Report Neural Networks

Fabio Fraschetti 1834942

October 2024

1 Introduction

In this notebook I tried to replicate the results of the paper "ClimateLearn: Benchmarking Machine Learning for Weather and Climate Modeling" where I try to forecast 3 features (climate variables): the temperature at 2 meters (t2m), the temperature at 850 hPa (t850) and the geopotential at 500 hPa (g500). I also compared different forecasting approaches: Direct, Iterative, and Continuous.

2 Dataset

To perform this task I downloaded the following data from ClimateBench: 2m temperature, temperature 850, geopotential 500, relative humidity, toa incident solar radiation, temperature , geopotential. That are hourly data from reanalysis of climate model. I joined all this data in a tensor of shape (18000, 13, 32, 64) where 18000 are the number of hours (about 2 years), 13 are the channels of the features presented above (temperature and geopotential have 3 different layers at [50, 600, 925]). While 32 and 64 are the shapes of the world map, so the world map is divided into a matrix of 32*64.

3 Input and output to the model

Starting from the above tensor I created a time series to feed the model, for each current hour (H) I add H-6 hour and H-12 hour. All the features of this hours are added on the second dimension of the tensor. Resulting into a tensor of shape (18000, 39, 32, 64) and in the end I added 3 constants features (latitude, land-sea mask and orography) always on the second dimension. Hence the final shape of the input tensor is (18000, 42, 32, 64). While for the output I took the values of the actual hour (H) plus 24 hours. In the end I divided the tensors into train, val and test set with percentage respectively 0.7,0.15,0.15.

4 Models

I developed 4 architectures for this task and each architecture is trained to forecast the t2m, t850 and g500 separately resulting in 12 models. The 4 architectures: ResNet of the paper, ResNet simplified, UNet and VIT. The first **ResNet** is based on the architecture summarized in the paper. The overall architecture starts with the periodic padding and an initial convolutional layer with a large kernel size to capture broad spatial patterns. Then there are batch normalization, relu function and dropout. After this there are 28 residual blocks, each one constitute by a convolution layer, batch norm, relu, another convolution layer, another batch norm, sequential, relu and dropout. Then there is a pool average to transform map into one value. In the end the network uses a fully connected layer to output the final prediction on a 32*64 grid, corresponding to the target spatial resolution of the climate data. In the second architecture, I attempted to create a lighter version of **ResNet** by eliminating the periodic padding and reducing the number of residual blocks from 28 to 2. Resulting in a faster model, in fact, the time to train all the features of the first network is 5.15 hours and for the second one is 45 minutes. The third model that I developed is a **UNet** composed by and encoder and decoder:
Encoder (Downsampling): The encoder progressively reduces the spatial resolution of the input while increasing feature depth. Channels increase from 64 to 512 through the encoder.
Bottleneck: This part sits between the encoder and decoder and consists of a convblock with 1024 output channels, capturing the most abstract features at the smallest spatial resolution.
Decoder (Upsampling): The decoder progressively upsamples the feature maps back to the original spatial dimensions. Channels reduce from 512 back to 64 as the spatial resolution increases. A final 1x1 convolution is applied to reduce the feature maps to the desired number of output channels. The last Model that I developed is a VIT composed by:
PatchEmbedding: This layer divides the input image into smaller patches (4x4 in this case) and applies a convolution to each patch to generate a patch embedding. It includes a class token, which is used for final prediction, and a positional embedding, added to retain the order of the patches.
MultiHeadSelfAttention: This implements multi-head self-attention, where the model learns relationships between different patches. The output is a weighted sum of values.
TransformerBlock: A basic building block for the transformer architecture. Each block consists of: Multi-head self-attention to learn patch relationships. Layer normalization before and after the attention and feed-forward layers. Feed-forward network (MLP) with two fully connected layers and

GELU activations, followed by dropout.
VisionTransformer: Is the main model, which consists of: PatchEmbedding to convert the image into a sequence of patch embeddings. A series of Transformer blocks to process the patch embeddings. A LayerNorm for normalization of the output. A fully connected layer to make the final prediction The final output is reshaped into a grid (32x64).

5 Training

I used as loss function the Root Mean Square Error (RMSE) and as Anomaly Correlation Coefficient (ACC) where $\hat{X}' = \hat{X} - C, \quad X' = X - C$ and $C = \frac{1}{N}\sum_k X$:

$$RMSE = \frac{1}{N}\sum_{k=1}^{N}\sqrt{\frac{1}{H \times W}\sum_{i=1}^{H}\sum_{j=1}^{W} L(i)\left(\hat{X}_{k,i,j} - X_{k,i,j}\right)^2}$$

$$ACC = \frac{\sum_{k,i,j} L(i)\hat{X}'_{k,i,j}X'_{k,i,j}}{\sqrt{\sum_{k,i,j} L(i)\hat{X}'^2_{k,i,j}}\sqrt{\sum_{k,i,j} L(i)X'^2_{k,i,j}}}$$

where $L(i) = \frac{\cos(H_i)}{\frac{1}{H}\sum_{i=1}^{H}\cos(H_i)}$. with optimizer Adam and a warmup of 5 epochs. If the RMSE of the val set doesn't decrease for 5 epochs, the training stops. The batch size is set to 32 because I see better results respect to 128 used in the paper.

6 Results

We have already seen that my ResNet is faster then the one presented in the paper, Now let's analyze the results. As we can see from Table 1 and Table 2 that both the RMSE and ACC for the temperatures (t2m and t850) are improved in my model, while for g500, my architecture performs slightly worse. I have also included the graphs for each ResNet model(Figure 1-6), focusing on the cell that roughly represents Europe (2 , 26 coordinates). But compared to the UNet model both perform very bad. As we can see from the tables and Figure(7,8,9) the Unet model reaches a very good prediction with 2 degrees of mean error in all the data in the world and also the g500 achieve a good result, that in general is the feature that is more difficult to predict. While the VIT model is the one that perform worse also respect to the ResNets.

	t2m	**t850**	**g500**
ResNet Paper	0.0941	0.1167	25.2514
ResNet Mine	0.0645	0.0842	25.3990
UNet	0.0209	0.0225	2.7658
VIT	0.1080	0.1171	28.0227

Table 1: RMSE Results

	t2m	**t850**	**g500**
ResNet Paper	0.8693	0.7239	0.6529
ResNet Mine	0.9393	0.8670	0.6401
UNet	0.9938	0.9911	0.9965
VIT	0.8206	0.7198	0.5358

Table 2: ACC Results

7 Continuous Vs Iterative Forecasting

Until now I predicted only one step forward for each model and so now I use UNet model, that is the model that performs better, for applying iterative and continuous forecasting on the t2m feature. In iterative Forecasting, the model predicts one step ahead, and then that prediction is fed back into the model as an input to predict the next step. This process is repeated for as many steps as needed. While before I used the model to predict a single feature at time, now I need to predict all the features to put into the model. In Figure 13 we can see that 3 lines the blue one is the temperature predicted based on the real values while the green line are the values predicted based on the blue one. Consider that the model performs worse because here we predict all the 13 features. While in Continuous Forecasting usually refers to models that make forecasts continuously over time without reusing predictions. In my model I predicted the next 24 hours has you can see from the Figure 14, always in the cell corresponding to Europe.

8 Conclusions

In the end I think I achieve a quite good result with UNet model. Considering that I used a tensor of 6GB respect to the paper that uses a tensor of about 80GB. I had to use a large amount of data because the initial models did not produce satisfactory results. At first, I had only included the t2m, t850 and g500 features, to predict the t2m, t850 and g500 the next day, but it was not enough, so I decided to include all the other features but reducing the length of the time series.

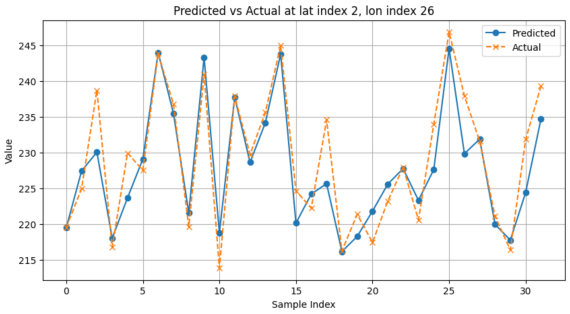

Figure 1: ResNet Paper t2m

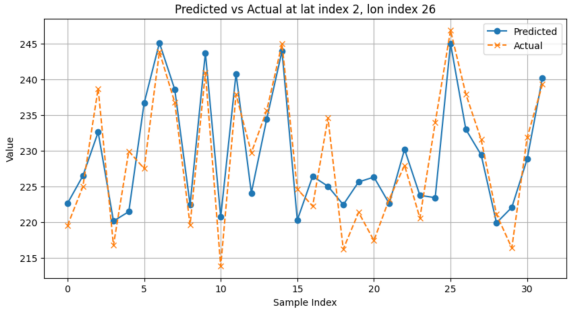

Figure 2: ResNet mine t2m

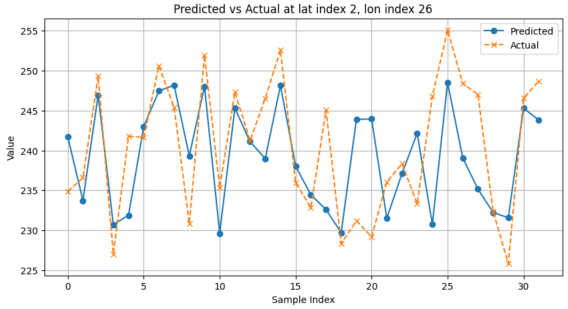

Figure 3: ResNet Paper t850

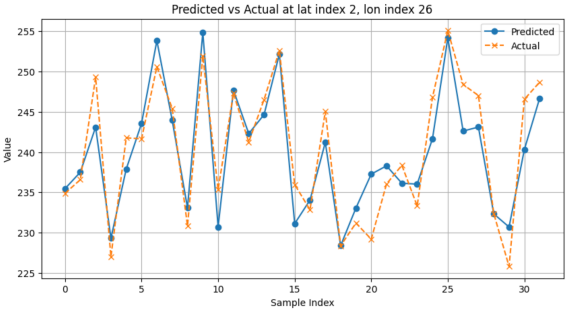

Figure 4: ResNet mine t850

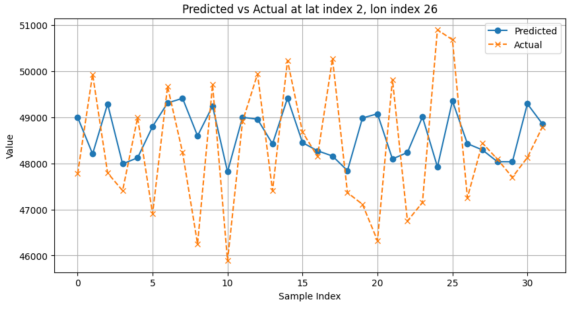

Figure 5: ResNet Paper g500

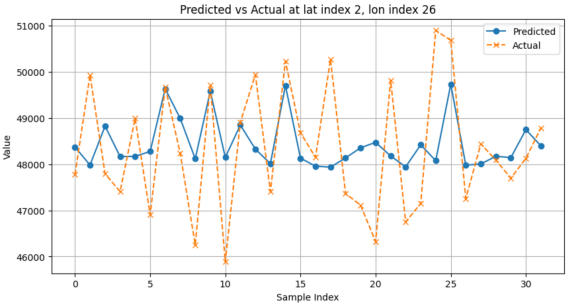

Figure 6: ResNet mine g500

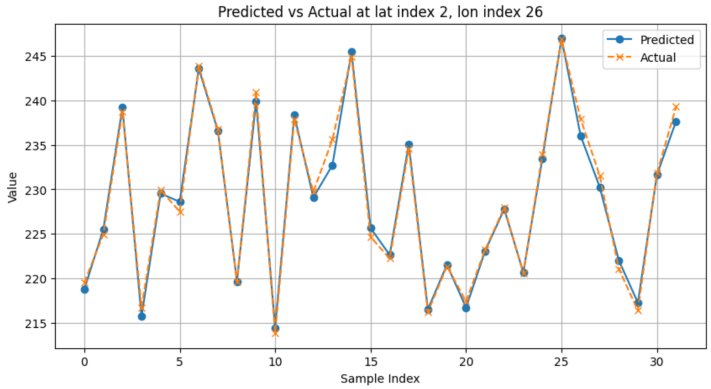

Figure 7: UNet t2m

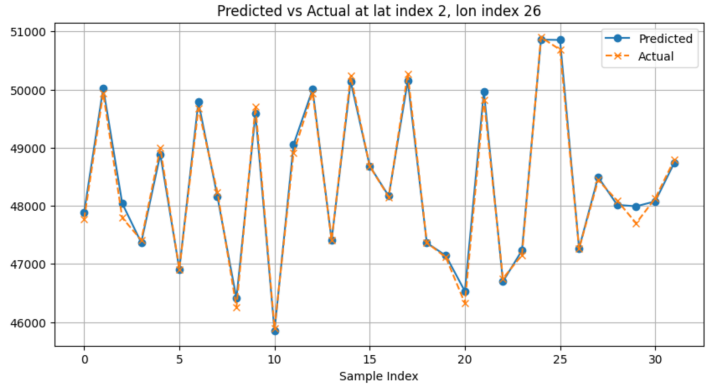

Figure 8: UNet g500

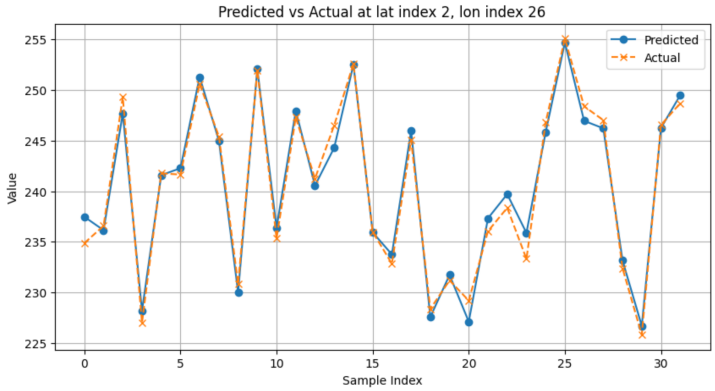

Figure 9: UNet t850

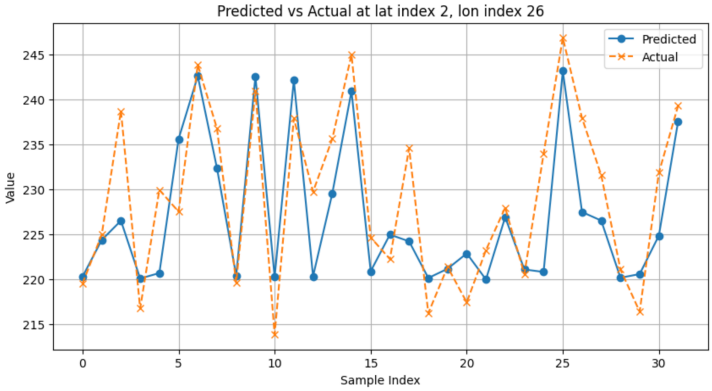

Figure 10: VIT t2m

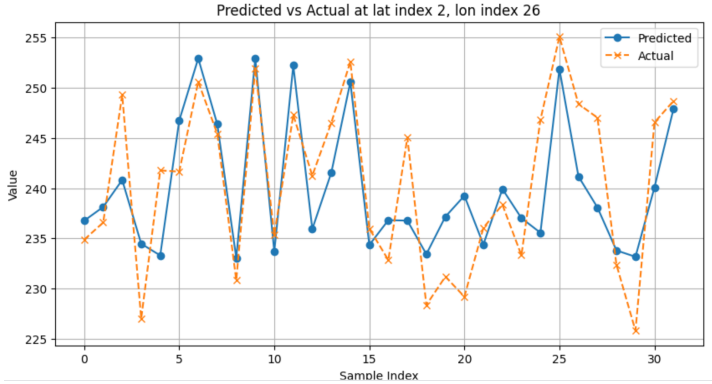

Figure 11: VIT t850

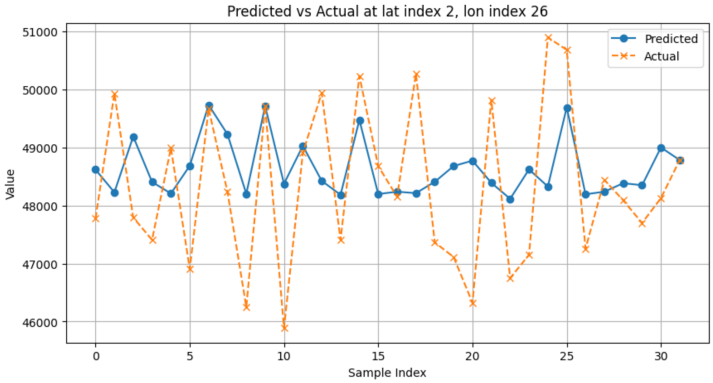

Figure 12: VIT g500

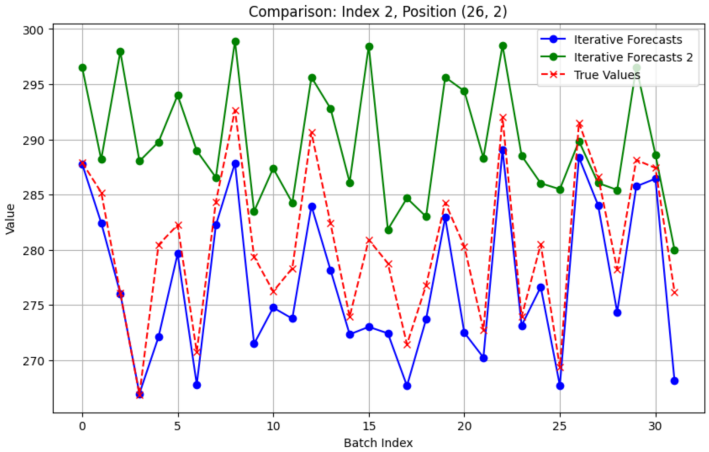

Figure 13: Iterative Forecasting

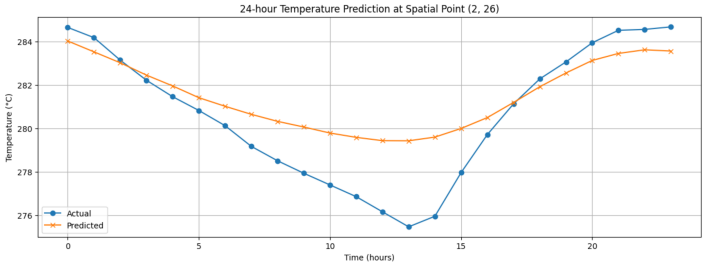

Figure 14: Continuous Forecasting