GEMv3 Forecast Dataset

GEMv3 is the latest in Salient's GemAI family of extended-range weather models featuring:

- state-of-the-art performance across its full 126-day forecast horizon
- an extensive set of output variables
- increased temporal resolution, up to 6-hourly timesteps
- early 4x daily operational delivery
- direct access to cloud-optimized Zarr stores

Access

The GEMv3 datasets are stored in Cloudflare R2 object storage, compatible with the S3 API.

The reforecast dataset, covering 2000-2025, is publicly available and free to access. The operational forecasts, beginning in January 2026, require a subscription. Contact us for information about access and pricing.

Dependencies

To access the GEMv3 forecasts using Python, a few dependencies are required: `xarray>=2025.6.0`, `zarr>=3`, `pcodec>=1`, and either `s3fs` or `obstore`.

Quickstart

```python
import xarray as xr

ds = xr.open_zarr(
    "https://gemv3-reforecast.salient-open-data.com/forecast",
    chunks=None,
)
t2m = ds.sel(
    forecast_date="2025-01-01",
    lat=42.36,
    lon=-71.06,
    method="nearest",
)["2m_temperature"]
t2m = t2m.assign_coords(
    valid_time=t2m.forecast_date + t2m.lead,
).swap_dims(lead="valid_time")
t2m.plot(hue="sample", add_legend=False);
```

Data Description

Spatial and Temporal Resolution

GEM models are trained on ERA5 data at 0.25° resolution, so all forecasts mirror the grid-cell average distributions of the ERA5 training dataset. The forecast dataset is stored on a (-180, 180) longitude and (-90, 90) latitude grid.

All GEMv3 forecasts use a non-uniform rollout timestep schedule, where the forecast is run in 6-hour steps out to 14 days, and then 24-hour steps for all remaining leads. Therefore, the first 56 leads in each Zarr store are distributed at a 6-hour resolution, and the remaining leads use 24-hour steps.

Operational Schedule

In the operational period, which begins on 2026-01-01, the forecasts are run 4x daily with 200 ensemble members on the following schedule:

- Daily 00z: 126 days
- Daily 06/12/18z: 14 days

All forecasts are uploaded within 6 hours of the initialization time, i.e. the 00z forecast is available by 06z. The entire forecast rollout is uploaded simultaneously rather than lead by lead.

Reforecast Schedule

The reforecast set contains 50 ensemble members and is configured in two parts, both combined into a single Zarr store:

The primary reforecast, covering 2020-01-01 to 2025-12-31, has the following schedule:

- Daily 00z: 21 days
- Weekly Wednesday 00z: 46 days
- 1st of the month 00z: 126 days

The extended reforecast, covering 2000-01-01 to 2019-12-31, includes additional sparsely sampled forecasts for S2S calibration:

- Weekly Wednesday 00z: 46 days
- 1st of the month 00z: 126 days

Variables

GEMv3 uses over 90 input and output variables in total to generate a forecast. To reduce storage volume, a limited subset of all variables is stored, primarily focused on diagnostic surface fields most relevant to end users:

Instantaneous snapshots:

- `geopotential@500`
- `mean_sea_level_pressure`
- `2m_temperature`
- `100m_u_component_of_wind`
- `100m_v_component_of_wind`

Sub-timestep aggregations:

- `minimum_2m_temperature_since_previous_post_processing`
- `maximum_2m_temperature_since_previous_post_processing`
- `mean_2m_dewpoint_temperature`
- `mean_2m_temperature`
- `mean_total_precipitation_rate`
- `mean_snowfall_rate`
- `mean_surface_downward_short_wave_radiation_flux`
- `mean_total_cloud_cover`
- `mean_10m_wind_speed`
- `mean_100m_wind_speed`
- `10m_wind_gust_since_previous_post_processing`

GEMv3 generally shares the naming conventions of the GCP ERA5 ARCO dataset, which themselves are sourced from the ECMWF CDS API names. GEMv3 uses a few modifications to handle aggregations:

- The `mean_` prefix indicates a temporal average over the timestep, trained on 6 or 24-hourly averages of the native hourly ERA5 values. `mean_sea_level_pressure` is an exception, which is delivered as an instantaneous snapshot but has the `mean_` prefix due to the underlying naming conventions.

- The `_since_previous_post_processing` suffix indicates a sub-timestep extrema, e.g. minima or maxima.
  - `minimum_2m_temperature_since_previous_post_processing` and `maximum_2m_temperature_since_previous_post_processing` are derived as timestep minima or maxima of the hourly ERA5 `2m_temperature` values, as recommended by the ECMWF documentation.
- The `@` delimiter indicates a vertical level, e.g. `geopotential@500` is the geopotential at 500 hPa.
- Each variable has a `step_type`, one of `instant`, `mean`, `min`, or `max`, which also describes the type of variable.

Time coordinates

All forecast datasets are stored in a "data cube" format that supports packing the entire forecast schedule into a single Zarr store, using the following coordinates:

- `forecast_date`: the initialization time for the forecast
- `lead`: the time delta from the initialization time

The particular time of any forecast slice can be easily obtained using `valid_time = forecast_date + lead`.

Note: The `lead` coordinate in the GEMv3 datasets follows standard GRIB conventions, where instantaneous values are defined at the valid time, and accumulations are a right-labeled aggregation since the previous timestep. This is different from the GEMv2 conventions, where all variables were daily aggregations with left-labeled timesteps, e.g. `lead=0` corresponding to hours 1-24 following the initialization date.

The stores contain two additional metadata coordinates for convenience:

- `n_leads`: 1D integer describing the number of leads a particular forecast was rolled out to
- `processed`: 1D boolean indicating whether the forecast has been run yet or not

The operational Zarr store has been pre-allocated with future dates extending to the end of 2027.

Data Format

The Zarr stores use modern features of Zarr v3, namely a sharded layout with smaller inner chunks to increase performance of point-wise access. The `pcodec` compression algorithm is paired with a per-variable scale/offset filter, configured to allow lossy compression at a maximum loss of 0.1% of the global standard deviation of each variable. This produces an average compression ratio of around 8x relative to raw `float32` with negligible precision loss.

Access Tips

Chunking

There are a few important notes for getting optimal read performance out of these datasets, particularly when using Dask. The data is stored in the following structure:

- inner chunks: `forecast_date=1, lead=7, sample=-1, lat=36, lon=36`
- outer chunks (shards): `forecast_date=1, lead=7, sample=-1, lat=288, lon=288`

When using Xarray's engine-preferred chunking (i.e. `chunks={}`), the store will be opened with the smaller 36x36 chunks, which creates a very large Dask graph.

In general, best performance is achieved by opening the dataset with `chunks=None`, subsetting down to the needed dates and leads using **slicing** specifically, and then applying spatial chunking only if needed for downstream processing. This minimizes the size of the Dask graph, and makes optimal use of Xarray's lazy indexing classes. Performance is also acceptable if the store is simply opened directly with `chunks=dict(lat=288, lon=288)`, but this does create a larger initial Dask graph due to the number of dates.

Another common processing step could be to homogenize the initial 6-hourly resolution data to daily values to match the extended-range rollout. The `step_type` attribute on each variable can be used to resample with the appropriate method, like so:

```python
import xarray as xr

date = "2025-01-01"
chunks = dict(lead=7, lat=288, lon=288)

ds_raw = xr.open_zarr(
    "https://gemv3-reforecast.salient-open-data.com/forecast",
    chunks=None,
)
ds_raw = ds_raw.sel(forecast_date=slice(date, date)).chunk(chunks)

ds_daily = xr.Dataset()
for v in ds_raw.data_vars:
    step_type = ds_raw[v].attrs.get("step_type")
    if step_type == "instant":
        ds_daily[v] = ds_raw[v].where(
            ds_raw.lead.dt.total_seconds() / 3600 % 24 == 0,
            drop=True,
        )
    else:
        resample = ds_raw[v].resample(
            lead="24h",
            label="right",
            closed="left",
            offset="18h",
        )
        ds_daily[v] = getattr(resample, step_type)()
```

If you have any trouble working with this dataset, please contact us with your use case and we'll be happy to help.

Model Details

Backbone

GEMv3 uses a neighborhood attention transformer architecture, which is a spatially continuous extension of the Swin transformer used in GEMv2. Probabilistic sampling is achieved by injecting a low-dimensional noise vector into the transformer blocks using AdaLN-Zero conditioning.

The adaptive timestep capability of GEMv3 is also achieved with AdaLN, where the desired timestep value is encoded as another low-dimensional conditioning signal that modulates the network activations.

GEMv3 uses a richer set of initial conditions including increased vertical levels and additional surface variables relative to GEMv2. It also incorporates a number of small tricks and optimizations from the modern transformer literature that increase relative performance.

Training

GEMv3 is trained exclusively on ERA5 data using a training period of 1979-2019. All reforecasts beyond 2020 are entirely out of sample. The extended reforecast from 2000-2019 may have small training dataset contamination in the early leads, but this should be negligible at the S2S horizon.

The model is trained using a fair-CRPS loss, combined with a spectral power regularization term which helps push the modeled outputs towards power spectral realism and better representation of joint spatial covariances.

Base training is done on single steps, followed by a rollout fine-tuning stage up to 4 steps. To make the model capable of taking multiple timesteps, training is done on a mixture of timesteps between 6 and 24 hours across all stages.

Output Heads

GEM models use a multi-head architecture with two explicit output categories:

- **prognostic**: 3-dimensional atmospheric state fields that are required to autoregressively evolve the weather state and are fed back into the model at each timestep
- **diagnostic**: other fields of interest to end users but not required to predict the dynamics, treated as outputs only in the autoregressive loop

This architecture enables training a standardized prognostic model core, while allowing for easy fine-tuning of additional diagnostic outputs. Joint optimization of the diagnostics maximizes predictive performance relative to external post-processing methods that derive these from instantaneous snapshots.

The diagnostics included in GEMv3 are all sub-timestep aggregations, such as the temporal mean, minimum, or maximum of surface fields, but in principle the multi-head framework supports a wide range of potential diagnostic quantities.

S2S Stability

GEMv3 uses several novel techniques to improve stability and performance of the S2S rollouts, notably:

1. Anomaly-space modeling: a subset of variables are transformed into seasonal anomalies for processing within the backbone, using a pre-computed climatology. This gives the model a more stable baseline distribution and reduces drift from plausible climate states.

2. Adaptive timestep: the configurable inference timestep allows the model to more finely resolve processes in the high-predictability medium-range horizon, while reverting to a coarser timestep for the remainder of the extended-range forecast. Coarser timesteps reduce the accumulation of numerical errors in autoregressive rollouts, also improving stability and reducing drift.

Additional Details

For more details on the multi-head architecture and lightweight probabilistic backbone introduced in GEMv2, see the following writeup. A more detailed writeup of the GEMv3 model is forthcoming.