), “extends power spectrum” but doesn’t handle forecast error
Lanczos = Lanczos interpolation
Deterministic CNN = neural network trained on MSE (not using GAN methodology)
Rank histograms plot (cGAN)

\[
\mathcal{R} = \frac{N_s}{N_p}
\]

- \( N_s \): no. samples where pixel value is smaller than truth
- \( N_p \): total no. predictions

**Figure (a):** Rank histograms plot for different methods.

**Figure (b):** CDF of normalized rank for different methods.
Rank histograms plot (cGAN)
Lead time assessment

\[
\text{CRPSS} = 1 - \frac{\text{CRPS}_{fc}}{\text{CRPS}_{bench}}
\]
More results available…

- Pooled CRPS scores
- RALSD plots
- ROC and precision-recall curves
- Fractions Skill Scores
- CRPS vs 0-72hr forecast lead time (without retraining on longer lead times)
Conclusions + Future Work

- GAN produces sharply varying but spatially coherent results
- Similar pixel-wise accuracy to ecPoint approach (better CRPS, worse calibration)
- Once trained, moderate computational cost (1s/sample on full image)

Future ideas:
- Temporally-consistent results
- Use ensemble information
- Incorporate into downstream hydrology model
questions
backup slides
This uses old orography
Spatial coherency

Motivation: similar point-wise properties (CRPS, etc.) to ecPoint approach. Why is a spatially-coherent image better?

Two things we have tried:

- Pooling (used with CRPS, also ROC)
- Fractions Skill Score

ecPoint approach: each pixel sampled from parent IFS gridbox’s PDF “ecPoint, no correlation”
Spatial coherency

Motivation: similar point-wise properties to ecPoint approach. Why is a spatially-coherent image better?

Two things we have tried:

- Pooling (used with CRPS, also ROC)
- Fractions Skill Score

Also power spectrum…

ecPoint approach: each 10x10 block sampled from parent IFS gridbox’s PDF “ecPoint part correlation”
Pooling

Used in Deepmind nowcasting paper

Idea: aggregate predictions and truth image over larger windows before calculating metric

- Average-pooling (~cumulative rainfall over a region)
- Max-pooling (max rainfall nearby)
- 4x4 and 16x16 windows

‘Poor man’s Fractions Skill Score’?
Fractions Skill Score

Original motivation: “on what spatial scale is a prediction skillful?”

1. For a precipitation threshold, binarise image
2. Average prediction and truth over window size
3. Compute MSE
4. Compute FSS

\[
MSE_{(n)} = \frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} [O_{(n)i,j} - M_{(n)i,j}]^2.
\]

\[
FSS_{(n)} = \frac{MSE_{(n)} - MSE_{(n)\text{ref}}}{MSE_{(n)\text{perfect}} - MSE_{(n)\text{ref}}} = 1 - \frac{MSE_{(n)}}{MSE_{(n)\text{ref}}}.
\]

\[
MSE_{(n)\text{ref}} = \frac{1}{N_x N_y} \left[ \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} O_{(n)i,j}^2 + \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} M_{(n)i,j}^2 \right].
\]

Have generalised into ensemble version too.
Solid = “ensemble skill”
Dotted = “individual prediction skill”
Solid = “ensemble skill”
Dotted = “individual prediction skill”

Can cheat on this metric:
Noisy images show artificially good individual prediction skill
Solid = “ensemble skill”
Dotted = “individual prediction skill”
Precision-recall curve, precip threshold 2.0, pooling type avg_4

- GAN (area = 0.30)
- ecPoint no-corr (area = 0.24)
- ecPoint part-corr (area = 0.22)
- IFS upscaled (area = 0.30)
Precision-recall curve, precip threshold 5.0, pooling type max_4

- GAN (area = 0.15)
- ecPoint no-corr (area = 0.07)
- ecPoint part-corr (area = 0.06)
- IFS upscaled (area = 0.07)
Improved model
cGAN model
Downsampled problem
(c.f. Leinonen)
ROC curve - downscaled problem

![ROC curve for Downscaled problem, 100 ensemble members, batch size 64](image)
deterministic model

trainable parameters
829,313
VAE intro - autoencoder
VAE intro - autoencoder
VAE intro - autoencoder
VAE intro - autoencoder
VAE intro - (conditional) variational autoencoder
VAE intro - (conditional) variational autoencoder

Loss function also penalises latent variable distributions far from $N(0, 1)$
- Acts as regularisation
- Implemented as KL divergence
VAE loss function - mismatch term

Have explored traditional options including MSE, MAE.

Now using MSSSIM: Multi-Scale Structural Similarity Image Measure
- Based on “pixel-wise” dot product of images, applied at multiple scales
- Slightly better results than MSE, MAE, etc.
- Not a magic bullet, has given good results in other image generation problems; perhaps too ‘deterministic’ here.

Lack of suitable “mathematically expressible” loss function limits quality of VAE results
GAN

generative adversarial network

learns a loss that tries to classify an output as real or fake, while simultaneously training a generative model to minimise this loss function

\[ \mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))] \]

The discriminator's estimate of the probability that real data instance \( x \) is real.
GAN

Generative Adversarial Network

A minimax game:

\[ \mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))] \]

The expected value over all real data instances.
GAN
generative adversarial network

learns a loss that tries to classify an output as real or fake, while simultaneously training a generative model to minimise this loss function

a minimax game:

\[ \mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))] \]
GAN
generative adversarial network

learns a loss that tries to classify an output as real or fake, while simultaneously training a generative model to minimise this loss function

\[ L = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))] \]

discriminator's estimate of the probability that a generated, fake instance is real.
GAN

Generative Adversarial Network

learns a loss that tries to classify an output as real or fake, while simultaneously training a generative model to minimise this loss function

\[ \mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))] \]

A minimax game:

the expected value over all random inputs to the generator
GAN
generative adversarial network

learns a loss that tries to classify an output as real or fake, while simultaneously training a generative model to minimise this loss function

\[ \mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))] \]

refers to real data instances
GAN

generative adversarial network

learns a loss that tries to classify an output as real or fake, while simultaneously training a generative model to minimise this loss function.

\[
\mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))]
\]

refers to **fake (generated)** data instances.
c-GAN

GAN loss:
\[ \mathcal{L}_{GAN} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(x|z)))] \]

cGAN loss:
\[ G^* = \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L_1}(G) \]
The RAPSD used in this study is defined as follows. The power spectrum of a 2-D image \( f(x, y) \) of dimension \( M \times N \) is defined as [33]

\[
P(f) = |F(u, v)|^2
\]  

(18)

where

\[
F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)e^{-j2\pi(ux/M+vy/N)}.
\]  

(19)

Fig. 3 shows how spectral estimates \( P(f) \) can be partitioned into annuli of width \( \Delta \) for regular rectangular grids. Each annulus has a central radius \( f_r \), a radial frequency, and \( N_r(f_r) \) frequency samples. The sample mean of the frequency samples of \( P(f) \) in the annulus \( ||f| - f_r| = \Delta/2 \) about \( f_r \) is defined as

\[
\overline{P}_r(f_r) = \frac{1}{N_r(f_r)} \sum_{i=1}^{N_r(f_r)} P(f_r, i)
\]  

(20)

where \( f_r = [\sqrt{u^2 + v^2}] \) and \([\cdot]\) represents the nearest integer operator.
Wasserstein loss

based on the earth-mover distance

\[ W(\mathbb{P}_r, \mathbb{P}_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r} [f(x)] - \mathbb{E}_{x \sim \mathbb{P}_g} [f(x)] \]

\( \mathbb{P}_r \) is the real data distribution, \( \mathbb{P}_g \) is the generated model distribution.
Wasserstein GAN

The loss functions themselves are simple:

\textbf{Critic Loss:} \( D(x) - D(G(z)) \)

The discriminator tries to maximise this function. In other words, it tries to maximise the difference between its output on real instances and its output on fake instances.

\textbf{Generator Loss:} \( D(G(z)) \)

The generator tries to maximize this function. In other words, it tries to maximise the discriminator's output for its fake instances.
WGAN
Wasserstein GAN

based on the earth-mover distance

\[ W(\mathbb{P}_r, \mathbb{P}_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_g}[f(x)] \]

\( \mathbb{P}_r \) is the real data distribution, \( \mathbb{P}_g \) is the generated model distribution

\( f \) is a K-Lipschitz function

\[ |f(x_1) - f(x_2)| \leq K|x_1 - x_2| \]
WGAN
Wasserstein GAN

based on the earth-mover distance

$$\mathcal{L} = \mathbb{E}_x[D(x)] + \mathbb{E}_z[D(G(z))],$$

$$D \in \mathcal{D}$$

$\mathcal{D}$ is the set of 1-Lipschitz functions
WGANGP
Wasserstein GAN with gradient penalty

\[ \mathcal{L} = E_x[D(x)] + E_z[D(G(z))] + \lambda E_{\hat{x}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2] \]

\(\hat{x}\) is a random sample from the probability distribution \(\mathbb{P}_{\hat{x}}\) which is implicitly defined by sampling uniformly along straight lines between a pair of points sampled from the real data distribution \(\mathbb{P}_r\) and the generated distribution \(\mathbb{P}_g\).
VAE model
full problem
● Too blurry
● Too little
variation
Rank histograms plot (cGAN)

Rank $\mathcal{R}$:

$$\mathcal{R} = \frac{N_s}{N_p}$$

- $N_s$: no. samples where pixel value is smaller than truth
- $N_p$: total no. predictions

(a) Norm. occurrence

(b) CDF

Normalized rank range: 0.0 to 1.0
ROC curve

ROC curve for IFS problem, 100 ensemble members, 800 images

ROC curve for ecPoint approach, batch size 16
Improving orographic resolution

Previously we were training on ~4km orography. Have obtained 1km orography and will re-train on 1km this weekend.

Motivation - check if model produces e.g. rain shadowing (could not see much detail at 4km).
So my initial impressions are these:

1. The GAN forecast realisations generally look quite reasonable / plausible, and better than I maybe expected, though I wasn't completely sure what to expect (!). However:
2. Sometimes there are patches of rain in areas where chances are probably so low that they should not be there (although I could have a better idea if you provided dates and times for all the cases)
3. The handling of moving convective cells leaves a lot to be desired - stripes of large totals should be the norm, with dry gaps inbetween, but you don't really see that at all (e.g. in the extremes case) - instead the picture is blurry.

Credibility of your GAN output in a forecaster environment would be hit somewhat by items 2 and 3, which would I'm sure be better handled by a LAM ensemble. Indeed it might be interesting for you to examine some LAM-EPS output of the same variable to see yourselves how it compares. By design ecPoint should do quite well with aspects 2 and 3, from a probabilistic perspective, even if it is not directly delivering high res totals plots like yours.