D1.2 Preliminary report on dataset compression and data optimisation methods and their effects on training

Due date of deliverable	31 January 2026
Submission date	30 January 2026
File Name	D1.2 Dataset Optimisation Report
Work Package /Task	WP1/T1.3
Organisation Responsible of Deliverable	UKMO
Author name(s)	Sam Denton (UKMO), Tom Dunstan (UKMO), Italo Epicoco (CMCC), Sebastian Hoffmann (MPG), Sam Madge (UKMO), Reza Shatery (CMCC), Jiayong Li (ETHZ), Langwen Huang (ETHZ), Alexandru Calotoiu (ETHZ), Torsten Hoefler (ETHZ)
Revision number	1.3
Status	Final
Dissemination Level	Public

Funded by the European Union

The WeatherGenerator project (grant agreement No 101187947) is funded by the European Union.
Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the Commission. Neither the European Union nor the granting authority can be held responsible for them.

1 Executive Summary

This preliminary report explores the potential of dataset compression and data optimisation methods to improve the efficiency and scalability of deep learning for weather and climate modelling within the WeatherGenerator project. The main objectives are to assess whether advanced data reduction techniques such as auto-encoder-based compression, lossy JPEG-2000 encoding, and batch curation strategies can reduce computational and storage demands while maintaining or enhancing model performance.

The findings suggest several promising opportunities, but also highlight important uncertainties:

Auto-encoder training: The development of a modular, reproducible pipeline for systematic experimentation is a key outcome, enabling rapid testing of new architectures and data strategies. This infrastructure lays the groundwork for future research.

Experiments on a small test dataset indicate that substantial reductions in data volume (up to 21:1 or higher) are possible with minimal loss of information, particularly when using convolutional auto-encoders on structured datasets. However, the effectiveness of these methods appears sensitive to the structure and preprocessing of the data. For non-gridded datasets, such as ERA5, alternative architectures may be more appropriate. Independently from the training framework described above, results are presented on the use of a VQGAN autoencoder model trained on remote sensing satellite data from Sentinel-2.

Lossy Compression for ML Training: Tests using JPEG-2000 compression on ocean forecast data show that deep learning models can retain strong predictive skill even at high compression ratios and reduced spatial resolutions. The impact of compression on model accuracy is generally modest, but the results may depend on the specific variables, tasks, and model architectures involved.

Error bounded compression for forecast initial conditions: This section evaluates the impact of using error-bounded lossy compression to initialise a trained machine-learning-based weather forecasting model. By using compressed fields as initial conditions for the Aurora forecasting model, predictions remain virtually unchanged - showing less than a 1% increase in RMSE even at compression ratios exceeding 50×.

Batch Curation: Batch selection techniques such as SALN can accelerate training and, in some cases, improve validation loss. However, their benefits are inconsistent, particularly for complex, multimodal, or highly dynamic prediction problems. The mechanisms underlying these effects are not yet understood, and so the applicability of such methods to large-scale, operational weather models is not yet clear.

While the results are encouraging, demonstrating that significant data reduction is possible without major loss of model skill on some tasks, many findings are preliminary and based on limited datasets or simplified tasks. The transferability of these approaches to full-scale, multimodal, and operational weather prediction will be the focus for the next phase of this task.

Table of Contents

2 Introduction

2.1 Background

Climate change brings huge risks for the wellbeing and prosperity of society in Europe and world-wide. Earth system models that provide a numerical representation of the various components of the Earth system with atmosphere, ocean, land surface, land ice, sea ice, lakes and atmospheric chemistry are currently the best available tools to understand and prepare for climate change and the associated weather extremes.

The WeatherGenerator project will build the world’s best generative Foundation Model of the Earth system – that will serve as a new Digital Twin for Destination Earth (DestinE). The WeatherGenerator will be based on representation learning and create a general and versatile tool that models the dynamics of the Earth system based on a large variety of Earth system data. At the same time, it will integrate observations and simulations at a previously unseen level and scale.

This project brings together Europe's leading scientific groups and research institutes, Small and Medium-sized Enterprises (SMEs), and industry partners in the area of Earth system modelling, high-performance computing (HPC) and machine learning to build the WeatherGenerator as a new Digital Twin of DestinE. Once trained, the WeatherGenerator will be applied for selected high-impact applications in the energy, food, water and health sectors.

The WeatherGenerator will lead to key innovations in weather and climate science and machine learning to enable Europe to establish and defend leadership with respect to machine-learning based Earth system modelling. The WeatherGenerator will define a new state-of-the-art in both machine learning and weather and climate sciences. Through its vastly improved efficiency and flexibility compared to current Earth system models, the WeatherGenerator will create new opportunities for fast DestinE services that allow testing of many different management options and can include new levels of interactivity for a large user base including, for example, city planners, regional and national authorities, architects, and engineering companies.

2.2 Scope of this deliverable

2.2.1 Objectives of this deliverables

This deliverable aims to describe initial investigations into the use of data compression and batch engineering/curation – collectively referred to here as ‘dataset optimisation’ – as a method for improving the efficiency of training the WeatherGenerator model. We describe the motivation and rationale for this task, outline the methods chosen and describe initial results, and provide recommendations for further work

2.2.2 Work performed in this deliverable

In this deliverable the work as planned in the Description of Action (DoA, WP1 T1.3) was performed. Preliminary investigations into the use of dataset compression and data optimisation methods and their effects on training deep learning models in a weather and climate context.

2.2.3 Deviations and counter measures

No deviations have been encountered.

2.2.4 WeatherGenerator Project Partners:

ECMWF	EUROPEAN CENTRE FOR MEDIUM-RANGE WEATHER FORECASTS
FZJ	FORSCHUNGSZENTRUM JUELICH
MetNor	NORWEGIAN METEOROLOGICAL INSTITUTE
MPG	MAX-PLANCK-GESELLSCHAFT ZUR FÖRDERUNG DER WISSENSCHAFTEN E.V.
KNMI	ROYAL NETHERLANDS METEOROLOGICAL INSTITUTE
MetFrance	MÉTÉO-FRANCE
SMHI	SWEDISH METEOROLOGICAL AND HYDROLOGICAL INSTITUTE
UKMO	UK METOFFICE
CMCC	CENTRO EURO-MEDITERRANEO SUI CAMBIAMENTI CLIMATICI
eScience	NETHERLANDS ESCIENCE CENTER
Buluttan	BULUTTAN
KAJO	KAJO SERVICES
LT	LATEST THINKING
Statkraft	STATKRAFT
ETHZ	EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE
MetSwiss	METEOSWISS

3 Motivation

In large-scale deep learning and representation learning, data are typically ingested directly into models, sometimes via lightweight encoders, into a shared latent space designed to capture relationships among inputs. For effective learning, this latent space must be sufficiently compact to prevent memorization of individual samples, thereby forcing the model to learn generalizable rules. This “information bottleneck” is a central concept in machine learning (Tishby, 1999).

Standard practice is therefore to present raw or minimally processed data to models to ensure that the compression is optimized to the task at hand. However, the WeatherGenerator project faces unique challenges due to the sheer volume of training data. Processing all data in raw form, if it is possible at all, can limit the inclusion of other sources and so reduce the diversity of data. Although ad hoc data reduction techniques (e.g., sub-selecting pressure levels from ERA5) are common, they are likely suboptimal. Recent work applying compression methods to weather and climate data (Han et al. 2024, Huang et al. 2025) have demonstrated that substantial reductions in data volume are possible without significant loss of information.

Beyond simple volume reduction, data pre-processing can enhance model training efficiency. Knowledge distillation methods show that trained models can serve as effective data pre-processors, denoising and conditioning data for subsequent training. Whether similar benefits can be achieved with lighter-weight or custom-built compressors remains an open question.

Data compression focuses on optimizing individual samples, but this principle can be extended to entire datasets through batch curation. Rather than randomly selecting samples for each batch, engineered batches may accelerate training or improve final model performance. In weather and climate applications, batch curation is an emerging area of research.

Section 4 introduces a prototype training and evaluation code for developing bespoke auto-encoder compressors, tailored to the needs of WeatherGenerator training. Separately, a VQGAN model is assessed for its ability to compress land use satellite imagery. In Section 5, an off-the-shelf image compressor, JPEG-2000While, is used to assess the suitability of a compressed dataset training a forecast model. Section 6 explores the use of compressed data for forecast initialisation of a trained model. Finally, in Section 7, initial work on the application of batch curation methods is tested on a selection of atmospheric forecasting tasks.

References

Tishby et al. (1999) Proceedings of the 37th Allerton Conference on Communication, Control and Computing, Urbana, Illinois 1999

Han et al. (2024) CRA5: Extreme Compression of ERA5 for Portable Global Climate and Weather Research via an Efficient Variational Transformer, https://arxiv.org/abs/2405.03376

Huang et al. (2025) Error bounded compression for weather and climate applications, https://arxiv.org/abs/2510.22265

4 Auto-encoder training and evaluation tool

4.1 Introduction

This study describes the creation of a training and evaluation code for building auto-encoders (AEs) to compress atmospheric data in order to create a smaller dataset for training the foundation model. Previous studies have demonstrated promising benefits in training speed, input/output operations, storage efficiency, and model performance when datasets are first compressed using various techniques. Examples include Zhao et al.'s (2025) latent ERA5 dataset and Nguyen et al.'s (2025) latent-space diffusion forecasting method. Building on these findings, this work delivers a framework to assess which AE architectures suit our specific needs.

The project is still in its early stages. For now, the report's goal is to demonstrate that the tools work effectively and deliver fair results across many architectures and configurations, rather than recommending a final approach. It details current implementations and suggests future architectures for evaluation. Tests have only been performed on a small portion of the dataset - between 100 and 1,000 time slices - representing about 150 GB, which is just 1.5% of the full dataset of 65,744 time slices. Therefore, it is unlikely that any models will perform exceptionally well at this point; instead, the results confirm that the tools function properly, can deliver detailed analysis and comparisons, and that the problem has been accurately defined.

4.2 Methodology

A Hydra-based sandbox was built to allow fair, concurrent testing of multiple architectures on the same problem. This setup enables large configuration sweeps, with different models trained in parallel under identical conditions for unbiased comparison.

Root Mean Square Error (RMSE) is currently used as the training and evaluation metric due to its simplicity, though it is acknowledged that alternative loss functions may produce better results for atmospheric data. As models improve and achieve satisfactory RMSE, more advanced metrics will be considered. For now, RMSE and visual reconstruction comparisons suffice.

Each model's training details -including memory usage, epoch time, number of epochs, dataset size, batch size, and compression ratio -are recorded, since some architectures require more resources than others.

All trained models compress latitude, longitude, and level together, chosen arbitrarily to maximize pattern discovery across dimensions. Alternative compression strategies should be explored; compressing by column or level restricts the model’s access to potentially helpful data from adjacent regions.

Dataset Preparation

The initial tests used a toy dataset which differed significantly from the final ERA5 dataset, particularly in dimensional structure. The toy dataset's structured grid (time, variables, levels, latitudes, longitudes) lent itself to the use of convolutional models, as they perform well with spatially correlated data. However, the full ERA5 dataset only has dimensions of time, variables, and points, complicating the application of convolutions. As a result, existing models required adaptation, and testing revealed that stacking all points into a single dimension disrupted spatial correlations -especially between latitudes -making it challenging for models to learn boundary discontinuities effectively. Although some success was achieved, this approach limited the models’ ability to capture other patterns. Therefore, it is recommended to shift away from convolutional architectures and consider alternatives more suited to the non-regular ERA5 grid.

4.3 Architectures Tested

The architectures consist of two main components: the 'backbone' (input processing and early network propagation) and the latent structure (deeper information propagation and latent space representation). Although models could be split into 'data ingestion,' 'backbone/down-sampling,' and 'latent structure,' in practice, all tested architectures use convolution layers for these stages.

While 3D convolution performed well on toy data, adapting it to the ERA5 dataset would require converting points to a regular grid and filling gaps, which produced artifacts and unusable reconstructions. Therefore, all results here use a 2D convolutional backbone.

Below is a summary of available configuration options for convolutional backbones and the investigated latent structure variants.

4.3.1.1 Configurable Features

- Activation Functions: ReLU, leaky_ReLU, and GELU were tested. ReLU is most efficient for small models, while leaky_ReLU and GELU perform better in larger networks.
- Residual Blocks: Adding residual connections consistently improves reconstruction loss, especially in deeper models, at the cost of increased memory and compute.
- Attention Mechanisms: Attention layers can enhance detail and pattern recognition but increase resource demands and training complexity.
- Strides, Channels, Kernels: Deeper networks with more channels and appropriate kernel sizes yield better compression and reconstruction, though trade-offs exist between performance and resource use.
- Regularization: Batch normalization and dropout (typically 0.1) are beneficial for training stability and generalization
- Learning rate: The learning rate determines the step size for updating model weights during optimisation, with most tests using 1e-3 and a reduction schedule, though adjustments may be needed for larger models or datasets.

4.3.2 Latent Space Structure

Summaries of architectures implemented and their effects on latent space:

Multi-Layer Perceptron

An MLP compresses features through dense layers, producing a latent vector that captures global feature interactions rather than local patterns.

Convolutional

A convolutional latent space forms a small feature map using convolutional blocks, preserving spatial information–the latent space retains the input's basic structure.

Transformer

Transformers use a sequence of tokens processed by self-attention, allowing non-local dependencies but requiring more memory.

Variational Auto Encoder

A VAE outputs parameters for a Gaussian distribution; sampling produces latent vectors. This approach creates smooth, continuous latent spaces suitable for generative models.

Disentangled Variational Auto Encoder

Disentangled VAEs (e.g., β-TCVAE) penalize correlation between latent variables, encouraging independent, interpretable features in the compressed space.

4.4 Results

Selected results are shown here to demonstrate the capabilities of the code for performing hyper-parameter sweeps. Only 100 data samples were used for training models within each sweep and some architectures and features are more suited to deeper networks, or require more data, or a larger batch size, and so the results are not suitable for quantitative comparisons.

4.4.1 Hyper-parameter Sweeps by Feature

Boxplot comparisons showing average RMSE, VRAM use and epoch time for each parameter covered by the feature.

4.4.1.1 Activation Function

This activation sweep serves as a basic check to confirm expected behaviour. Relu is generally the most efficient in compute and memory, delivering around 30% lower RMSE than leaky_relu and 50% lower than gelu for smaller models. However, with medium models (4-5 strides), leaky_relu slightly outperforms, and with the largest models (6 strides), Gelu and leaky_relu perform equally. Gelu also uses about twice as much VRAM and takes roughly 10% longer to train.

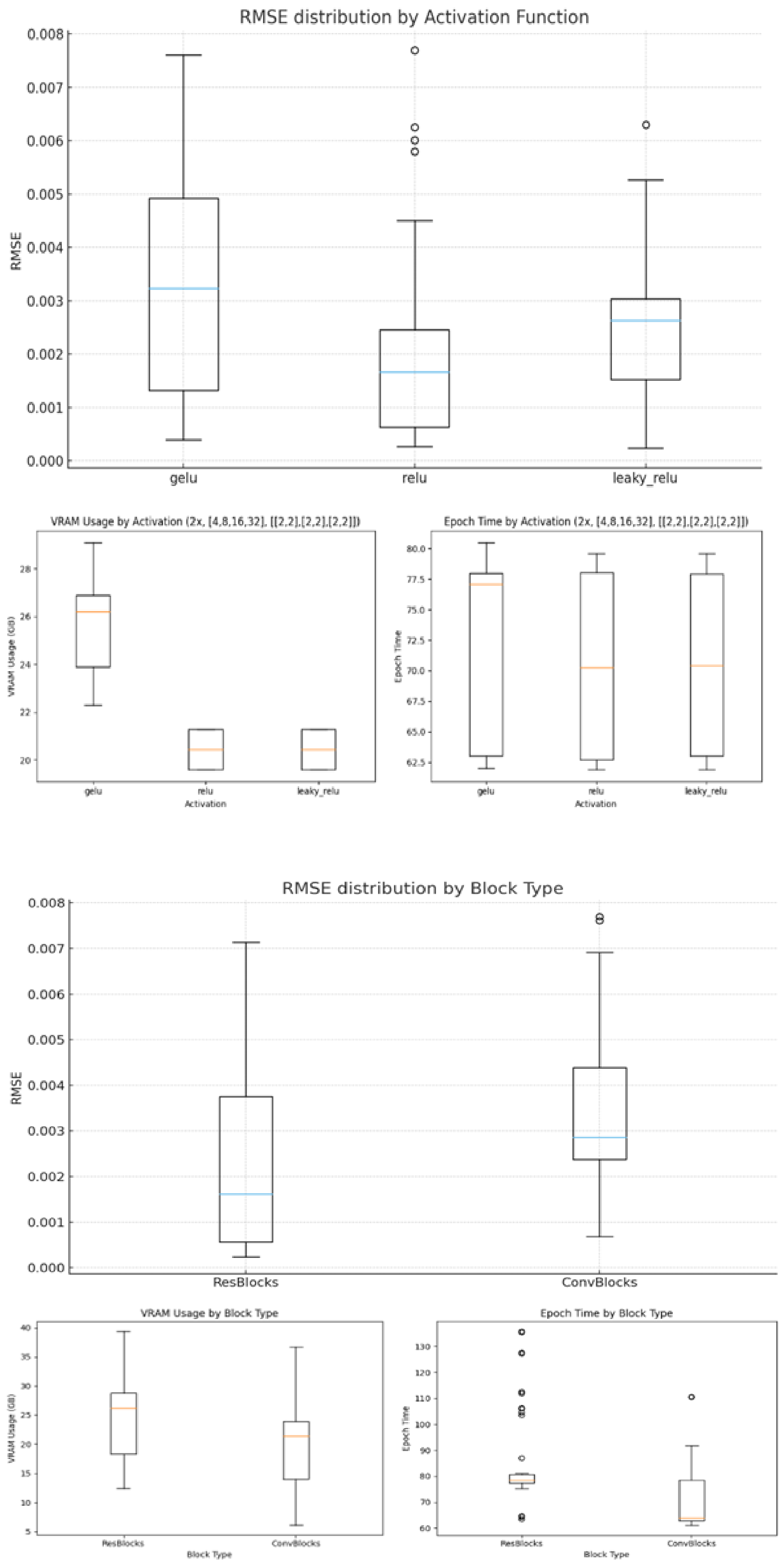


Figure 1 Box and whisker plots for block type sweep

4.4.1.2 Residual Blocks

In every scenario tested, adding residual connections reduced reconstruction loss, with larger models showing greater improvement. However, this came at a cost of about 20% more VRAM usage and 10% longer computation time.

4.4.1.3 Attention

Results for attention layers are less clear. Models with attention layers needed much more memory and time to train, which limited fair comparisons and convergence. While some attention models struggled to find useful patterns, those that did appeared to produce more realistic details, even with lower RMSE. These models seemed better at maintaining variation between regions, though this has not been quantified. However, they tended to create finer detail, sometimes misplaced, while models without attention blurred outputs towards average values.

No definitive conclusions can be drawn about attention versus non-attention, but among tested configurations, omitting attention in the first layer was most effective. Increasing attended layers raised memory and training time, but less so after the initial increase. Self-attention introduced grid artifacts aligned with window borders, consistent with other studies (Li et al. 2024), though distinction was difficult when checkerboard or ringing artifacts were also present.

The use of dropout was tested within the attention layers. As expected, dropout improves loss consistently with no impact to memory or compute. The standard value of 0.1 worked well, however a wider range of values could be explored, as different architectures and configurations work better with different dropout values.

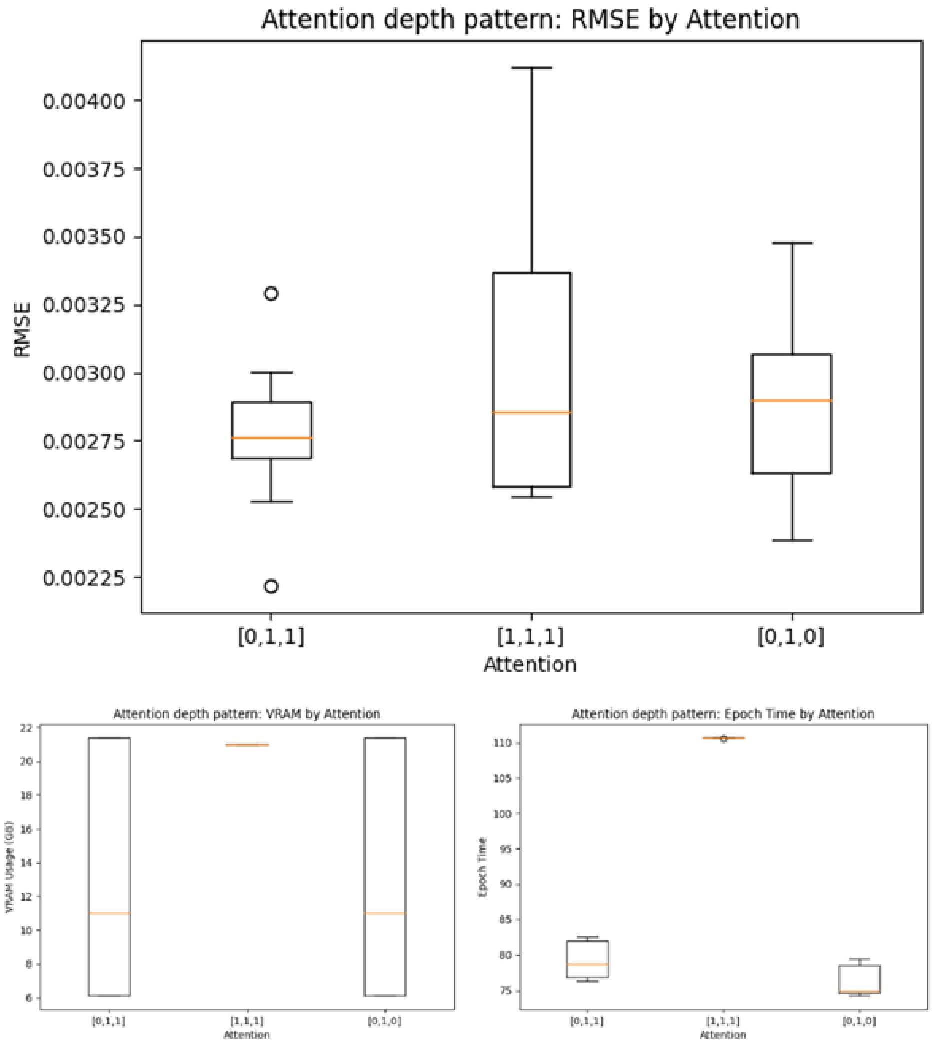


Figure 2 Box and whisker plots for attention depth sweep

4.4.2 Latent space architectures

4.4.2.1 Multi-Layer Perceptron (MLP)

Inserting a multi-layer perceptron bottleneck layer within a convolutional model added significant errors and artifacts to reconstructions. Convolution layers maintain spatial relations between the latent variables, but an MLP breaks these relations and forces all data represented in a single vector, and so does not give sufficient control over the latent space for use in downstream training tasks.

4.4.2.2 Convolutional

The majority of models have been trained with a convolutional latent space. This is the natural result of using convolutional blocks for down-sampling so requires no manipulation. The main advantage of a convolutional latent space is that spatial relations are maintained, so the latent space is just a reduced version of the data, with the additional of channels. Results showed that for data already regularly structured in 2D or 3D grids, convolution worked well. Applying convolutional layers to unstructured datasets will require further work to develop efficient methods of recovering the structure before applying convolution.

4.4.2.3 Transformer

Transformer based models trained so far showed similar if slightly worse RMSE, however the compression ratio was ambitious at 128x, and transformer architectures typically require more data to converge during training compared to other, comparably sized, architectures.

4.4.2.4 Variational Auto Encoder

Results for VAE's, both on ERA5 and a small test dataset showed worse reconstruction loss than standard auto-encoders. This is to be expected, since VAE penalises a model for correlation between encoded factors within latent variables as well as the reconstruction loss, and so has a more tightly constrained optimum that requires more data and training to converge compared to deterministic methods.

4.4.2.5 Disentangled Variational Autoencoders (β-TCVAE)

A sweep was attempted with different configurations of disentangled VAE, however, due to their requirements for a large batch size for effective training, the computational requirements were greatly in excess of the other architectures, and a fair test could not be carried out at this time.

4.4.3 Example reconstructions

Examples given below were produced with a model trained only on the temperature fields using a ResBlock Auto-Encoder architecture, using leaky_relu as the activation function. Each was trained on 1000 samples, with a batch size of 25, where each sample is a single time slice containing the entire atmosphere for each variable. The table shows the best RMSE recorded at each compression ratio, with reconstruction images below (Figure 3, Figure 4) for the lowest and highest ratios.

Ratio	RMSE
2:1	8.2e-5
4:1	1.79e-5
12:1	1.14e-4
21:1	1.46e-4

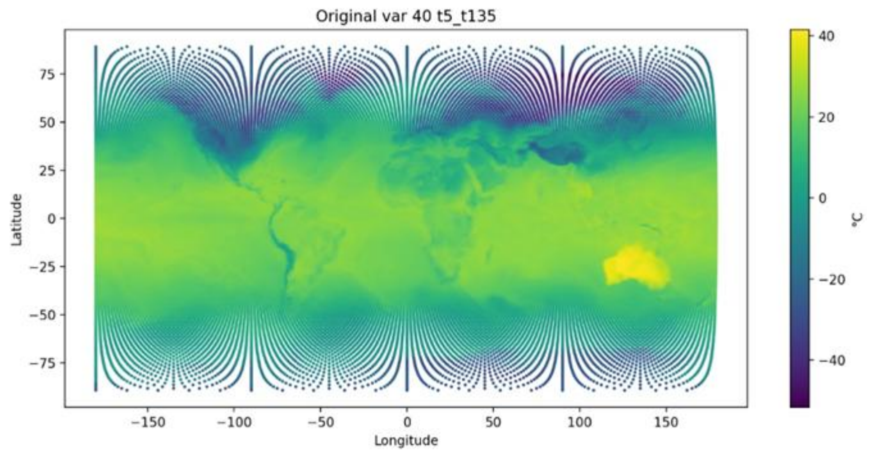


Figure 3 Original t_135 variable at timestep 5

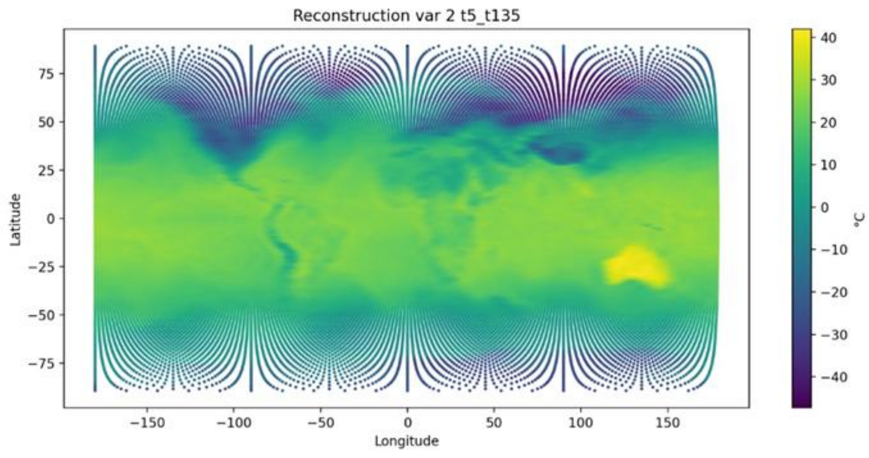


Figure 4 Reconstruction from a 21.3:1 compression ratio

While the best-performing 4:1 model achieved a marginally lower RMSE than the best 2:1 model, this difference is within the expected variance arising from stochastic training effects. When considering averages across all models trained with comparable data volume, reconstruction error increases with compression ratio, as shown in the following table.

Comp Ratio	Average RMSE
2:1	0.000203
4:1	0.000213
12:1	0.000224
21.3:1	0.000269

Overall, these results demonstrate that high-fidelity reconstruction is achievable at substantial compression ratios using the current framework, even before adopting architectures better suited to unstructured atmospheric data.

Discussion

The work presented in this report represents an early stage in a longer-term investigation into compressing high-resolution atmospheric datasets for downstream machine-learning applications. As such, the results should be interpreted primarily as evidence that the methodological approach and tooling developed are sound, rather than as definitive conclusions regarding optimal architectures or compression strategies. The primary contribution to date is the establishment of a flexible and reproducible experimental framework capable of supporting large-scale architectural sweeps and detailed performance analysis

One of the clearest outcomes of this study is the importance of dataset organisation and representation. Clearly, care must be taken when using architectures that rely on local correlations, such as convolutional networks, to ensure that the data are organised appropriately and convolutions are not applied across discontinuities.

The results highlight limitations of convolutional architectures when applied to irregularly gridded atmospheric data. While convolution performed well once the dataset was manipulated to better align with its assumptions, this alignment required non-trivial compromises. In the absence of a regular spatial grid, convolutional ingestion layers must implicitly learn spatial relationships that would otherwise be encoded in the data structure. This supports the conclusion that convolution is unlikely to be the most appropriate choice for data ingestion in this domain, even if it remains useful for down-sampling or refinement stages. Architectures that natively operate on unstructured data, such as graph-based models, are therefore a logical next direction.

An additional consideration raised by this work is the role of overfitting in the context of data compression. In conventional predictive modelling, overfitting is undesirable due to its negative impact on generalisation. However, for compression of a fixed dataset, some degree of specialisation may be acceptable or even beneficial. That said, excessive overfitting introduces a risk in this application: information may become encoded in the model parameters rather than the latent representation itself. Since the downstream foundation model will be trained solely on the compressed data, any information not present in the latent space would be irretrievably lost. This reinforces the need for compression methods that explicitly preserve all relevant information within the latent representation.

Several additional directions for improving reconstruction quality were identified but not fully explored. One example is the idea of storing a residual or difference map alongside the compressed representation, enabling near-lossless reconstruction while maintaining overall data reduction. While this approach introduces additional complexity, it may offer a useful trade-off between compression efficiency and fidelity and warrants further investigation.

The scale of the experiments conducted was also constrained by practical limitations. Due to access restrictions on the HPC system used, only a small fraction of the full ERA5 dataset was available for training. Most architectural sweeps were conducted using 100 time slices, with the best-performing models retrained on 1,000 samples. Given that the full dataset contains over 65,000 time slices, these experiments used approximately 1.5% of the available data. Increasing the training set size consistently reduced reconstruction error, with losses decreasing by roughly a factor of two when moving from 100 to 1,000 samples, suggesting that further gains are likely achievable without architectural changes.

Finally, as reconstruction error decreased, it became increasingly difficult to meaningfully compare models using RMSE alone or through qualitative visual inspection. Future work will therefore require more sophisticated evaluation metrics tailored to atmospheric data. These include alternative loss functions that penalise spatial or spectral artefacts, diagnostics focused on specific variables or regions, and physically motivated analyses such as spectral energy distributions and zonal wavenumber spectra. Ultimately, the most informative evaluation will be end-to-end: training a predictive weather model on both the original and compressed datasets and comparing forecast skill directly.

4.5 VQGAN for WeatherGenerator-Land

WeatherGenerator-Land is a geospatial foundation model that is being developed at the Max Planck Institute for Biogeochemistry (MPI-BGC) for modelling land surface processes and their interaction with the atmosphere. Its goal is to ingest Earth observations pertaining to the land surface from various heterogenous sources and assimilate them to a single representation vector. Similar to the atmospheric WeatherGenerator model, a majority of Earth observations that will be used for training are remote sensing data. However, over the course of a decade, a single, high-resolution Earth observation satellite mission, such as Sentinel-2, can generate more than 10 petabytes of data. While this wealth of information offers a unique opportunity for self-supervised pre-training, storing and processing such massive datasets is challenging, even for powerful HPC systems.

Over the past year, work has been conducted at the MPI-BGC to explore the use of VQGAN (Esser et al., 2021) for lossy neural compression. Our goals are two-fold:

1. Use VQGAN to reduce dataset size and, hence, enable scaling to more training data.
2. Exploit the learned codebook to construct a probabilistic masked autoencoder similar to Li et al. (2023) and Mizrahi et al. (2023). Such a formulation should result in better representations as it can, for instance, account for bimodality and uncertainty under cloudy conditions and better deal with the very high reflectance values that are observed in clouds.

Up until now, the primary focus lay on gathering experiences with VQGAN and establishing its usefulness for compression of remote sensing data. That is, no experiments have been conducted yet on the efficacy of compressed data for self-supervised learning, nor on using the codebooks to construct a probabilistic masked autoencoder. The results that we report here were also presented as a poster at the "*Third Workshop on Machine Learning for Earth System Modelling*" that took place in August, 2025 in Bonn, Germany.

4.5.1 Preliminary Experiments on ImageNet

Preliminary experiments with the VQGAN architecture have been conducted on ImageNet (Russakovsky et al., 2015) and we present reconstructions in Figure 6. Additionally, we report the following findings:

First, as shown in Figure 7, normalization of the input vector as well as the codebook vector prior to quantization lead to significantly improved reconstruction quality and codebook utilization. This result agrees with findings from Yu et al. (2021). Moreover, we find that it makes training more robust with regards to hyperparameter choices, particular the codebook and commitment loss weights. Normalization also leads to much more diverse outputs directly after initialization. Codebook collapse, i.e. the sudden collapse to only a single used codebook vector, has been completely absent when using normalization. We find that layer normalization (Ba et al., 2016) results in slightly better performance than projecting the vector onto the unit sphere.

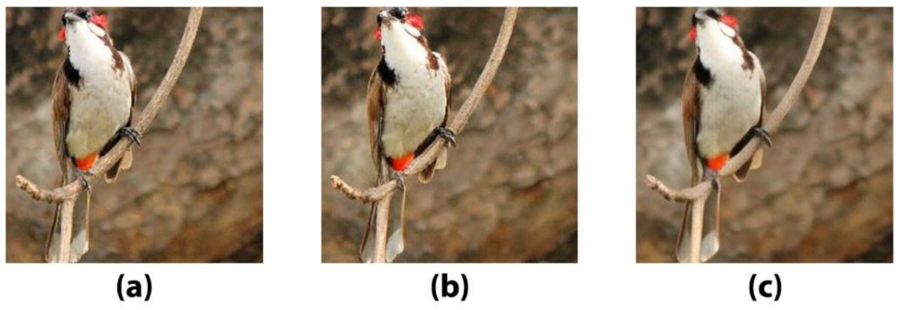


Figure 6: Original image (a), reconstruction produced by VQGAN-f8 (b), and reconstruction produced after pre-training, i.e. trained without an adversarial loss term (c). Without the adversarial loss, reconstructions lack fine-scale features and textures. The compressed versions are ~160x smaller than the original.

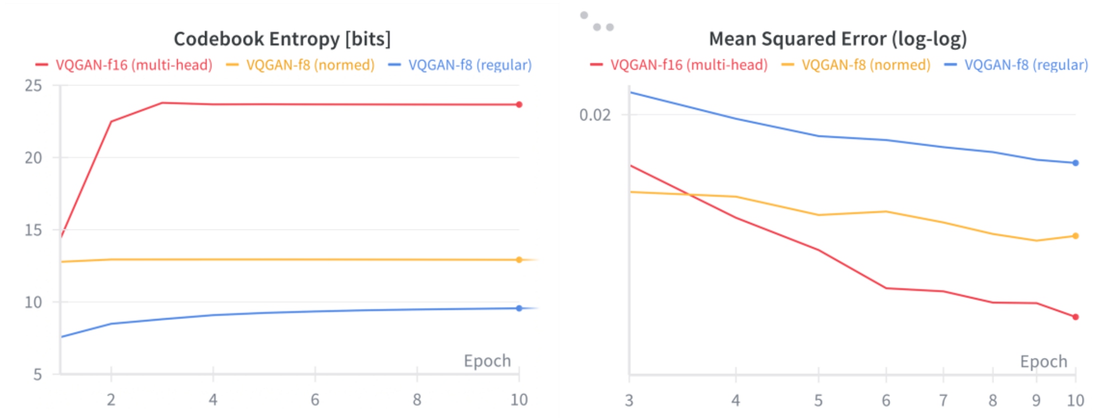


Figure 7: Codebook entropy (left) and validation mean squared error (right) during VQGAN pre-training on ImageNet. Despite using a more aggressive spatial downscaling factor of 16, the VQGAN-f16 model with multiple heads (red) has a smaller errors than models with a downscaling factor of 8 (orange and blue). It compensates for the lower latent space resolution by using more codewords.

Second, pre-training the generator for longer before introducing the discriminator for adversarial training significantly improves final reconstruction quality, agreeing with the observations from Esser et al. (2021).

Third, we find that the generator has a tendency to construct adversarial patches, i.e. small, noisy image regions which are nonetheless classified as being a real image patch by the discriminator. Examples for Sentinel-2 data are presented in Figure 8. While we didn’t find a solution, separating the training into two distinct runs, pre-training and GAN, allowed us to easily restart the GAN training in such a case. Additionally, we found that slowly ramping up the learning rate when introducing the discriminator allows it to better discriminate between fake and real images and makes the model less susceptible to this phenomenon.

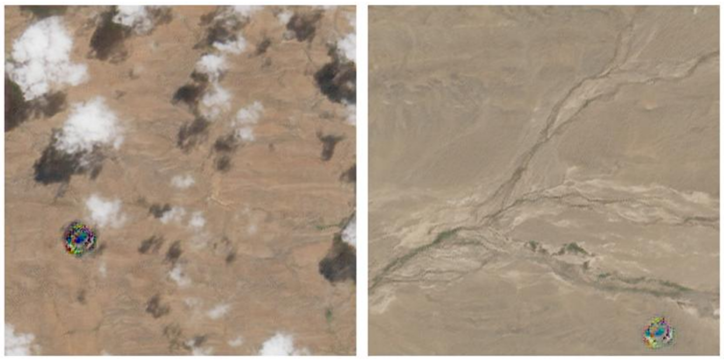


Figure 8: Two examples of adversarial patches that can spontaneously occur during GAN training. Once training diverges, they are found in every reconstruction produced by the autoencoder.

Fourth, performing quantization multiple times using different codebooks and projections, a method known as product quantization (c.f. Jegou et al., 2010), allows us to scale to much larger effective codebook sizes. As shown in Figure 6, in extreme cases we were able to scale to codebook sizes with more than 13.000.0000 used entries (or, equivalently, 23-24 bits of entropy). In comparison, when using a single codebook, scaling beyond 13 bits of entropy is often infeasible (c.f. Mentzer et al., 2023). Product quantization thus allows choosing the compression ratio of the model, which is determined by an architecture-dependent spatial downscaling factor and the entropy of the codebook, more freely.

Lastly, on the validation set, we are able to achieve a reconstruction Frechét inception distance (r-FID) (Heusel et al., 2017) of 0.73, simply by improving the training pipeline and hyperparameters, i.e. without modifying the original VQGAN model architecture. For comparison, Esser et al. (2021) report an r-FID of 1.49 despite training on the larger OpenImages dataset, while Yu et al. (2021) report an r-FID of 1.29 when training a vision transformer with the same downscaling factor and codebook size on ImageNet. The improvements mainly stem from introducing linear ramp up for the learning rate, cosine annealing, separating pre-training and GAN-training, pre-training for longer, and tuning the codebook, commitment, and adversarial loss weights, as well as the learning rate.

4.5.2 Sentinel-2 Experiments

To verify the suitability of VQGAN for the compression of remote sensing satellite data, we train both a single-codebook and multi-codebook model on Sentinel-2 L2A data over the African continent. Africa spans roughly 70 degrees of latitudes and multiple climate zones and, hence, provides a diverse, but still spatially bounded test bed for our compression method.

The input to our model is a 504x504x12 (Height x Width x Band) array of multi-spectral surface reflectance values in a local Universal Transverse Mercator projection and 10m resolution. A single input sample roughly covers a 5-by-5km area. To simplify initial experiments, the 20m and 60m bands are upscaled to 10m resolution.

For training, we sample 100.000 distinct month-location-pairs between 2017 and 2021. Within each month, we use the three least cloudy observations with at least 30% noncloudy pixels, resulting in roughly 270.000 remaining training samples (~690GB). Evaluation samples are obtained from the year 2022 using the same method but at distinct locations.

For the VQGAN model, we choose a downscaling factor of 4, resulting in 58M trainable parameters. The single-codebook model uses a codebook of size of 8192, while the multi-codebook model utilizes 4 codebooks with 1024 entries each (4 times 10 bits) which are progressively faded in during training. Both models are trained for 40 epochs on 8 A100 over the course of three days using AdamW, a global batch size of 64, mixed-precision training, and cosine annealing. The initial global learning rate is set to 2.88e-4. To account for clouds, which are significantly brighter than the land surface, the models are trained with a variant of an L1-loss that linearly reduces the weight of pixels with reflectances above 50%.

4.5.3 Results

Due to difficulties of training with an adversarial loss term, we only report results for pre-training, i.e. models solely trained with an L1-loss. Furthermore, as clouds exhibit much higher reflectance values and we are primarily interested in the land surface, any reported metrics are calculated on cloud-free pixels unless otherwise mentioned.

Figure 9 shows reconstructions produced by the multi-codebook model. Qualitatively, the reconstructions stay faithful to the input data but fine-scale features are slightly blurred out.

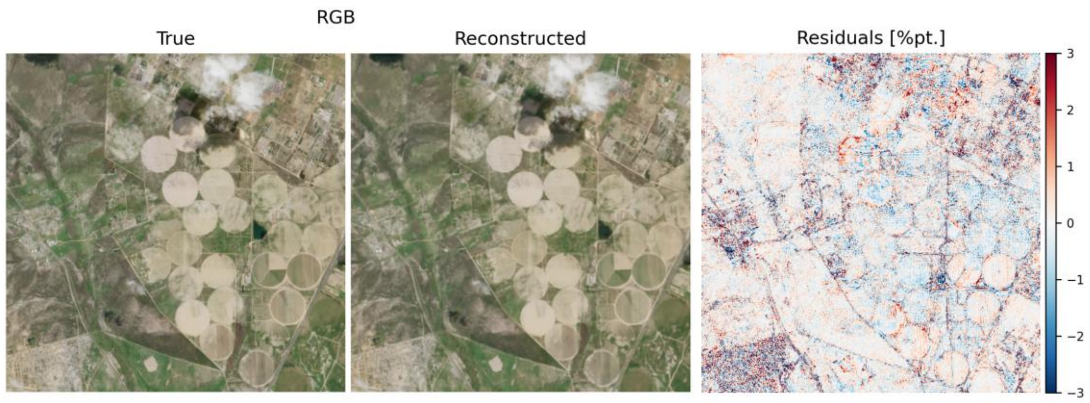


Figure 9: Sentinel-2 scene in Africa (left) and its corresponding reconstruction (center). The compressed scene is ~65x smaller than the original. The largest errors (right) are observed for clouds or in areas with many fine-scale features such as cities.

Quantitatively, we compare VQGAN to JPEG-2000 in Figure 10 and find that it achieves a 2-4x higher compression ratio for the same reconstruction error. Importantly, the reported compression ratios are with respect to the actual data volume of the data and not with respect to the data volume of the upscaled data that is used as input for the model. Consequently, a possible multi-resolution model would likely be able to achieve higher rates.

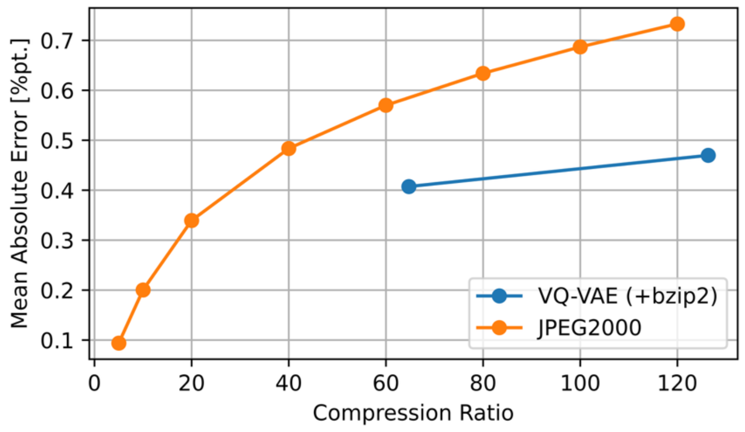


Figure 10: Mean reconstruction error as function of compression ratio for JPEG-2000 (orange) and our VQGAN models (blue). Under equal compression ratio, the VQGAN model achieves a significantly smaller error. Likewise, the compression ratio of the VQGAN is 2-4x larger than JPEG-2000 if the desired reconstruction error is kept constant. Codewords are further entropy-coded using bzip2 and the reported compression ratio is derived with respect to the true size of a Sentile-2 tile without upscaled 20m and 60m bands.

An important question is how to entropy-code the obtained codeword indices after compression with VQGAN. For the single-codebook model, we observe that bzip2 is able to reduce the average size of a single index to 10.17 bits while the actual entropy is 9.64 bits. As a result, by using a modern array storage format that provides compression out-of-the-box, such as zarr, optimal compression ratios can be obtained without extra effort.

Finally, we also inspect the reconstruction errors per individual band and report our findings in Figure 11. Despite the fact that the 20m and 60m have been upscaled to 10m beforehand, errors are roughly in the same order of magnitude for all bands. This indicates that the model is not able to adequately exploit these artificially introduced redundancies (namely, all 2x2 or 6x6 blocks have the identical value for 20m and 60m bands respectively).

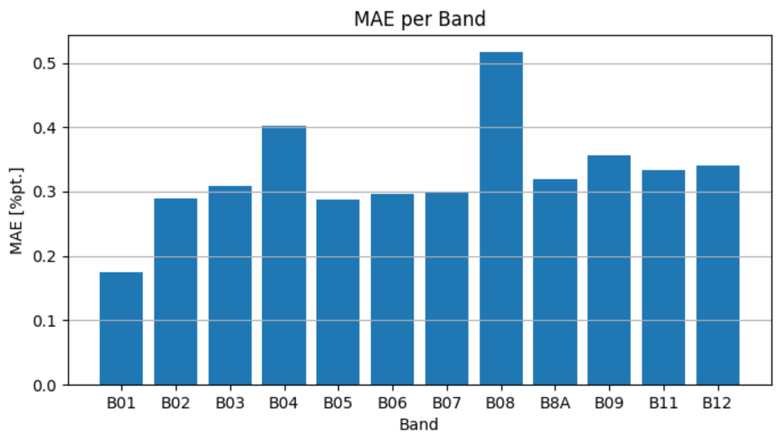


Figure 11: Mean absolute error per band for the multi-codebook model. Errors are distributed evenly across bands and resolutions, with band B08 (near infrared, 10m resolution) and band B01 (violet, 60m) being notable outliers.

4.5.4 Outlook

Our findings demonstrate the feasibility of the VQGAN architecture for the compression of Sentinel-2 remote sensing data. In the future, we plan to extent our results by

1. Using a vision transformer architecture to exploit the multi-resolution structure of the data.
2. Exploring simpler quantization schemes such as finite scalar quantization (Mentzer et al., 2023), possibly extending them with surrogate gradient methods.
3. Re-iterating on the GAN-training for Sentinel-2 data.
4. Integrating additional remote sensing data, such as Sentinel-1, to demonstrate feasibility beyond Sentinel-2.
5. Exploring the trade-offs of compressed vs uncompressed data for self-supervised training.
6. Investigating the possible benefits of a probabilistic masked autoencoder trained directly on discrete codewords.

4.6 Conclusions

The primary outcomes of this work are the successful development of a modular, reproducible training and evaluation pipeline built around a highly configurable Hydra-based workflow, and testing of the VQGAN architecture for compressing satellite imagery data. The framework enables rapid and systematic experimentation across architectures, datasets, and training strategies with minimal code changes, and provides a scalable foundation for continued investigation.

Early experiments using convolutional autoencoders indicate that data representation and preprocessing choices have a larger impact on reconstruction accuracy than architectural variation alone. This highlights the importance of aligning data layout and ingestion strategy with the inductive biases of the chosen architecture and suggests that dataset preparation should be treated as a first-order design decision rather than a secondary concern.

Although convolutional architectures were used throughout this initial phase, they were found to be a suboptimal choice for data ingestion due to the non-regular grid used in the ERA5 dataset and the use of a single spatial point dimension. While convolution may remain effective for down-sampling or refinement stages, alternative ingestion mechanisms are likely to be more appropriate. In particular, prior work suggests that graph-based architectures offer a more suitable inductive bias for unstructured atmospheric data, making them a clear priority for future investigation.

Despite these limitations, the results achieved are encouraging. Using a small fraction of the available dataset and modest computational resources, models were able to achieve high-fidelity reconstructions at compression ratios exceeding 21:1 for the convolution based auto-encoders. This demonstrates that substantial data reduction with low information loss is feasible and suggests that significantly improved performance should be attainable when training on larger datasets with architectures better matched to the underlying data structure.

No definitive conclusions can yet be drawn regarding optimal latent-space formulations. Deterministic latent representations and convolutional or MLP-based approaches outperformed variational methods in these early experiments, at least using RMSE for comparisons; however, this is likely influenced by limited training data, tuning constraints, and the additional complexity of variational objectives. Using more comprehensive analysis and comparison methods are likely to introduce some subtlety to that conclusion, given, for example, an MLP-based latent representation breaks spatial correlation introducing challenges for a downstream training task. VQGAN models show very promising results on 2D satellite imagery, and the next steps will be to adapt this to handle 3D atmospheric data.

Overall, this work establishes both the technical infrastructure and the initial empirical understanding required to support a longer-term research programme. Future work will focus on scaling training to larger datasets, adopting ingestion architectures suited to unstructured data, and developing more physically meaningful evaluation metrics, with the ultimate goal of enabling effective downstream prediction from highly compressed atmospheric representations.

References

1. Zhao, S., et al. (2025). Transforming Weather Data from Pixel to Latent Space.
2. Nguyen, T., et al. (2025). OmniCast: A Masked Latent Diffusion Model for Weather Forecasting Across Time Scales.
3. Cheng, S., et al. (2025). Machine learning for modelling unstructured grid data in computational physics: A review
4. Odena, A., et al. (2016). Deconvolution and Checkerboard Artifacts
5. Li, Y., et al. (2024). Spatial relaxation transformer for image super-resolution
6. Li, T., et al. (2023). Mage: Masked generative encoder to unify representation learning and image synthesis
7. Mizrahi, D., et al. (2023) 4M: Massively multimodal masked modelling
8. Esser, P., et al. (2021) Taming transformers for high-resolution image synthesis
9. Yu, J., et al. (2021) Vector-quantized image modeling with improved vqgan
10. Ba, J., et al. (2016) Layer normalization
11. Jegou, H., et al. (2010) Product quantization for nearest neighbor search
12. Mentzer, F., et al. (2023) Finite scalar quantization: Vq-vae made simple
13. Russakovsky, O., et al. (2015) Imagenet large scale visual recognition challenge
14. Heusel, M., et al. (2017) Gans trained by a two time-scale update rule converge to a local nash equilibrium
15. Mentzer, F., et al. (2023) Finite scalar quantization: Vq-vae made simple

5 Training ML models with compressed data

Introduction

Modern forecasting systems often rely on high-resolution reanalysis products, which are computationally expensive to store, distribute, and process. As climate datasets continue to grow in spatial and temporal coverage, the demand for efficient data reduction techniques without compromising the performance of downstream machine learning models has become increasingly urgent.

A promising approach for large-scale climate data compression is the use of JPEG-2000, an established wavelet-based codec capable of providing tuneable lossy compression with controllable bit rates. JPEG-2000 natively supports reduced-resolution decoding, enabling users to retrieve downsampled versions of the data directly from the compressed codestream without decompressing the full-resolution field. This offers significant benefits for bandwidth-limited or distributed machine learning workflows, where fetching only a subset of the data resolution can substantially reduce I/O cost.

However, it remains unclear to what extent deep learning models can maintain predictive skill when trained on JPEG-2000 compressed and reduced-resolution inputs. Compression removes fine-scale texture and attenuates gradients, while reduced-resolution decoding yields systematically coarser spatial fields. Both operations could potentially degrade the ability of neural networks to learn the physical structure and variability of ocean temperature.

This study investigates the robustness of deep learning–based ocean forecast when the input fields are compressed with the lossy JPEG-2000 (Taubman et al., 2002) compression and reduced-resolution encoding. The main goal is to investigate the accuracy of an ML model trained by using compressed data. The ML architecture used for this investigation is based on a U-Net with the aim to extend the same analysis to the more complex WeatherGenerator architecture developed in WP3 and WP4.

Methodology

The main objective of the investigation is to train the U-Net model to forecast the next day using the JPEG2000 compressed data as training dataset and compare the forecast skill with the same U-Net architecture trained with the original dataset (uncompressed). The comparison is repeated by changing the compression ratio (c-ratio) and the resolution level (r-level). The resolution level ranges from full, to half and quarter resolution while the compression ratio (c-ratio) ranges in the interval [16, 32, 64]. Each forecast obtained by the model trained with the compressed data has been compared with the forecast obtained by the same model architecture but trained with the original data using the following evaluation metrics: Root Mean Square Error (RMSE); Peak Signal-to-Noise Ratio (PSNR) to measure the quality of reconstructed or compressed images and Structural Similarity Index Measure (SSIM) used in computer vision for measuring the similarity between two images. The comparison has been repeated with multi-seed ensembles to assess training stability.

5.1.1.1 Dataset

We used the Mediterranean reanalysis dataset[1] with daily mean value of the ocean variables at 1/24′ horizontal resolution and 141 vertical levels. For our experiment we

[1] Escudier, R., Clementi, E., Omar, M., Cipollone, A., Pistoia, J., Aydogdu, A., Drudi, M., Grandi, A., Lyubartsev, V., Lecci, R., Cretí, S., Masina, S., Coppini, G., & Pinardi, N. (2020). Mediterranean

sampled only 18 vertical levels and considered four three-dimensional variables, salinity, temperature, eastward northward velocity, and one two-dimensional variable, sea surface height (SSH). The original data was converted and stored in numpy format to speedup data loading.

All input variables (3D and 2D) are encoded into the JPEG-2000 codestream using three compression ratios: 16, 32, 64. Each variable generates: a JP2 codestream file; a JSON metadata file containing mean, standard deviation, and clipping range; a binary mask indicating valid ocean point. The compression is applied independently to each depth level and variable. A key feature of JPEG-2000 is that low-resolution versions of the image can be decoded directly from the codestream without reconstructing the full-resolution image.

For each day t, we have a 73-channel input tensor: 72 channels from the 4 three-dimensional variables (18 levels each) and one channel from SSH at the same spatial resolution. Masks are resized using nearest-neighbor interpolation to preserve ocean/land boundaries.

Before compression, data is normalized with a standard scaler. All channels are stored in standardized z-score. The input shape for the U-Net depends on r-level:

- r-level=0: full resolution 380×1016
- r-level=1: half resolution 190×508
- r-level=2: quarter resolution 95×254

Model

For the investigation we used a U-Net designed to ingest JPEG2000 compressed input Mediterranean state at time t and forecast the Mediterranean state full resolution at time t+1. The model architecture is made of residual MultiConv blocks with GroupNorm and SiLU activation in the encoder, dilated convolutions to increase receptive field at the bottleneck, and ConvTranspose2d upsampling layers in the decoder.

5.1.1.2 Training setup

The model was trained using four years of daily mean values, one year for the validation and one year for testing. The masked MSE loss function evaluated only on the ocean points was used to ensure that the land points do not influence training and to better suit the model for the evaluation that is performed only on the ocean points. To ensure reproducibility we used fixed seeds for data loaders, workers, and CUDA generators. During the training we used the Adam optimizer (lr = 1e-4), a batch size of 1, a maximum of 50 epochs, early stopping with patience 5, and triggered when validation loss fails to improve by at least 1e-4. Finally, we repeated the training with a multi-seed sweep to assess model stability (4 repetitions were used).

Results

This section presents a comprehensive evaluation of the U-Net model across all dataset configurations, beginning with the uncompressed ("normal") baseline and followed by JPEG-2000 compressed data at three wavelet resolution levels (r-level 0, r-level 1 and r-level 2) and three compression ratios (16, 32, and 64). For each configuration, we report training and validation curves averaged across four random seeds and summarize model skill on the test set using three complementary metrics: RMSE, PSNR,

Sea Physical Reanalysis (CMEMS MED-Currents) (Version 1) [Data set]. Copernicus Monitoring Environment Marine Service (CMEMS). https://doi.org/10.25423/CMCC/MEDSEA_MULTIYEAR_PHY_006_004_E3R1 and SSIM. These metrics quantify, respectively, the absolute physical error, the reconstruction fidelity relative to dynamic range, and the preservation of structural patterns within the ocean fields. Finally, aggregated bar-plot comparisons highlight the overall impact of compression ratio and resolution level on prediction accuracy. Together, these results allow us to systematically assess how lossy JPEG-2000 compression affects predictive skill under different resolution levels.

5.1.1.3 Baseline

The uncompressed dataset provides an upper-performance baseline for evaluating the impact of JPEG-2000 compression. Across four random seeds, the U-Net model trained on full-resolution data (380 × 1016) achieves consistently strong convergence and highly stable validation behaviour. The average validation loss is 0.00372 ± 0.00009, with extremely small variance across seeds, indicating that the model is both stable and largely insensitive to initialization when trained on clean inputs.

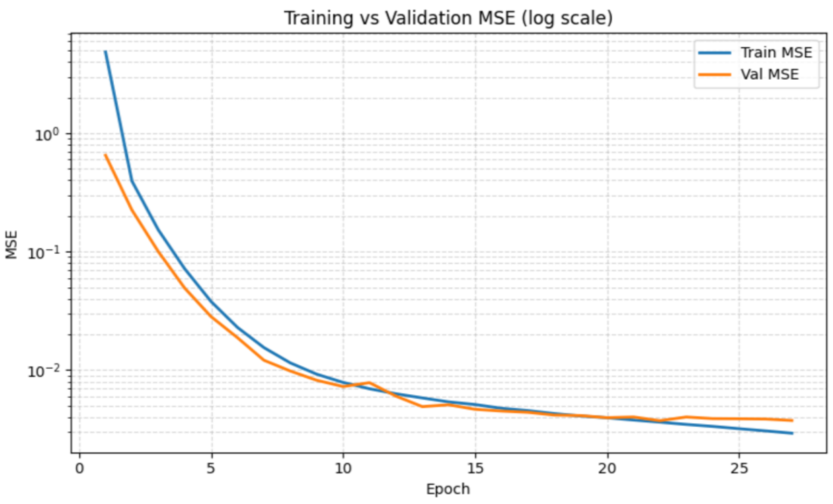


Fig12. training and validation curves for the model trained with original dataset

The training and validation curves exhibit the expected pattern for a well-behaved supervised learning system: a rapid initial drop during the first few epochs followed by a smooth, gradual decrease until early stopping around epoch ~25. Importantly, the validation loss closely tracks the training loss throughout, showing no signs of overfitting, which confirms that the model generalizes well when provided with high-quality data.

Test set results further validate this conclusion. The model achieves an RMSE of 0.1644 ± 0.0034 °C, which is the best performance among all experiments. This error range corresponds to extremely accurate reconstructions of next-day temperature fields, with small errors. The corresponding PSNR of 42.74 ± 0.18 dB is the highest across all configurations and reflects excellent fidelity relative to the expected temperature dynamic range. In addition, the SSIM score of 0.9956 ± 0.0002 indicates that the model preserves nearly all structural and spatial patterns present in the ground truth fields.

Overall, the normal dataset results establish the upper bound of predictive skill in this study. This performance serves as a reference point for interpreting the degradation observed under compressed JP2 (r-level 1 and r-level 2) conditions.

The expanded comparison across all compression settings provides a clear picture of how JPEG-2000 compression and wavelet resolution jointly influence forecasting accuracy. r-level=0 corresponds to no wavelet transformation; only JPEG-2000 quantization is applied. The compression itself introduces very little error, even at extreme compression. Moreover, the difference between uncompressed and compressed data at r-level=0 is only ~0.02 °C, confirming that JP2 quantization preserves most physically relevant information.

This demonstrates that wavelet quantization is not the primary source of degradation but rather the resolution level.

5.1.1.4 Compressed data

At resolution r-level 1, the model receives data reduced by a factor of 2 in each spatial dimension. This level delivers the best trade-off between compression efficiency and predictive accuracy. Errors are only ~0.05–0.07 °C higher than the uncompressed baseline. The performance degrades smoothly as compression increases. Finally, the spatial structures are still well preserved as witnessed by the SSIM value, enabling the U-Net to recover detailed temperature patterns. r-level 1 is therefore the most robust and practical compression level for operational forecasting.

At r-level 2, the model receives heavily downsampled inputs (¼ spatial resolution). Two important trends emerge: resolution dominates compression ratio moving from r-level 1 to r-level 2 and it introduces the largest increase in RMSE, far greater than adjusting c-ratio. Moreover, higher compression slightly smooths the data and reduces error. RMSE decreases as c-ratio increases, suggesting that stronger quantization sometimes removes noise-like details that are hard for the model to learn. Overall, the poor accuracy with r-level 2 is driven primarily by loss of spatial information, not due to the JPEG-2000 quantization.

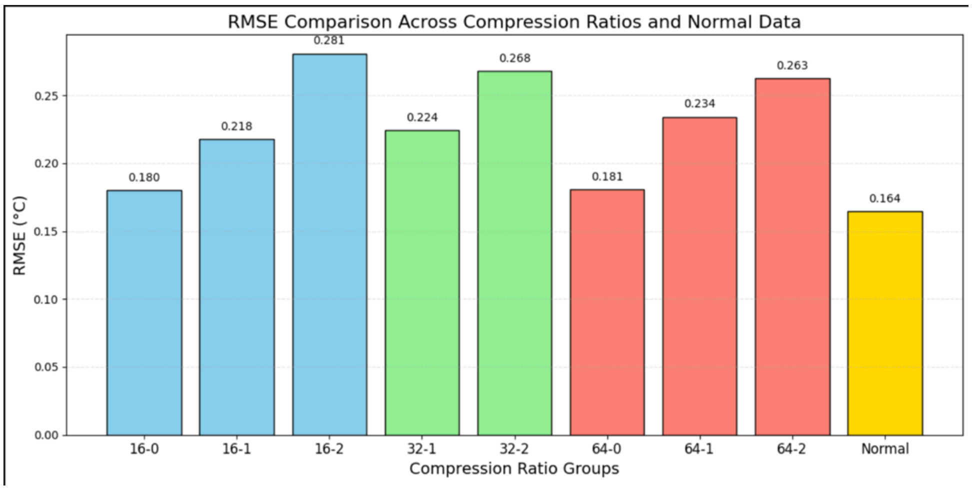


Fig 13. RMSE for temperature obtained by models trained with the dataset compressed at different resolution levels and compression ratios.

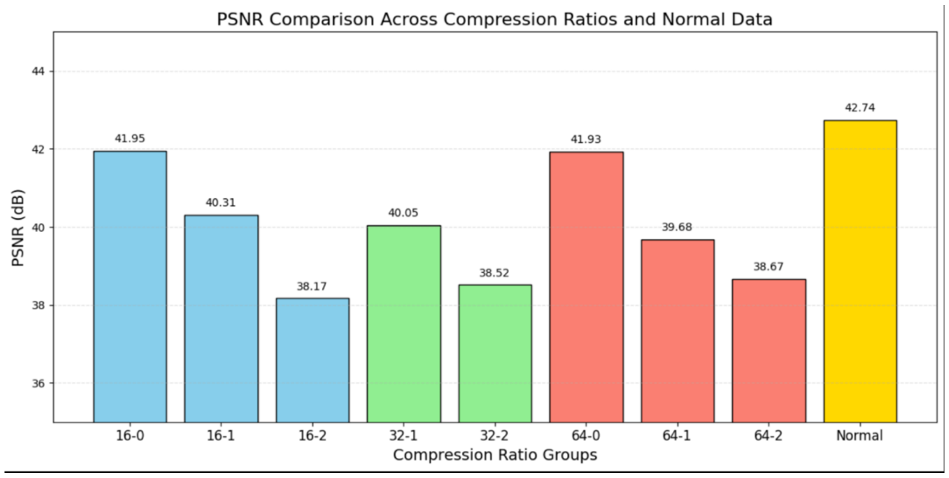


Fig. 14 PSNR for temperature obtained by models trained with the dataset compressed at different resolution levels and compression ratios.

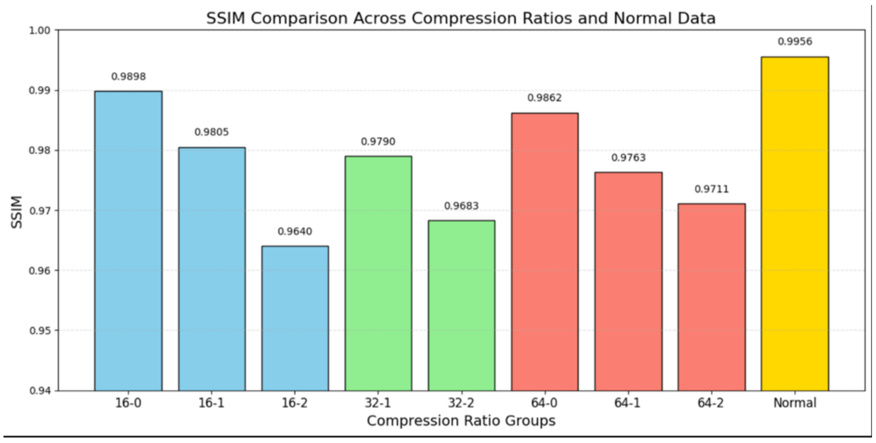


Fig. 15 SSIM for temperature obtained by models trained with the dataset compressed at different resolution levels and compression ratios.

5.1.1.5 Generalization Test

To assess the robustness of the compression-aware U-Net, a model trained on JP2 data compressed at c-ratio=64 and r-level=1 was evaluated on datasets compressed at c-ratios 16 and 32 (without re-training the model). Results showed almost identical RMSE, and PSNR across all cases. This demonstrates that the model does not overfit to compression-specific artifacts and instead learns physically meaningful structures that are preserved across compression levels. The finding has practical importance: a single trained model can reliably operate on multiple JP2 compression settings without retraining, simplifying deployment in operational pipelines.

Two factors could explain this behaviour: training at c-ratio 64 exposes the model to stronger smoothing and quantization noise and this acts like regularization, making the model tolerant to a wide range of compression artifacts. At r-level=1, structural information is preserved extremely well across all ratios as reported by SSIM and PSNR results, this means the effect of compression ratio on temperature fields is relatively small compared to true ocean variability and to model uncertainty, therefore, the model is operating in a regime where compression noise is small relative to the predictive task.

5.1.1.6 Storage Efficiency

A central motivation for this study is to reduce the exceptionally large volume of full-resolution reanalysis data. The 3D temperature-salinity-velocity fields alone exceed 1.4 terabytes, and the 2D SSH dataset adds another 20 gigabytes. Operating machine-learning prediction pipelines with such data is computationally expensive and often infeasible for real-time or resource-constrained systems. JPEG-2000 compression dramatically reduces storage requirements while retaining high predictive skill, as shown earlier in the evaluation sections.

Below we summarize the achieved storage compression for the three tested settings all evaluated at r-level=1.

Raw Size	c-ratio	JP2 Size	Reduction
1465.04 GB	16	8.48GB	172×
	32	4.62GB	317x
	64	2.69GB	544x

Conclusion

Results show that the U-Net model maintains strong predictive skill under moderate compression and half resolution (r-level = 1, c-ratio = 16), providing performance close to that of the uncompressed baseline. As spatial resolution decreases, the model requires more training epochs and performance exhibits greater variability, though still retaining adequate forecasting accuracy. Surprisingly, extremely high compression (c-ratio = 64, r-level = 2) does not always degrade performance, suggesting that the dominant predictive features may remain preserved even under aggressive JPEG-2000 quantization.

Overall, this work demonstrates that deep learning ocean forecast models are resilient to significant data compression, offering a practical pathway for large-scale climate applications where storage, network bandwidth, and computational cost are limiting factors.

References

Taubman et al. 2002: "JPEG2000 Image Compression Fundamentals, Standards and Practice" Springer Nature Link - https://link.springer.com/book/10.1007/978-1-4615-0799-4

6 Running ML weather-forecasting on point-wise error bounded compressed initial data

6.1 Introduction

In this study, we investigate the impact of using error-bounded compressed weather data as initial conditions for machine learning based weather forecasting models. The objective is to determine whether error-bounded lossy compression can significantly reduce storage and I/O requirements while preserving forecast skill and fidelity.

We introduce an error-bounded compression framework tailored to atmospheric data. The framework supports both standard global maximum error bounds and point-wise error bounds. With the point-wise error bound compression mode, we are able to align compression error with the intrinsic uncertainty in reanalysis datasets such as ERA5.

We achieve a 51.2x compression ratio when compressing ERA5 dataset in point-wise error bound mode using the ERA5 uncertainty data. When applying the compressed dataset to the Aurora ML forecasting model, we achieve a nearly identical forecast with less than 0.5% increase in RMSE of forecast error while other established compression methods lead to much larger forecast errors of more than 1% increase in RMSE.

This experiment demonstrates our compression framework is capable of significantly reducing the input data volume for weather forecasting while keeping the compressed data within the uncertainty range and achieving a near identical forecast result as the original data.

6.2 Methodology

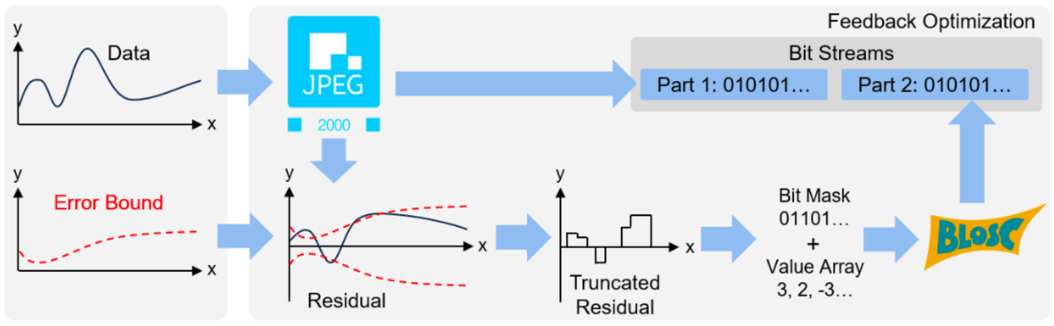


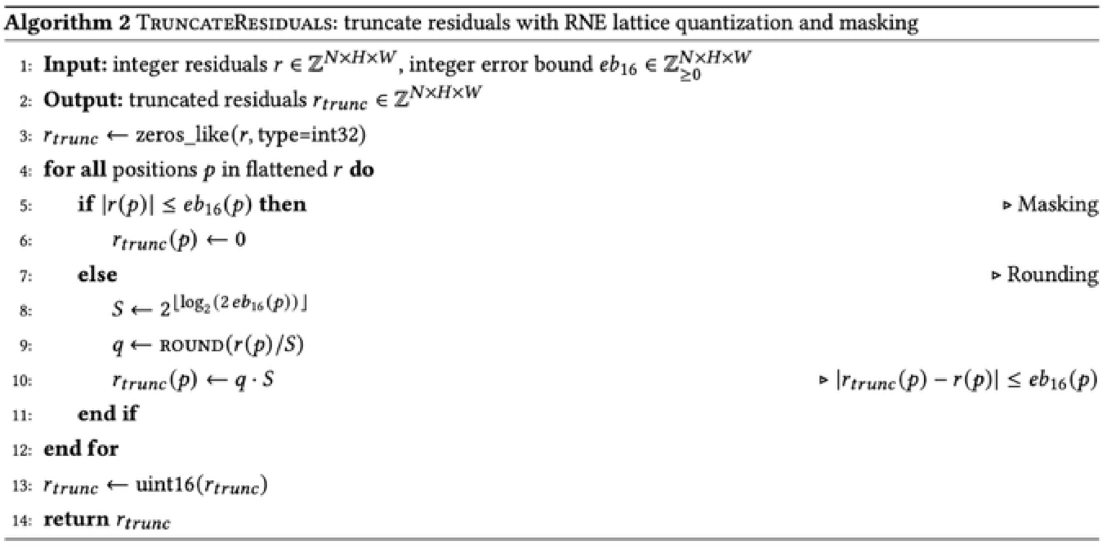
Fig. 16 Overview of compression framework. It first applies JPEG2000 to compress the original data, and then truncates the residual using a pointwise error-bound target. The truncated residual is represented by (i) a bitmask of nonzero residual locations and (ii) an array of the corresponding nonzero residual values. These two arrays are compressed with the Blosc library, using bit-shuffling and the ZSTD compressor.

We employ an error-bounded compression method combining JPEG2000 as a base encoding with an adaptive residual encoding guided by error bounds. As shown in the overview figure, our method consists of the following core stages:

- Base compression via JPEG 2000. This stage applies JPEG2000 to obtain a strong overall compression ratio with a low RMSE metric. A predefined compression level (parameter cratio) controls the target compression ratio in this stage.
- Adaptive residual encoding via truncation (Algorithm 2). This stage encodes the base-layer compression residual. Residuals are rounded and masked according

to the error-bound target. The resulting sparse truncated residuals are then compressed losslessly using bit-shuffle and ZSTD algorithm.

- Optional feedback optimization for improved compression ratio. Although the residual stage enforces the pointwise error bound regardless of the base-layer cratio, there exists an optimal cratio parameter that maximizes the overall compression ratio. This parameter can be set a priori or obtained via feedback optimization, where we use the golden-section search algorithm to find the cratio that maximizes the overall compression ratio.



Algorithm 2 TruncateResiduals: truncate residuals with RNE lattice quantization and masking

```
Input: integer residuals $r \in \mathbb{Z}^{N\times H\times W}$, integer error bound $eb_{16} \in \mathbb{Z}_{\geq 0}^{N\times H\times W}$
Output: truncated residuals $r_{trunc} \in \mathbb{Z}^{N\times H\times W}$
$r_{trunc} \leftarrow$ zeros_like($r$, type=int32)
for all positions $p$ in flattened $r$ do
    if $|r(p)| \leq eb_{16}(p)$ then                                  ▷ Masking
        $r_{trunc}(p) \leftarrow 0$
    else                                                              ▷ Rounding
        $S \leftarrow 2^{\lfloor \log_2(2\, eb_{16}(p)) \rfloor}$
        $q \leftarrow \textsc{Round}(r(p)/S)$
        $r_{trunc}(p) \leftarrow q \cdot S$                            ▷ $|r_{trunc}(p) - r(p)| \leq eb_{16}(p)$
    end if
end for
$r_{trunc} \leftarrow$ uint16($r_{trunc}$)
return $r_{trunc}$
```

We use a per-chunk min-max scaling to convert both original data and the pointwise error bound from floating-point format to 16-bit unsigned integer format prior to compression. Similarly, we apply the inverse scaling to recover the original dynamic range and floating-point format after decompression.

To account for any rare discrepancies introduced by integer rounding or dequantization, we apply a final check as the last step of compression. We detect any positions $\boldsymbol{p}$ where $|\boldsymbol{x}(\boldsymbol{p}) - \widehat{\boldsymbol{x}}(\boldsymbol{p})| > \boldsymbol{eb}(\boldsymbol{p})$ and store a losslessly compressed bitmask of such positions and the original values in floating-point format at those positions. At decompression time, we overwrite these positions in the decompressed result, thereby guaranteeing that the compression error is bounded by the pointwise error bound in the original floating-point format.

6.2.1 Residual truncation

We define the residual array as $\boldsymbol{r} = \boldsymbol{u} - \widehat{\boldsymbol{u}} \in \mathbb{Z}^{N\times H\times W}$, where $\boldsymbol{u}$ is the original data in unsigned integer format, $\widehat{\boldsymbol{u}}$ is the JPEG2000 reconstructed data, $\boldsymbol{N}$ is the number of channels of the data (e.g., number of timesteps, or variables), $\boldsymbol{H}$ and $\boldsymbol{W}$ are sizes of spatial dimensions (e.g., number of latitude and longitude grid points in a regular grid). We truncate it using the error bound $\boldsymbol{eb_{16}}$ as follows

$$r_{trunc}(\boldsymbol{p}) = \begin{cases} \boldsymbol{0}, |\boldsymbol{r}(\boldsymbol{p})| \leq \boldsymbol{eb_{16}}(\boldsymbol{p}), (\textbf{Masking}) \\ \boldsymbol{Q_{S(p)}}(\boldsymbol{r}(\boldsymbol{p})), |\boldsymbol{r}(\boldsymbol{p})| > \boldsymbol{eb_{16}}(\boldsymbol{p}), (\textbf{Rounding}) \end{cases}$$

, where $\boldsymbol{p}$ is the position index, and $\boldsymbol{Q_S}$ denotes round-to-nearest-even (RNE) quantization on a lattice with step size $\boldsymbol{S}(\boldsymbol{p}) = \boldsymbol{2}^{\lfloor \log_2(2\, \boldsymbol{eb_{16}}(\boldsymbol{p})) \rfloor}$. Residuals that are already within the point-wise bound are masked to zero, and the remaining values are rounded to a coarser lattice determined by $\boldsymbol{S}$.

We claim that whenever we choose the masking or rounding branch the point-wise error bound is always satisfied. In the masking branch, residuals already within the pointwise bound are set to zero, which trivially satisfies the bound. In the rounding branch, we have following lemma:

Lemma 1. For residual at position $\boldsymbol{p}$, $\boldsymbol{r}(\boldsymbol{p}) = \boldsymbol{u}(\boldsymbol{p}) - \hat{\boldsymbol{u}}(\boldsymbol{p})$, the RNE quantizer $\boldsymbol{Q_S}$ on a lattice with step size $\boldsymbol{S} = \boldsymbol{S}(\boldsymbol{p}) = 2^{\lfloor \log_2(2\, eb_{16}(\boldsymbol{p})) \rfloor}$ has a rounding error bounded by $\boldsymbol{eb_{16}}(\boldsymbol{p})$:

$$|\boldsymbol{Q_S}(\boldsymbol{r}) - \boldsymbol{r}| \leq \frac{\boldsymbol{S}}{2} \leq \boldsymbol{eb_{16}}(\boldsymbol{p}).$$

As a result, the final reconstruction $\hat{\boldsymbol{u}} + \boldsymbol{r_{trunc}}$ has compression error bounded by the point-wise error bound $\boldsymbol{eb_{16}}$.

6.2.2 Residual compression

Because the truncation step sets as many residuals to zero as possible, the truncated residual field $\boldsymbol{r_{trunc}}$ is typically sparse. We store $\boldsymbol{r_{trunc}}$ in a sparse format by decomposing it into a packed bitmask $\boldsymbol{r_{mask}} = \boldsymbol{1}\{\boldsymbol{r_{trunc}} \neq \boldsymbol{0}\}$ and then compress it losslessly with bit-shuffling and ZSTD at the highest compression level. The base-layer bitstream and encoded residuals are concatenated to form the final compressed bitstream.

6.2.3 Feedback optimization

The only tweakable parameter in the compression pipeline is the JPEG2000 compression level cratio. Although the residual truncation step keeps the final compression error within the point-wise error bound for any cratio, the overall compression ratio depends on this parameter. For a small cratio, the base layer achieves a lower compression ratio, but the residuals are very sparse and require little storage. For a large cratio, the base layer achieves a higher compression ratio, but the residuals require substantially more storage to satisfy the error bound.

Consequently, there exists an optimal cratio that maximizes the overall compression ratio. We implement a search procedure based on the golden-section search to find the optimal cratio. In practice, this step can be run on a small subset of the data to pre-determine a near-optimal cratio when compressing large datasets such as ERA5, thereby reducing the overhead of feedback loops.

Experiment

6.2.4 Experiment

We evaluate the impact of compression on ML weather forecasting. We use a leading ML forecasting model Aurora and measure changes in forecast errors when using compressed data as inputs. We report the latitude-weighted root mean squared error (RMSE), which is consistent with Aurora's evaluation loss function. We feed Aurora with compressed data at timestep -6 hour and 0 hour and produce forecasts at future 8 steps (up to 48 hours). The experiment uses the ERA5 data in the first week of 2024 December and we report the RMSE of forecast result using different compression methods against the original uncompressed ERA5 and the future steps. We choose SPERR, SZ and SZ3 as the baseline methods. We use a search loop to search the error bound setup for them such that their compression ratios are matched with our method. In the current setup, the compression ratio is 60x for all methods.

Our method shows negligible impact in the RMSE while others introduce significant increase in the forecast error compared to using original ERA5. The increase in the RMSE is less than 1% in all forecast steps using our compressed ERA5 dataset.

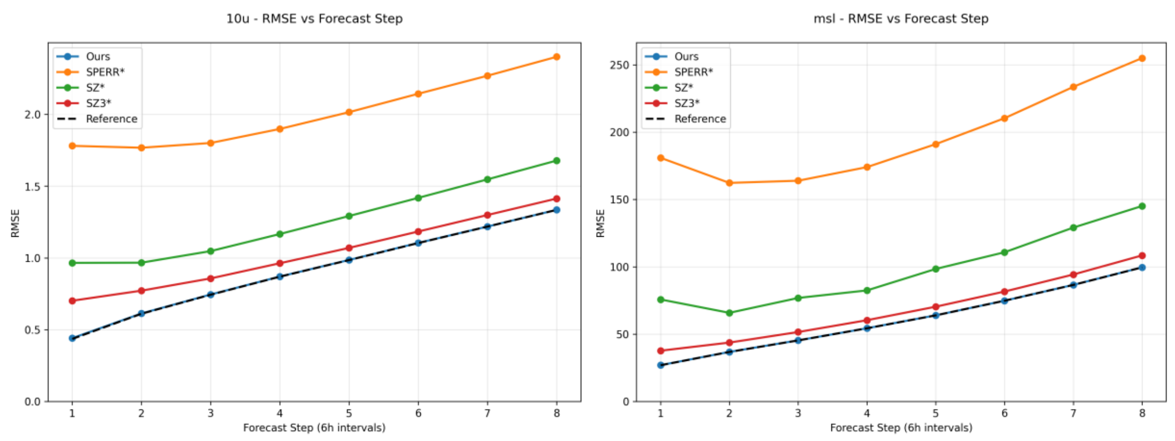


Fig. 17 Forecast RMSE at different future steps. For 10m u component of wind (left) and mean sea-level temperature (right). All methods are at same compression ratio of 60x, star symbol (*) means using constant global error bound.

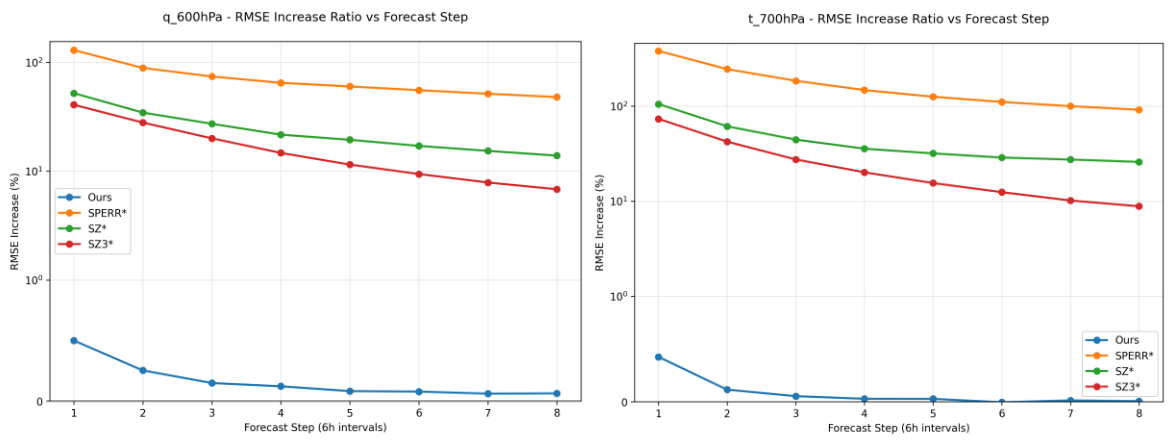


Fig. 18 Forecast RMSE increase percentage over forecast on original ERA5. For specific humidity at 600hPa (left) and temperature at 700hPa (right). All methods are at same compression ratio of 60x, star symbol (*) means using constant global error bound.

References

Bodnar, Cristian, et al. "Aurora: A foundation model of the atmosphere." *arXiv preprint arXiv:2405.13063* 1.8 (2024).

Collet, Yann, and Murray Kucherawy. *Zstandard compression and the application/zstd media type*. No. rfc8478. 2018.

Taubman, David S., and Michael W. Marcellin. "JPEG2000: Standard for interactive imaging." *Proceedings of the IEEE* 90.8 (2002): 1336-1357.

Huang, Langwen, et al. "Error bounded compression for weather and climate applications." *arXiv preprint arXiv:2510.22265* (2025).

Di, Sheng, and Franck Cappello. "Fast error-bounded lossy HPC data compression with SZ." *2016 ieee international parallel and distributed processing symposium (ipdps)*. IEEE, 2016.

Liang, Xin, et al. "Sz3: A modular framework for composing prediction-based error-bounded lossy compressors." *IEEE Transactions on Big Data 9.2* (2022): 485-498.

Li, Shaomeng, Peter Lindstrom, and John Clyne. "Lossy scientific data compression with sperr." *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2023.

7 Batch Curation

7.1 Background

Conventional wisdom suggests that training a model with more data is better. This has meant that compute environments have had to scale to handle increased data volumes. However, Sorscher et al., 2022 showed that with data pruning the typical scaling laws can be beaten. There have since been several examples of smaller, heavily curated datasets outperforming the 'more is more' approach. Both LIMO (Ye et al., 2025) and s1 (Muennighof et al., 2025) showed greatly increased performance during LLM fine-tuning against the AIME benchmark with dataset sizes of 1% and 1.7% of their compared methods. These suggest that there are ways to reduce dataset size whilst maintaining or improving model performance. These methods are however only verified for fine tuning. Well et al. (2021) showed that there was potential for curation to take place with respect to a random batch and the model state whilst improving model accuracy. Google Deepmind's JEST (Evans et al., 2024) furthered this, showing increased average accuracy whilst training on less data and requiring less computation. These all focus on text and image data which are popular and where the issues of scaling and collapse are particularly notable. Weather prediction models also suffer from scaling issues and so investigating further data curation methods could be beneficial.

A relatively generic and simple batch curation method, Spectral Analysis and Joint Batch Selection (SALN) (M. Sharifi, 2024), was tested for its applicability. SALN aims to increase accuracy and decrease training time even compared to using JEST for the image classification tasks of the OxfordIIIPet and CIFAR10 datasets.

7.2 SALN

SALN is a relatively simple batch curation technique that assesses the informational similarity between members of a random batch. A filter is applied to remove members that do not contain the desired relative information quality.

The similarity is calculated using the cosine similarity between the vectorised members of the batch. A graph is then created with these cosine similarities as weighted edges. Calculating the eigenvectors and values for this graph, specifically the Fiedler vector, yields the connection across batch members. Ordering these members by their connectedness then allows slices to be taken with a desired informational structure. For example, taking only the most similar data may provide a more consistent signal covering only the most probable portions of the data distribution. Taking the least connected data may then cover extremes, unique situations or noisy elements. This allows for some amount of customisation in training instead of solely relying on the random batch to provide good gradient estimates.

SALN has the potential to improve training speed in two main ways,

1. Faster convergence due to clearer training signal
2. Less data pushed through the model

As SALN only cares about the data it can be run either as a preprocessing step or online as the model trains. In particular SALN cares about random batches of data and so wants batch sizes to hit a sweet spot between enough data to compare for meaningful batch selection and not so much data that these comparisons take too long. As SALN compares all N batch members it has O(N^2)complexity which is also affected by the size of each member. This is important when the model is large and so can only be run with a small batch size. In this situation, SALN can be run as a preprocessing step on super batches of the dataset. As long as the resulting SALN batch lengths are exact multiples of the training batch size and the training order is restricted to doing super batches in clusters SALN should perform similarly. Gradient accumulations will then also want to span these super batches.

7.3 Experiments

Two core experiments were conducted to explore the capabilities of SALN. Firstly SALN was applied to a relatively simple dataset taking temperature, relative humidity and vertical velocity as inputs to predict a mask of dry intrusions. Following this, SALN was tested within FastNet v1.1 - the UKMO and Alan Turing Institutes' data driven medium range weather prediction system (Daub et al., 2025). It is based on graph neural networks and is under active development.

The application of SALN to the dry intrusions dataset consistently provided a small speedup across multiple runs for simple convnet models and a greater speedup for more complex U-Nets. This increased speed, including the runtime for SALN as well as training time and occurred whilst improving the models train, test and validation losses.

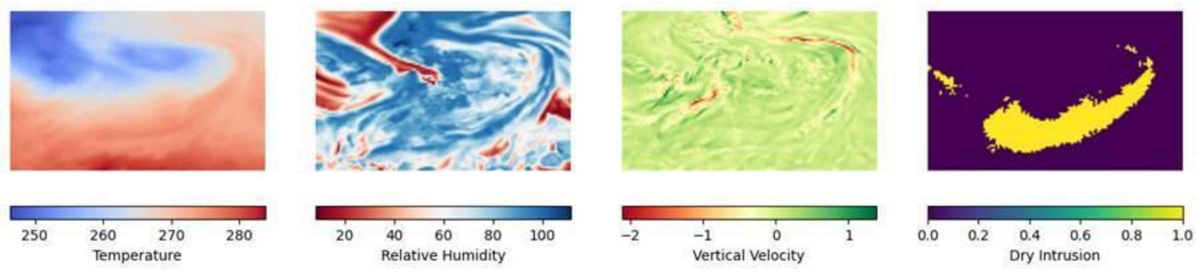


Figure 19. A sample of inputs and output for the dry intrusion models

	Average Val loss (10 runs)	Time for 10 runs
ConvNet	0.1965	2m44s
SALN ConvNet	0.1631	2m32s
Inverse SALN ConvNet	0.1638	2m35s
U-Net	0.1717	3m41s
SALN U-Net	0.2033	2m30s
Inverse SALN U-Net	0.1610	2m44s

Table 1. A comparison of models using SALN and inverse SALN. A run here is both SALN batching and a training run. Losses are averaged over and time is the total for 10 runs.

Notably, both SALN and inverse SALN improve convnet performance, both in training time and validation losses. Only the SALN U-Net appears worse and it in fact never met the threshold to predict any dry intrusion. It is unclear why both versions of SALN should offer such similar benefits to the convnet. The code was all modular, i.e. the SALN U-Net used the same code as the approaches, no bugs were found, and it remains unclear why it performs so much worse.

To check that these improvements were not trivially achievable by removing data the models were also run with randomly discarded data. An average validation loss of 0.1973 for the convnet and 0.1932 for the U-Net were found. This means that it wasn’t simply data removal being beneficial. As this was time series data there was a potential that the distribution shift could play a role. Training only on the data closest to validation and test data provided an average validation loss of 0.1995 for the convnet and 0.1751 for the U-Net. This suggests that SALN is doing something reasonable for this dataset.

By comparing mean fields of the model output to the mean fields of the target, the effects of SALN can be more clearly seen.

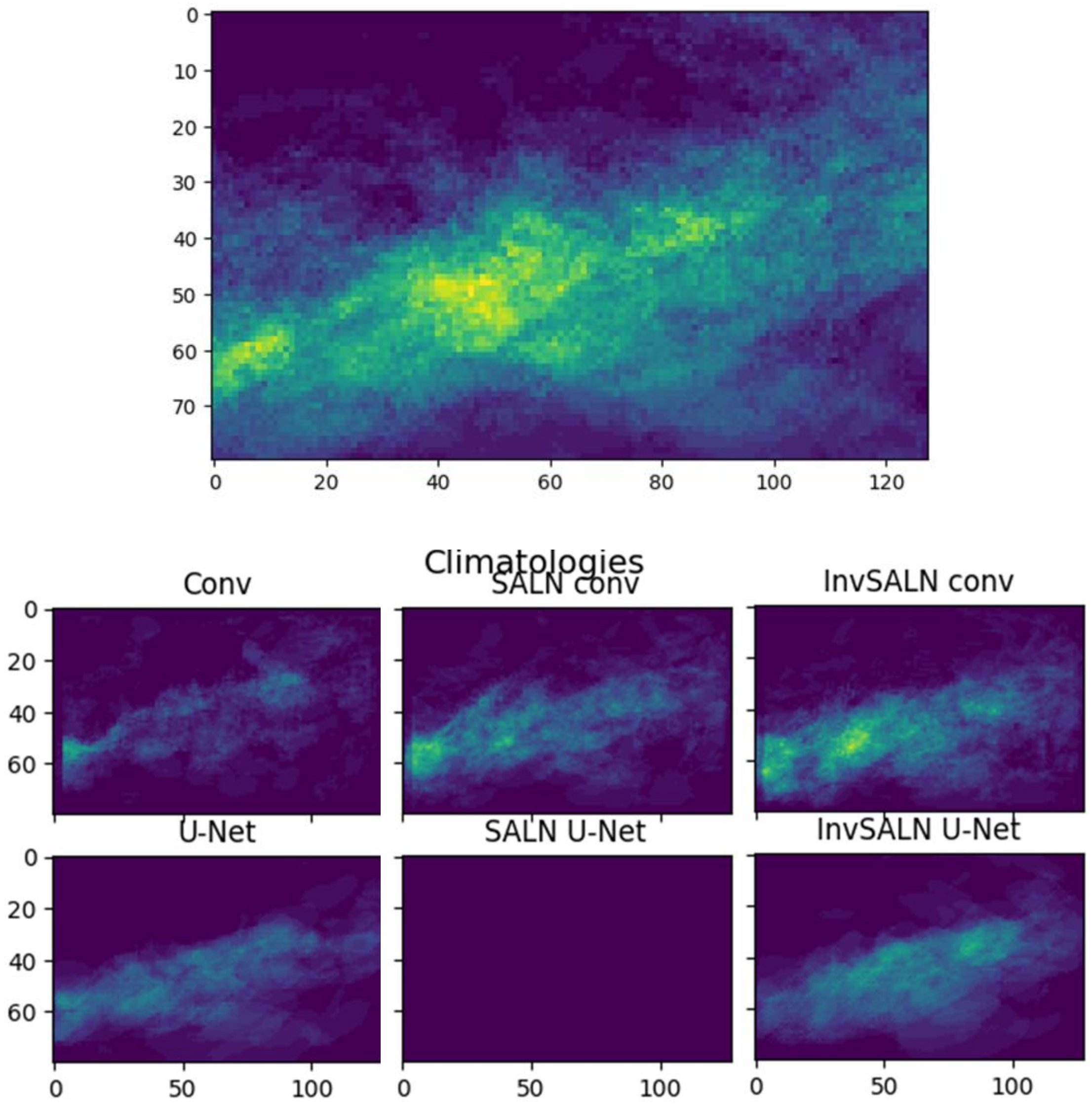


Figure 20. Comparing average prediction for each model against the average truth

In Figure 20 it can clearly be seen that the SALN U-Net has no skill, however all the other models correctly represent the large-scale patterns in the mean fields. The SALN convnets look closer to the truth especially in intensity whereas it is less convincing for the U-Net.

The above has been looking into SALNs application to dry intrusion identification, and while this is a meteorological use case, it is significantly different to a weather prediction system. SALN was applied to FastNet for a better idea on its applicability to WeatherGenerator. The training time nearly halved for one run configuration, however the resulting model did not perform as well under any metric and when including the time it took SALN to run had taken longer overall. The larger overall runtime is likely due to SALN not having been parallelised in the same way that the training code has, overall the GPU time is decreased by about a quarter. SALN should be parallelisable and so a wall clock speedup should be attainable. The validation MSEs for these were 0.0140 for the regular run and 0.0142 for SALN. Inverse SALN performs almost identically to regular SALN, more runs would need to be done to see if there are any discernible differences. On a more fully featured run SALN was much worse with a validation loss of 0.0034 compared to 0.0027 and taking 85% of the time. This means that at a higher scale SALNs speedup decreased and its losses became less comparable to a standard training.

7.4 Understanding SALN

Whilst SALN improves losses and training times for some tasks it does not for the most similar task to WeatherGenerator. With an understanding of what SALN is doing, it is possible that adjustments could be made to enable its benefits. To investigate SALNs effects, further tests were performed.

Firstly, as the dry intrusions dataset is a time series and SALN selects data by similarity, it is possible that it could introduce significant temporal clustering. To see how SALN handled this it was given unshuffled batches. By looking at SALNs importance rankings it can be seen how batch members get culled under different filter ratios, and this should reveal any potential temporal clusters being selected. The results of this test did indeed indicate some temporal clustering, although the selected sub-batches were also interspersed with non-adjacent samples, indicating that cosine similarity function is not only identifying temporal correlations.

Inverse SALN works in the opposite way, taking the most unique data. The effects on the temporally ordered batches are, however, similar. Inverse SALNs most important members are less often in adjacent pairs and tend to have pairs appear later in importance than for SALN. The groups important to SALN are then seen in Inverse SALNs least important ratings. SALN and Inverse SALN are not strongly identifiable by this, especially with non-extreme filter ratios.

To explore how the SALN-selected batches differ from the original batches, difference maps between SALN and original batches were inspected. It could be expected that since SALN looks for similar data, it may lead to regions with large differences to the original batches, however, visual inspection revealed no obvious systematic differences between SALN-selected sub-batches and randomly selected sub-batches.

Comparing histograms of the original random batches to the SALN and Inverse SALN batches shows that Inverse SALN tends to include more of the extremes than SALN for the same filter ratio. These histograms also show that, overall, the distributions remain similar to the original apart from the representation of extremes.

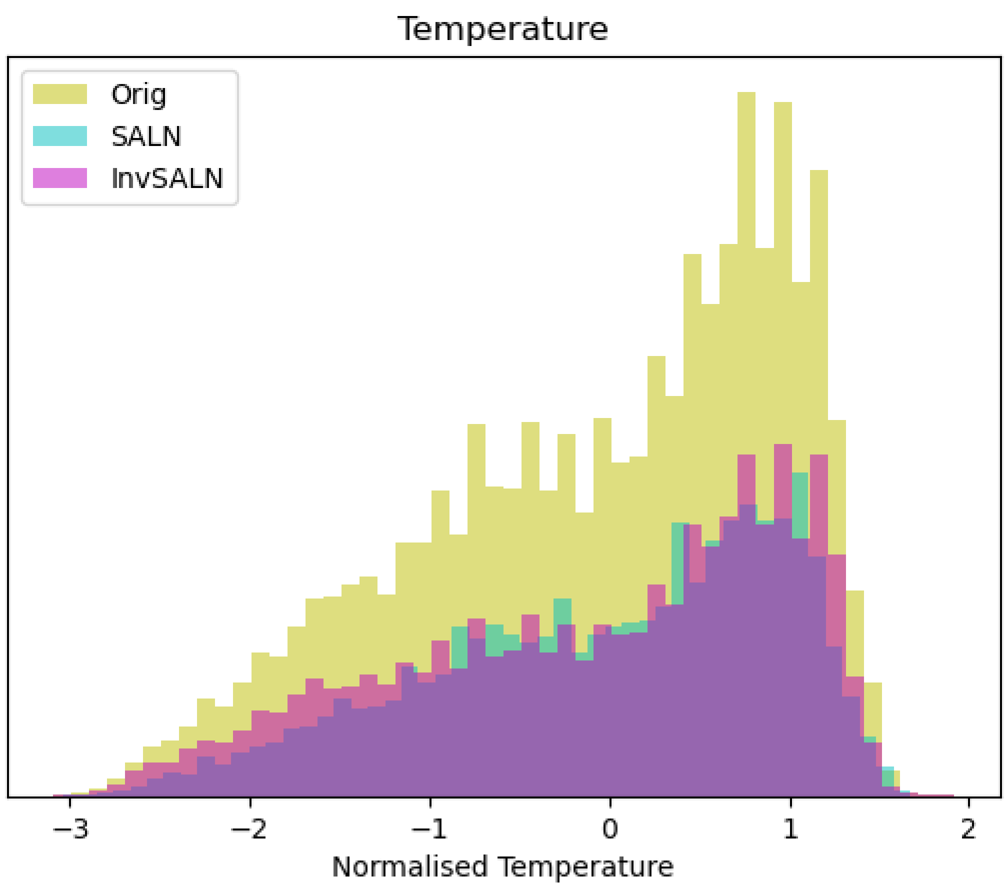


Figure 21. A histogram showing a random batch with the SALN and Inverse SALN filter ratios at 0.5.

7.5 Discussion

The reasons for SALNs efficacy on different tasks remain uncertain, further work must be done to understand exactly why it is working for these tasks but not others. Data complexity likely plays a role. SALN shows no benefit when applied to the Cifar10 dataset with an image resolution of 32x32, but it does for the Oxford III Pet (M. Sharifi, 2024) and dry intrusions datasets which have much higher resolutions. One reason for this may be that SALN works particularly well for classification and problems which use convolution when there is enough information to select meaningful batches. Whilst that appears true for problems of intermediate complexity, within increasing complexity, such as for the FastNet use case, SALN performance degraded proportionally increasing training time and validation losses. One possible reason for this is that the learning target in FastNet is the temporal increment of the forecast variables, rather than the forecast variables themselves. However, a test where SALN was applied to the increments still showed no benefits from SALN on FastNet training.

Dohmatob et al., (2025) provide a theoretical framework for when using less data is optimal for machine learning. If this framework is applicable to regression-based tasks it offers an explanation of SALNs utility. They propose that with a good quality generator/model and data selector/pruning oracle keeping only hard-to-learn data is best. When the model is weak and the oracle is good, a less aggressive pruning of the hard data and keeping only the easy data works best. This aligns well with the intuition of the SALN process, SALN selects the most similar data, making it easier to learn, and Inverse SALN selects the most unique or hardest data. This then raises the question of how we would assess the quality of our model and pruner for this framework. The core assumption of SALN is that the cosine similarity is a good measure of similarity which may not be the case for weather data or some specific configuration of it.

The speed up for training could, however, still be useful should training on the same dataset be required. Any time there is a change of inputs SALN will need to be run again with the new data but if model parameters are changed SALN will still apply. This could make SALN an effective tool for hyperparameter sweeps or testing, providing that SALN batches result in similar hyperparameters and testing results to the regular batches.

Whilst SALN is generic in its applicability, only requiring that you can vectorise the data, it only works when the data is the same size and type. This means that SALN is inappropriate for being applied directly to multimodal data and would have to be applied separately to different sources. This would also mean batches would have to be stitched together using groups that were selected as SALN batch cohabitants across data sources, relinquishing any idea that groups from different sources have benefit to each other. If different sources are embedded into a single space before training, SALN should be able to be applied to the embedding space. There is the potential for SALNs informational analysis to perform better on embedded or compressed data as there may be higher information density. An alternative to SALN would be Google's JEST which is explicitly made for use on multimodal data (Evans et al 2024), which may be addressed in future works.

The fact that SALN and inverse SALN both perform similarly is interesting when they identify importance in opposite ways. They both appear to be doing something more useful than just removing or skewing data as tested on the dry intrusions dataset. Explainability-based tests could help to understand how SALN selection relates to physical processes. An intuitive argument can be made for both directions containing required information with less connected data containing extremes and less likely scenarios, whilst the most connected data contains broadly useful information, this seems to be confirmed to a degree with *Fig 21*. This also raises the possibility for a combined SALN which takes some combination of both directions. This could come in the form of a single batch or some batch scheduling to ensure the model remembers extreme scenarios with inverse SALN batches. A brief test showed that a singular combined batch of 85% most connected and 15% least connected with a buffer to avoid particularly noisy data performed slightly better than all other SALN configurations. There is also the possibility of using inverse SALN as a method for selecting batches for fine tuning on unique data.

7.6 Conclusion

There is much more that could be investigated with SALN, we are however cautious to recommend much further experimentation on it without first being sure on its applicability to WeatherGenerator specifically. This is especially true for how it would potentially handle multimodal data. The tasks it worked for were mostly classification, and whilst one use case it worked for was meteorological it did not work particularly well for the most similar task, FastNet. JEST looks to be a tool that would be more certainly useful; this would require development more closely tied to the model training as it uses the model state to select batch members.

References

1. B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli, and A. Morcos. "Beyond neural scaling laws: beating power law scaling via data pruning. Advances in Neural Information Processing Systems", 35:19523–19536, 2022.

2. Y. Ye, Z.Huang, Y. Xiao, E. Chern, S. Xia, and P. Liu. “Limo: Less is more for reasoning”. arXiv preprint arXiv:2502.03387, 2025.

3. N. Muennighoff, Z.Yang, W. Shi, X.Li, L. Fei-Fei, H. Hajishirzi, L. Zettlemoyer, P. Liang, E.Candès, and T. Hashimoto. “s1: Simple test-time scaling”. arXiv preprint arXiv:2501.19393, 2025.

4. M.C. Welle, P. Poklukar, and D. Kragic. "Batch curation for unsupervised contrastive representation learning." *arXiv preprint arXiv:2108.08643* (2021).

5. T. Evans et al. "Data curation via joint example selection further accelerates multimodal learning." *Advances in Neural Information Processing Systems* 37 (2024): 141240-141260.

6. M. Sharifi, "Optimizing Data Curation through Spectral Analysis and Joint Batch Selection (SALN)." *arXiv preprint arXiv:2412.17069* (2024).

7. E. Dohmatob, M. Pezeshki, and R. Askari-Hemmat. "Why Less is More (Sometimes): A Theory of Data Curation." *arXiv preprint arXiv:2511.03492* (2025).

8 Conclusions

This deliverable demonstrates that dataset compression and batch optimisation are promising tools for enabling scalable deep learning within the WeatherGenerator project. Early experiments with convolutional autoencoders show that substantial data reduction - exceeding 20:1 ratios - is achievable while preserving high-fidelity atmospheric reconstructions. Complementary work using VQGAN for high-volume remote-sensing data confirms that neural compression can outperform traditional codecs, offering significantly higher compression ratios with controlled reconstruction error.

Model training experiments using JPEG-2000 compressed data further show that ocean-forecasting models remain robust under moderate to strong compression, particularly when data are down sampled at half-resolution, enabling large reductions in storage and GPU memory requirements with minimal loss of predictive skill.

The use of error-bounded compressed data to initialise trained models for inference also demonstrates that compression ratios up to 60x can be applied with only a small (<1%) increase in forecast error, further suggesting that raw NWP model or reanalysis data contains significant redundancy in the data.

Batch-curation techniques such as SALN can accelerate training and in some cases improve performance, though results vary widely with task complexity and are not yet reliable for WeatherGenerator-scale prediction.

Overall, the work establishes a flexible experimental pipeline, provides initial evidence that substantial data reduction is feasible, and identifies key research directions to support future scaling of WeatherGenerator training.

Document History

Version	Author(s)	Date	Changes
1.0	Sam Denton (UKMO), Tom Dunstan (UKMO), Italo Epicoco (CMCC), Sebastian Hoffmann (MPG) Sam Madge (UKMO), Reza Shatery (CMCC)	13/01/2026	Initial version
1.1	Tom Dunstan	26/01/2026	1st revision
1.2	Tom Dunstan	28/01/026	2nd revision
1.2	Tanya Warnaars	29/01/2026`	3rd edits
1.3	Sam Denton (UKMO), Tom Dunstan (UKMO), Italo Epicoco (CMCC), Sebastian Hoffmann (MPG) Sam Madge (UKMO), Reza Shatery (CMCC) Jiayong Li (ETHZ) Langwen Huang (ETHZ) Alexandru Calotoiu (ETHZ) Torsten Hoefler (ETHZ)	03/02/206	Added section on error-bounded compression for inference initialisation
1.3	Tanya Warnaars	09/02/2026	Fixed formatting issues

Internal Review History

Internal Reviewers	Date	Comments
Marieke Wesselkamp	19/01/2026	
Cristian Lussana	19/01/2026	