Python and AI/ML for Weather, Climate and Environmental Applications

Let us enjoy playing with Python and AI/ML!

Five-Day Schedule Overview

Time	Day 1	Day 2	Day 3	Day 4	Day 5
09:00–10:00	Opening by ECMWF DG, Start: Coding & Science in the Age of AI	Neural Network Architectures	Diffusion and Graph Networks	MLOps Foundations	Model Emulation, AIFS and AICON
10:00–11:00	**Lab:** Python Startup: Basics	**Lab:** Feed-forward and Graph NNs	**Lab:** Graph Learning with PyTorch	**Lab:** Containers and Reproducibility	**Lab:** Emulation Case Studies
11:00–12:00	Python, Jupyter and APIs	Large Language Models	Agents and Coding with LLMs	CI/CD for Machine Learning	AI-based Data Assimilation
12:00–12:45	**Lab:** Work environments, Python everywhere	**Lab:** Simple Transformer and LLM Use	**Lab:** Agent Frameworks	**Lab:** CI/CD Pipelines	**Lab:** Graph-based Assimilation
12:45–13:30		**Lunch Break**			
13:30–14:30	Visualising Fields and Observations	Retrieval-Augmented Generation (RAG)	DAWID System and Feature Detection	Anemoi: AI-based Weather Modelling	AI and Physics
14:30–15:30	**Lab:** GRIB, NetCDF and Obs Visualisation	**Lab:** RAG Pipeline	**Lab:** DAWID Exploration	**Lab:** Anemoi Training Pipeline	**Lab:** Physics-informed Neural Networks
15:30–16:15	Introduction to AI and Machine Learning	Multimodal Large Language Models	MLflow: Managing Experiments	**The AI Transformation**	Learning from Observations Only
16:15–17:00	**Lab:** Torch Tensors and First Neural Net	**Lab:** Radar, SAT and Multimodal Data	**Lab:** MLflow Hands-on	**Lab: How work style could change**	**Lab:** ORIGEN and Open Discussion
17:00–20:00				Joint Dinner	

What is Multimodal Artificial Intelligence?

Single-modality AI

- Classical ML processes one modality at a time
- Text-only LLMs operate on sequences of tokens
- Images, audio, or fields are handled separately

Limitation

- No explicit link between different data modalities
- Limited representation of real-world information

Multimodal AI

- Jointly processes multiple modalities
- Learns relationships *across* modalities
- Typical inputs:
  - text
  - images
  - audio
  - spatial or physical fields

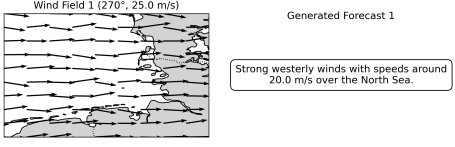
Key idea: Meaning emerges from the interaction of modalities.

Why Multimodal AI in Weather and Climate?

Nature of meteorological data

- Numerical model fields
- Maps and visualisations
- Radar and satellite images
- Textual forecasts and warnings

Weather information is inherently multimodal.



Human forecasters

- Integrate multiple information sources
- Combine perception and physical reasoning
- Translate observations into language

Key question: How can AI systems learn this integration?

Example A: Symbolic Multimodality

Basic idea

- Start from a physical wind field
- Extract a small set of **symbolic descriptors**
- Use a language model to generate text

The model does *not* see images or full fields.

Representation pipeline

- Numerical wind field
- $\rightarrow$ symbolic spatial summary
- $\rightarrow$ text prompt
- $\rightarrow$ language model

Multimodality is achieved through **representation design**.

From Wind Fields to Symbols

Physical input

- Two-dimensional wind field
- Zonal and meridional components (u, v)
- Defined on a spatial grid

This information is high-dimensional and continuous.

Symbolic reduction

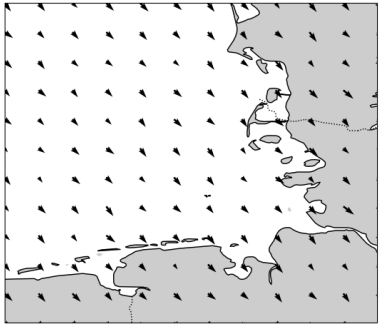
- Mean wind speed
- Dominant wind direction
- Categorical intensity classes

Use a small number of *interpretable* features are retained.

```
=== SYMBOLIC INPUT ===
Wind field summary:
Mean speed: 9.1 m/s
Speed variability: low
Dominant direction: northwesterly (316°)
Directional spread: ±5.8°
Stronger winds in the northern sector.
Overall intensity: moderate

=== TARGET TEXT ===
Northwesterly winds over the German Bight with moderate intensity and speeds around 10 m/s.
```

Example Wind Field — German Bight



Text Generation from Symbolic Descriptors

LLM input

- Short symbolic prompt
- Encodes physical properties
- Passed directly to the language model

```
Mean wind speed: 15.2 m/s
Dominant direction: northwesterly
Region: North Sea
```

LLM output

- Natural-language forecast text
- Generated solely from symbolic input

```
Strong northwesterly winds with
speeds around 15 m/s over the
North Sea.
```

Example A: Core Implementation Idea

Step 1: Symbolic preprocessing

- Reduce the wind field to a few key descriptors
- Physics-informed, human-designed
- Low-dimensional and interpretable

This step encodes *what matters* for the language model.

Step 2: Prompt-based text generation

Symbolic prompt and text generation

```
summary = f"Mean speed: {vmean:.1f} m/s, "
summary += f"Direction: {direction}"

inputs = tokenizer(
    summary,
    return_tensors="pt"
)

outputs = model.generate(**inputs)
```

In this example the LLM sees only text, not physics.

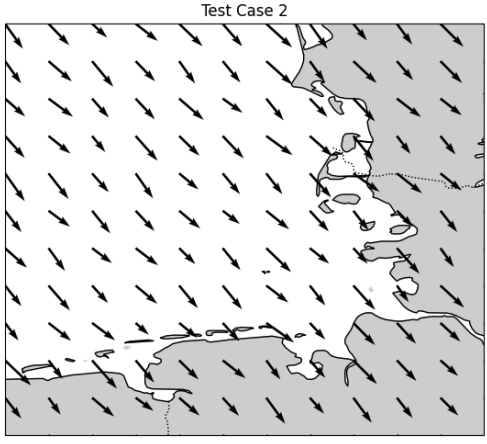
Example A: Result on a Concrete Wind Field

Physical input

- Synthetic wind field over the North Sea
- Used to compute symbolic descriptors
- Image shown here for illustration

Generated text (Example A)

- Language model sees only **symbolic input**
- No image is passed to the model



Northwesterly winds over the German Bight with strong intensity and speeds around 20 m/s.

Example B: Visual Multimodality

Key difference to Example A

- No symbolic preprocessing
- No explicit speed or direction given
- The model receives an **image only**
- Converted to RGB pixel values

All physical structure must be inferred visually.

```
# Draw the canvas
fig.canvas.draw()

buf = np.asarray(fig.canvas.buffer_rgba())
image = buf[..., :3]    # drop alpha channel
# Normalize to [0, 1]
```

Multimodal processing pipeline

- Wind field rendered as image
- Vision encoder extracts features
- Features are mapped to language space
- Language model generates text

This is *true* multimodal learning.

```
# ----------------------------------------
# Render the wind field as an image
# This image is the ONLY input seen by
# the multimodal model.
# ----------------------------------------
img = render_wind_image(LON, LAT, U, V)
```

From Image Tensor to Vision Embedding

Vision encoder (ViT)

- Input: image tensor $(1, 3, 224, 224)$
- Image is split into patches
- Self-attention models spatial relations

The vision model extracts a high-level representation of the entire image.

Forward pass and feature extraction

Vision Transformer

```
with torch.no_grad():
    vision_out = vit(
        pixel_values=pixel_values )

# Global image representation
vision_feat = (
    vision_out.last_hidden_state[:,0,:]
)  # CLS token
```

The CLS token summarizes the full image content.

$\texttt{img} \in \mathbb{R}^{224\times224\times3} \rightarrow \texttt{vision\_feat} \in \mathbb{R}^{1\times768}$

CLS token $\in \mathbb{R}^{768}$ — a learned global image representation

From Vision Embedding to Language Model Input

The modality gap

- Vision encoder outputs visual embeddings
- Language models expect text embeddings
- Both live in different representation spaces

A learned projection is required to connect vision and language.

Vision–language bridge

Project vision features into language space

```
# vision features -> embedding
encoder_embed = bridge( vision_feat
  ).unsqueeze(1)

# Pass embeddings to the T5 encoder
outputs = t5(
    inputs_embeds=encoder_embed,
    labels=labels )
```

The language model receives embeddings, not tokens.

$$\text{CLS} \in \mathbb{R}^{768} \xrightarrow{\text{Linear projection}} \text{encoder\_embed} \in \mathbb{R}^{1\times 1\times 512}$$

Example B: Results on Unseen Wind Fields

Model input

- Wind field rendered as an image
- No symbolic or numerical input
- Image only, unseen during training

The model infers direction and strength **from visual structure alone.**

Input wind field
Base direction: 315°, mean speed: 25.2 m/s

Northwesterly winds over the German Bight with strong intensity and speeds around 20 m/s.

Vision Transformer: Patch Embedding

Image representation

An input image is represented as

$$\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$$

- H, W: image height and width (pixels)
- 3: RGB color channels

The image is split into non-overlapping patches of size $P \times P$.

Number of patches:

$$N = \frac{H \cdot W}{P^2}$$

Patch embedding

Each image patch is flattened:

$$\mathbf{x}_p \in \mathbb{R}^{P^2 \cdot 3}$$

and projected into an embedding space:

$$\mathbf{x}_p \xrightarrow{E} \mathbf{z}_p \in \mathbb{R}^D$$

The full input sequence is:

$$\mathbf{Z}_0 = [\mathbf{z}_{\text{CLS}}, \mathbf{z}_1, \ldots, \mathbf{z}_N] \in \mathbb{R}^{(N+1) \times D}$$

D is the transformer embedding dimension.

Vision Transformer: Self-Attention

Transformer processing

The patch embedding sequence is processed by L stacked transformer layers:

$$\mathbf{Z}_{\ell+1} = \text{Transformer}(\mathbf{Z}_\ell), \quad \ell = 0, \ldots, L-1$$

Each layer consists of:

- Multi-head self-attention
- Feed-forward networks
- Residual connections $x + \mathcal{F}(x)$
- Layer normalization

CLS token as global image embedding

Self-attention enables each patch to attend to all other patches.

After the final layer, the CLS token is:

$$\mathbf{z}_{\text{CLS}}^{(L)} \in \mathbb{R}^D$$

This vector represents the entire image and can be used for:

- Image classification
- Regression tasks
- Multimodal conditioning of language models

Radar Reflectivity as Multimodal Input

Weather radar data

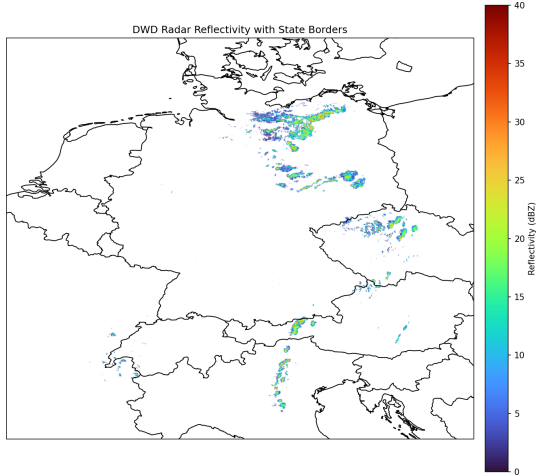
- Active remote sensing (microwave)
- Measures backscattered power
- Proxy for precipitation intensity

Radar reflectivity is expressed in

$$\mathrm{dBZ} = 10 \log_{10}(Z)$$

with Z the radar reflectivity factor.

Radar images are routinely interpreted visually by forecasters.



DWD radar composite with state borders. Colors indicate reflectivity intensity (dBZ).

From Radar Product to AI-Readable Image

Operational radar data

- Provided as HDF5 files
- Contains data and metadata
- Scaling and masking required

Physical decoding:

- apply gain and offset
- mask undetectable values
- remove non-physical signals

Result: a georeferenced reflectivity field.

Decode radar reflectivity

```
raw = f["dataset1/data1/data"][:]

gain   = what.attrs["gain"]
offset = what.attrs["offset"]

data = raw.astype(np.float32) *
     gain + offset
data[(raw == 0) | (raw == 65535) |
      (data < 0)] = np.nan
```

This step preserves physical meaning before visualization.

Radar as Image Modality

Why visualization matters

- Radar fields are spatial data
- Interpretation relies on patterns
- Humans read radar as images

For multimodal AI, the radar field is therefore converted into a **visual representation**:

- color encodes intensity (dBZ)
- white indicates missing / no signal
- borders provide spatial context

The image becomes the AI input modality.

Render radar reflectivity map

```
masked = np.ma.masked_invalid(data)
plt.figure(figsize=(12, 10))
ax = plt.axes(projection=ccrs.
    PlateCarree())
ax.set_extent([3, 17, 44, 56])
im = ax.imshow(
    masked, origin="lower",
    cmap=cmap, vmin=0, vmax=40,
    transform=ccrs.PlateCarree() )

ax.add_feature(cfeature.BORDERS)
ax.coastlines(resolution="10m")
plt.colorbar(im, ax=ax, label="
    Reflectivity (dBZ)")
plt.savefig("radar_map_germany.png",
     dpi=150)
```

Multimodal AI Interpretation of Radar Images

Multimodal model input

The AI model receives:

- a radar image (visual modality)
- a textual instruction (language modality)

The model jointly reasons over:

- spatial precipitation patterns
- reflectivity intensity gradients
- mesoscale structure

The **output** is a natural-language meteorological interpretation.

Multimodal AI request

```
query = "Please interpret this radar reflectivity image ..."
response = client.chat.completions.create(
  model="gpt-4-turbo",
  messages=[{ "role": "user",
    "content": [
    {"type": "text", "text": query },
    {"type": "image_url",
    "image_url": {"url": f"{encoded}"
        }} ] }],
  max_tokens=800 )
```

Example: AI-Generated Radar Interpretation

Input to the model

- Radar reflectivity image (dBZ)
- Geographical context (Germany)
- Task-oriented meteorological prompt

The AI has no direct access to:

- numerical radar grids
- timestamps or motion
- physical radar equations

This radar reflectivity image shows precipitation distribution and intensity across Germany and parts of its neighboring countries, indicated by different colors reflecting different reflectivity values measured in dBZ (decibels of Z). Here's a detailed analysis:

[...] Stratiform Rainfall: The majority of the precipitation patterns, especially the widespread areas with uniform green shades in the northern and southwestern parts, suggest stratiform precipitation structures. These are typically smooth and consistent patterns associated with frontal systems or broad-scale upward motions that produce persistent and uniform rainfall.

Capabilities and Limitations of Multimodal Radar Interpretation

What the model can do well

- Identify precipitation regions
- Distinguish weak vs. strong reflectivity
- Recognize spatial patterns
- Describe convective vs. stratiform structures

This closely mirrors human visual interpretation.

Key limitations

- No access to raw radar measurements
- No temporal information (single snapshot)
- No physical thresholds or calibration
- Interpretation depends on visualization choices

The model provides *descriptive insight*, not quantitative meteorological analysis.

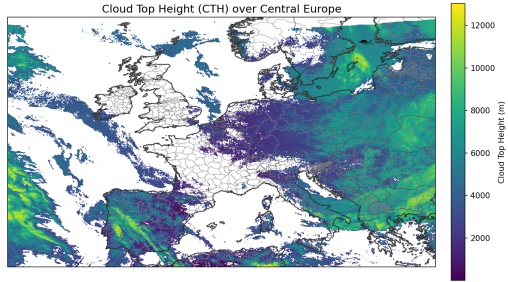
Cloud Top Height as Multimodal Input

What is Cloud Top Height (CTH)?

- Satellite-derived cloud product
- Estimates height of the upper cloud boundary
- Derived from thermal infrared observations

CTH is typically expressed in meters above sea level and provides information on:

- vertical cloud structure
- convective depth
- storm intensity



Satellite-derived Cloud Top Height (CTH) over Central Europe. Higher values correspond to deeper cloud systems.

High CTH values often indicate deep convection.

Accessing Cloud Top Height (CTH) Data

Operational satellite product

- Provided by DWD Open Data
- Derived from geostationary satellites
- Updated at regular intervals

CTH data is distributed as:

- compressed NetCDF files (`.nc.bz2`)
- regular latitude–longitude grids
- physical units: meters

The latest available file is selected automatically.

Download latest CTH product

```
base_url = (
  "https://opendata.dwd.de/weather/"
  "satellite/clouds/CTH/")
response = requests.get(base_url)
soup = BeautifulSoup(response.text, "
    html.parser")
cth_files = sorted([ link.get("href")
  for link in soup.find_all("a")
  if link.get("href", "").endswith(".
    nc.bz2") ])
my_f = cth_files[-1]
url = urljoin(base_url,my_f)
with open("cth.nc.bz2", "wb") as f:
    f.write(requests.get(url).content)
```

Multimodal AI Interpretation of Cloud Top Height

Model input

The AI model receives:

- a Cloud Top Height image
- a task-oriented text prompt

The task focuses on:

- identifying regions of high cloud tops
- distinguishing deep convection from stratiform clouds
- describing large-scale cloud structures

Submit CTH image to multimodal LLM

```
query = "This is a satellite-derived Cloud Top Height image [...]"
response = client.chat.completions.create(
  model="gpt-4-turbo",
  messages=[{"role": "user",
   "content": [
    {"type": "text", "text": query},
    {"type": "image_url", "image_url":{
    "url": f"{encoded}" }} ] }],
    max_tokens=800,)
```

Multimodal AI in Weather: Comparison and Takeaways

Four multimodal examples

- **Synthetic wind fields (A)**
  - physical feature extraction
  - numerical fields → text
- **Synthetic wind fields (B)**
  - image-based representation
  - learned visual perception
- **Radar reflectivity**
  - precipitation structure
  - surface-reaching hydrometeors
- **Cloud Top Height (CTH)**
  - vertical cloud structure
  - deep convection indicators

Key insights

- Multimodal AI reasons over images and text
- Physical meaning enters via representation
- Visualization choices matter
- Models provide *interpretation*, not physics
- How Vision Transformers (ViT) work

Outlook

- combining multiple modalities
- adding temporal context
- integrating AI with NWP systems

Multimodal Models Available at OpenAI

GPT-4 family

- First widely deployed multimodal models
- Text and image input
- Text output
- Strong vision-language reasoning

Typical use cases:

- image interpretation
- document understanding
- visual question answering

Used extensively in early multimodal weather and geoscience applications.

GPT-5 family

- Current OpenAI foundation model generation
- Native multimodal reasoning
- Text and image input
- Text output with improved reasoning depth

Key characteristics:

- stronger cross-modal alignment
- improved robustness and consistency
- better handling of complex visual scenes

Multimodal AI on IONOS

IONOS AI Model Hub

Provides a curated set of foundation models via an OpenAI-compatible API endpoint.

Available model categories

- Text-only language models
- Embedding models
- Image generation models
- **One vision–language model**

Most models remain text-only.

Multimodal image understanding

Mistral Small 24B

- Input: text + image
- Output: text
- Tool calling supported
- OpenAI-compatible chat schema

Successfully used to interpret:

- radar reflectivity images
- cloud top height (CTH) maps

Selective but production-grade multimodality.