SAPIEN

Skillful Atmospheric Predictions with IntelligEnt Networks

Primary Goal

LDcast on LUMI:

- Containerize the LDcast model and run it efficiently on a single GPU of LUMI.
- **Stretch Goal:** Adapt the code to run on multiple GPUs for scalability.

Scaling Up:

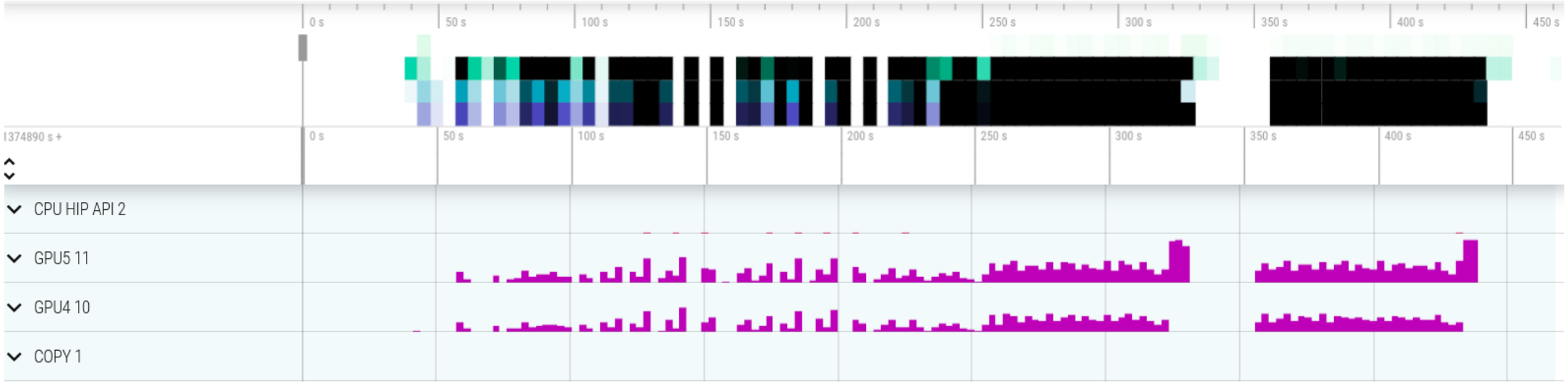
- Begin scaling LDcast across multiple GPUs, focusing on optimizing for LUMI's architecture.
- If time allows, explore deploying SHADEcast after achieving a scalable LDcast pipeline.

Achievements, day-by-day: Tuesday

Ran LDcast on **1 GPU** on LUMI in a container with rocm6.0, and profiled the run with rocprof

Achievements, day-by-day: Wednesday

Ran and profiled training runs on **2 and 4 GPUs**.



Achievements, day-by-day: Wednesday

Ran and profiled training runs on **2 and 4 GPUs**.

The GPUs seem to be all active and working at the same time.

BUT: we didn't find a significant decrease in training time
(on a very small training dataset and with very few training steps)

Achievements, day-by-day: Thursday

Ran and profiled training runs on multiple GPUs and on larger dataset + with more epochs and larger batch size

Found that the batch size that we specify is local (per GPU), so when scaling up we are also doubling the batches --> training time is then scaling down linearly if this is true (and we need to check if the loss goes down more quickly)

Challenges we faced

Perfetto issue on one GPU

- Solution: run perfetto locally with older version

Run on multiple GPUs

- Magic command #SBATCH --ntasks-per-node=2

Perfetto issues on multiple GPUs

- Created a script that stores each GPU run on different folder

Memory issues when increasing number of batches

- Magic command: accumulate_grad_batches=2

Goals that we accomplished

Familiarize with pytorch lightning on LUMI

Profiling on multiple GPUs

Familiarize with scaling up

Performance Optimization through Batch Size Adjustment

Setup: Initial configuration with 32 batches.

Optimization: Increased the number of batches from 32 to 64 using the "magic" command.

Results:

Epoch duration reduced:

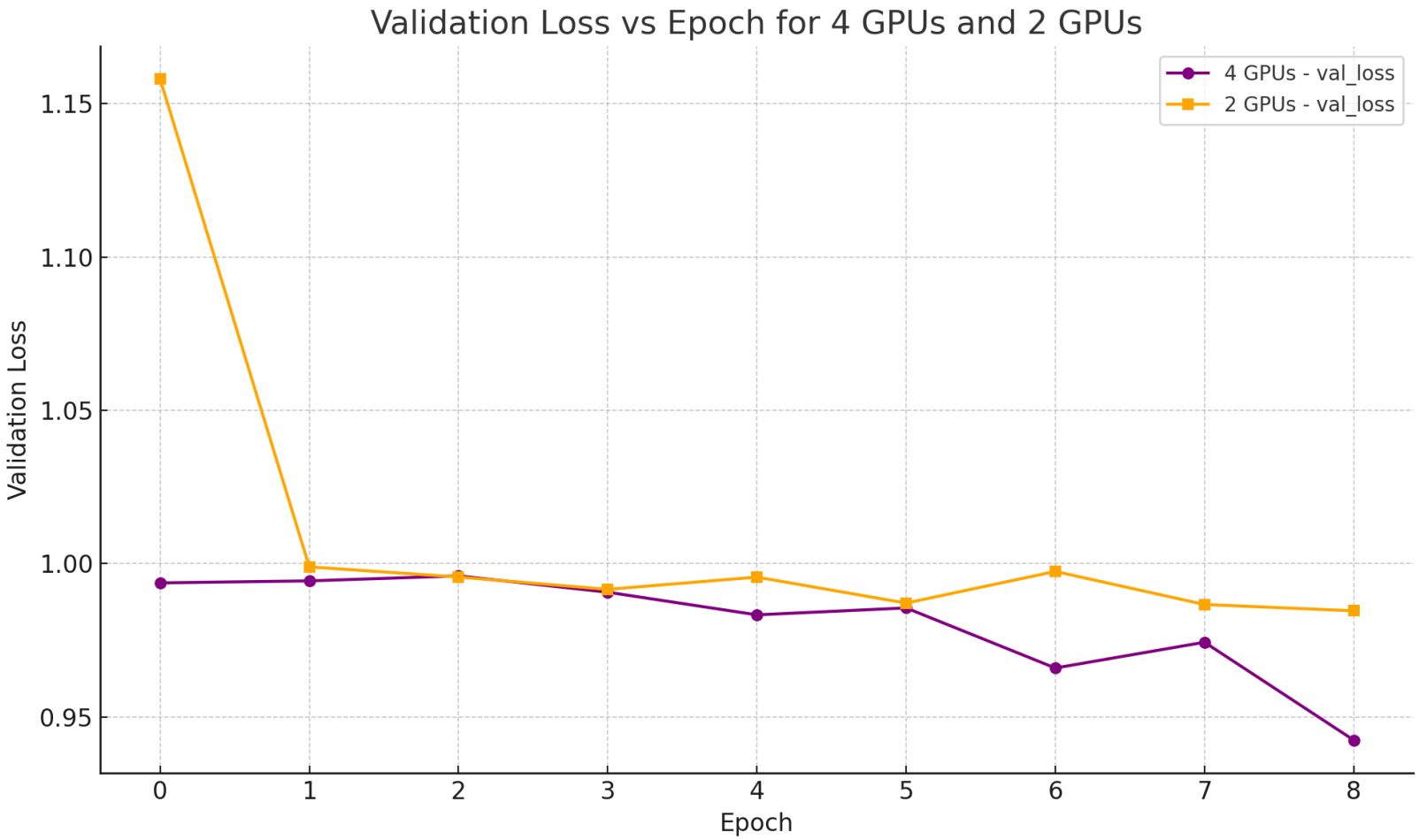
- From 72 seconds per epoch
- To 50 seconds per epoch

Impact:

Improved training efficiency.

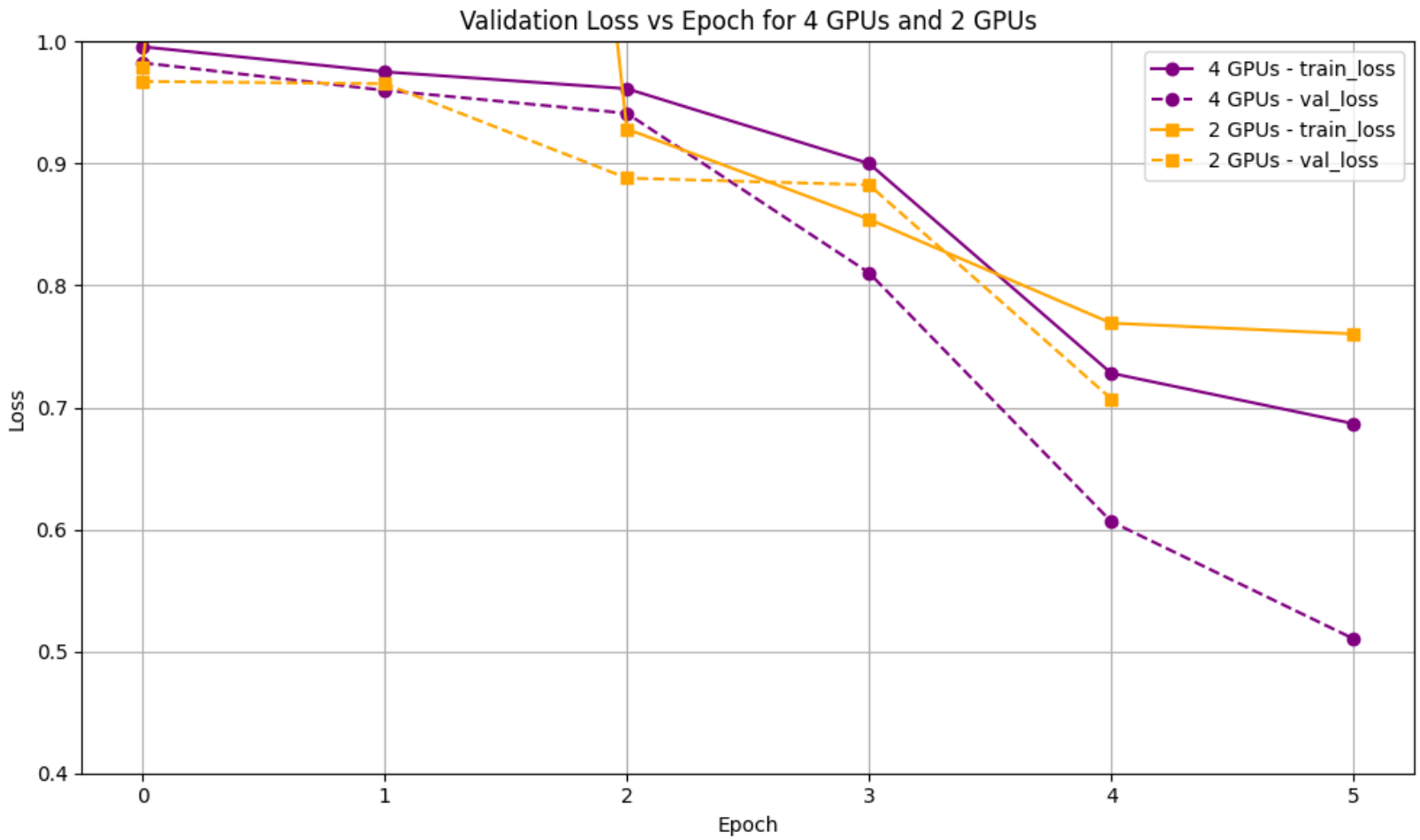
Significant time savings per epoch with a simple batch size change.

32 batches 1 month dataset 20 epoch steps

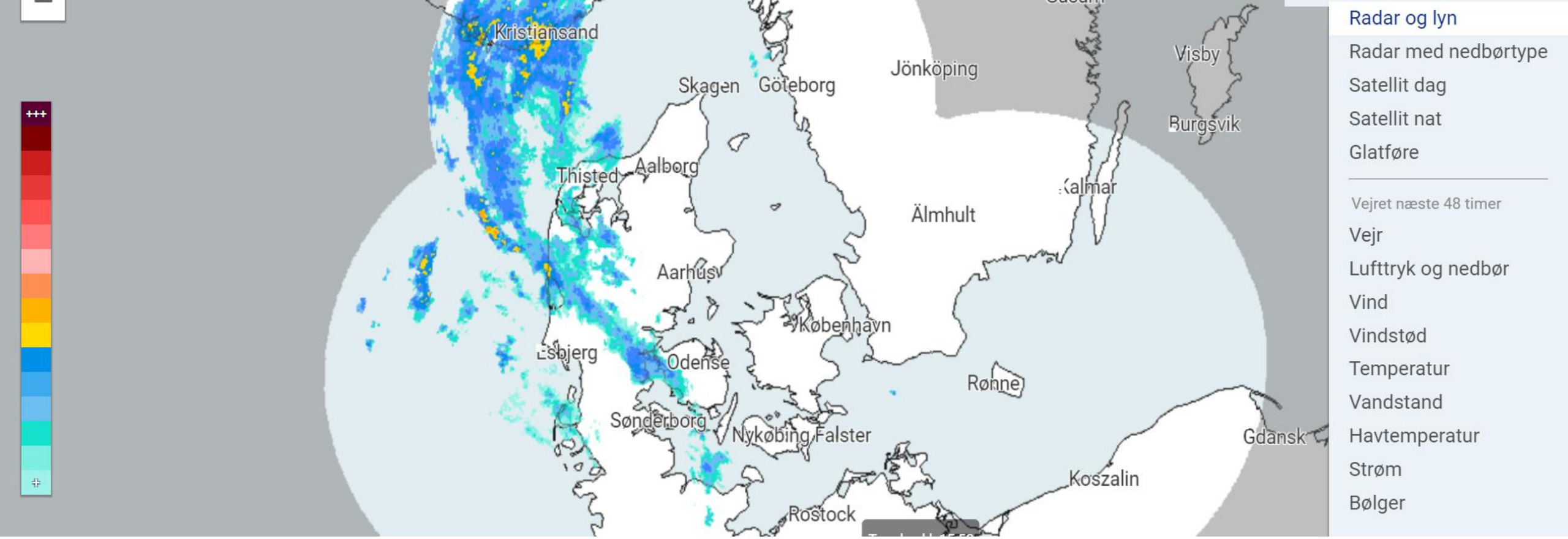


Training time:
~ 33 minutes both

32 batches 4-month dataset, 100 epoch steps



Training time:
~ 1 hour and 30 minutes both

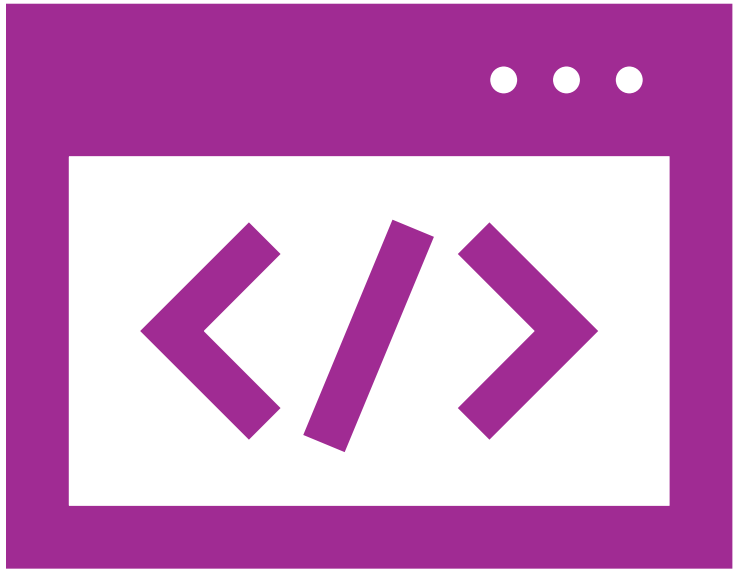


Next steps

Apply all we learnt to train LDcast on our full Danish radar dataset of 96 months!

Resources

- LDCast
  https://github.com/MeteoSwiss/ldcast
- SHADECast
  https://github.com/EnergyWeatherAI/GenerativeNowcasting



Thank you :)