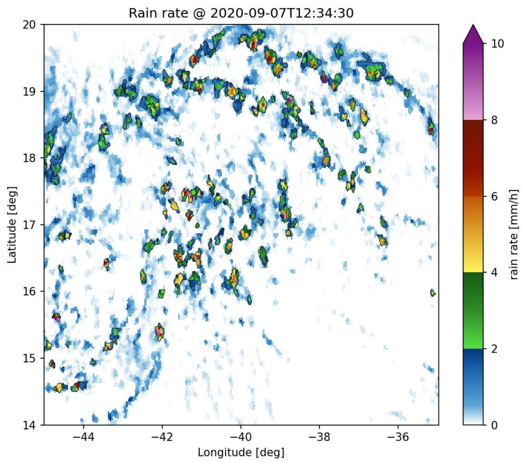
Project Report

Cloud Development Prediction

Matthias Faust
Jahne Schütz
Thorsten Köhler

Introduction



Data

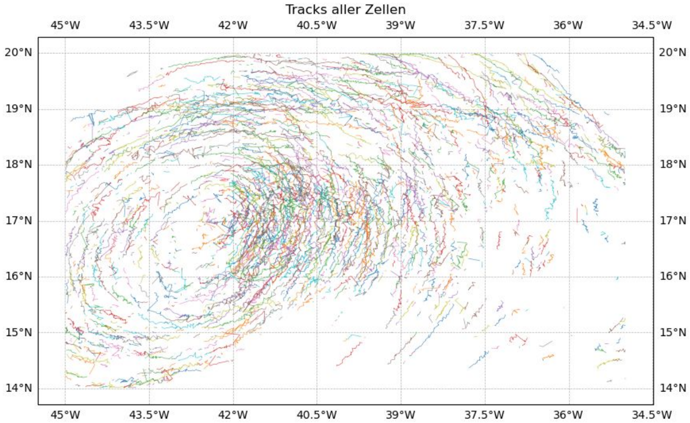
- Simulation of Atlantic hurricane Paulette (2020)
- Cloud features and tracks
- Splitting and merging events
- Vertical meteorological profiles

Prediction

- Lifespan
- Rain formation
- Position

How many timesteps are needed?

How many timesteps can be predicted?



Problem Overview

Original Dataset

- Several terabyte
- Weather model and tracking tool
- 800.000 cloud objects
- Data extraction not trivial

Used Dataset

- Each cloud stored as CSV file containing a 2D matrix
- Each row a timestep of meteorological variables
- Task type: Regression using RNN

Literature Review

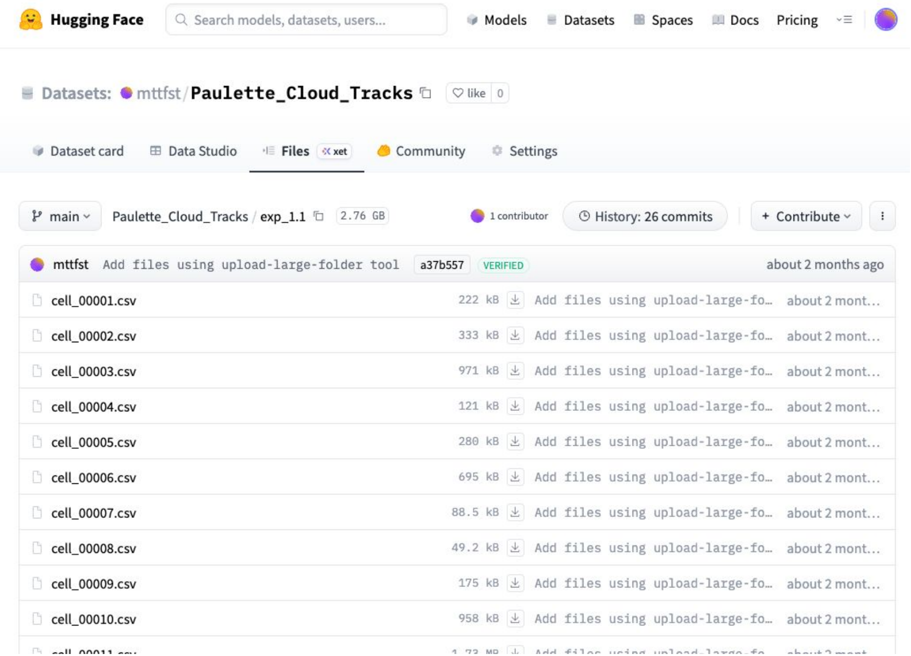
Highlights

- Two papers with similar data (vertical profiles) and papers with similar problems (e.g. prediction of ice formation)

Solutions

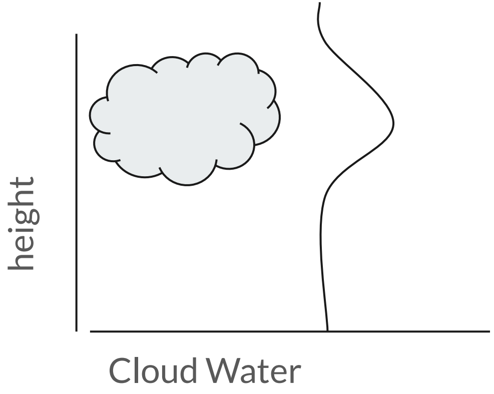
- RNNs, GRUs & LSTMs widely used for time series forecasting and climate modelling

Dataset Characteristics

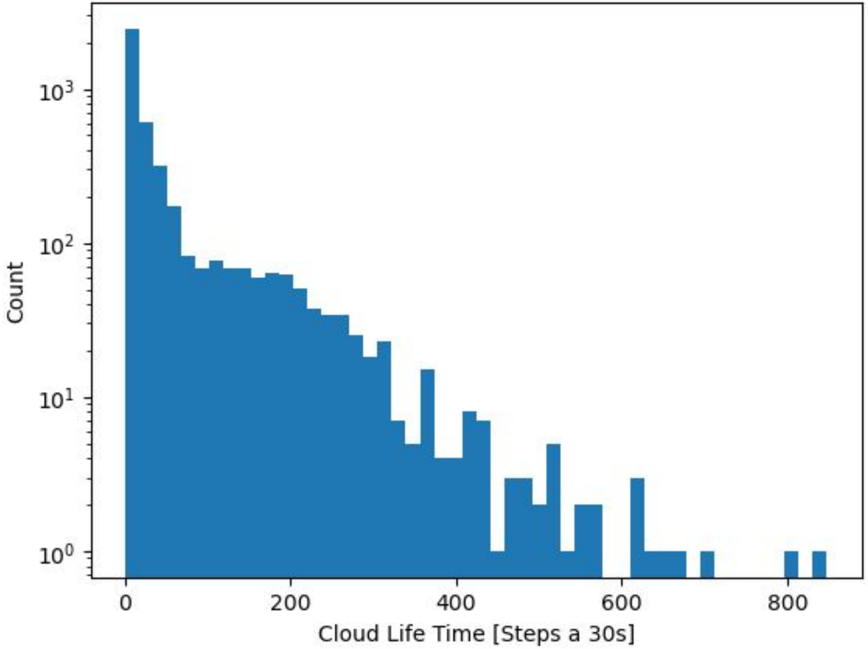


Data Structure

- 9000 individual cloud tracks
- Time series with a 30s timestep
- Meteorological data of the air column at cloud center



Dataset Characteristics



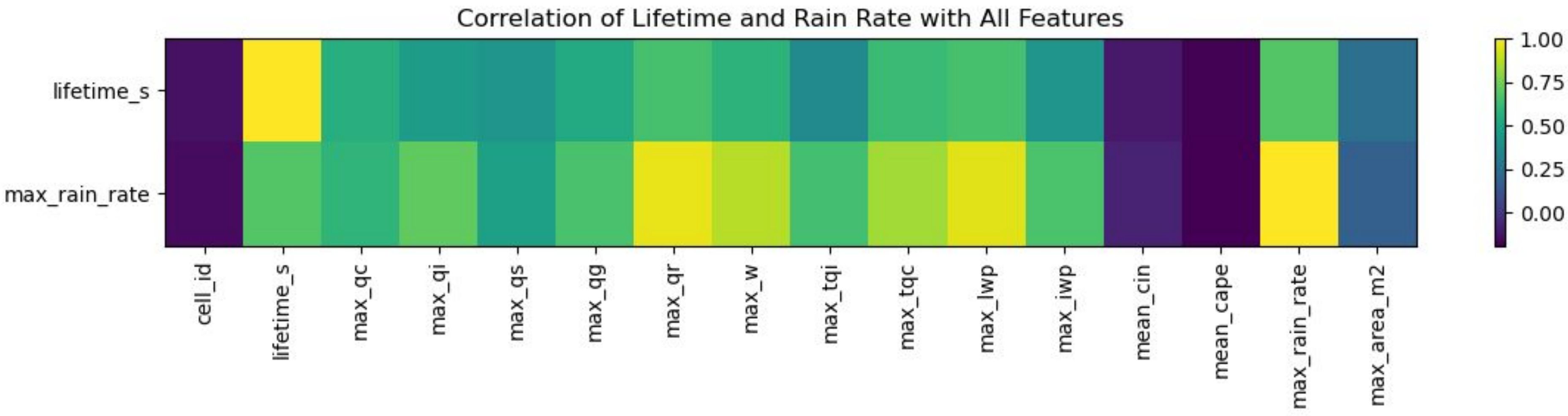
Data Properties

- Dataset skewed towards short living clouds (logarithmic scale!)
- No missing values

Dataset Characteristics

Feature Engineering

- Vertical profiles were flattened to min/max values



Baseline Model

```
=== Evaluation Task A (Snapshot Baselin
Train – MAE:  19.16 s
Train – MSE:  6571.95 s^2
Train – RMSE: 81.07 s
Val – MAE:  44.32 s
Val – MSE:  19937.47 s^2
Val – RMSE: 141.20 s
Test – MAE:  46.86 s
Test – MSE:  19430.32 s^2
Test – RMSE: 139.39 s
```

Random Forest

- From each track 3 random points selected to predict total track length
- Model predicts track length precisely to a couple of timesteps
  - Strong overfitting
  - Maybe biased by the skewed track length distribution

Model Definition and Evaluation

Model

- Start with baseline RNN
- Gradual increase in complexity
- Final goal: multivariate LSTM

Layer (type)	Output Shape	Param #
input_layer_4 (InputLayer)	(None, 665, 9)	0
not_equal_4 (NotEqual)	(None, 665, 9)	0
masking_4 (Masking)	(None, 665, 9)	0
any_4 (Any)	(None, 665)	0
simple_rnn_8 (SimpleRNN)	(None, 665, 64)	4,736
batch_normalizatio… (BatchNormalizatio…	(None, 665, 64)	256
simple_rnn_9 (SimpleRNN)	(None, 665, 32)	3,104
time_distributed_4 (TimeDistributed)	(None, 665, 1)	33

Model Definition and Evaluation

Evaluation

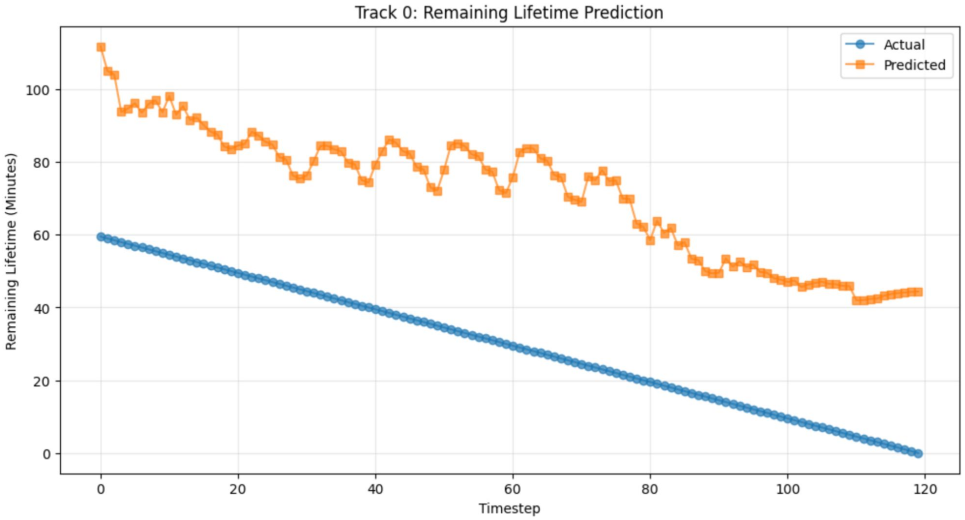
- MSE or RMSE usually used for regression tasks
- Masked loss to excluded padding timesteps
- Later: Customized loss & metric

Results

Training

- Constantly higher validation loss
- The model is overfitting.

Results



Track Forecast

- The model has understood the basic temporal concept.
- The model is systematically biased upwards.
- The model does not know any explicit position in the track.

Challenges

Data and Coding Infrastructure

- Sharing big dataset with the team
- Making the dataset available
- Working on Colab Kernel

Solution

- Hugging Face CLI for upload
- Hugging Face token
- VS Code with Colab Extension

Challenges

Many Short Lifespan Clouds

- Skewed data
- MSE better than MAE
- RMSE better for interpretation, but faster calculations with MSE

Solution

- Modify MSE loss function
- Use third or higher powers instead of second powers

Errors

NaN Losses during Training

- Vanishing/exploding gradients?
- Masking problem?

Solution

- Found a division by zero
- Treat constant variables correct in normalization

Discussion

Expectation Management

- We gradually learned what our dataset can realistically predict — and where its limits are.
- Initially, we assumed we could predict total lifetime, rainfall and even cloud positions.

Limitations

- The full potential of vertical profiles is difficult to leverage — the model mostly relies on averaged values.
- Using high-temporal-resolution data turned out to be too ambitious.

Plans before Submission

Sliding Windows

- Compare models with or without sliding windows
- Compare different lengths of sliding windows

Documented Comparisons

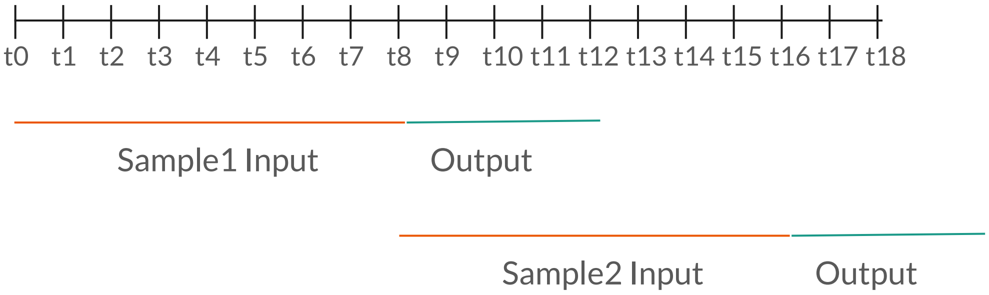
- Save hyperparameters and performance statistics automatically
- Modify hyperparameters automatically

Update: Custom loss metric

Evaluation

- Background: own loss function to penalize deviations more severely (see slide 10 - Model Definition and Evaluation)
- Approach: MSE basis used and power increased
  → The result was exploding gradients
- 2nd approach: hybrid approach, i.e., proportional combination of MSE and the custom loss function (see notebook: 3_Model/Prototypes/model_definition_evaluation_JS.ipynb)
  → The result did not improve so significantly that it would justify the effort.

Update: From Cloud Tracks to Training Samples



- Cloud tracks converted into overlapping temporal samples
- Fixed input window (history) and output window (forecast)
- Multiple samples extracted per cloud lifecycle
- Enables supervised sequence forecasting

Update: Final Multi-Task Forecasting Architecture

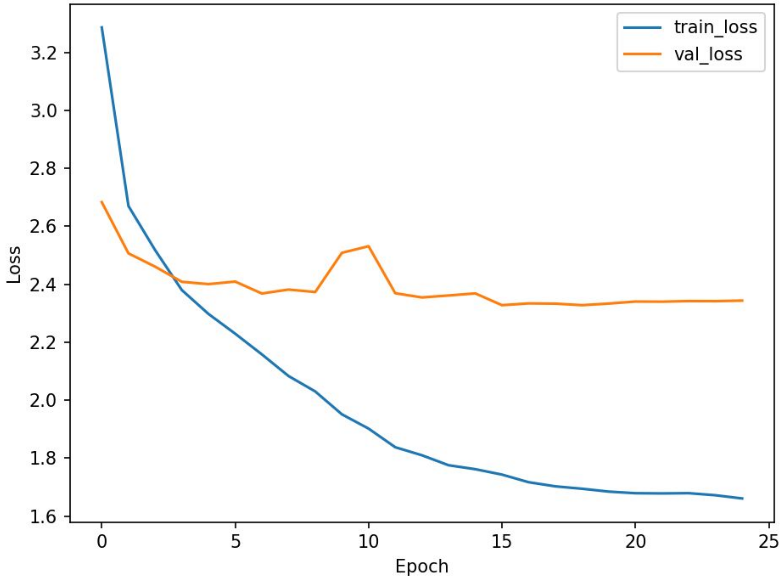
Model: "cloud_multitask"

Layer (type)	Output Shape	Param #	Connected to
X (InputLayer)	(None, 40, 27)	0	-
gru_1 (GRU)	(None, 40, 128)	60,288	X[0][0]
ln_1 (LayerNormalizatio…	(None, 40, 128)	256	gru_1[0][0]
gap (GlobalAveragePool…	(None, 128)	0	ln_1[0][0]
gmp (GlobalMaxPooling1…	(None, 128)	0	ln_1[0][0]
pool_concat (Concatenate)	(None, 256)	0	gap[0][0], gmp[0][0]
shared_dense (Dense)	(None, 64)	16,448	pool_concat[0][0]
shared_dropout (Dropout)	(None, 64)	0	shared_dense[0][…
cloud_base (Dense)	(None, 40)	2,600	shared_dropout[0…
rain_gsp_rate_L00 (Dense)	(None, 40)	2,600	shared_dropout[0…
tqc_L00 (Dense)	(None, 40)	2,600	shared_dropout[0…
tqi_L00 (Dense)	(None, 40)	2,600	shared_dropout[0…

Total params: 87,392 (341.38 KB)
Trainable params: 87,392 (341.38 KB)
Non-trainable params: 0 (0.00 B)

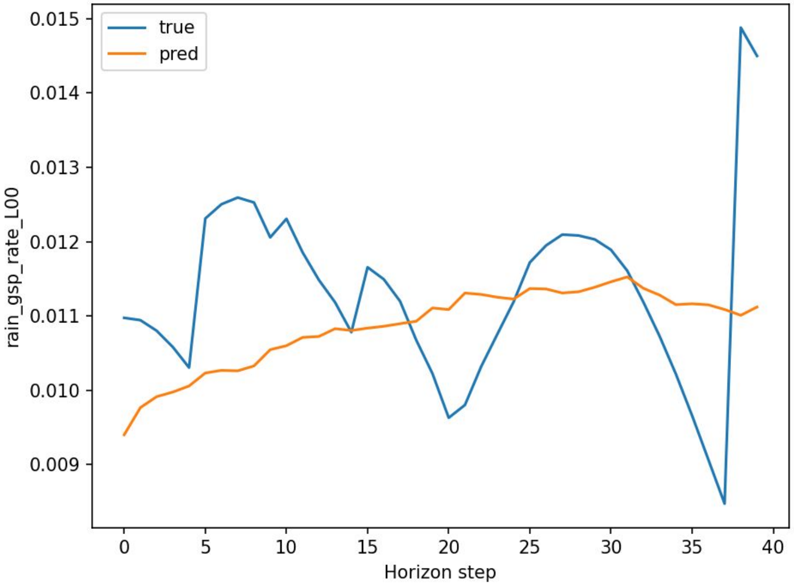
- Shared temporal encoder (GRU, 128 units)
- Layer normalization and pooled representation
- Separate prediction heads for rain, cloud base, TQC, TQI
- Architecture selected via systematic performance sweeps (~87 runs)

Update: Training Behaviour and Generalization



- Rapid learning during early epochs
- Validation loss stabilizes after ~5 epochs
- No instability or divergence observed
- Model reaches intrinsic predictability limit

Update: Example Forecast Simulation



- Model captures mean evolution of cloud properties
- Exact small-scale fluctuations are not reproduced
- Forecasts remain physically plausible
- Consistent improvement over persistence baseline

Update: Conclusions

- Cloud evolution is partially predictable
- Predictability timescale ≈ 20 minutes
- Motion dynamics provide strongest signal
- Temporal representation dominates performance
- Cloud systems predictable in tendency, not in detail

Thank you!

Questions?