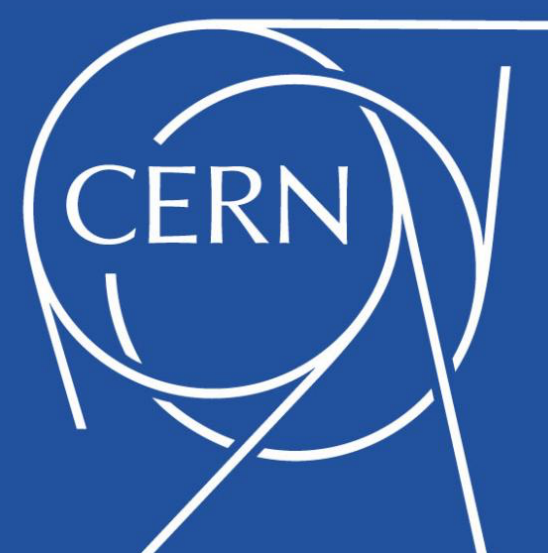


A CERN physicist's perspective on some of your questions and where Deep Learning could help

Maurizio Pierini



Your Questions

- ① *How to build confidence in raw data and any calculations carried on that raw data?*
- ① *How to detect sensor failures/anomalies?*
- ① *How to robustly delete data and have confidence that good data hasn't been deleted?*
- ① *How to limit access to different datasets generated by different groups?*
- ① *Techniques for intuitively linking different datasets together?*
- ① *Application of A.I. / Machine Learning for real time analysis of live data streams?*
- ① *Real time X-ray / C.T. / other scanning to allow us to view combustion processes.*
- ① *A.R. / V.R. techniques for data visualisation and manipulation*

What I will try to address

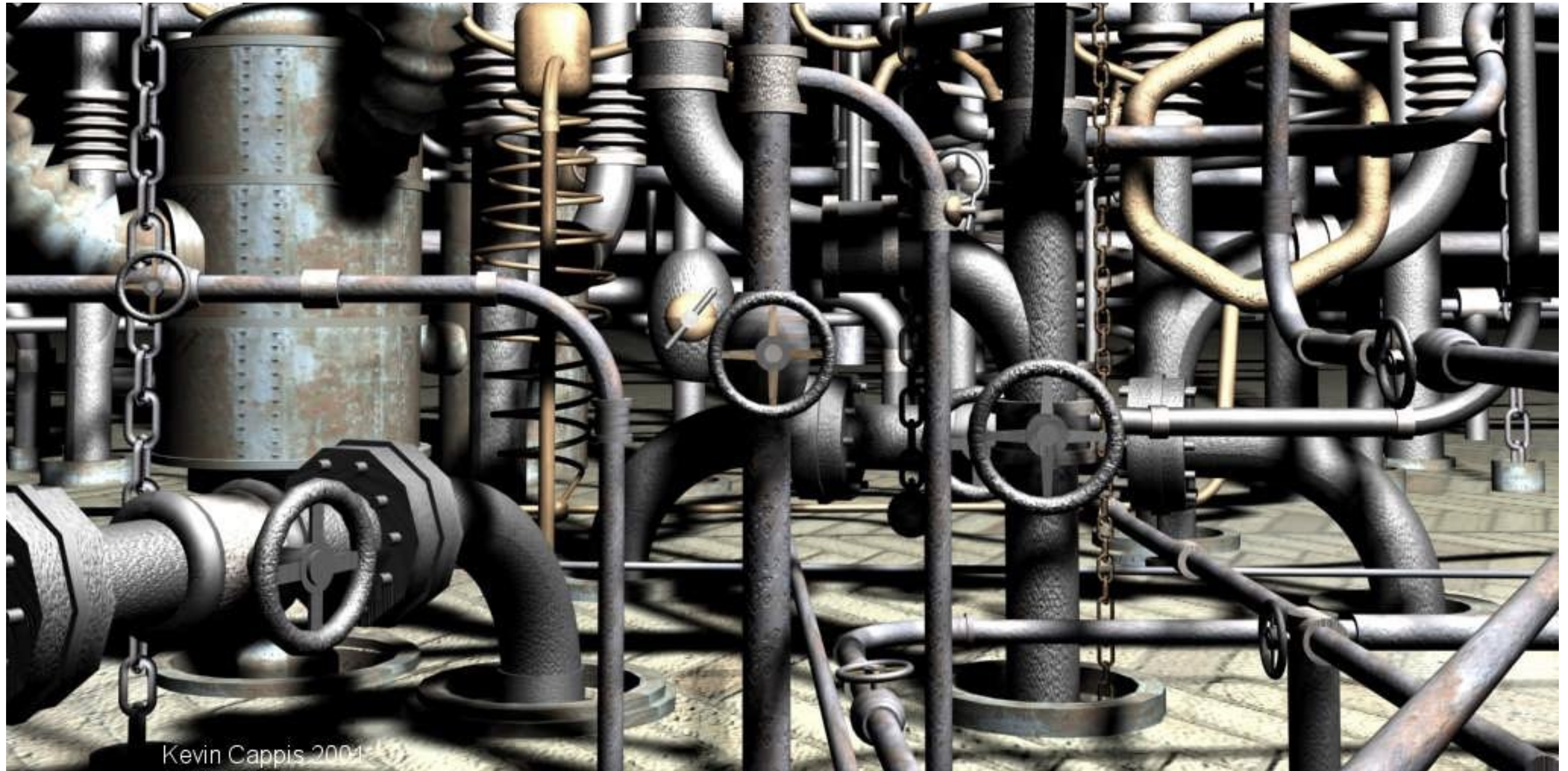
- ◎ *How to build confidence in raw data and any calculations carried on that raw data?*
- ◎ *How to detect sensor failures/anomalies?*
- ◎ *How to robustly delete data and have confidence that good data hasn't been deleted?*
- ◎ *How to limit access to different datasets generated by different groups?*
- ◎ *Techniques for intuitively linking different datasets together?*
- ◎ *Application of A.I. / Machine Learning for real time analysis of live data streams?*
- ◎ *Real time X-ray / C.T. / other scanning to allow us to view combustion processes.*
- ◎ *A.R. / V.R. techniques for data visualisation and manipulation*

...in this order

- ◎ *Techniques for intuitively linking different datasets together?*
- ◎ *How to limit access to different datasets generated by different groups?*
- ◎ *How to robustly delete data and have confidence that good data hasn't been deleted?*
- ◎ *Application of A.I. / Machine Learning for real time analysis of live data streams?*
- ◎ *How to detect sensor failures/anomalies?*
- ◎ *How to build confidence in raw data and any calculations carried on that raw data?*
- ◎ *Real time X-ray / C.T. / other scanning to allow us to view combustion processes.*
- ◎ *A.R. / V.R. techniques for data visualisation and manipulation*

How I will address the questions

- ◎ *The slides are written trying to explain you how we deal with the same problems without entering in the details of the physics problems behind*
- ◎ *To do so, I will use a race-related language (race, lap, sensor) than a particle physicist language (run, event, detector hit)*
- ◎ *The content will look very trivial & naive (at least it does to me) but I hope it will help to establish a discussion*
- ◎ *I am not implying that your problems are as trivial as my language is*

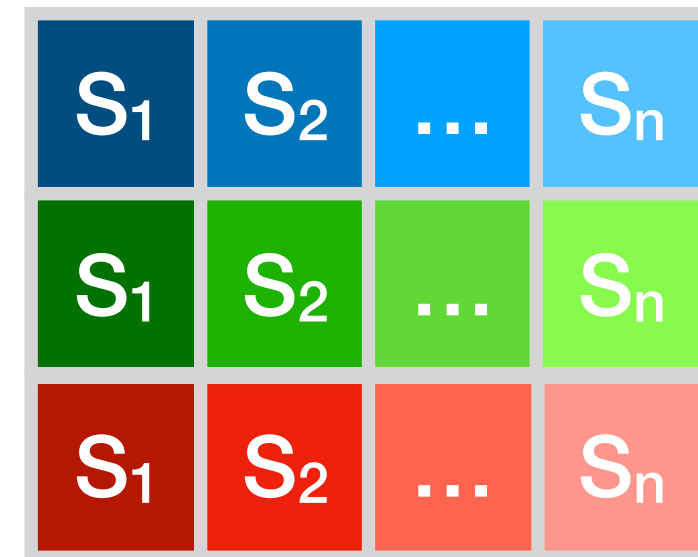


Data plumbing

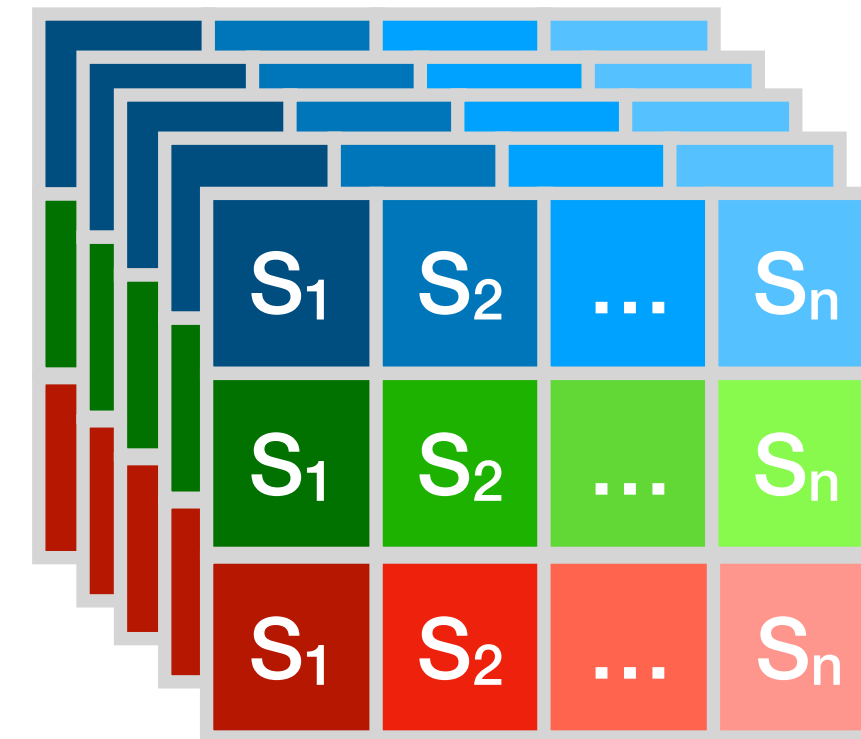
Structured Dataset



At some point you read out your car sensors.

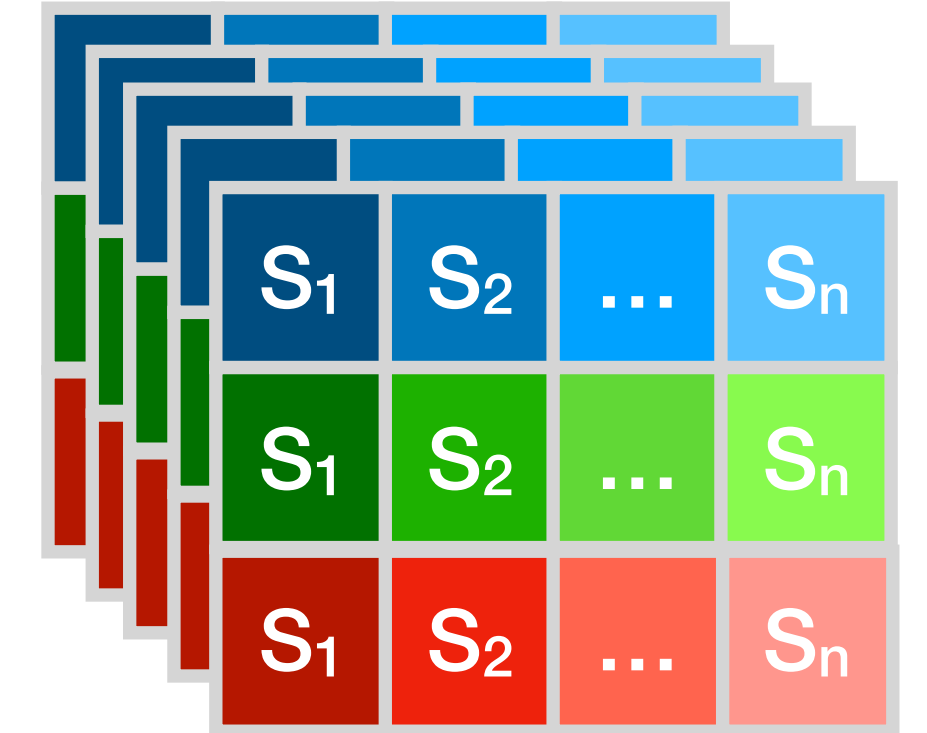


You do it every second across one lap

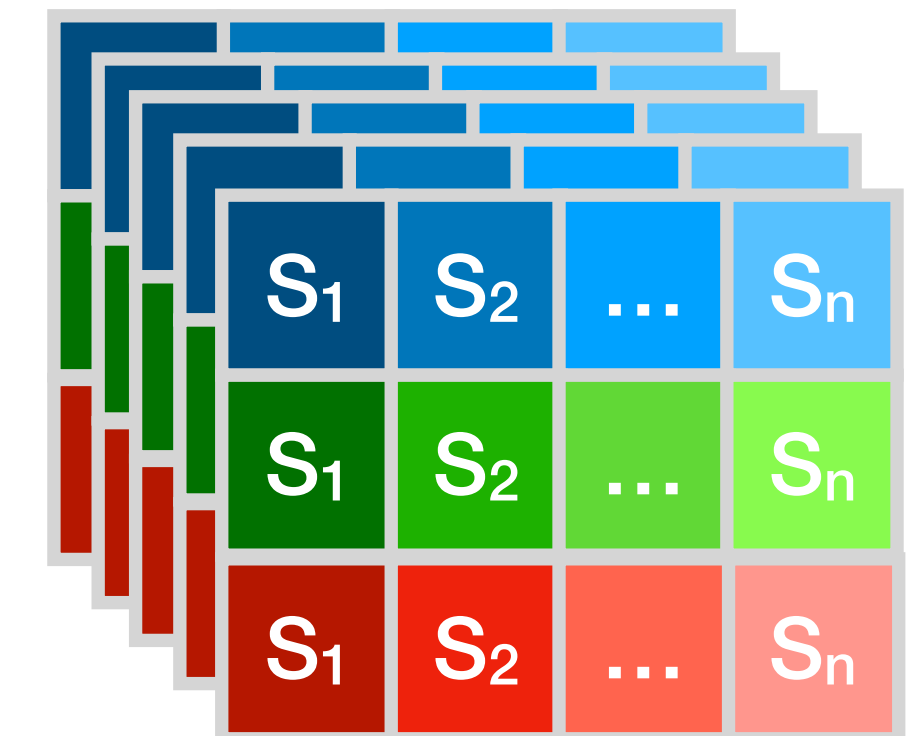


You have n laps in a session (race, test, etc)

You have many sessions



...



the sensor

$$D(i, j, k, d)$$

The session-labelling index

The lap number

the readout number

This is a unique data identification

Structured Dataset

- ◎ *You can represent your dataset as a (numpy, ROOT, etc.) array*
- ◎ *You can store arrays in files (HDF5, ROOT) file*
- ◎ *You can store and distribute data through a distributed filesystem (e.g., CERN EOS) for Big Data Applications*

This is the entry point to build your python-system data science software ecosystem

From here, any data scientist could take it over and help you solve your problems

Machine Learning Applications



Statistical analysis

Data Visualisation

How to get there

- As far as I understand, you have datasets spread across different platforms (data bases, excel spreadsheets, etc.)
- Each of these sources can be turned into numpy arrays or similar kinds of data frames
- In case you have to combine datasets from different sources, you need to use some universal data ID

lap some data

D₁(i, j, k, sd)

Session readout

S ₁	S ₂	...	S _n
S ₁	S ₂	...	S _n

lap other data

D₂(i, j, k, od)

Session readout

D ₁	D ₂	...	D _n
D ₁	D ₂	...	D _n

How to get there

- As far as I understand, you have datasets spread across different platforms (data bases, excel spreadsheets, etc.)
- Each of these sources can be turned into numpy arrays or similar kinds of data frames
- In case you have to combine datasets from different sources, you need to use some universal data ID

$$D_{1+2}(i, j, k, \text{sdod})$$

lap all data

Session readout

S ₁	S ₂	...	S _n	D ₁	D ₂	...	D _n
S ₁	S ₂	...	S _n	D ₁	D ₂	...	D _n

Some complication

⊙ *Structured datasets are not necessarily fixed size*

⊙ *Different sessions at different circuits will have different number of laps*

⊙ *Number of readouts/lap is not a fixed number*

⊙ *...*

⊙ *One can certainly fill the dataset with zeros until a fixed size is reached*

⊙ *waste disk space, need post-processing, ...*

⊙ *Or, one could deal with a variable-size data frame, using adequate software (e.g., CERN's ROOT)*

S ₁	S ₂	...	S _n
S ₁	S ₂	0	0
S ₁	S ₂	...	0

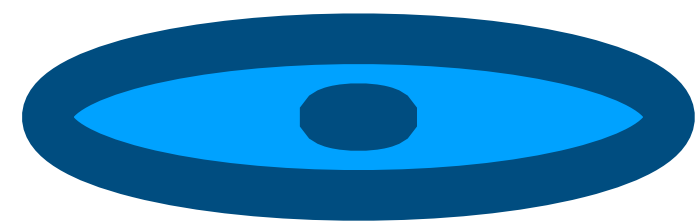
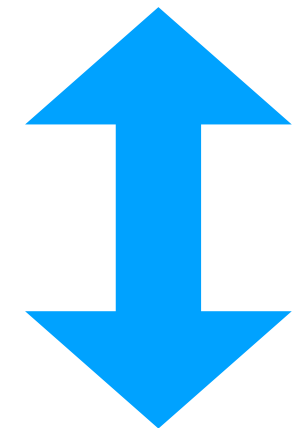
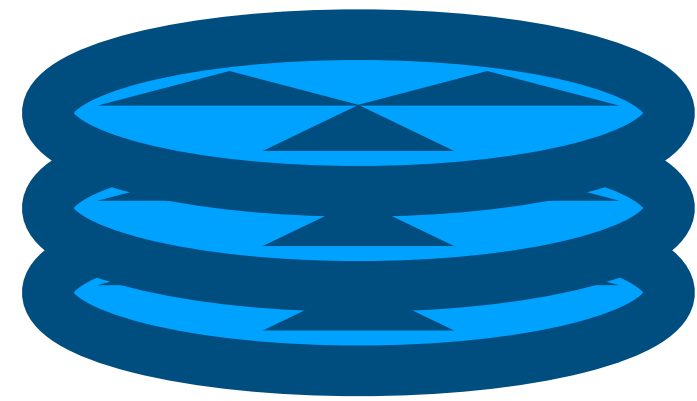
S ₁	S ₂	...	0
S ₁	S ₂	...	S _n
S ₁	0	0	0

S ₁	0	0	0
S ₁	S ₂	...	S _n
0	0	0	0

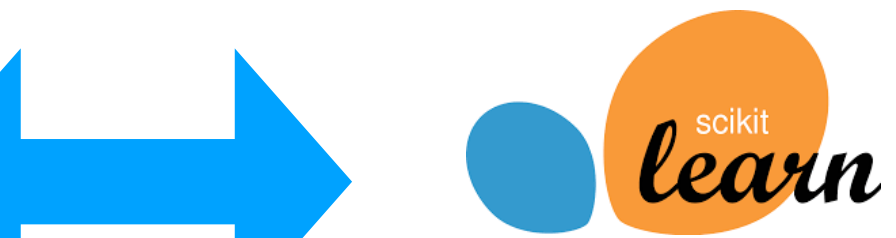
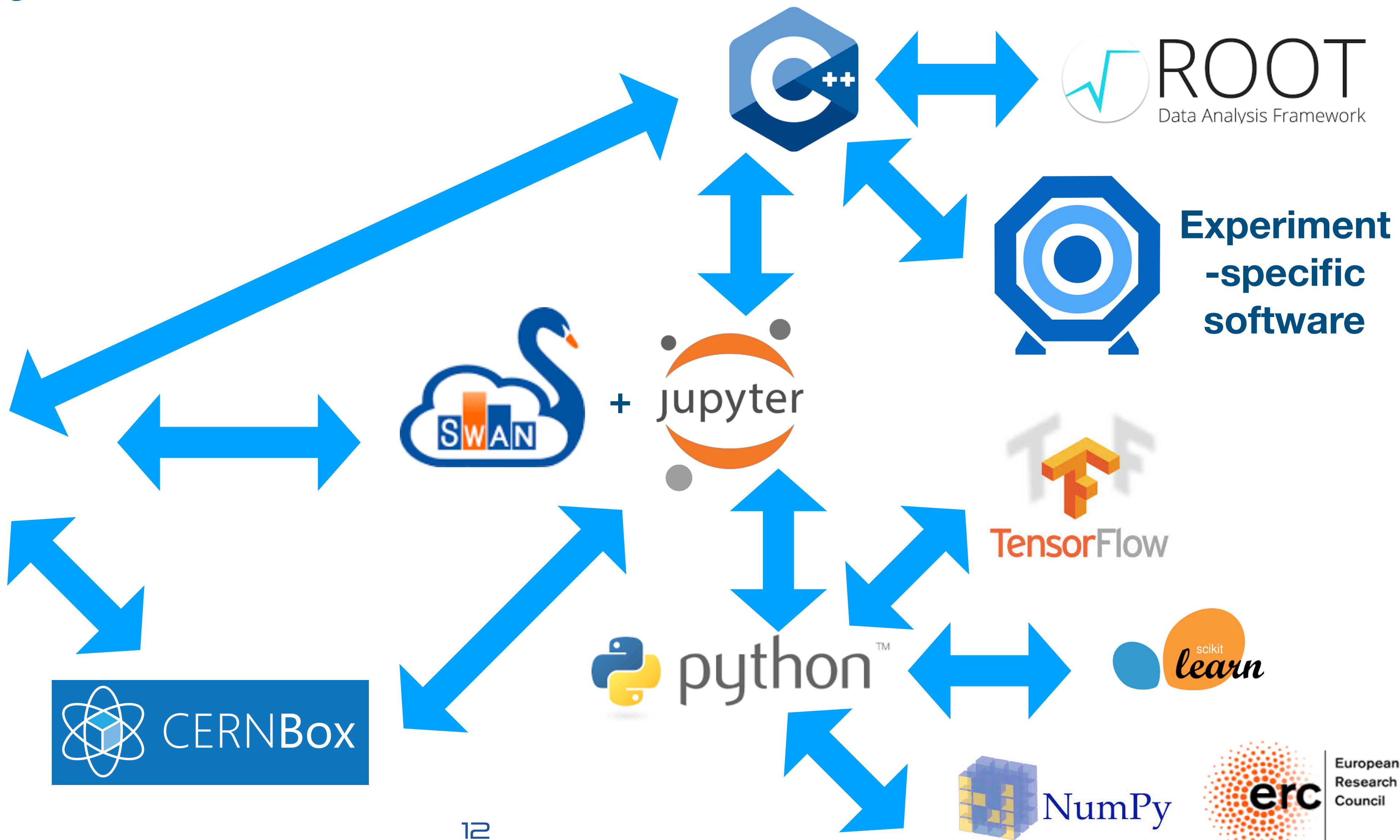


How we do it at CERN

Long-term storage
(Tape)

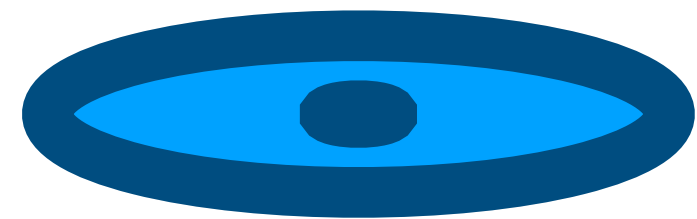
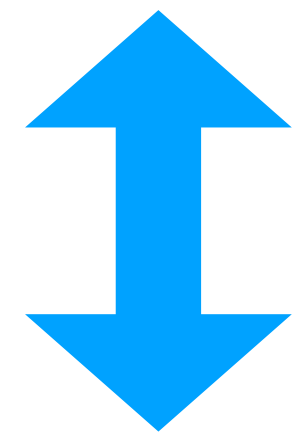
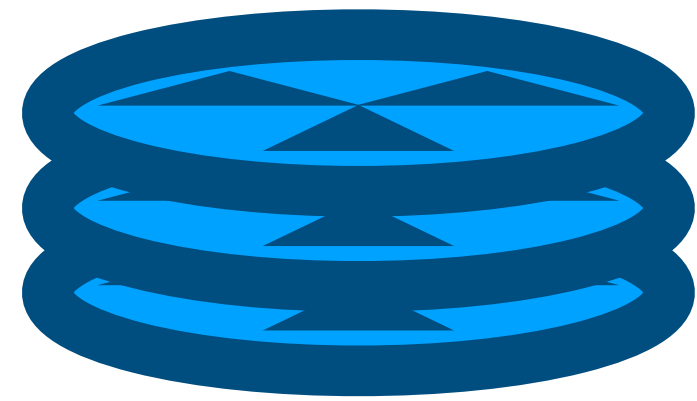


Short-term storage
(Disk)



How we do it at CERN

**Long-term storage
(Tape)**



**Short-term
storage
(Disk)**

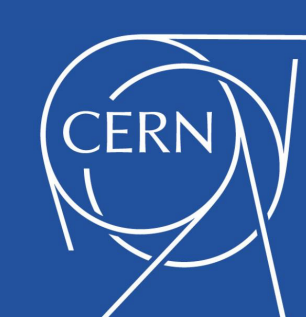
Once we take data, we don't delete them ever.
We store them on tape to have a cheap long-term storage

A dedicated software decides which datasets to keep on disk and which datasets to be moved to tape, based on popularity statistics (number of accesses, latest access, etc)
Algorithm can be rule-based or tuned with Deep Learning (e.g. Reinforcement Learning)

Datasets can be stored in multiple copies and different data tiers:

- RAW data (1 copy at most)
- High-Level data (functions of raw data, smaller size, multiple copied)

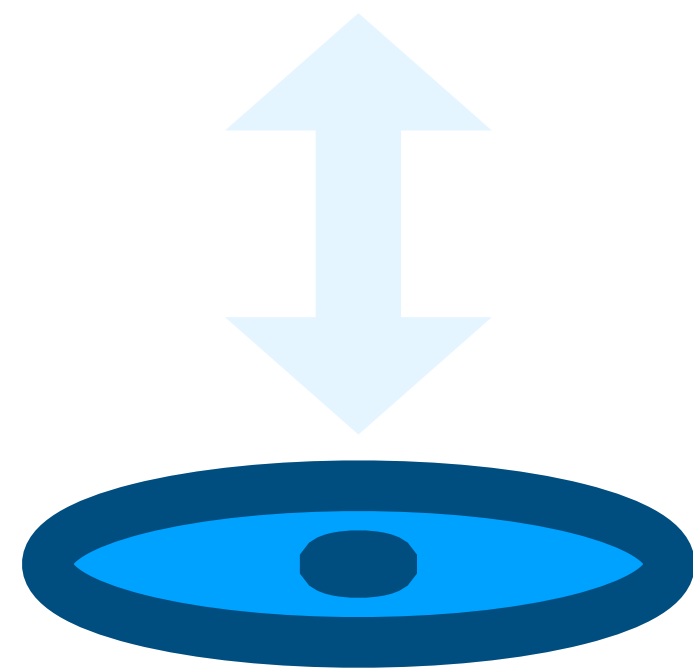
Data are stored in compressed (gzip, etc) data formats (HDF5, ROOT)



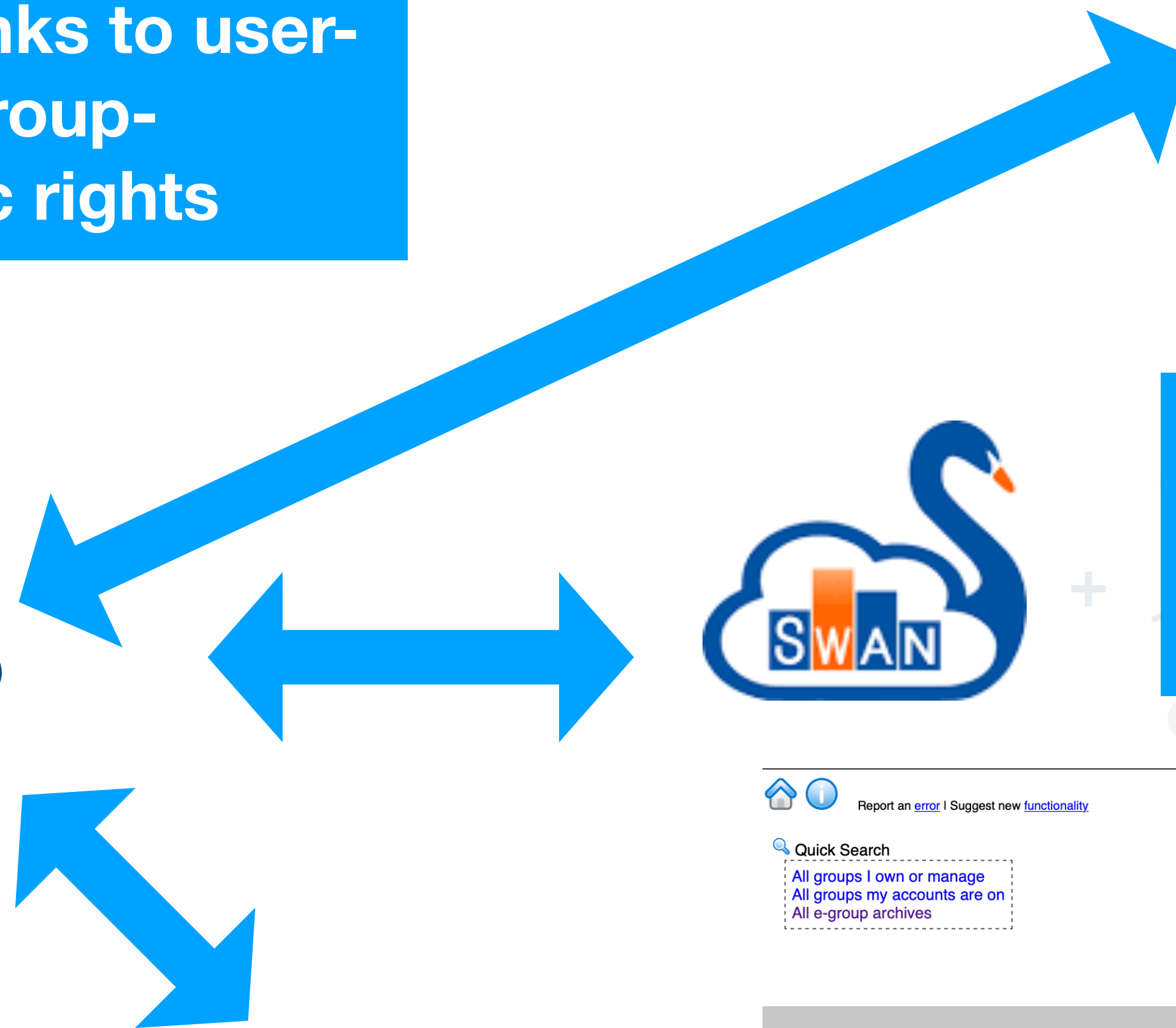
How we do it at CERN

Long-term storage

Datasets access is granted thanks to user- and group-specific rights



Short-term storage (Disk)



Experiment-specific software

Access rights are managed through a web interface (e-groups)



Report an [error](#) | Suggest new [functionality](#)

e-groups Maurizio PIERINI | Group Memberships: 382 | [Logout](#)

Quick Search
 All groups I own or manage
 All groups my accounts are on
 All e-group archives

e-group name: contains: Search




All e-groups ALICE ATLAS CMS LCG LHCB LHCf MoEDAL TOTEM

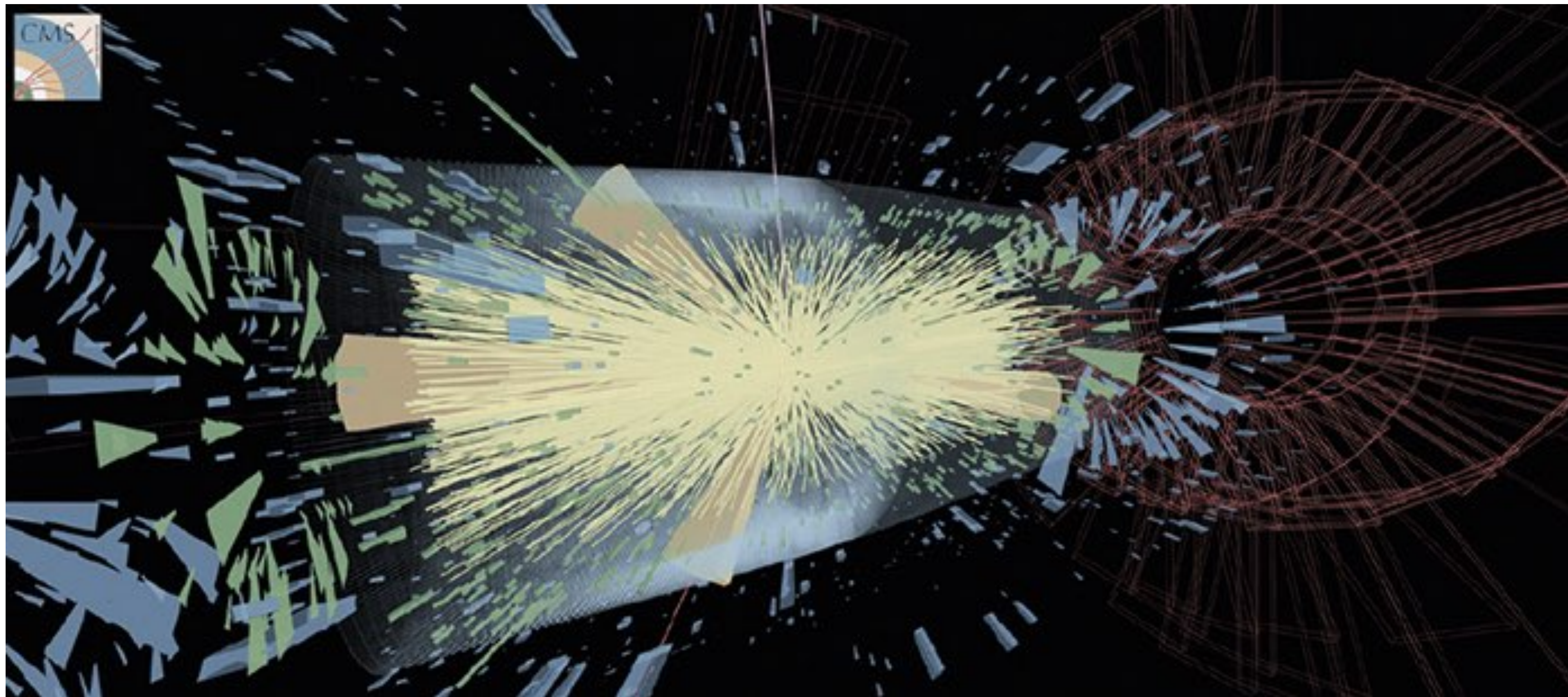
Only groups I own or manage | Only groups I am on | Page Size:

[+ Create new static group](#) [+ Create new dynamic group](#) [Show groups for one member](#) [Manage groups for one member](#) [Manage owner/admin](#)

E-groups								
Name	Type	Topic	Description	Status	Owner	Actions	Archive	
cernbox-project-dshep-admins	Static		CERNBOX PROJECT DSHEP ADMINS	Active	luca.mascetti@cern.ch	Subscribe		
cernbox-project-dshep-readers	Static		CERNBOX PROJECT DSHEP READERS	Active	luca.mascetti@cern.ch	Unsubscribe		
cernbox-project-dshep-writers	Static		CERNBOX PROJECT DSHEP WRITERS	Active	luca.mascetti@cern.ch	Unsubscribe		
dshep-lcd	Static	Data Science at High Energy Physics	DSHEP LCD dataset working group	Active	Maurizio.Pierini@cern.ch	Unsubscribe		

Answered (?) questions

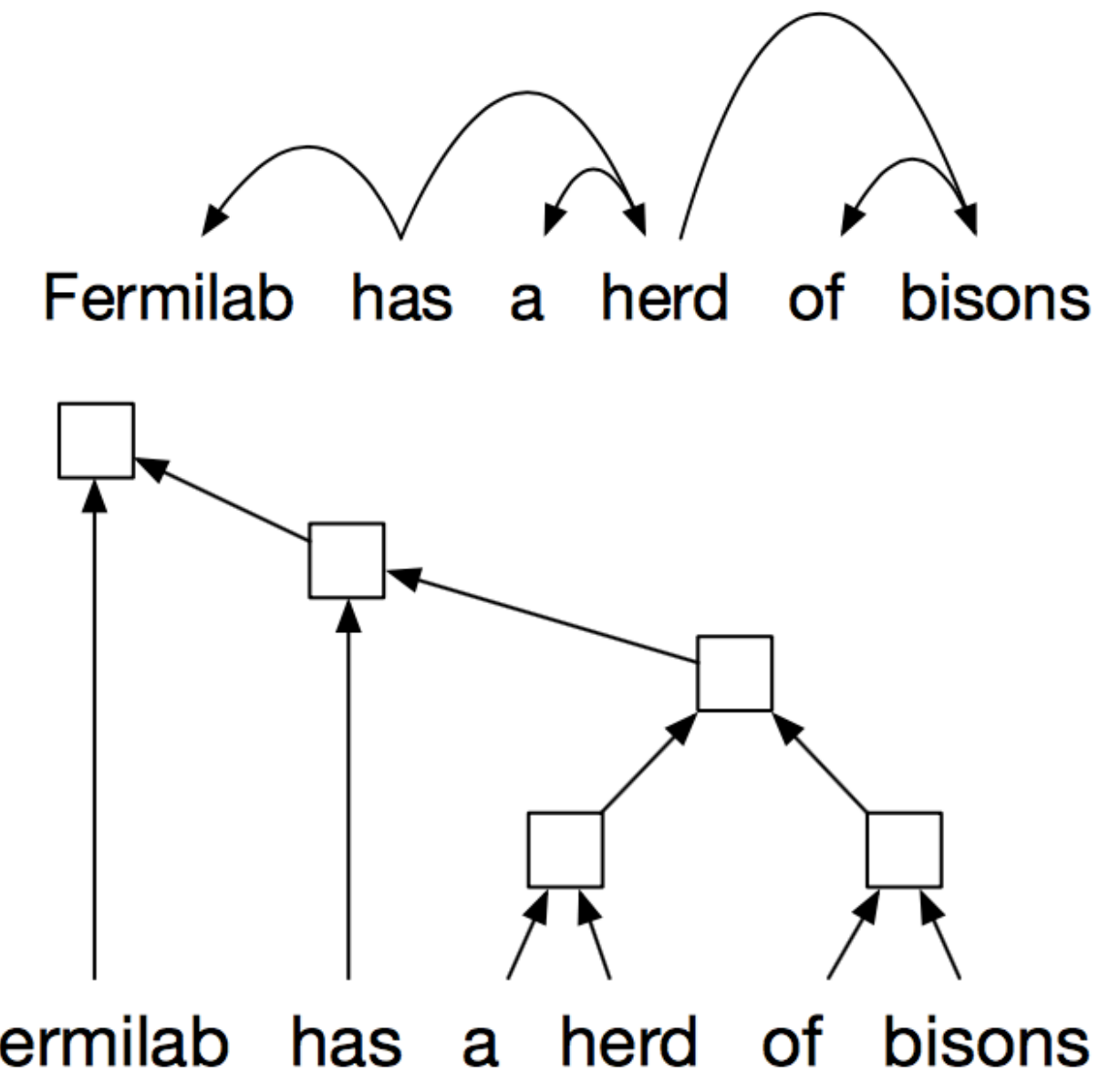
- ◎ *Techniques for intuitively linking different datasets together?* 
- ◎ *How to limit access to different datasets generated by different groups?* 
- ◎ *How to robustly delete data and have confidence that good data hasn't been deleted?* 
- ◎ *Application of A.I. / Machine Learning for real time analysis of live data streams?*
- ◎ *How to detect sensor failures/anomalies?*
- ◎ *How to build confidence in raw data and any calculations carried on that raw data?*
- ◎ *Real time X-ray / C.T. / other scanning to allow us to view combustion processes.*
- ◎ *A.R. / V.R. techniques for data visualisation and manipulation*



Real-time data analysis with Deep Learning

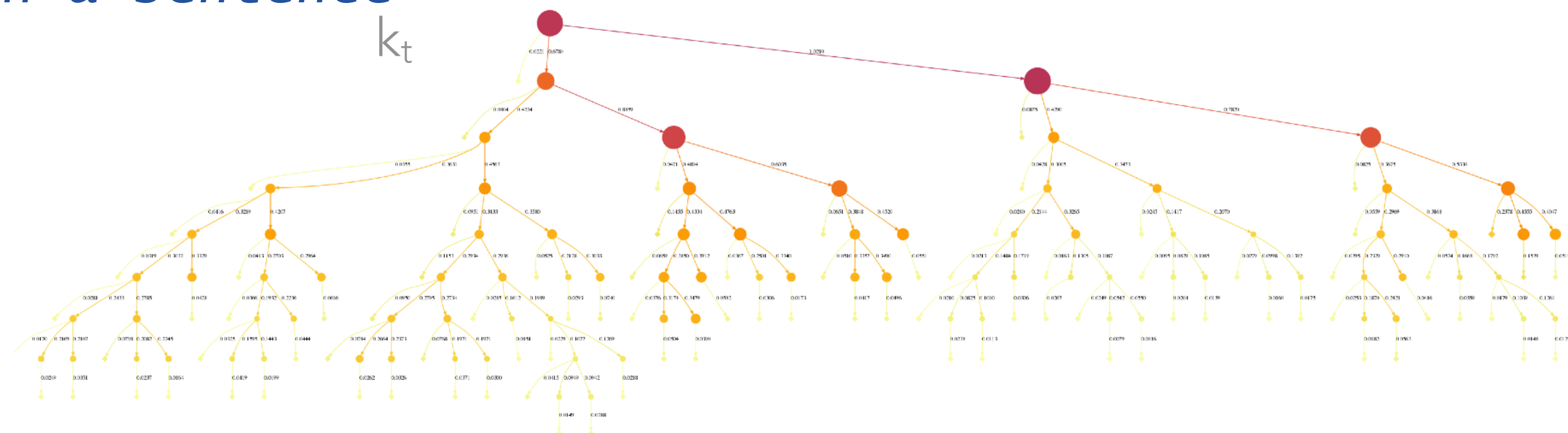
LHC events & language processing

- *PF reco is not the best match for computing vision techniques (e.g., convolutional neural networks) don't work*
- *one would have to convert the particles to a pixelated images, loosing resolution*
- *Instead, list of particles can be processed by Deep Learning architectures designed for natural language processing (RNN, LSTMs, GRUs, ...)*



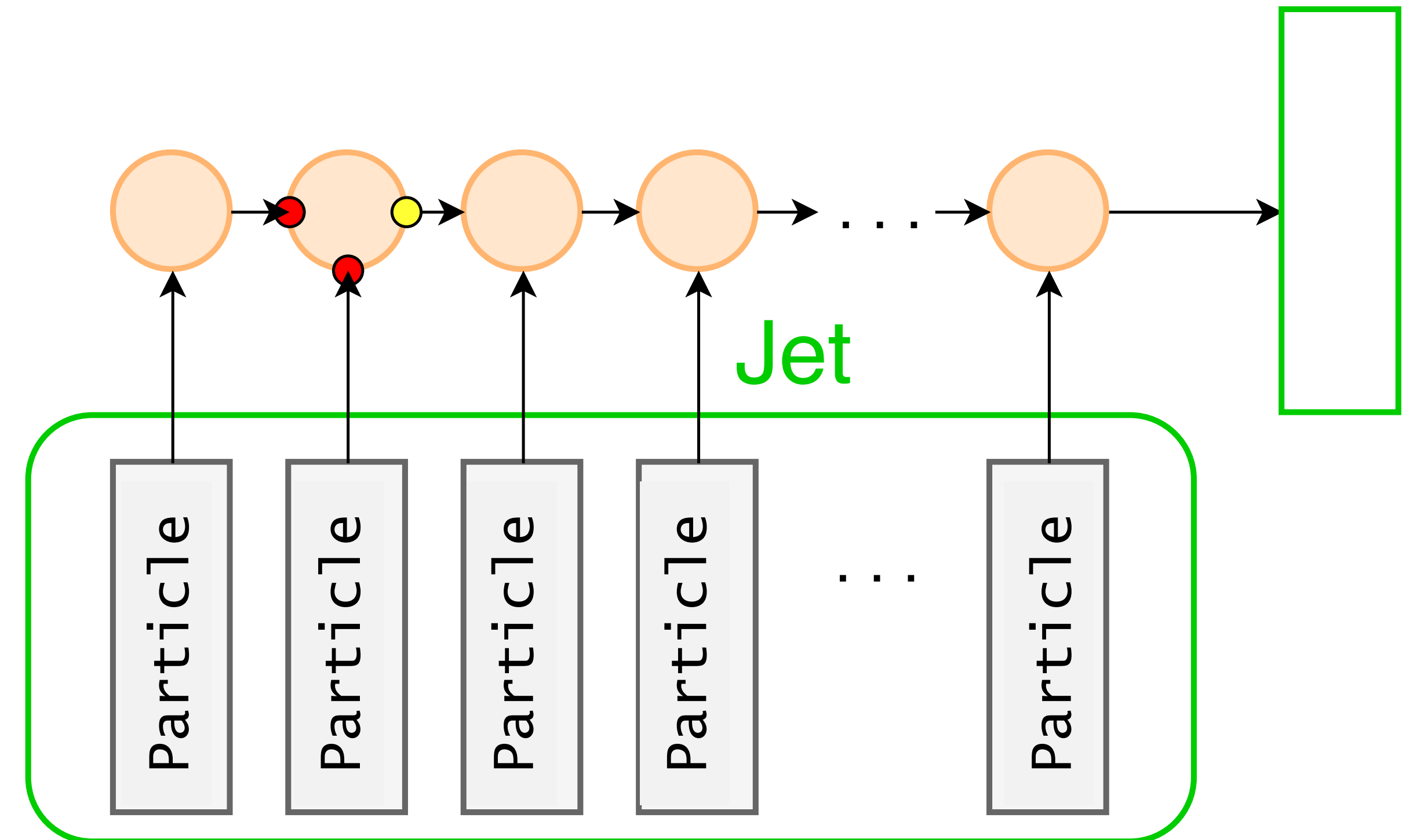
● *particles as words in a sentence*

● *QCD is the grammar*



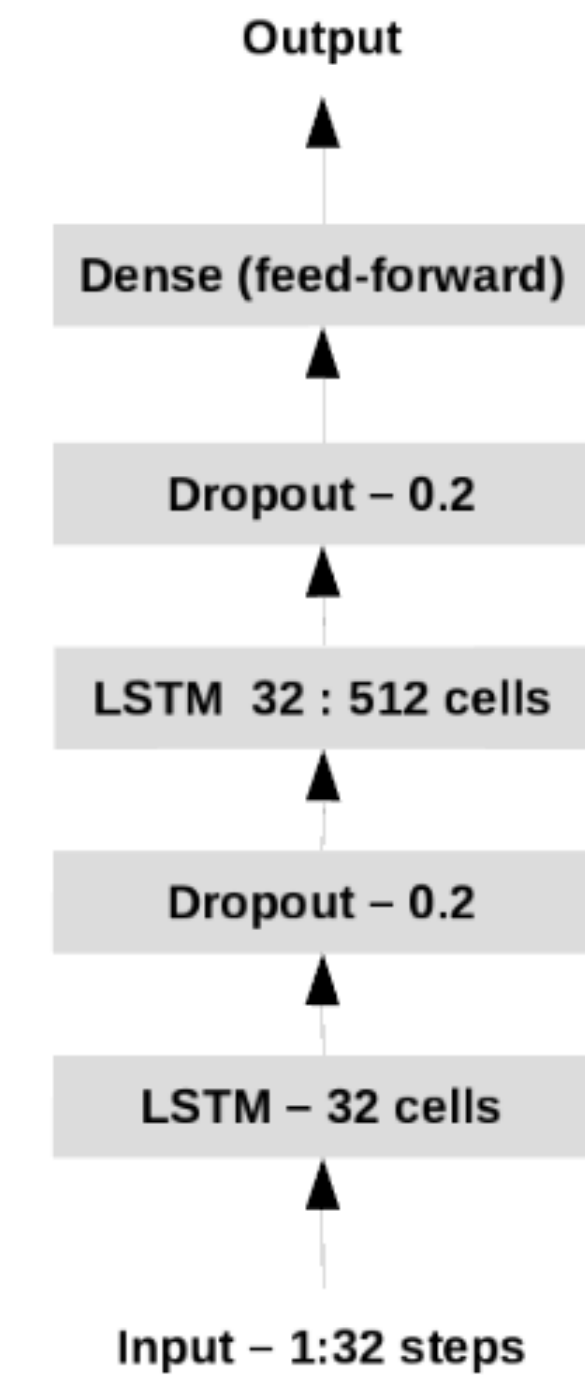
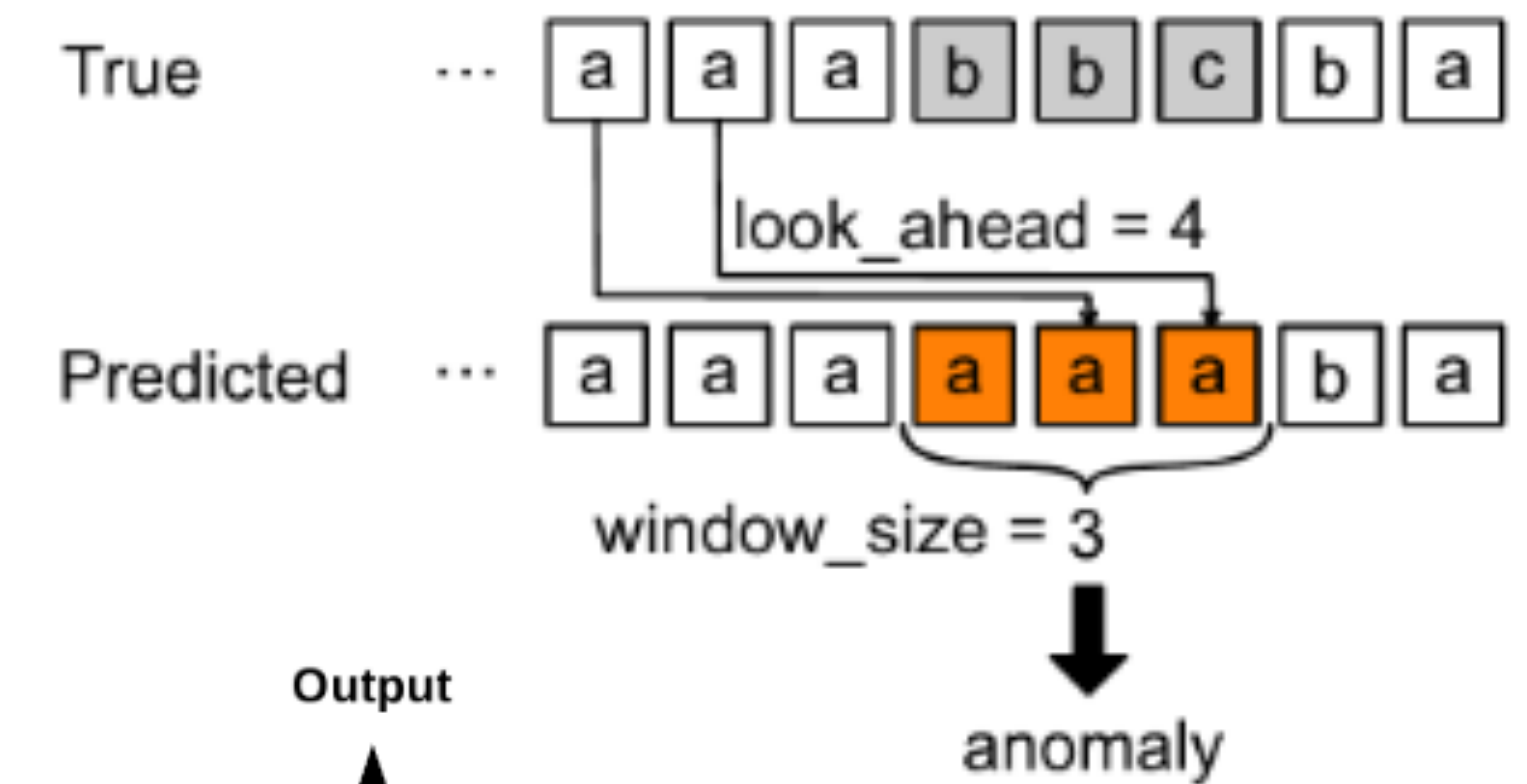
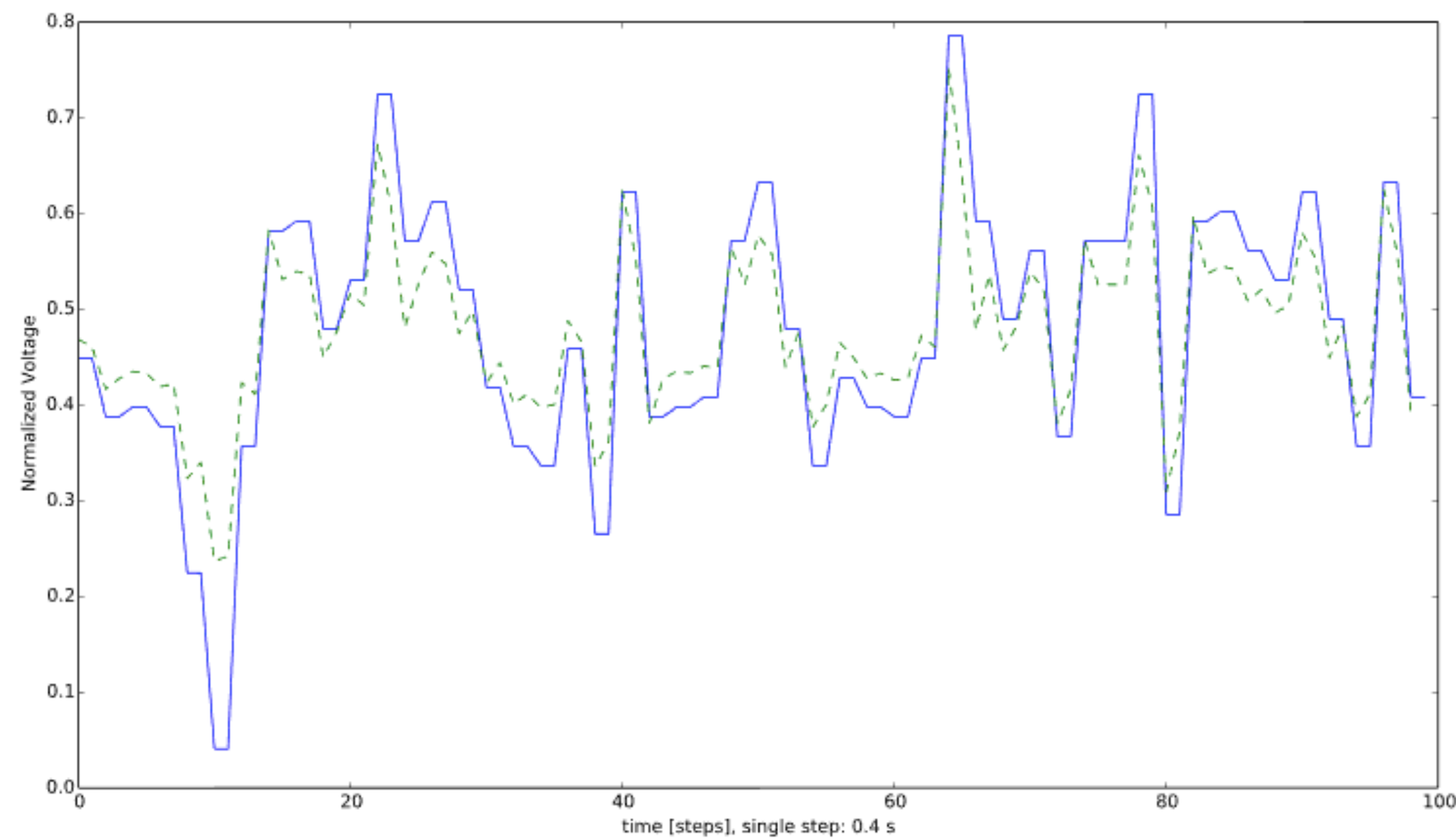
Recurrent Neural Networks

- A network architecture suitable to process an ordered sequence of inputs
- words in text processing
- a time series
- particles in a list
- Could be used for a single jet or the full event
- Next step: graph networks (active research direction)



LSTM networks for LHC magnets

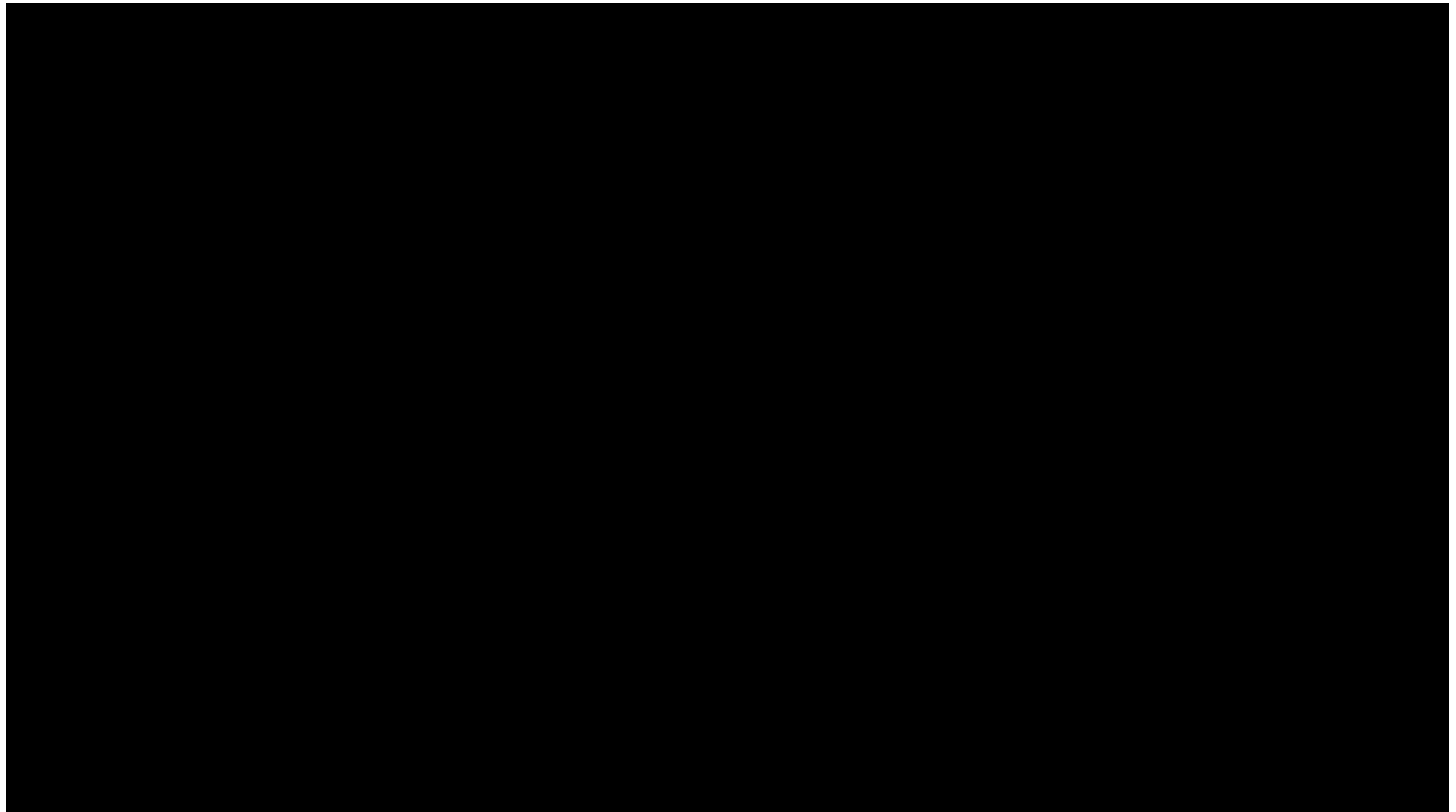
- Study time series of magnet voltages
- target: predict voltage changes



Maciej WIELGOSZ's work on LHC magnets

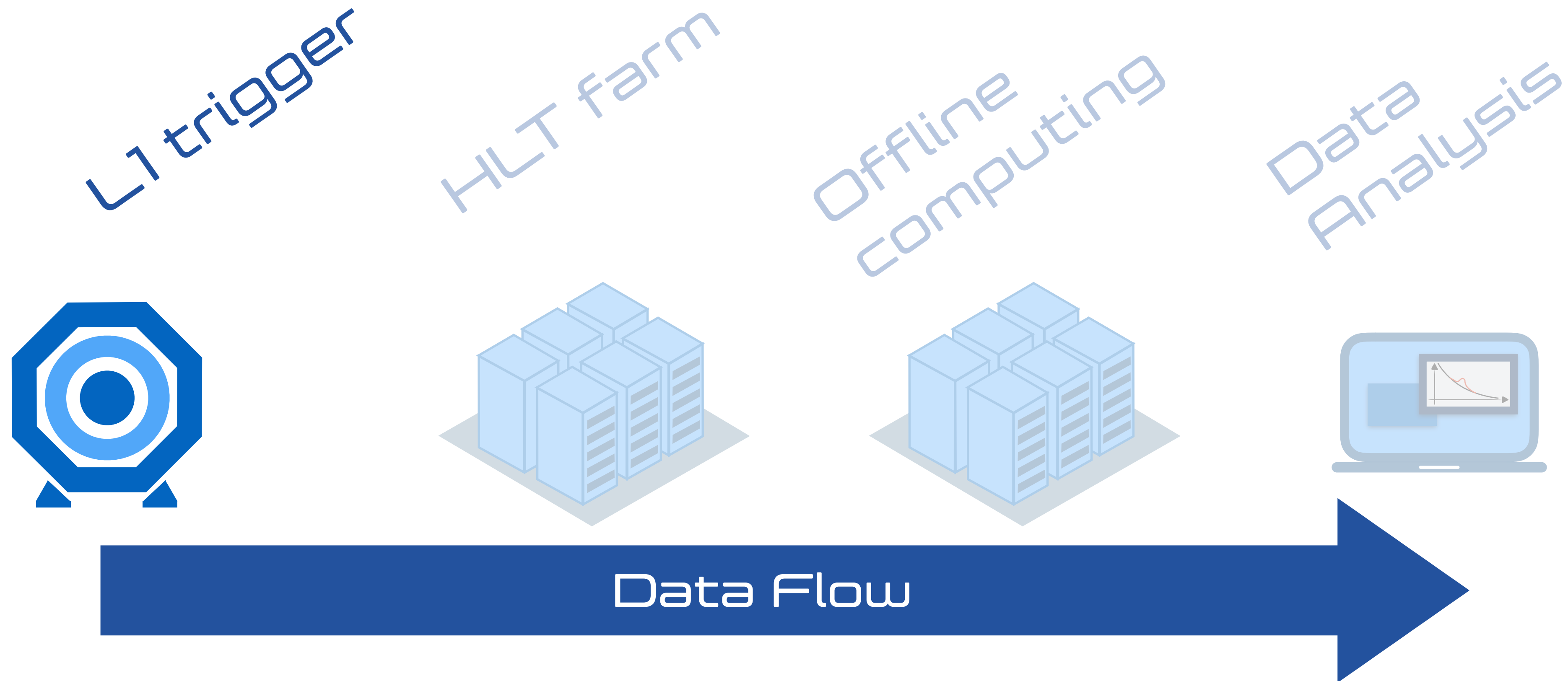


Real-time @LHC



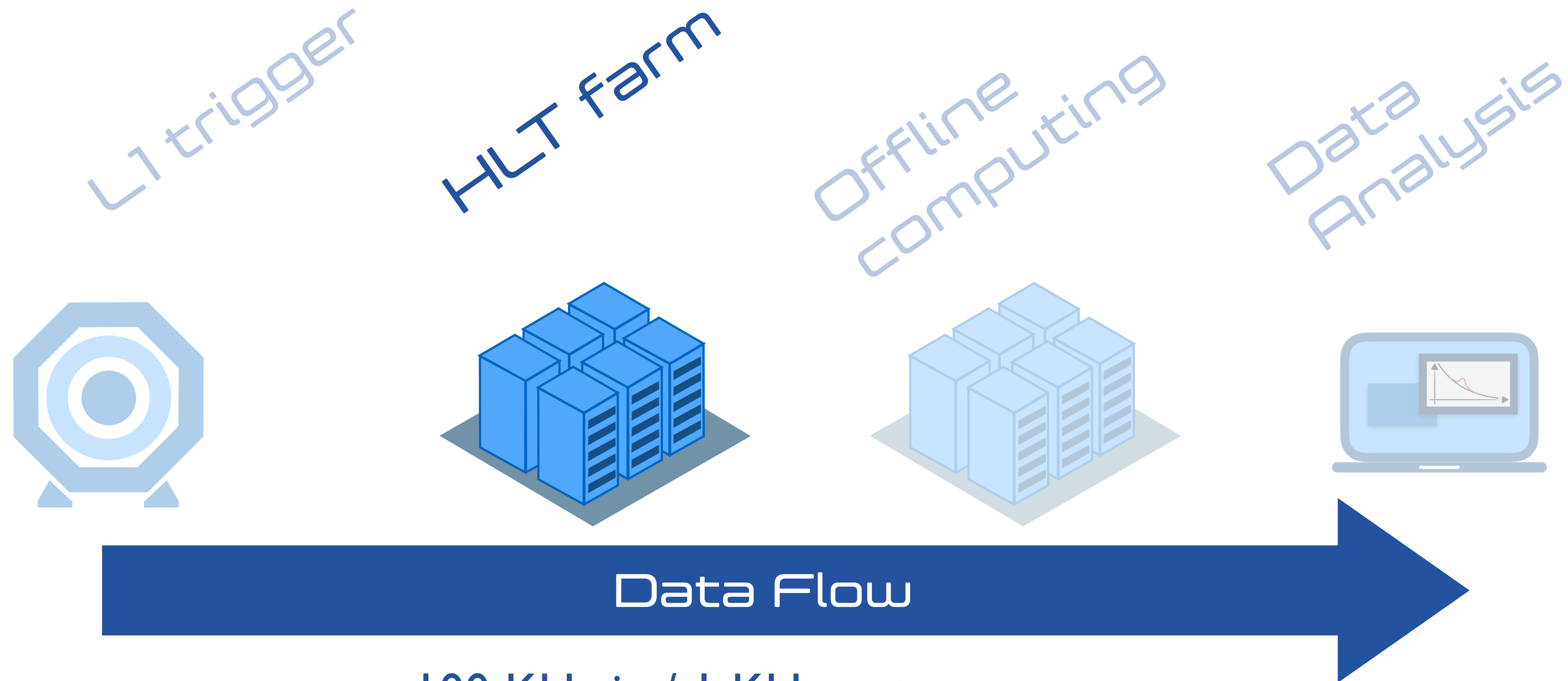
<https://www.youtube.com/watch?v=jDC3-QSiLB4>

The LHC Big Data problem



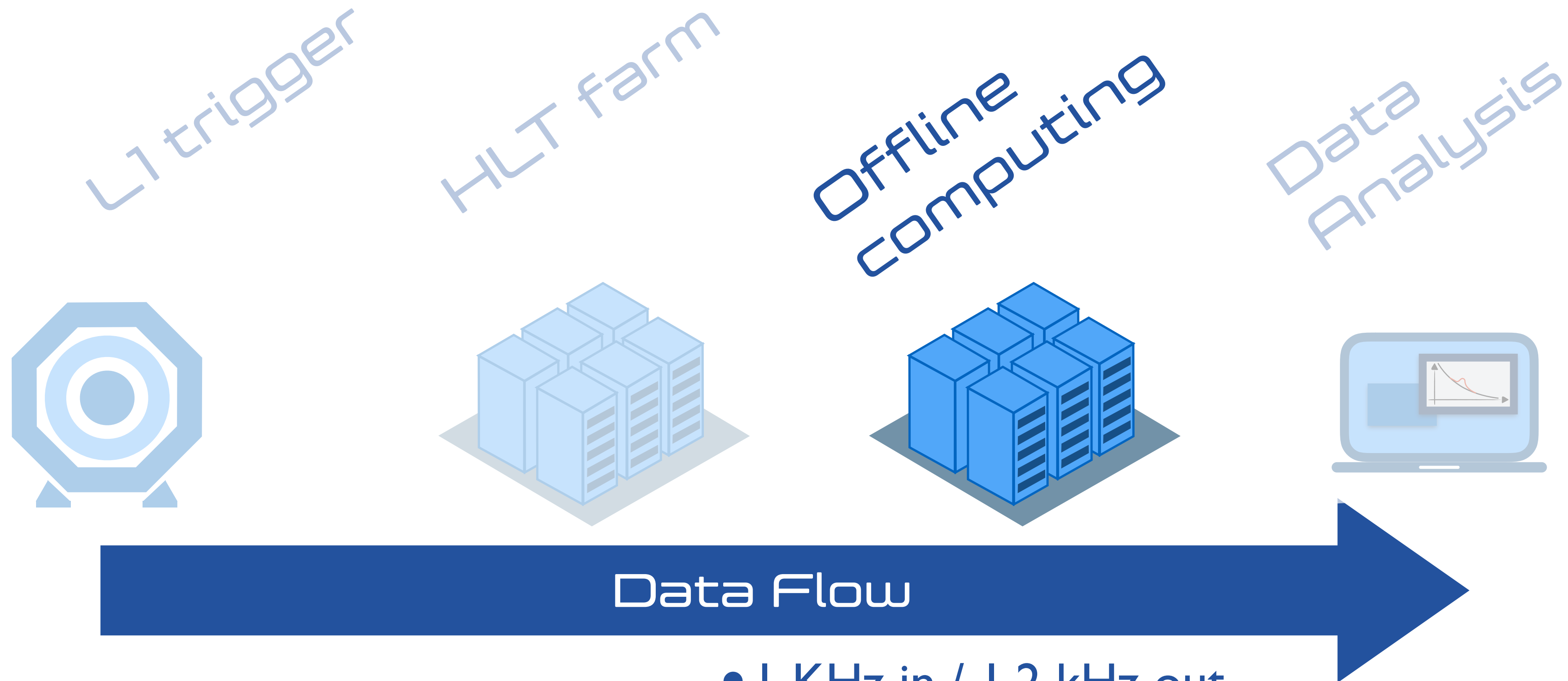
- 40 MHz in / 100 KHz out
- ~ 500 KB / event
- Processing time: ~10 μ s
- Based on coarse local reconstructions
- FPGAs / Hardware implemented

The LHC Big Data problem



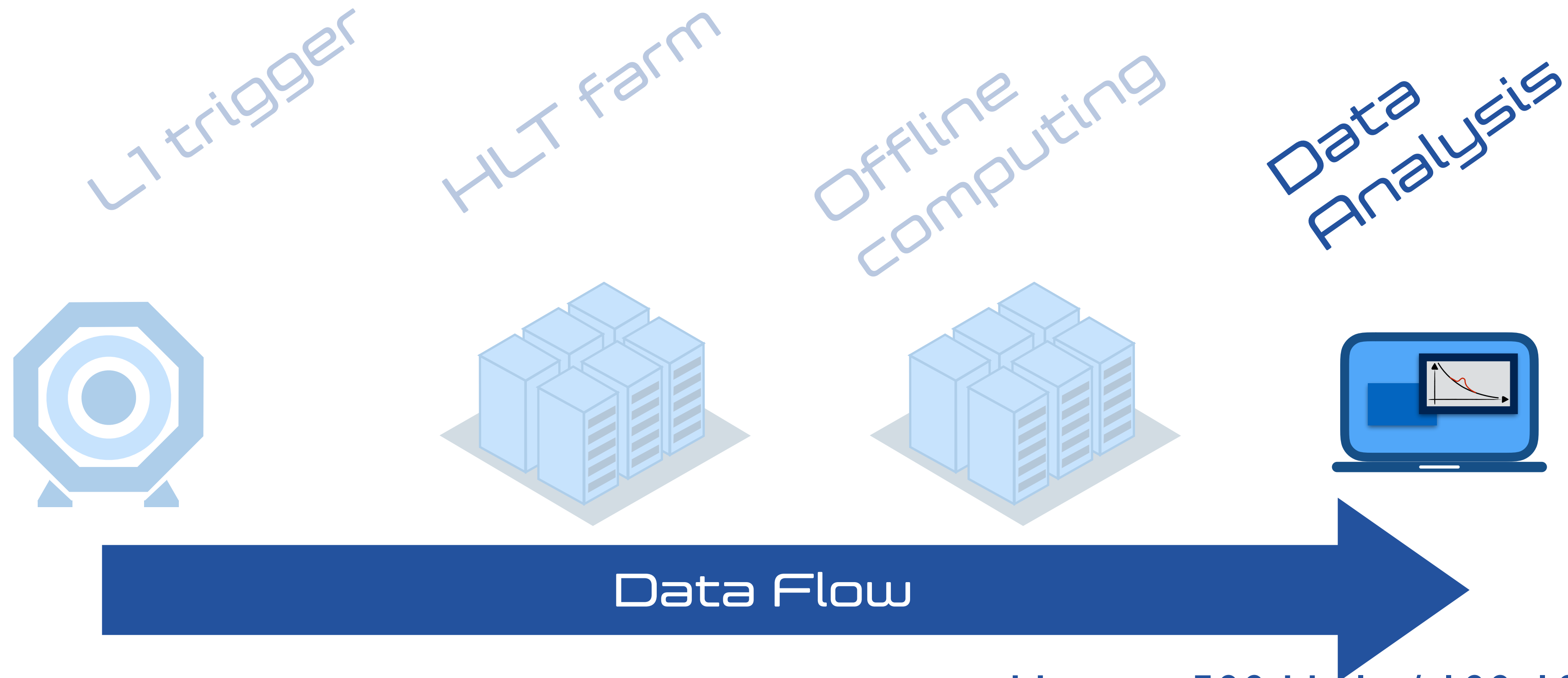
- 100 KHz in / 1 KHz out
- ~ 500 KB / event
- Processing time: ~100 ms
- Based on simplified global reconstructions
- Software implemented on CPUs

The LHC Big Data problem



- 1 KHz in / 1.2 kHz out
- ~ 1 MB / 200 kB / 30 kB per event
- Processing time: ~20 s
- Based on accurate global reconstructions
- Software implemented on CPUs

The LHC Big Data problem



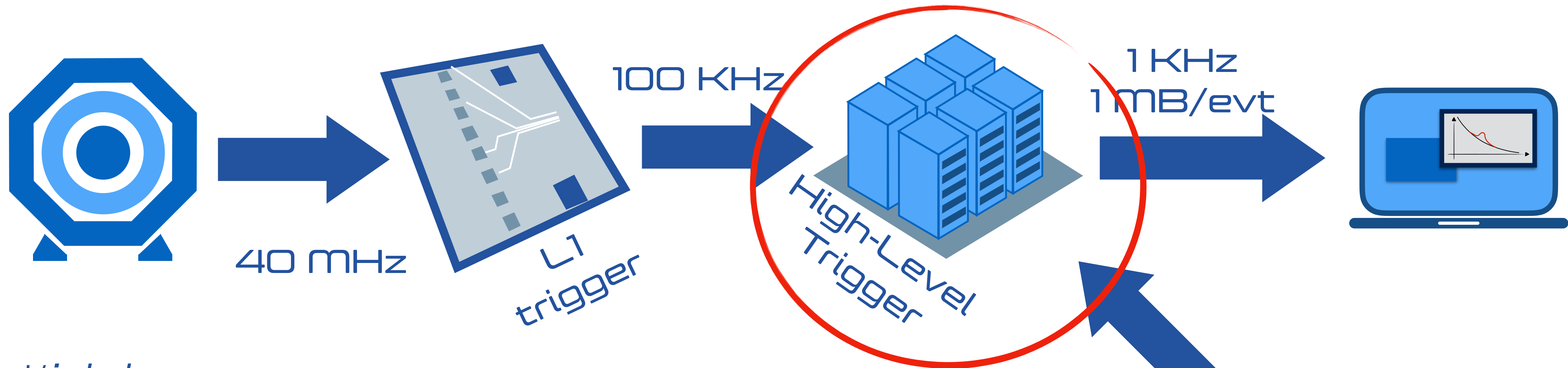
- Up to ~ 500 Hz In / 100-1000 events out

- < 30 KB per event

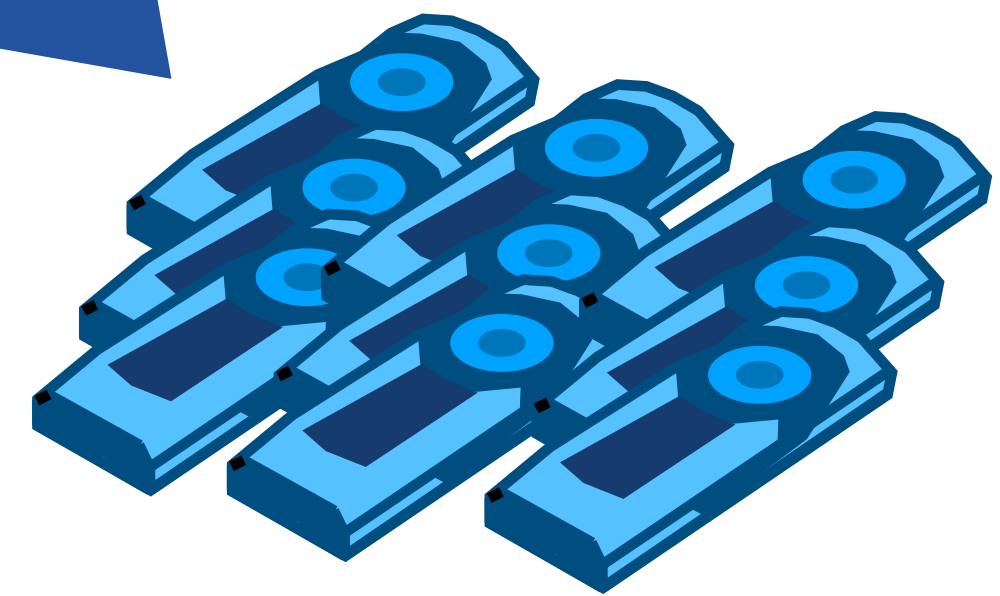
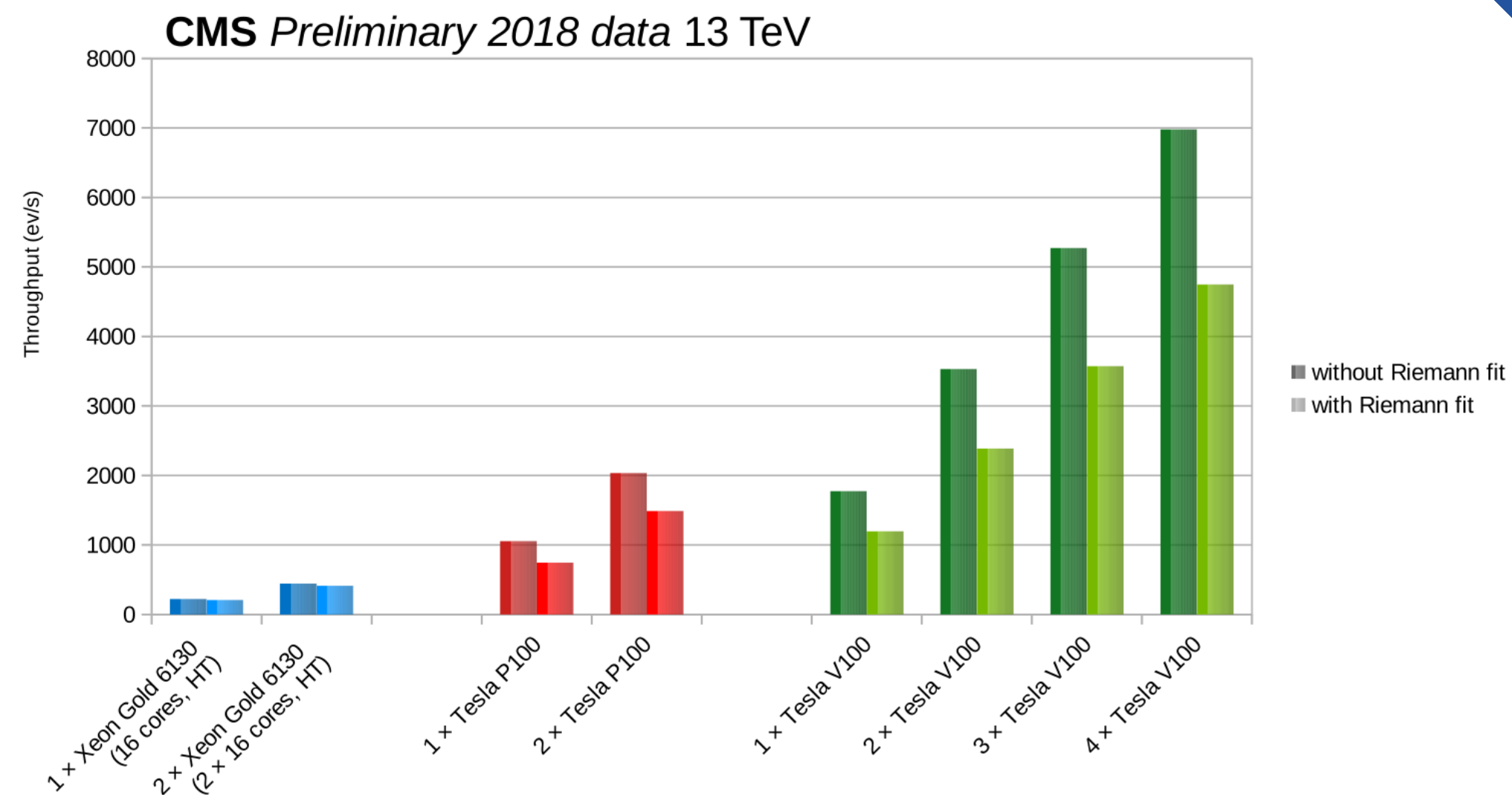
- Processing time irrelevant

- User-written code + centrally produced selection algorithms

Heterogeneous HLT

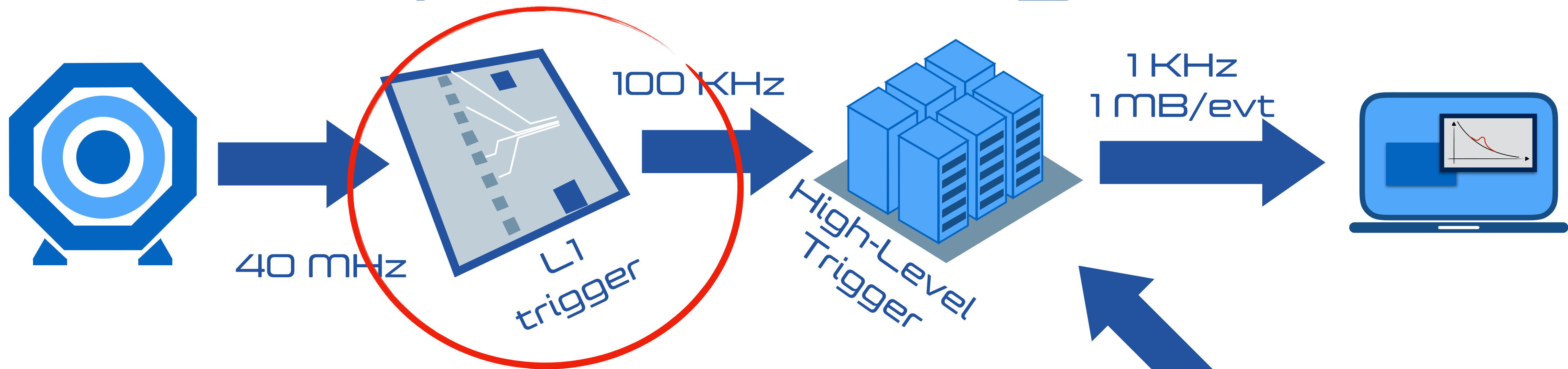


- With heterogenous hardware in place (for other reasons) Deep Learning inference @HLT quite easy
- This will happen no matter what, to speed up traditional algorithms
- Deep Learning @HLT will benefit of it

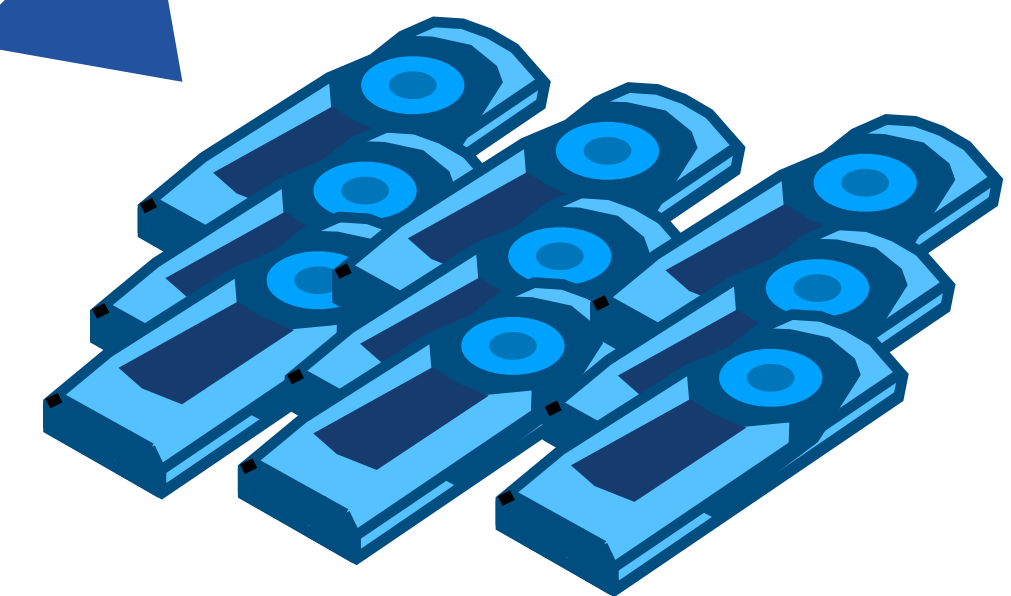


Patatrack project for CMS HLT on GPUs

Deep Learning at L1

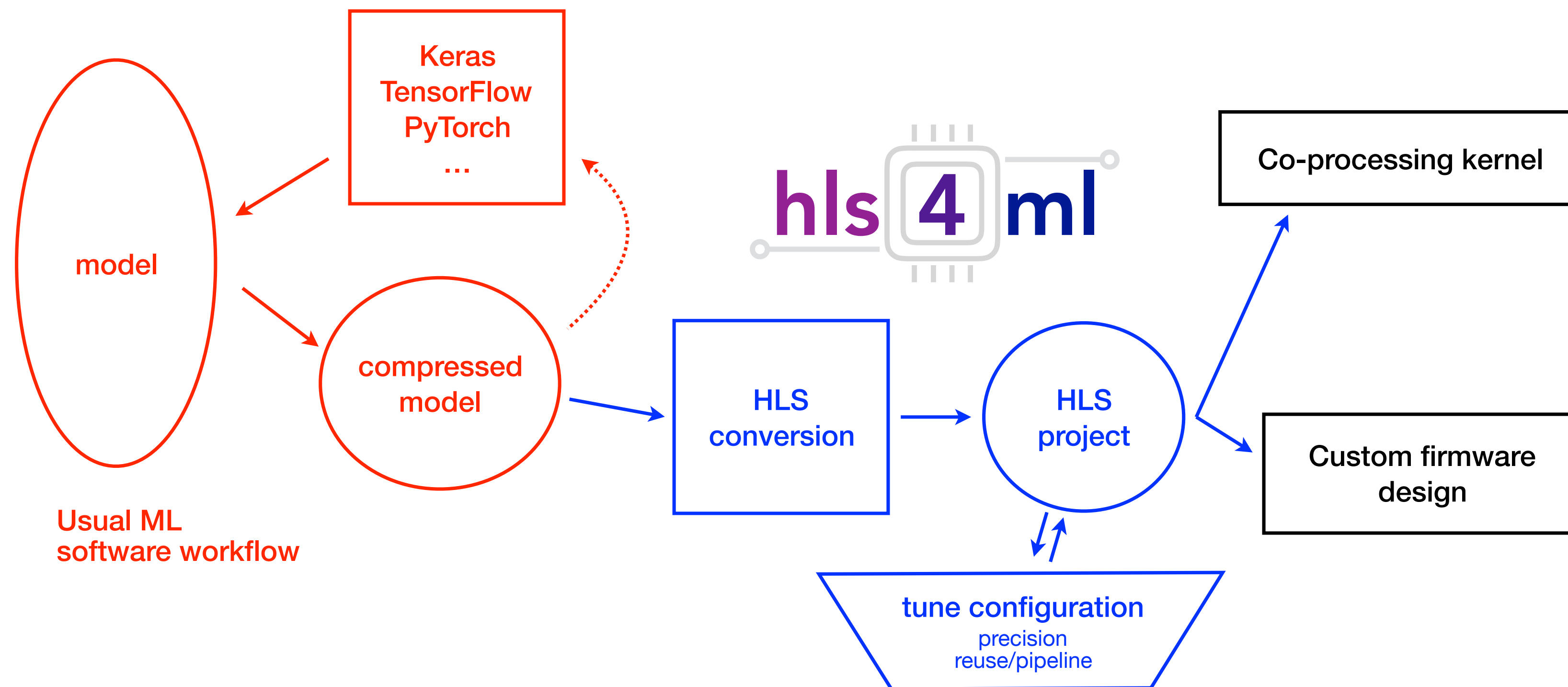


- ◉ *Situation at L1 is different, mainly due to the typical latency (<math><10 \mu\text{sec}</math>)*
- ◉ *Custom cards connected to detector electronics by optic links*
- ◉ *Data flow in the cards one by one*
- ◉ *Networks need to be implemented in FPGA firmware*
 - ◉ *advanced design by expert engineers (not common resource in HEP)*
 - ◉ *automatic translation tools doing the job*

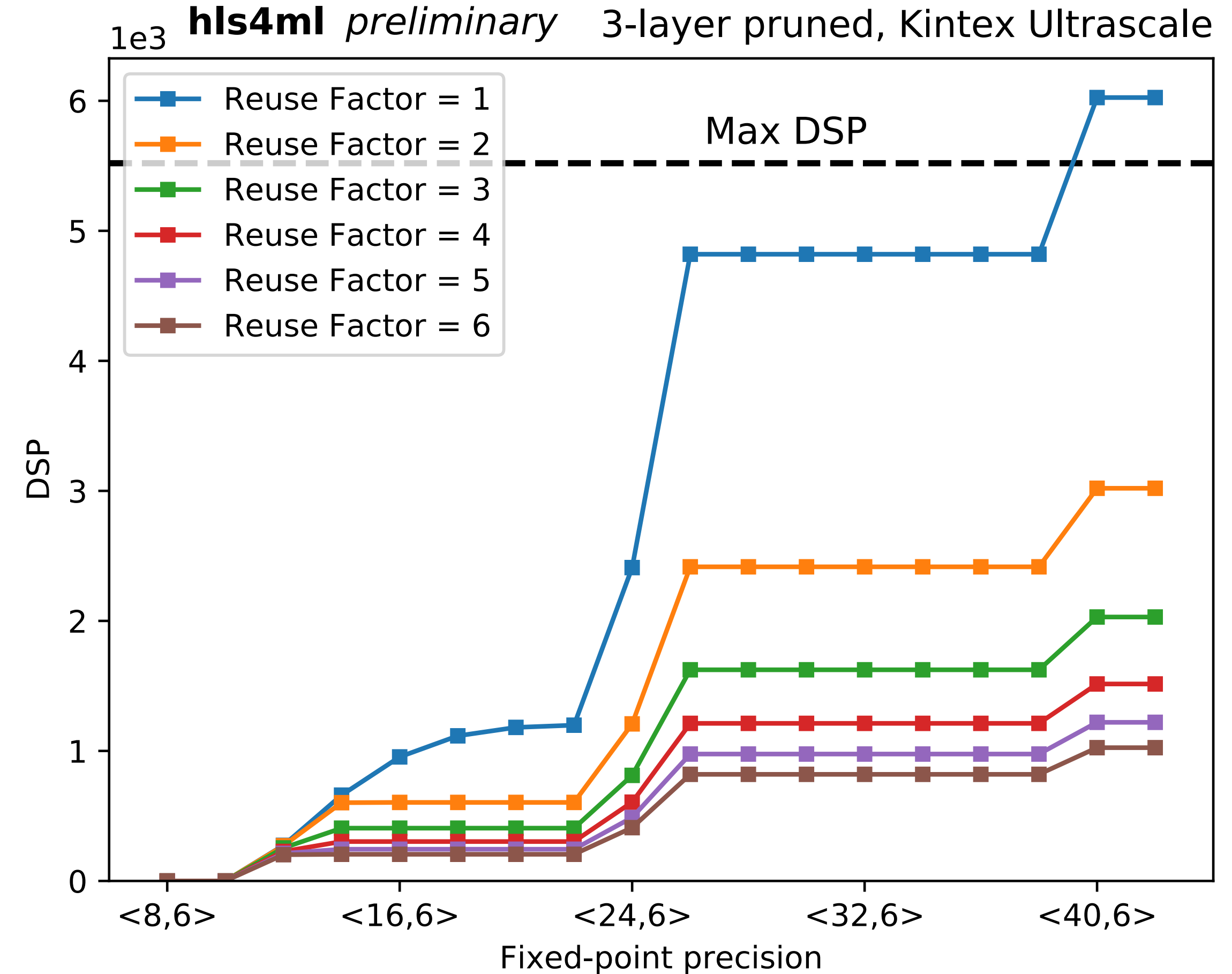
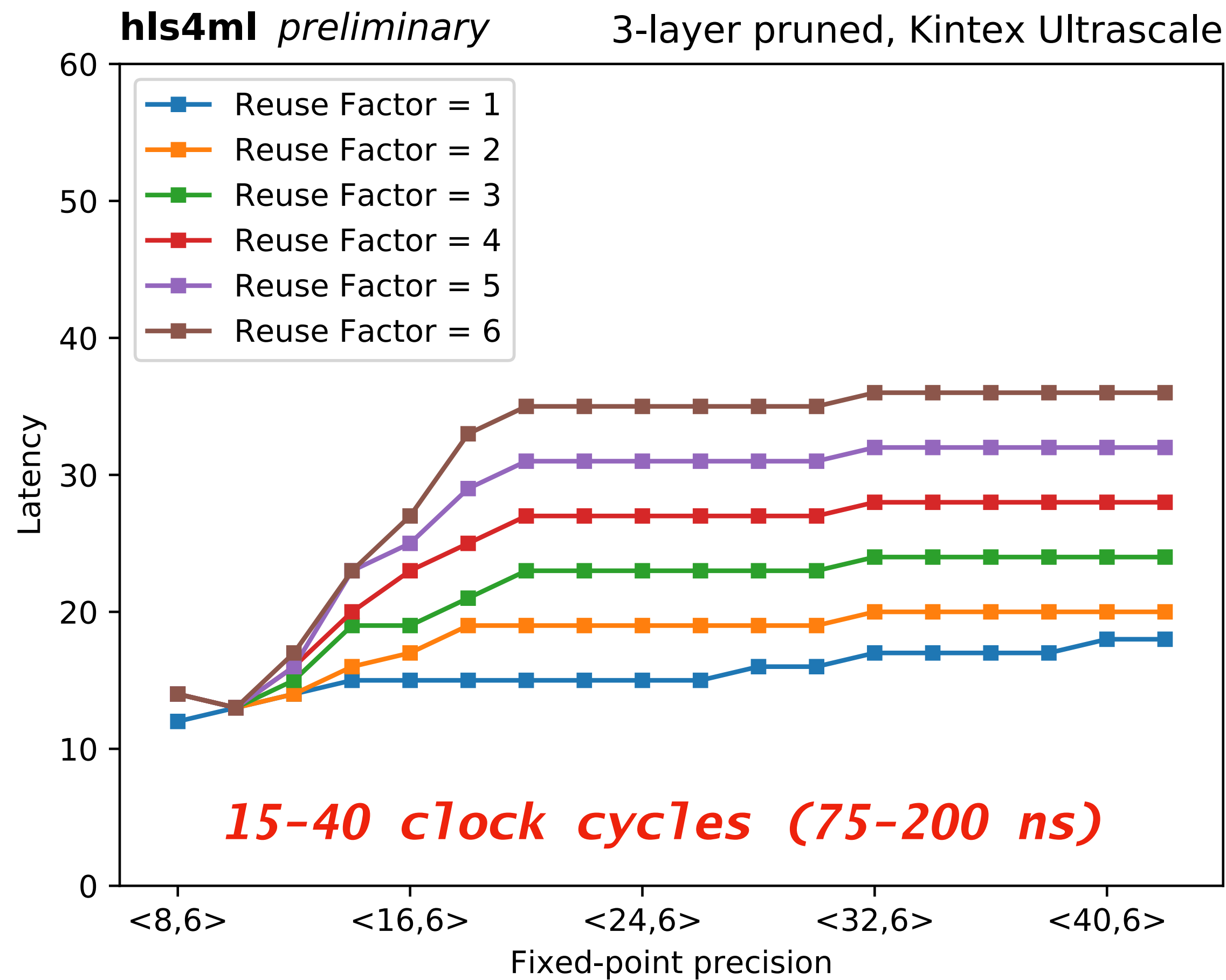


HLS4ML

- *HLS4ML aims to be this automatic tool*
 - *reads as input models trained on standard DeepLearning libraries*
 - *comes with implementation of common ingredients (layers, activation functions, etc)*
 - *Uses HLS softwares to provide a firmware implementation of a given network*
 - *Could also be used to create co-processing kernels for HLT environments*

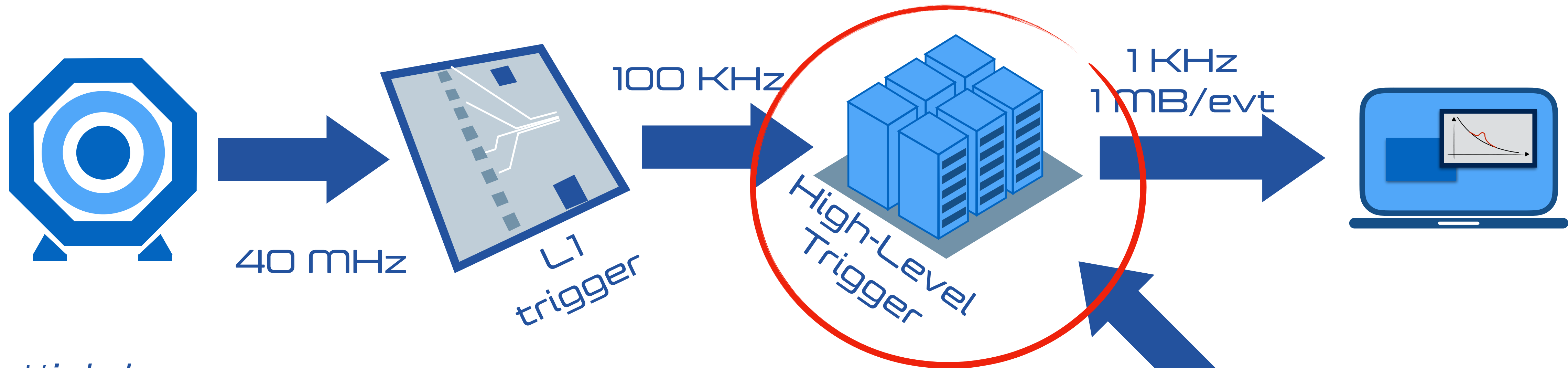


(ns) Fast Machine Learning

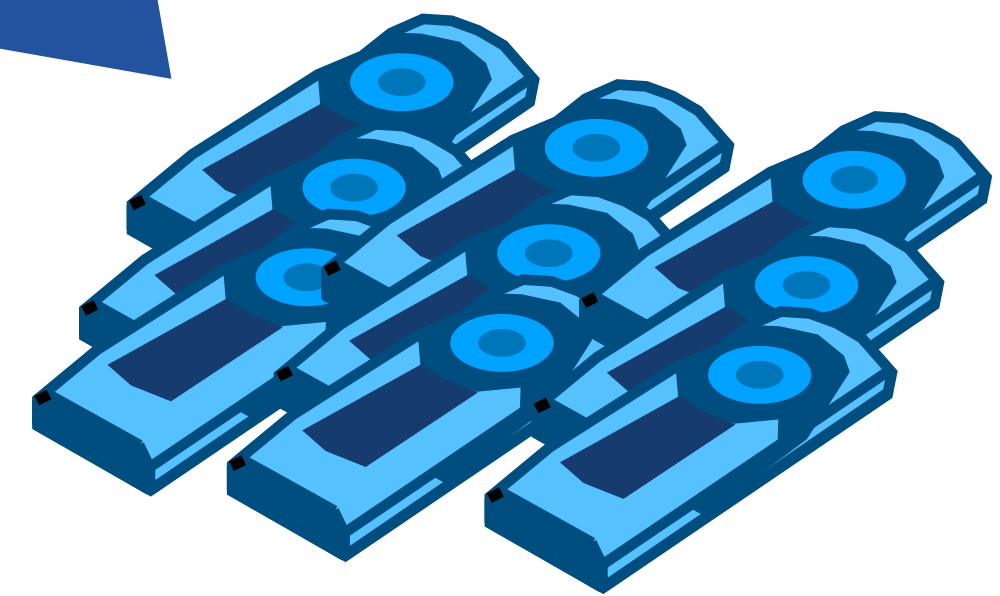
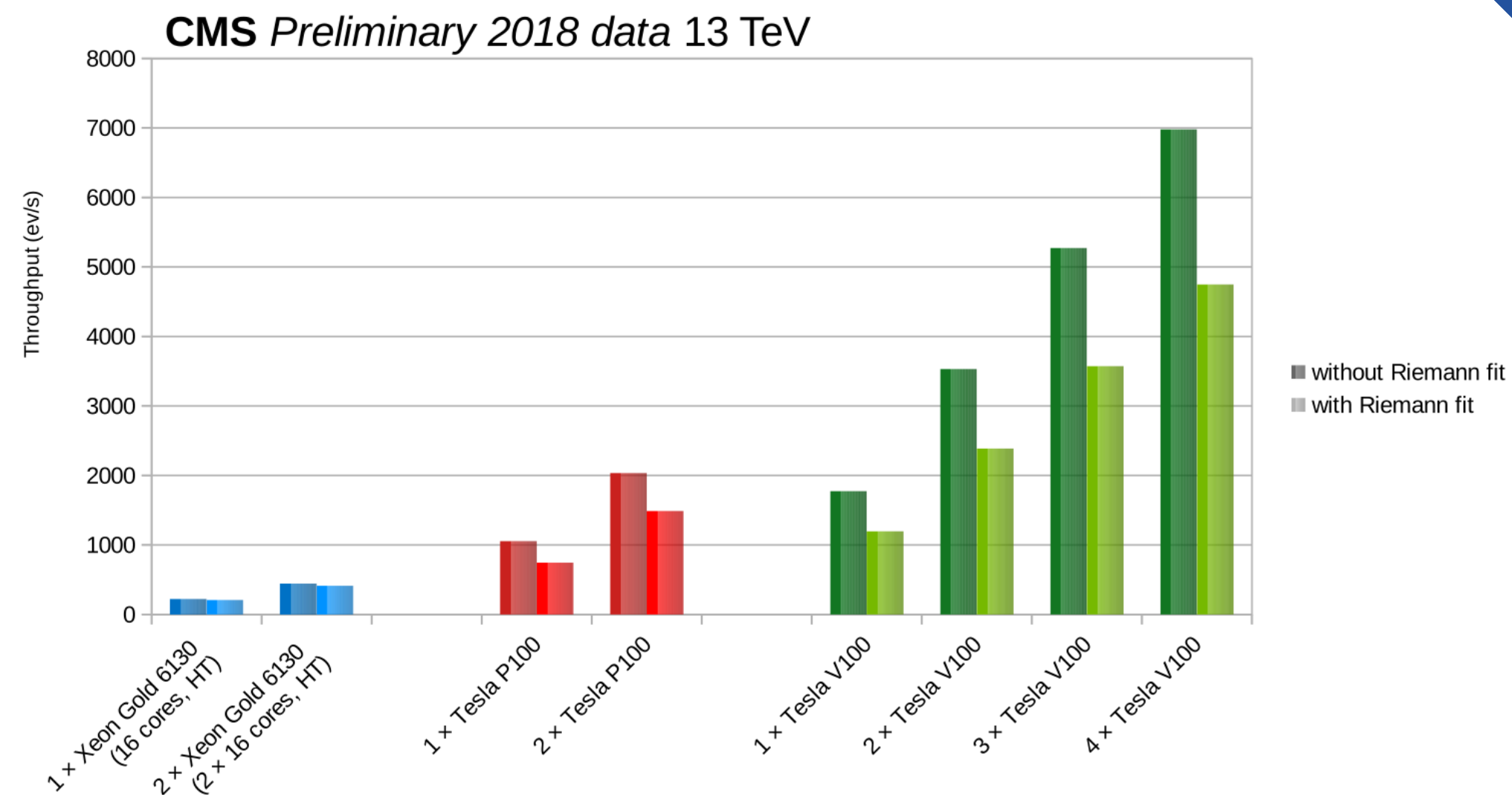


*Foreseen architecture (FPGAs) will handle these networks
 Inference-optimized GPUs could break the current paradigm
 Looking forward to R&D projects with nVidia & E4 on this*

Heterogeneous HLT

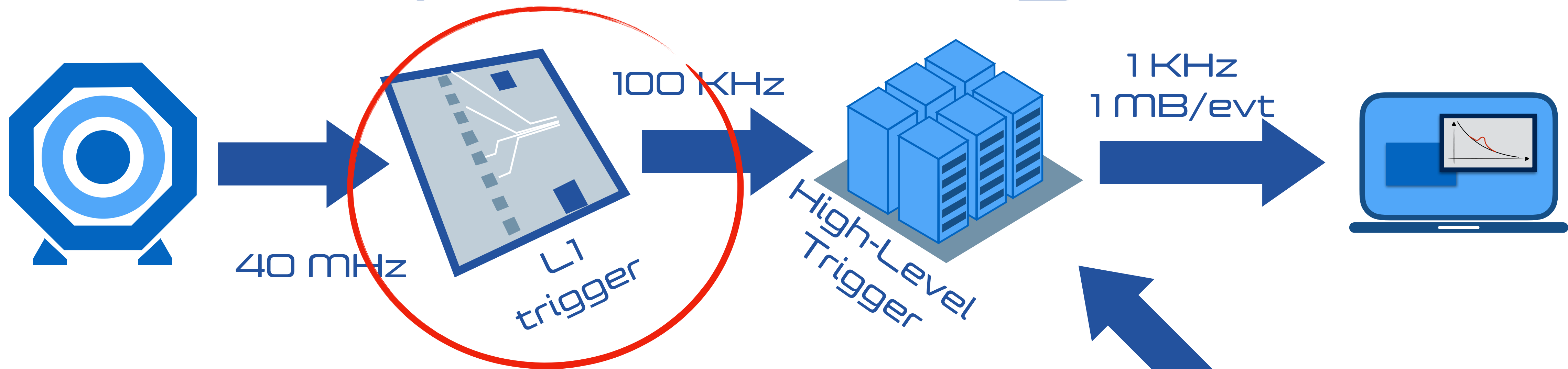


- With heterogeneous hardware in place (for other reasons) Deep Learning inference @HLT quite easy
- This will happen no matter what, to speed up traditional algorithms
- Deep Learning @HLT will benefit of it

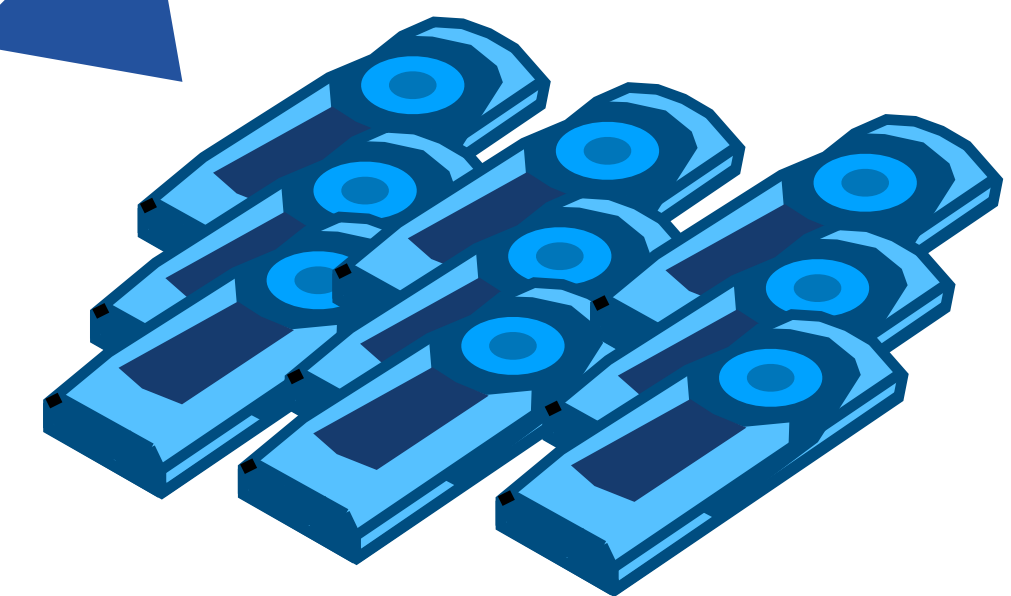


Patatrack project for CMS HLT on GPUs

Deep Learning at L1

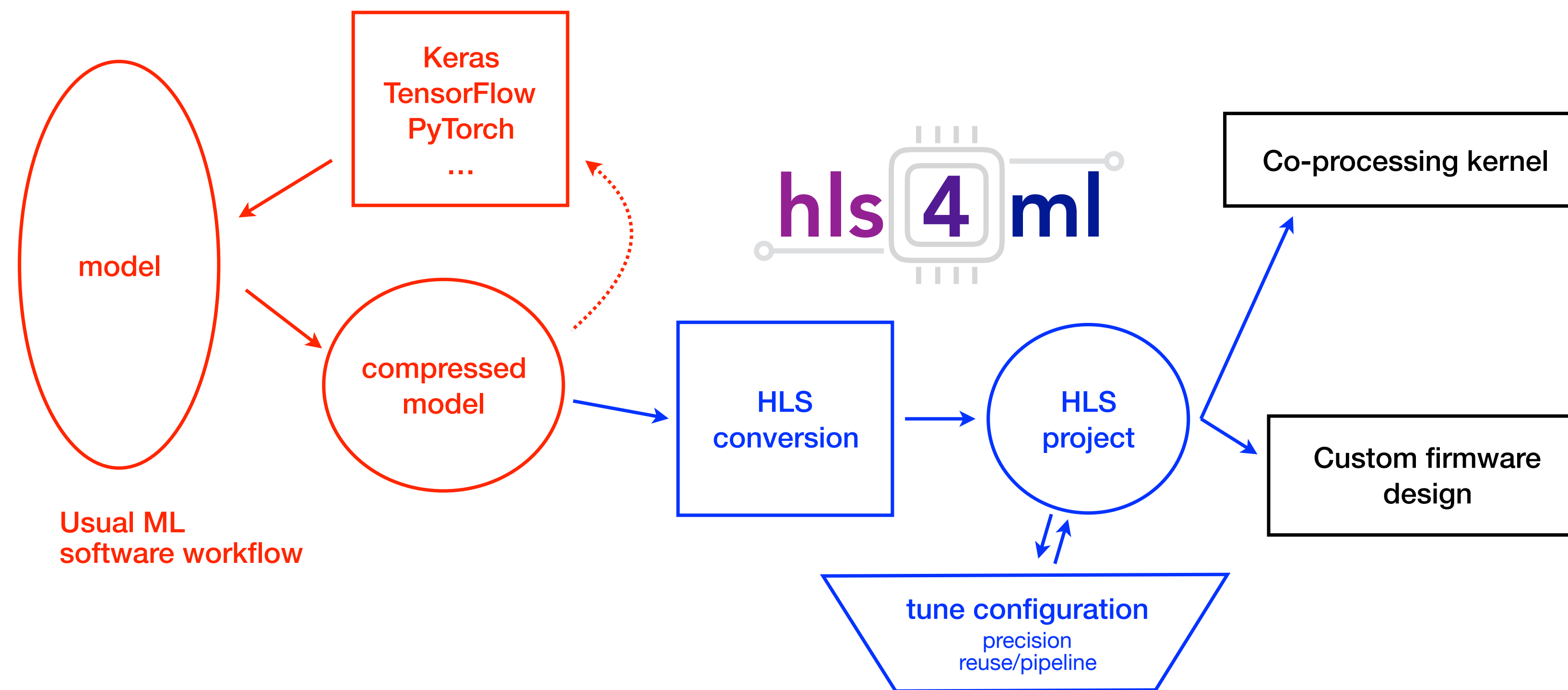


- ◉ *Situation at L1 is different, mainly due to the typical latency (<math><10 \mu\text{sec}</math>)*
- ◉ *Custom cards connected to detector electronics by optic links*
- ◉ *Data flow in the cards one by one*
- ◉ *Networks need to be implemented in FPGA firmware*
 - ◉ *advanced design by expert engineers (not common resource in HEP)*
 - ◉ *automatic translation tools doing the job*

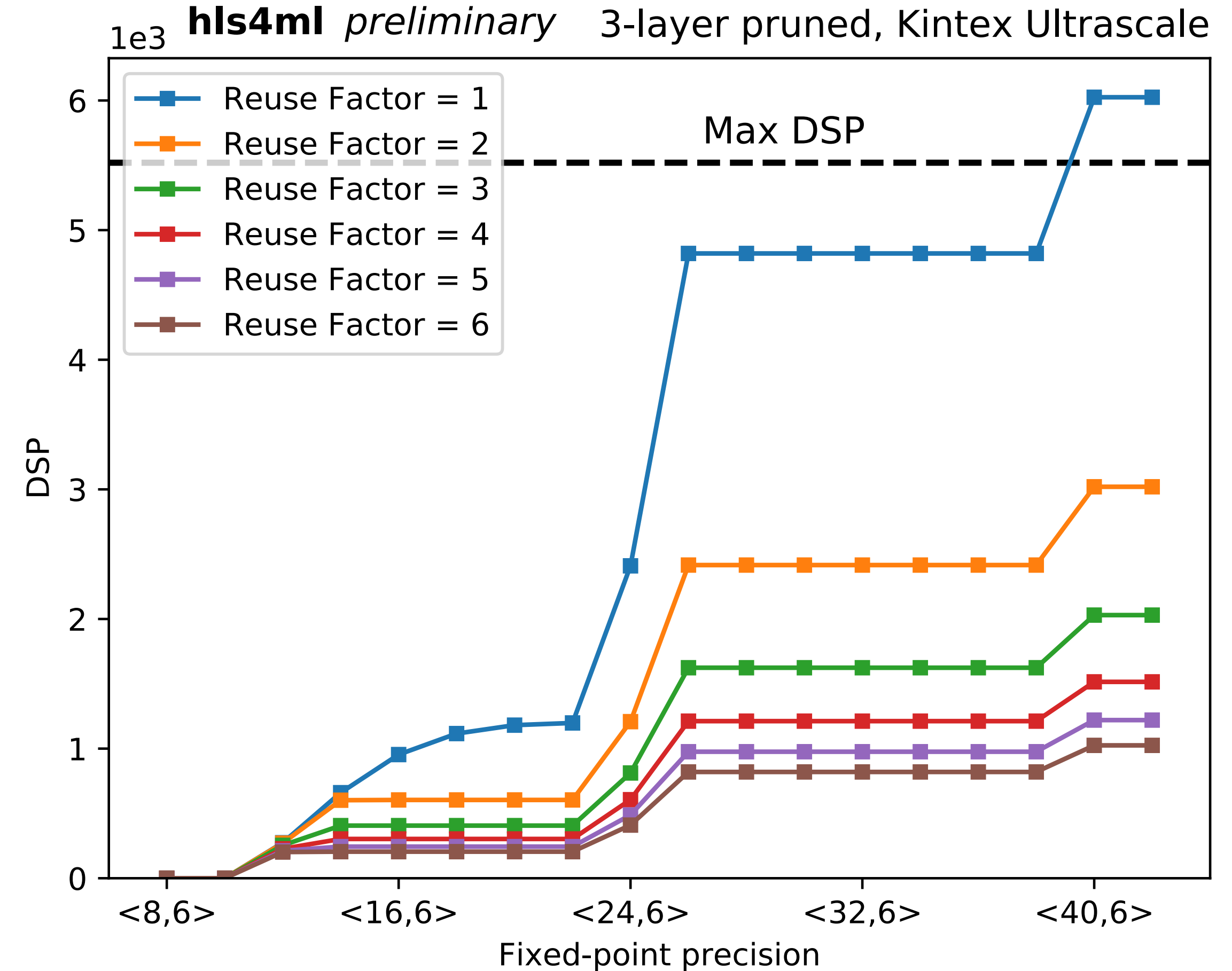
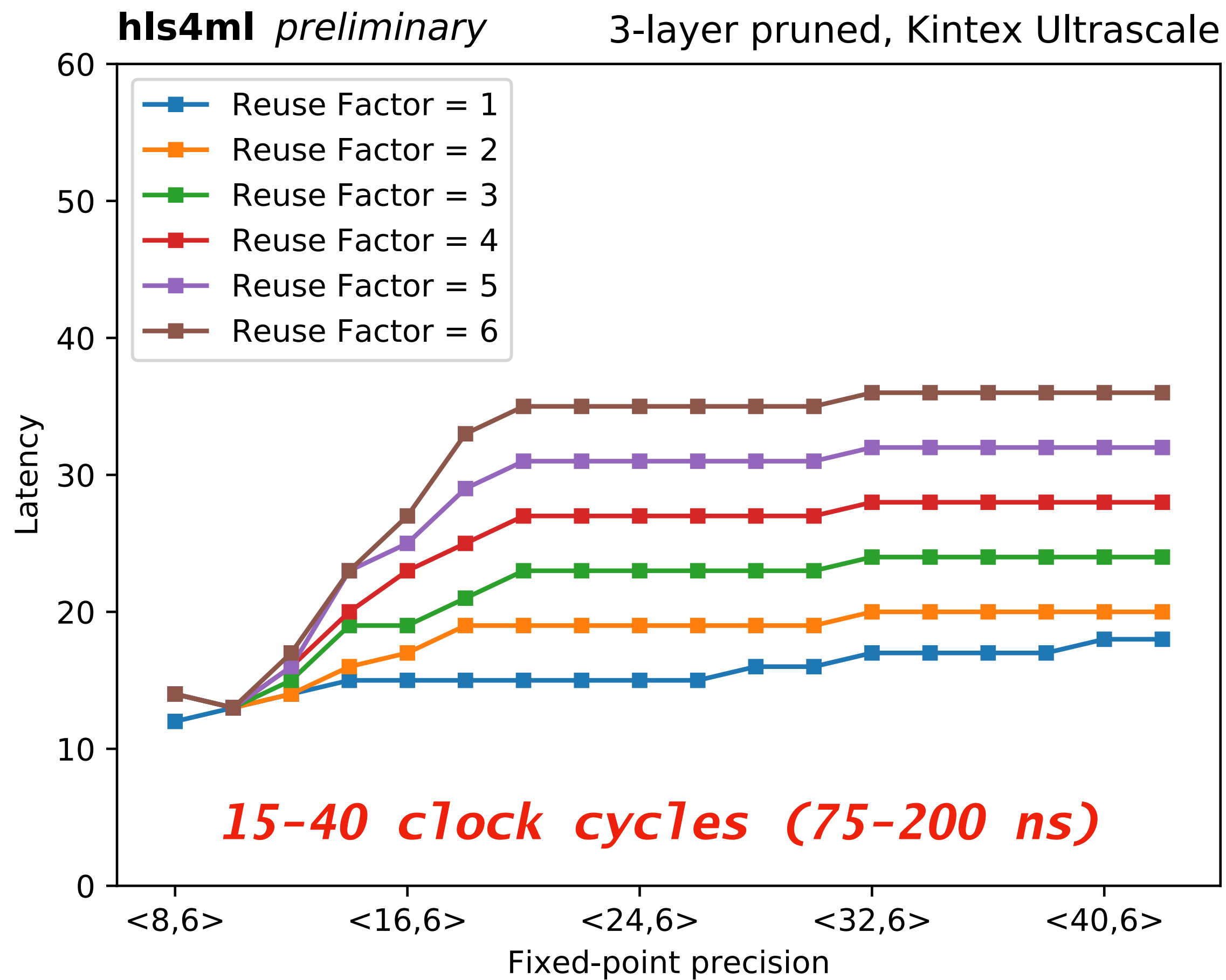


HLS4ML

- *HLS4ML aims to be this automatic tool*
 - *reads as input models trained on standard DeepLearning libraries*
 - *comes with implementation of common ingredients (layers, activation functions, etc)*
 - *Uses HLS softwares to provide a firmware implementation of a given network*
 - *Could also be used to create co-processing kernels for HLT environments*



Fast Inference



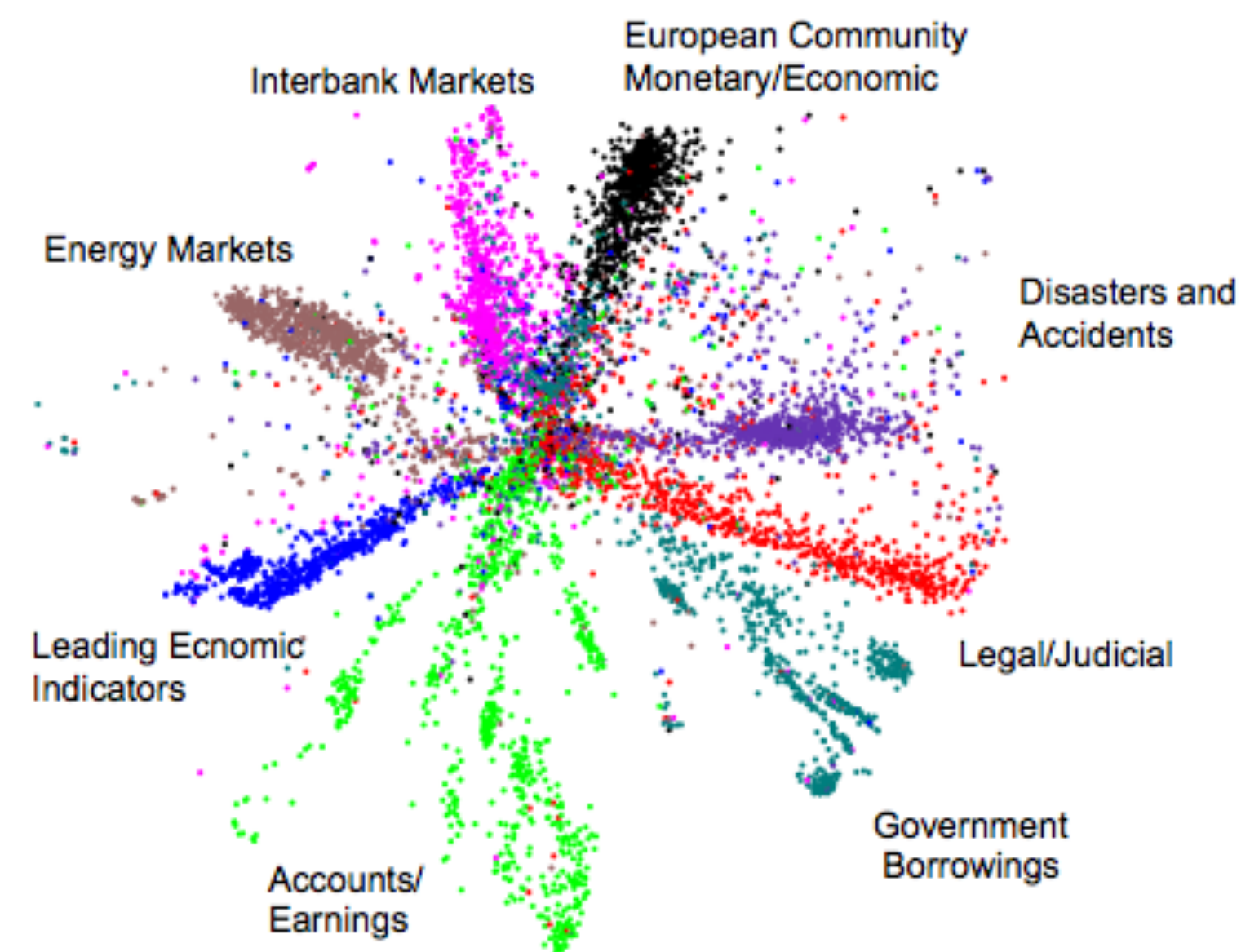
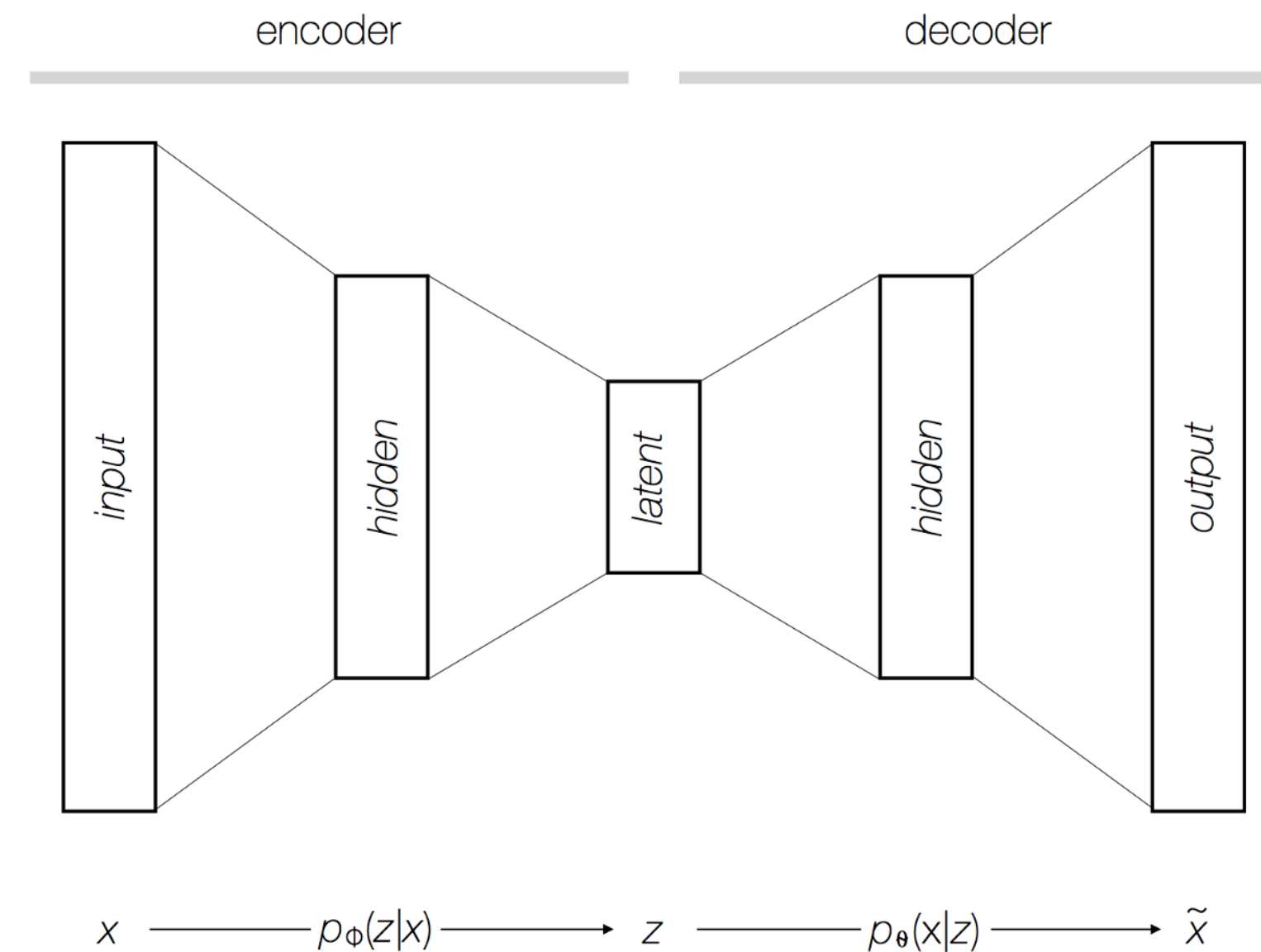
*Foreseen architecture (FPGAs) will handle these networks
 Inference-optimized GPUs could break the current paradigm
 Looking forward to R&D projects with nVidia & E4 on this*



Anomaly Detection

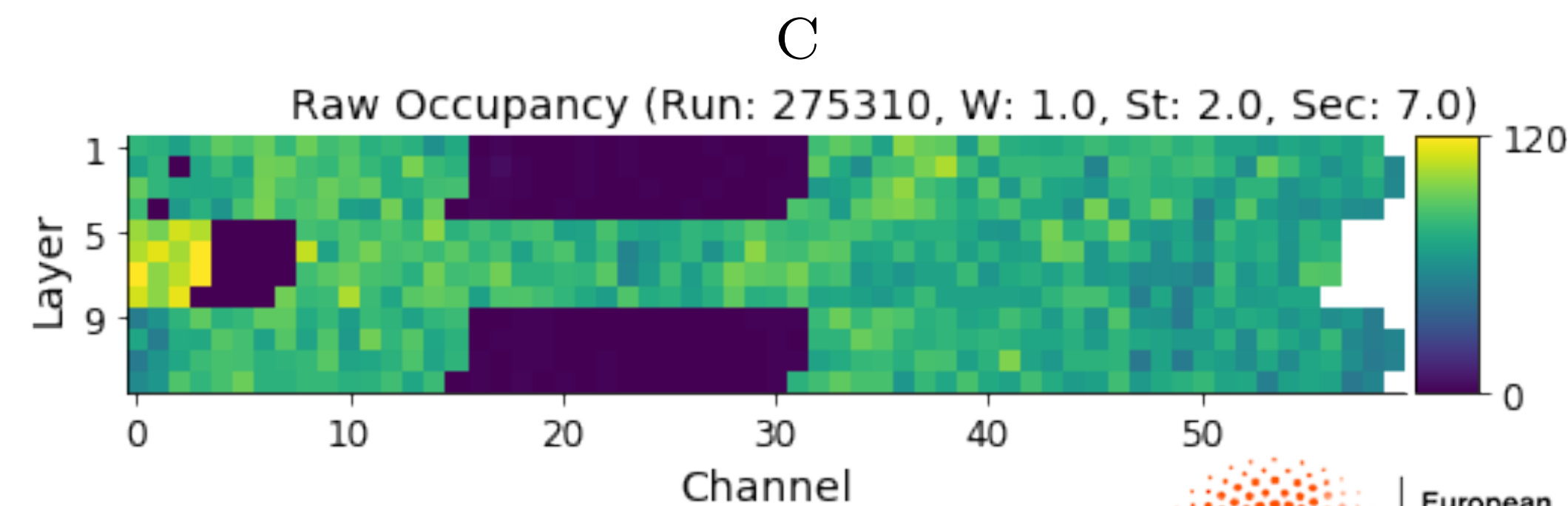
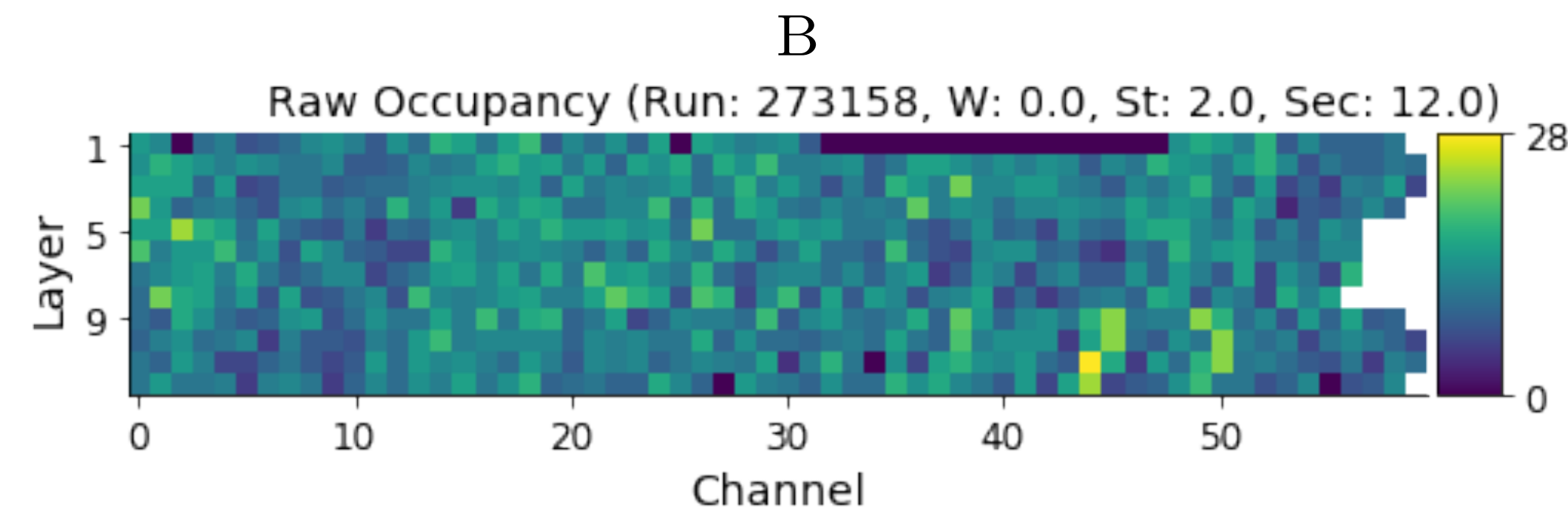
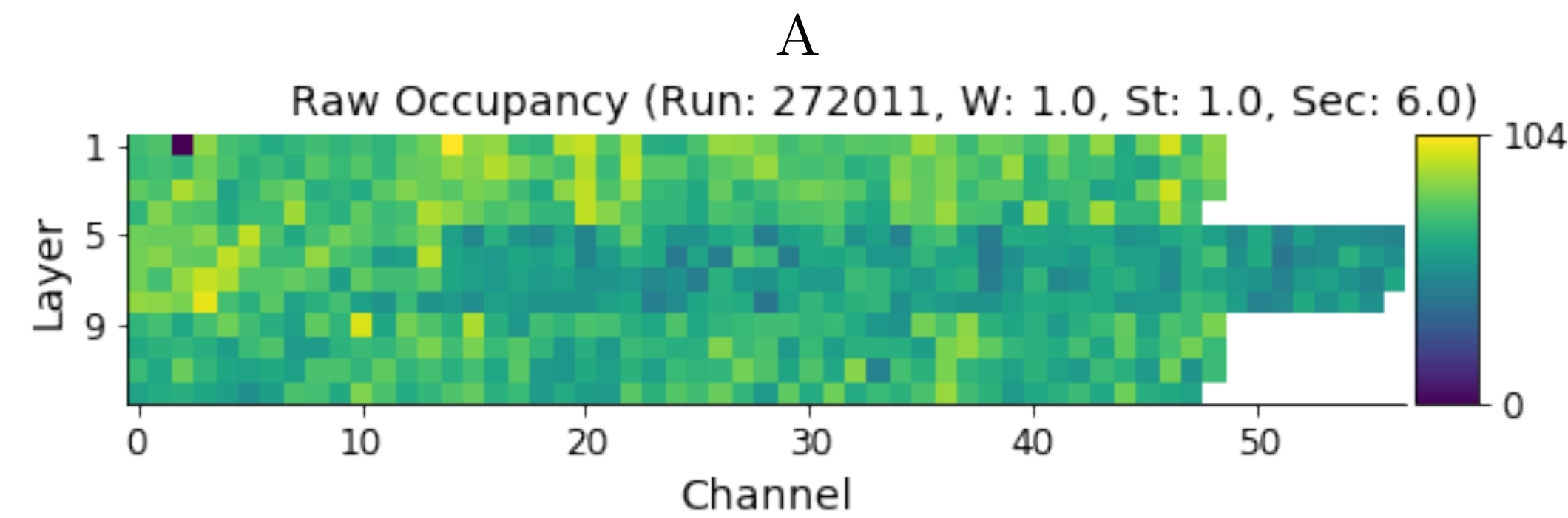
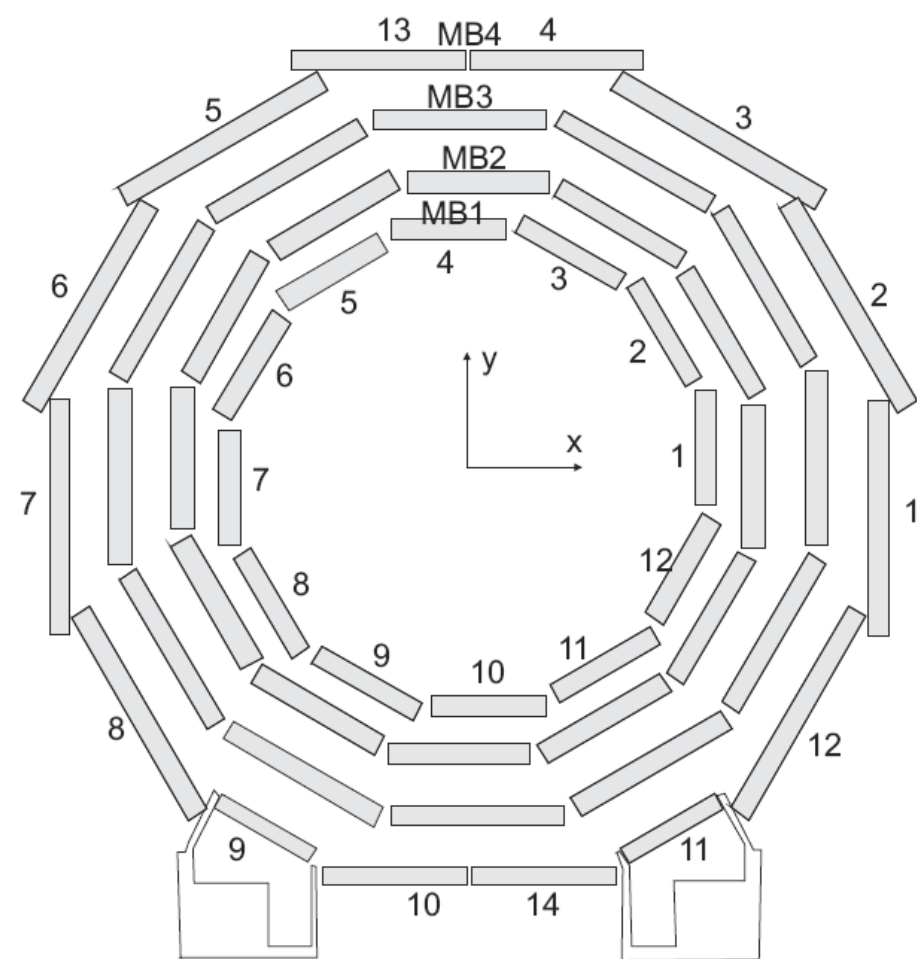
Autoencoders in a nutshell

- Autoencoders are compression-decompression algorithms that learn to describe a given dataset in terms of points in a lower-dimension latent space
- UNSUPERVISED algorithm, used for data compression, generation, clustering (replacing PCA), etc.
- Used in particular for anomaly detection: when applied on events of different kind, compression-decompression tuned on refer sample might fail
- One can define anomalous any event whose decompressed output is “far” from the input, in some metric (e.g., the metric of the auto-encoder loss)



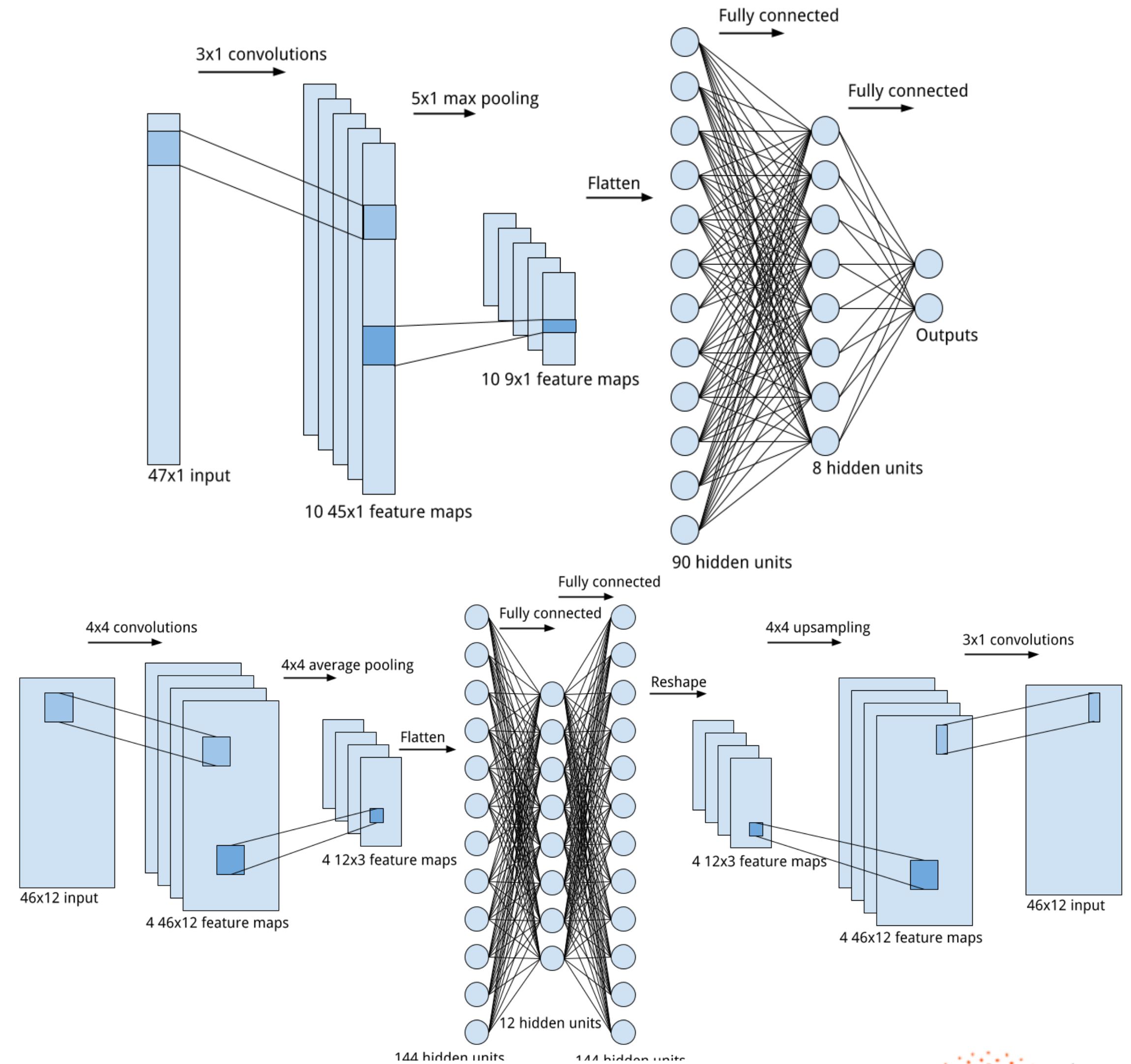
Example: Data Quality Monitoring

- When taking data, >1 person watches for anomalies in the detector 24/7
- At this stage no global processing of the event
- Instead, local information from detector components available (e.g., detector occupancy in a certain time window)



Example: Data Quality Monitoring

- Given the nature of these data, ConvNN are a natural analysis tool. Two approaches pursued
- Classify good vs bad data. Works if failure mode is known
- Use autoencoders to assess data “typicality”. Generalises to unknown failure modes

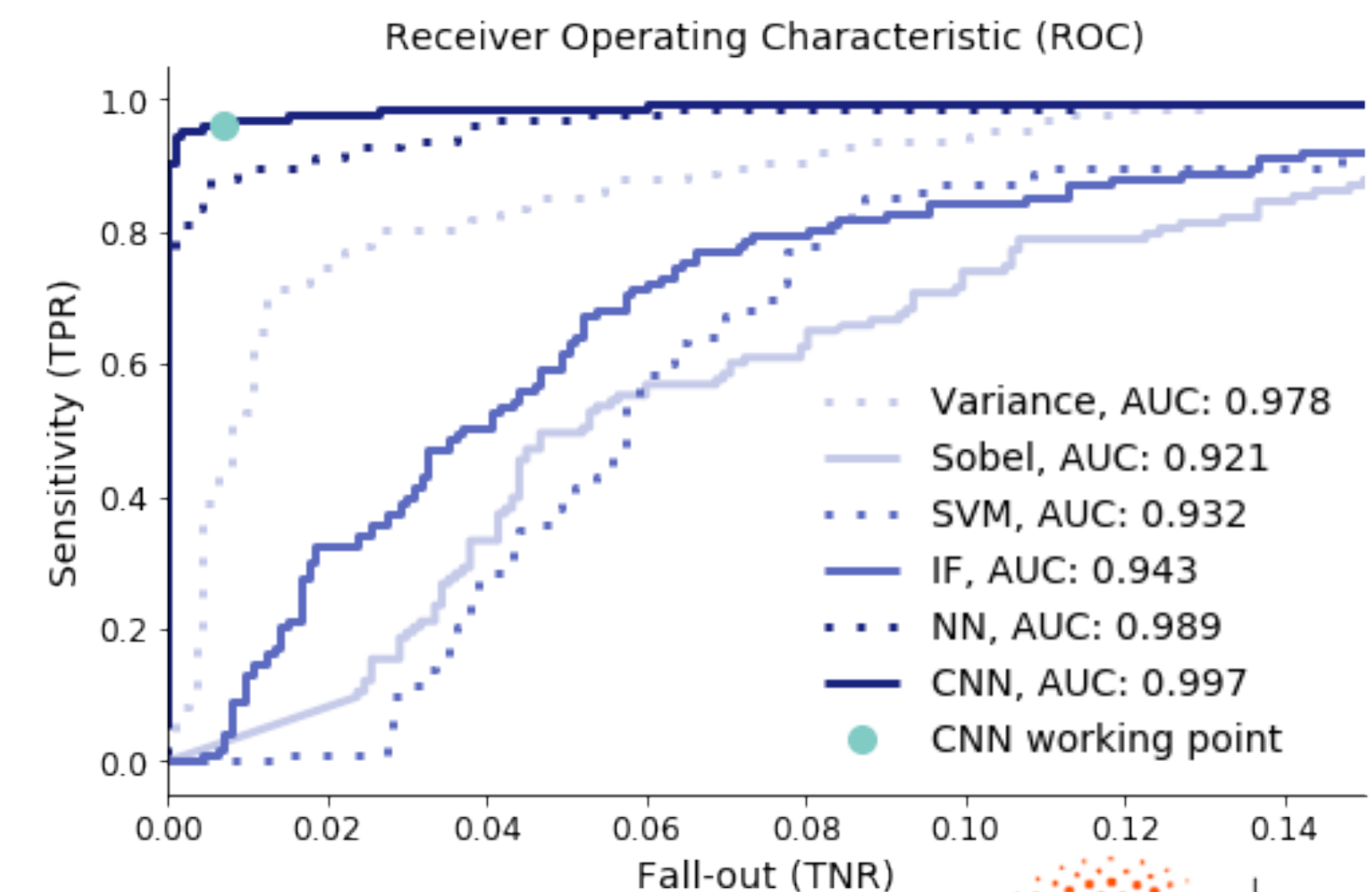
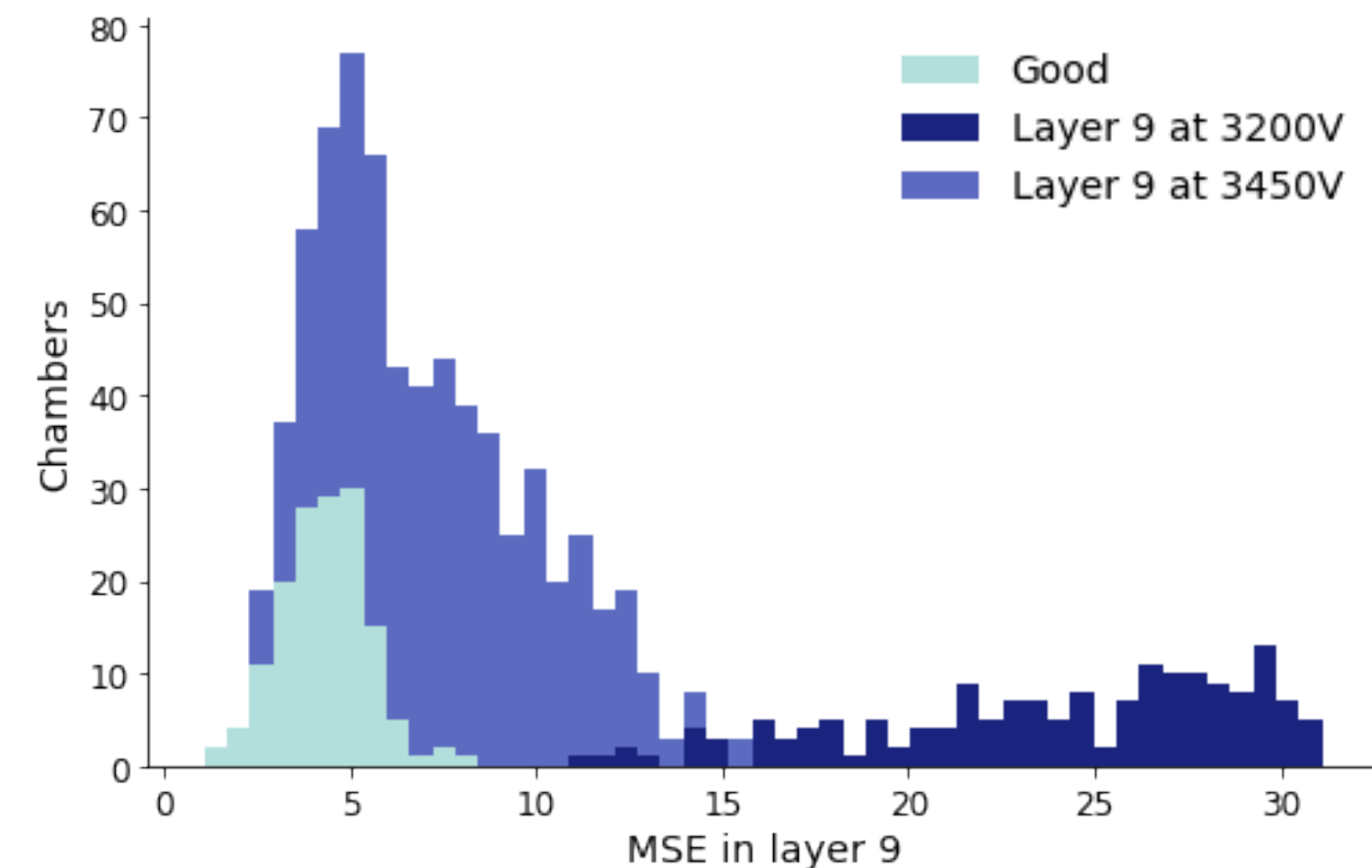


A. Pol et al., to appear soon

[Pol, G. Cerminara, C. Germain, MP and A. Seth arXiv:1808.00911](#)

Example: Data Quality Monitoring

- Given the nature of these data, ConvNN are a natural analysis tool. Two approaches pursued
- Classify good vs bad data. Works if failure mode is known
- Use autoencoders to assess data “typicality”. Generalises to unknown failure modes

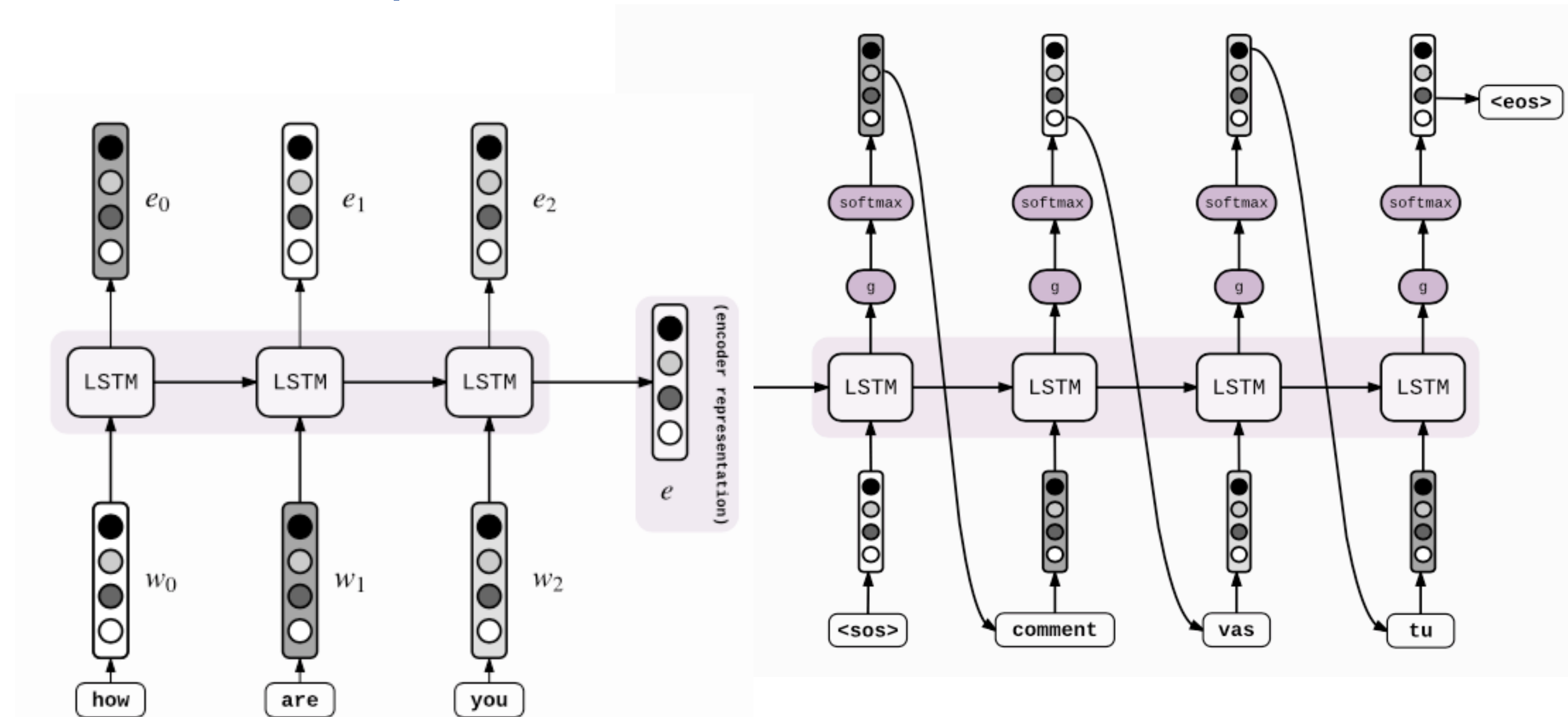


A. Pol et al., to appear soon

[Pol, G. Cerminara, C. Germain, MP and A. Seth arXiv:1808.00911](#)

VAE with PF particles

- *Issues:*
 - *variable number of particles/event as input*
 - *need to return particles as output*
- *Networks used for translation*
 - *start from a sentence in language*
 - *code its meaning in some latent space z*
 - *translate to some other language, generating words from z*

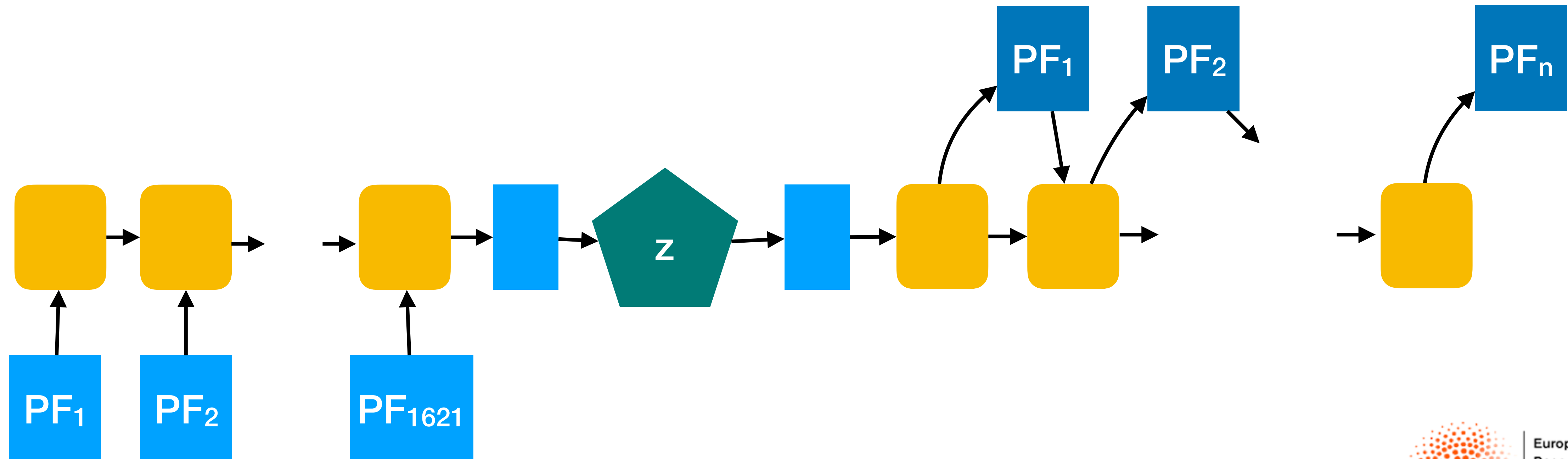


VAE with PF particles

⦿ *Issues:*

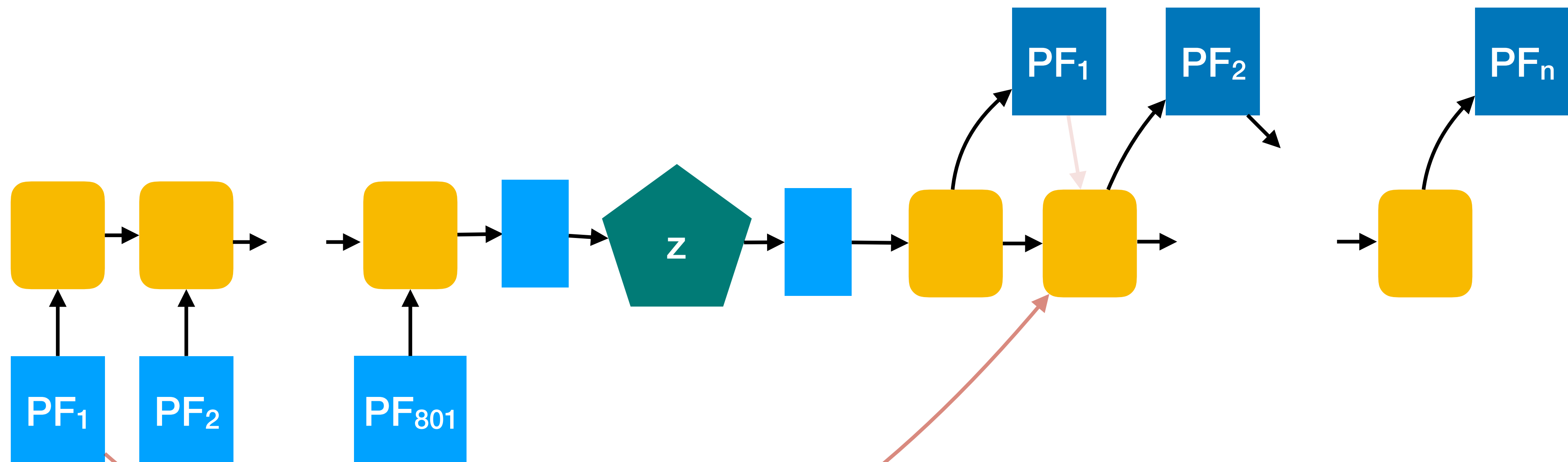
⦿ *variable number of particles/event as input*

⦿ *need to return particles as output*



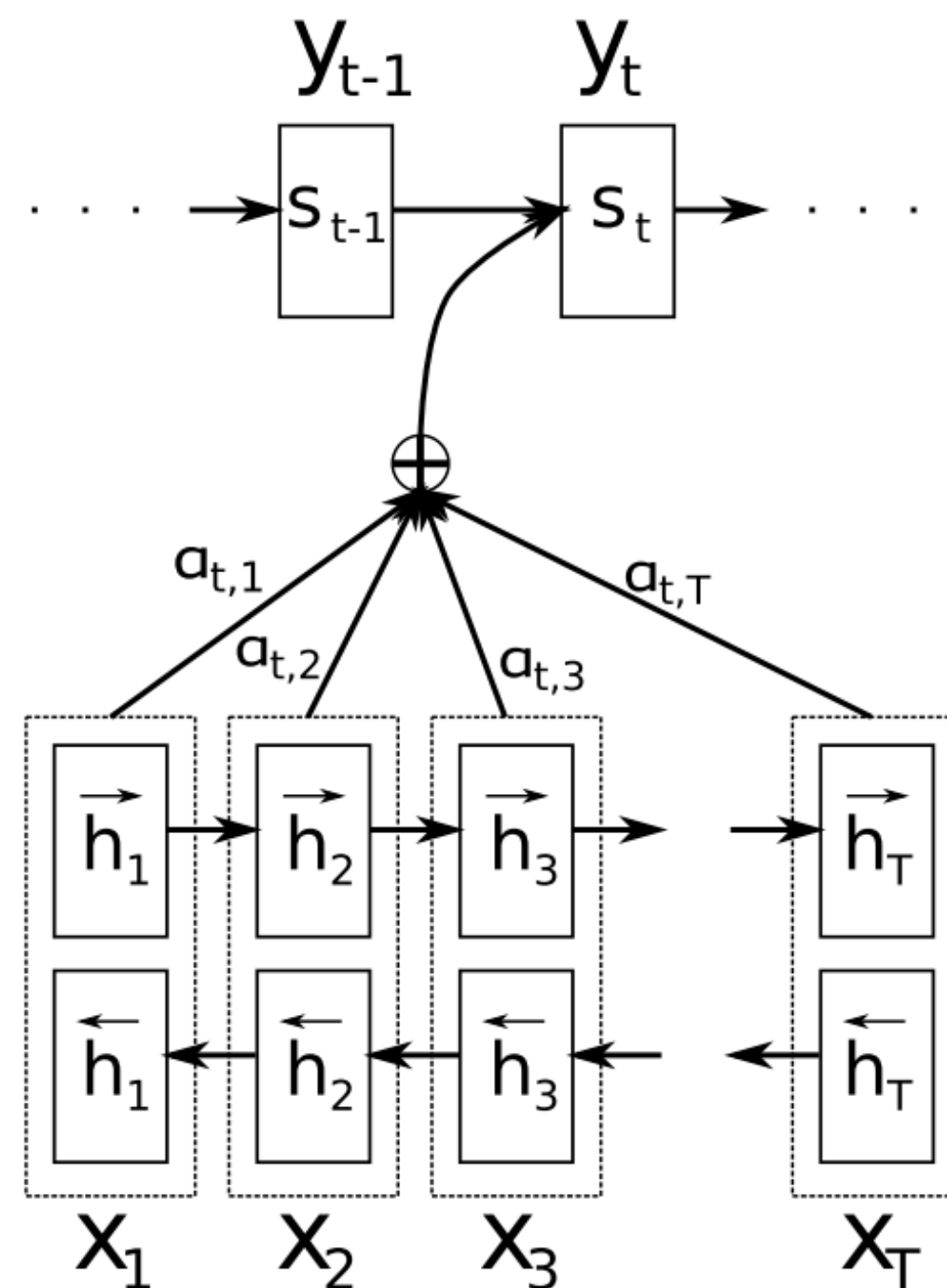
Teacher forcing

- At early stage of training, the decoder can't reconstruct a reasonable first PF candidate; autoregressive mechanism propagates it into a wrong chain of particles.
- Teacher-forcing:** under some probability k , feed the target as the next input instead of using the previous prediction. k decreases as the epoch number increases.



Adding Attention

- Attention allows the decoder to focus on which part of the inputs is relevant to the next prediction.



the **Encoder** generates $h_1, h_2, h_3, \dots, h_T$ from the inputs $X_1, X_2, X_3, \dots, X_T$

a is the **Alignment model** which is a **feedforward neural network** that is trained with all the other components of the proposed system

$$e_{ij} = a(s_{i-1}, h_j)$$

The **Alignment model** scores (e) how well each encoded input (h) matches the current output of the decoder (s).

The alignment scores are normalized using a **softmax function**.

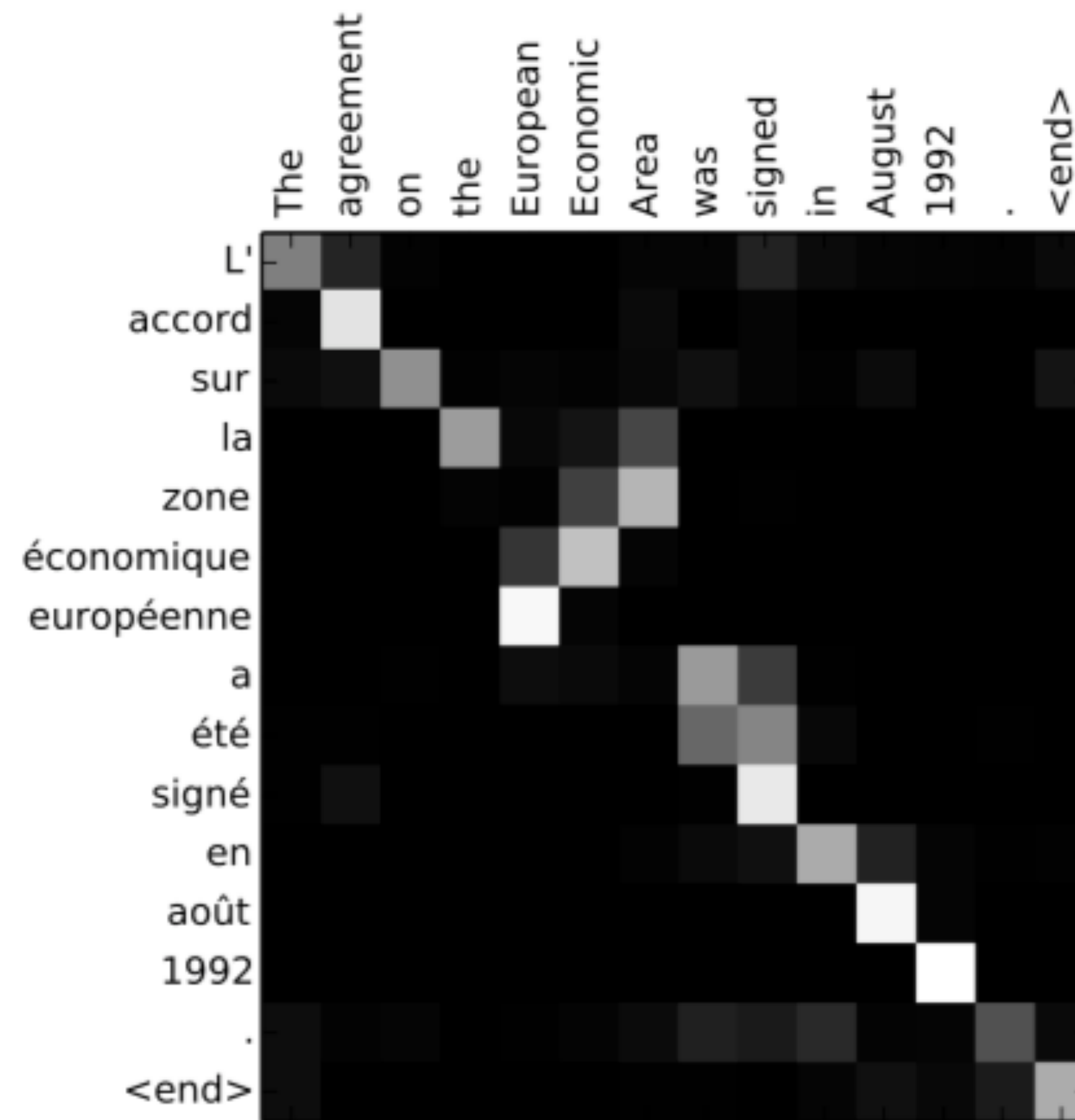
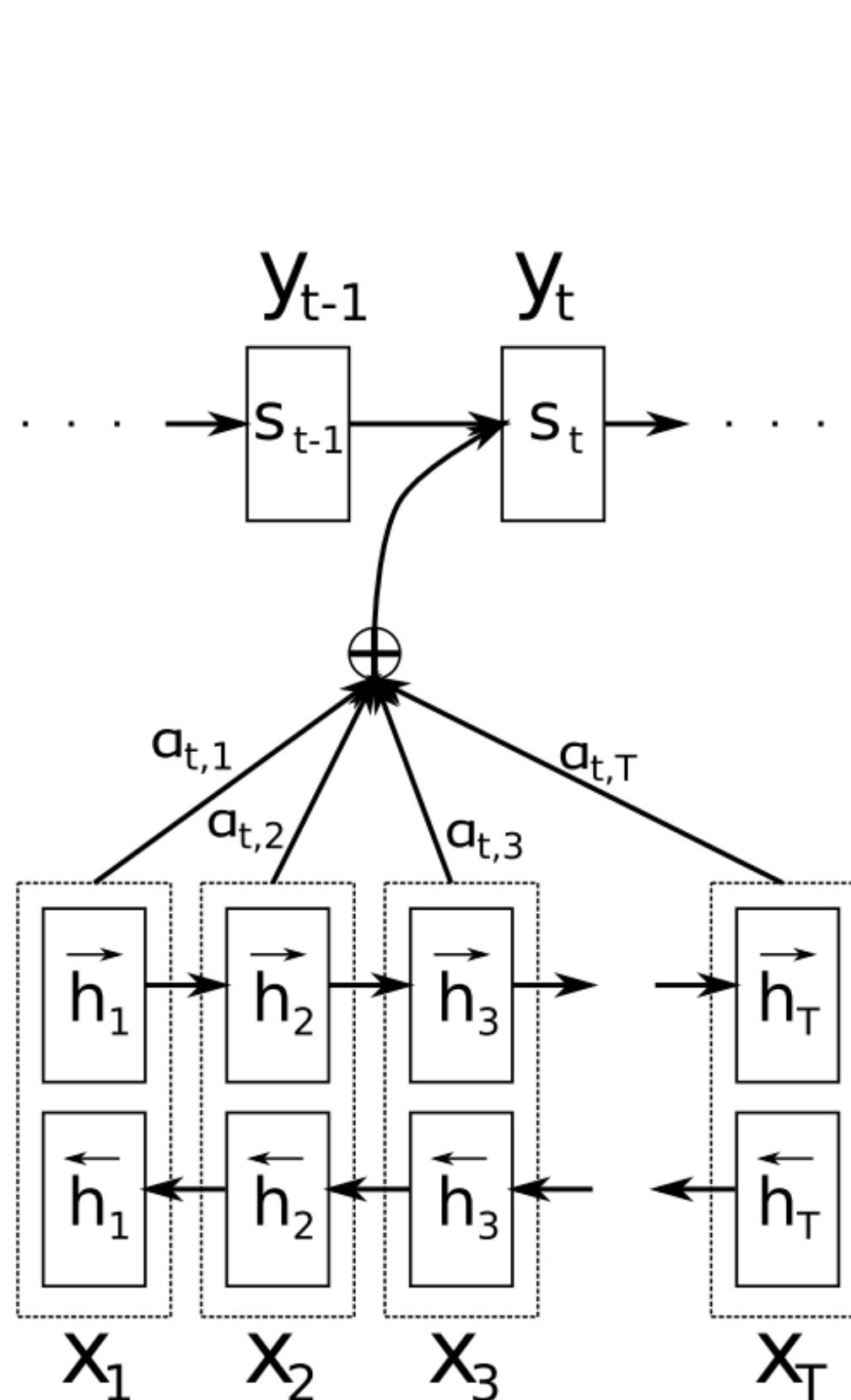
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

The context vector is a weighted sum of the **annotations** (h_j) and **normalized alignment scores**.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

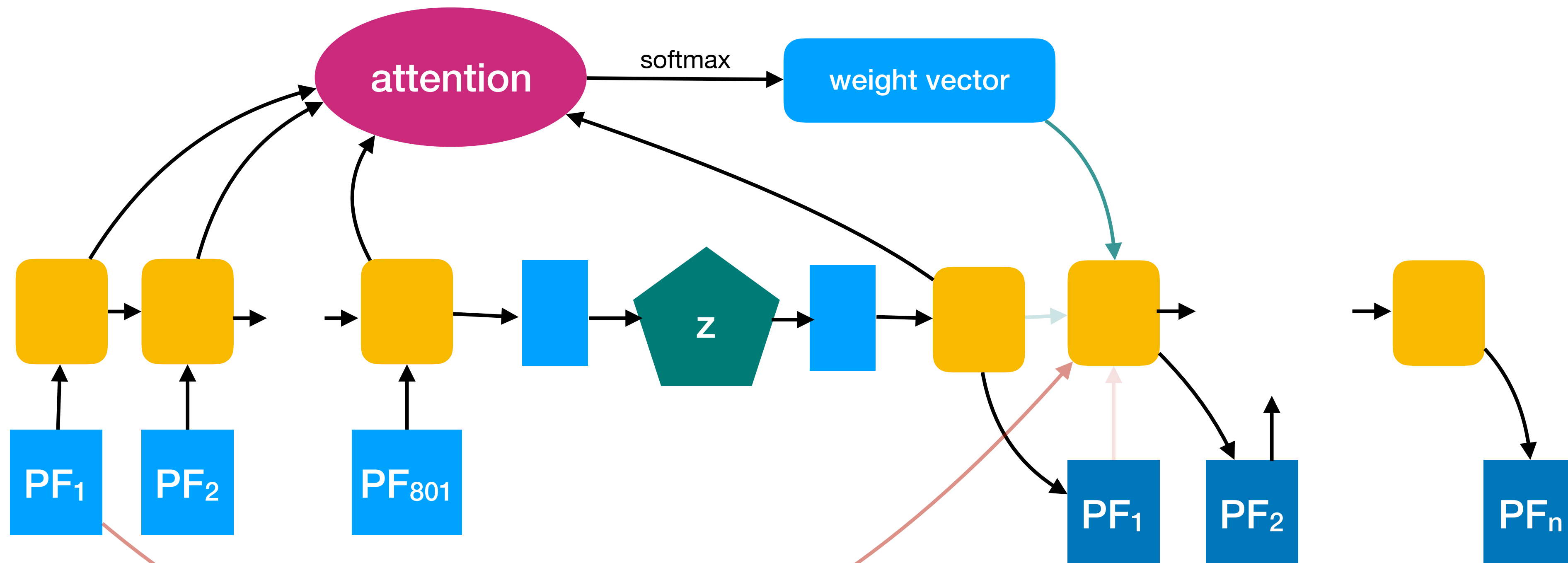
Adding Attention

- Attention allows the decoder to focus on which part of the inputs is relevant to the next prediction.



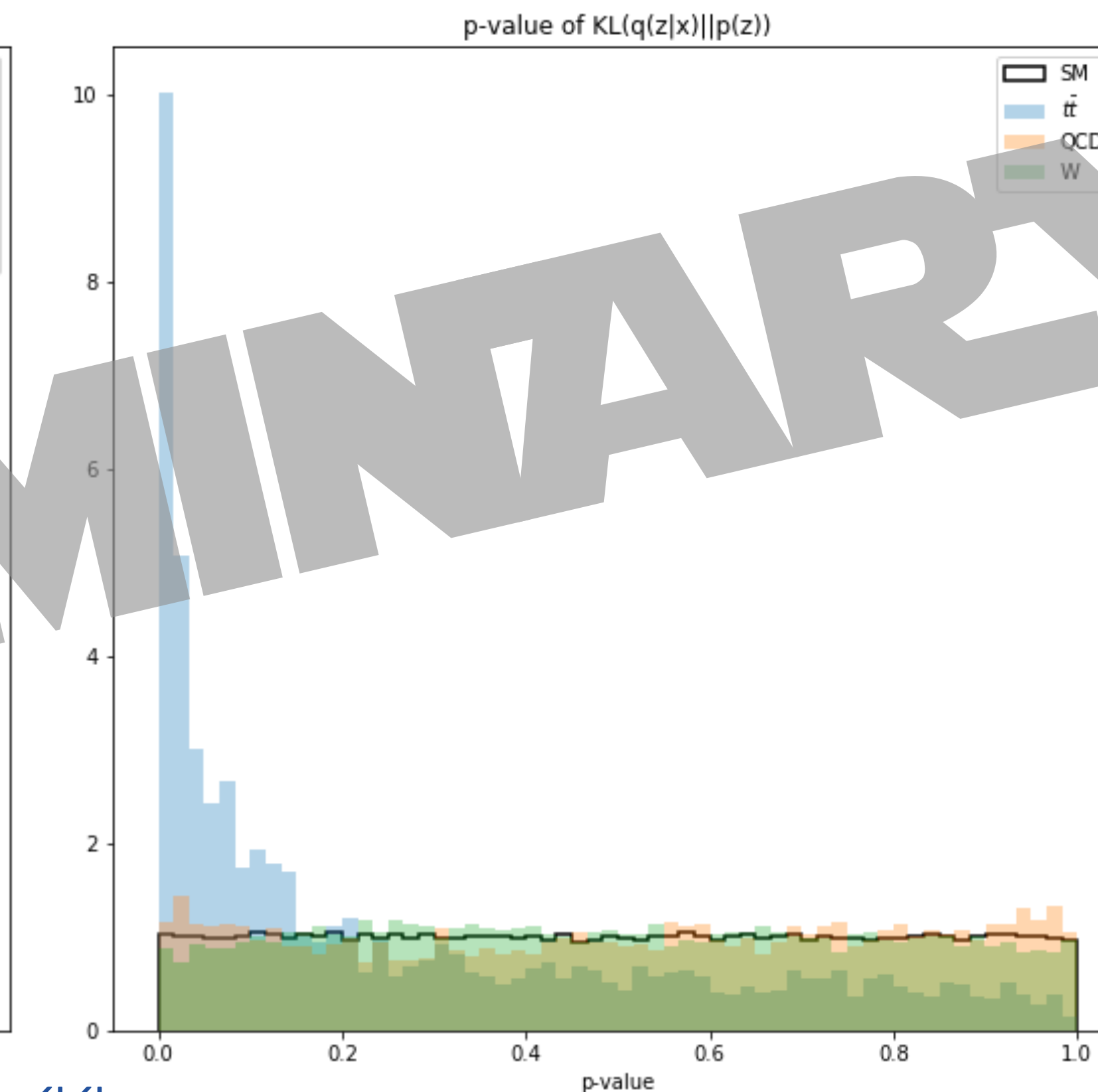
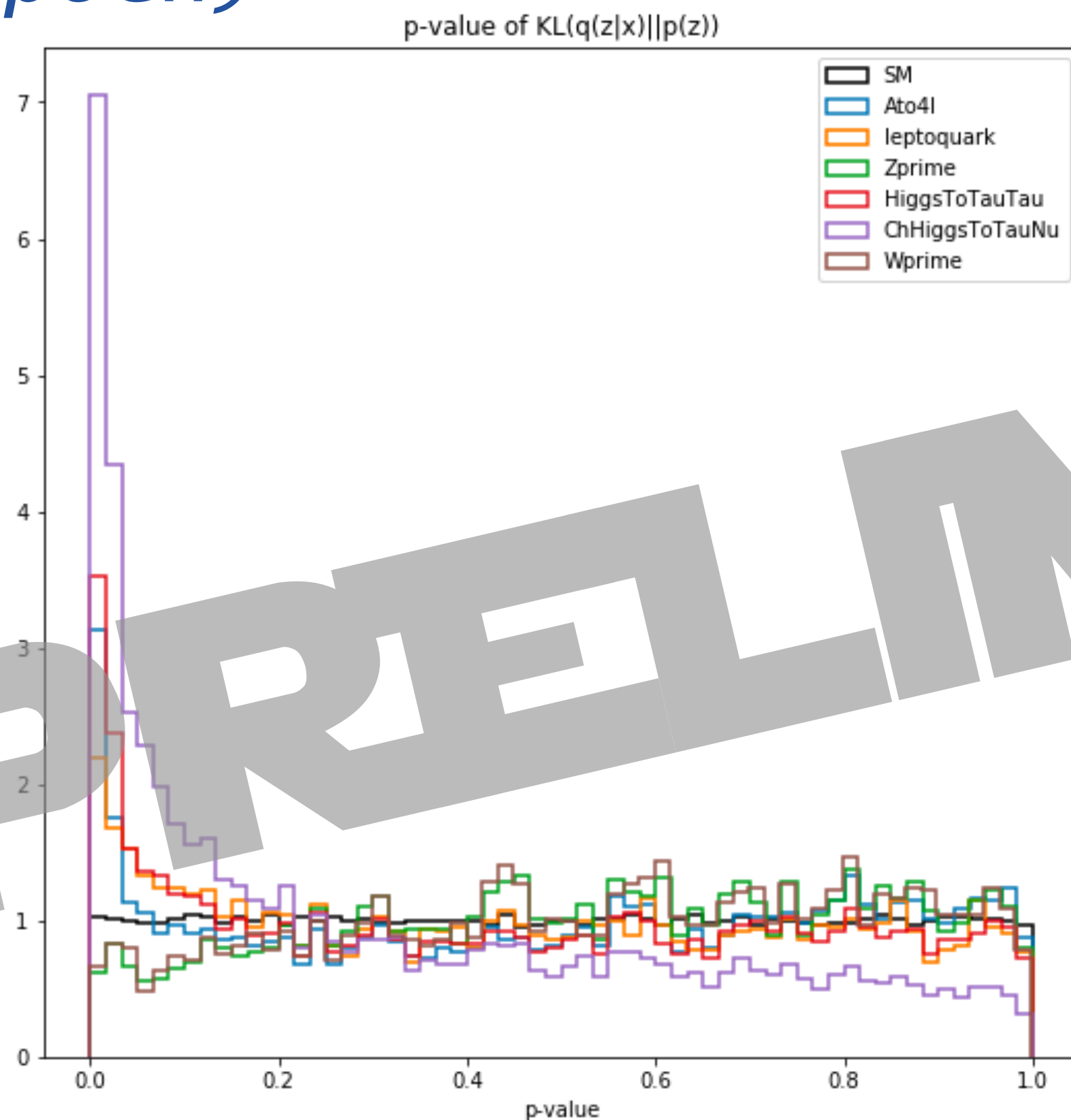
Adding Attention

- Attention allows the decoder to focus on which part of the inputs is relevant to the next prediction.



Performances

- (Preliminary) results trained on a small subset of the initial dataset (90K events)
- Due to architecture complexity, training is much slower (6h/epoch)



PRELIMINARY

Confidence on data

I am not sure I understand

- ◎ *If you mean “confidence in the quality of the data” then the autoencoder approach would serve you there*
- ◎ *We say that “data are data”. The confidence should be in your expectations of the data*
 - ◎ *We have a ~5% precise simulation of the full process (from collision to electronic signal)*
 - ◎ *I assume that you have your own simulation tools (for aerodynamic etc.)*
- ◎ *Is the question related to data/simulation agreement?*



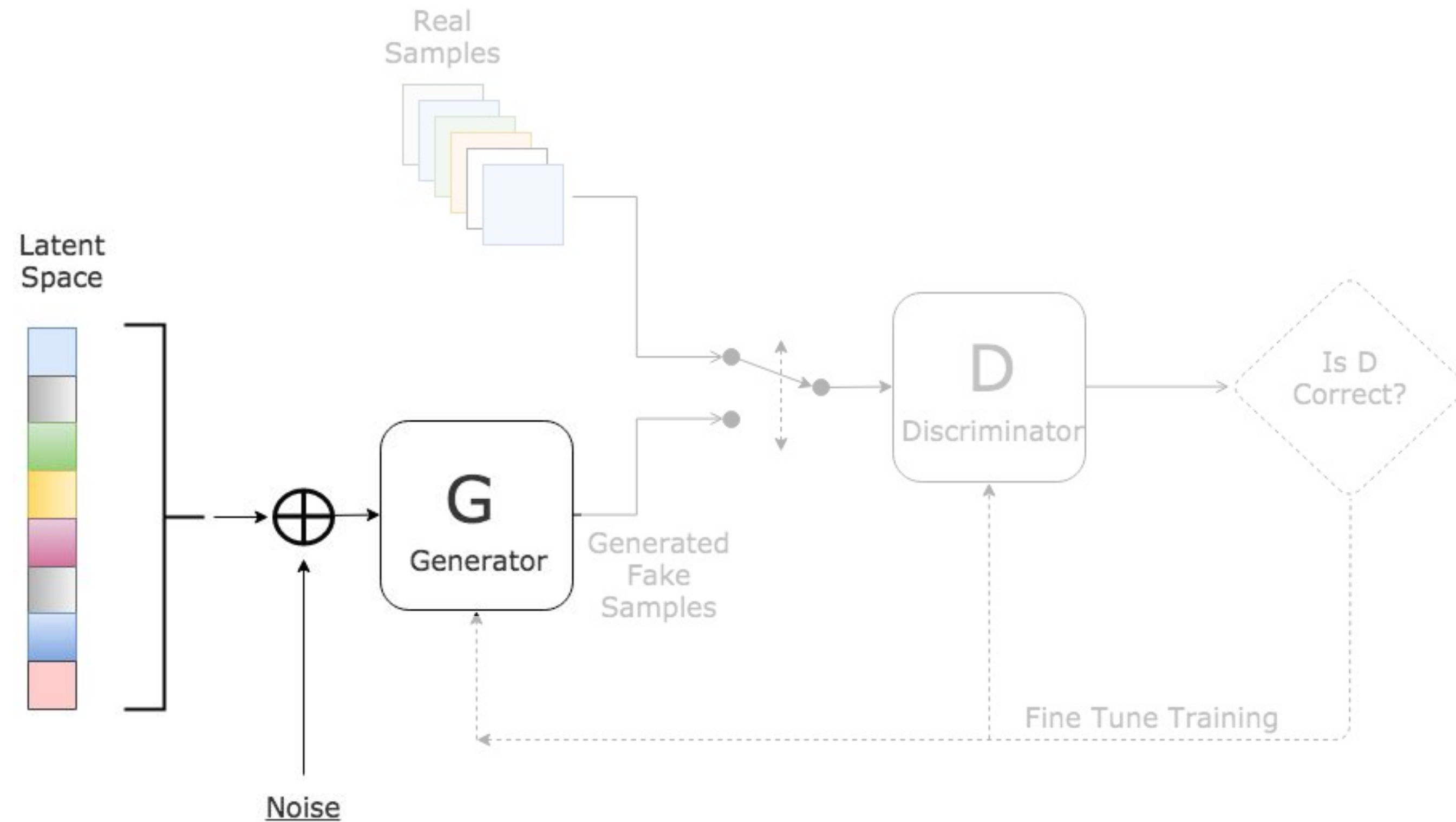
Generating large datasets
with small resources

Generative Adversarial Training

- Two networks trained against each other

- Generator:** create images (from noise, other images, etc)

- Discriminator:** tries to spot which image comes from the generator and which is genuine



- Loss function to minimise: $Loss(Gen) - Loss(Disc)$

- Better discriminator -> bigger loss

- Better generator -> smaller loss

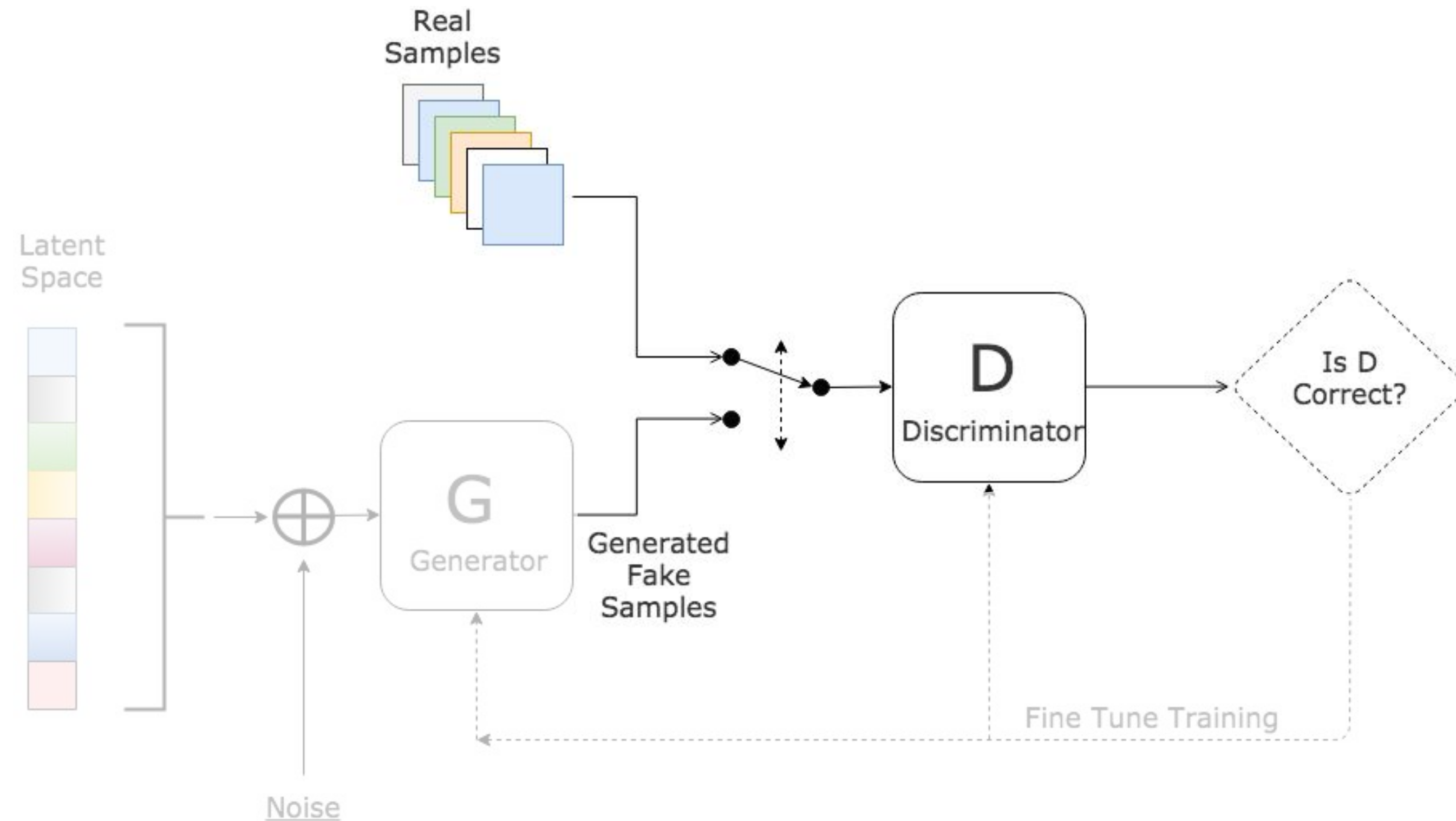
- Trying to fool the discriminator, generator learns how to create more realistic images

Generative Adversarial Training

- Two networks trained against each other

- Generator:** create images (from noise, other images, etc)

- Discriminator:** tries to spot which image comes from the generator and which is genuine



- Loss function to minimise: $Loss(Gen) - Loss(Disc)$

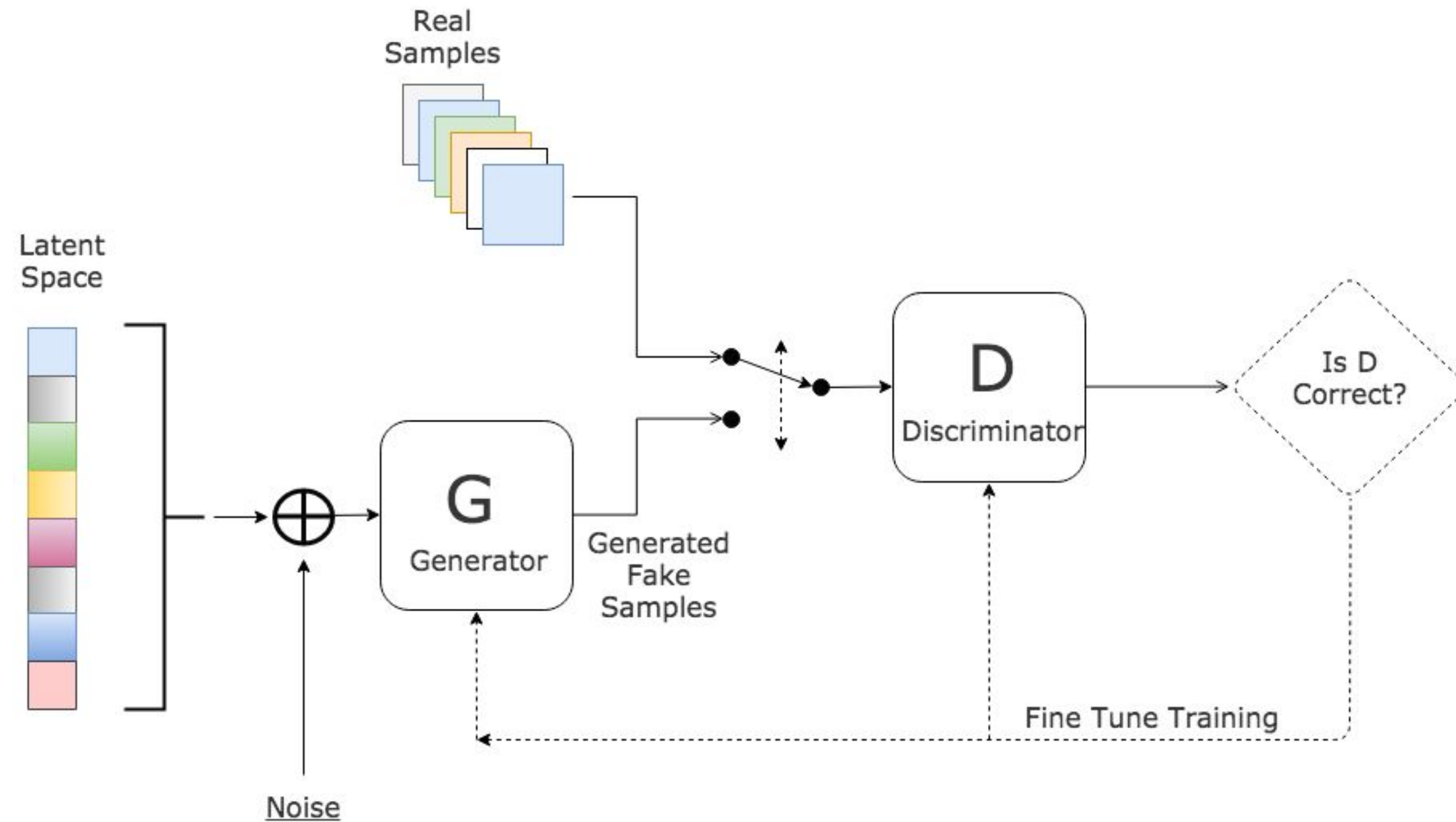
- Better discriminator -> bigger loss

- Better generator -> smaller loss

- Trying to fool the discriminator, generator learns how to create more realistic images

Generative Adversarial Training

- Two networks trained against each other
- Generator: create images (from noise, other images, etc)
- Discriminator: tries to spot which image comes from the generator and which is genuine



- Loss function to minimise: $Loss(Gen) - Loss(Disc)$
 - Better discriminator -> bigger loss
 - Better generator -> smaller loss
- Trying to fool the discriminator, generator learns how to create more realistic images

Generative Adversarial Training

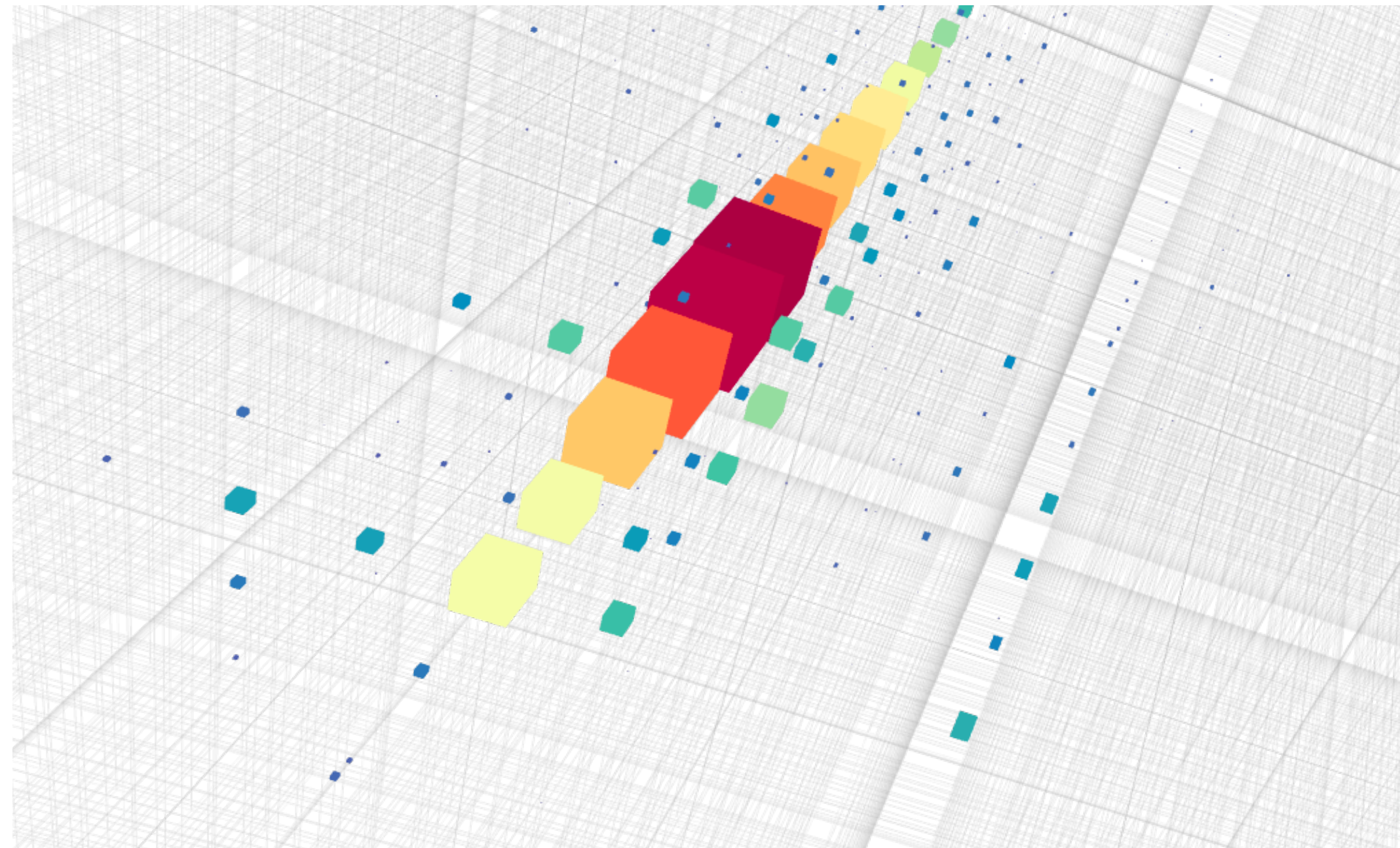
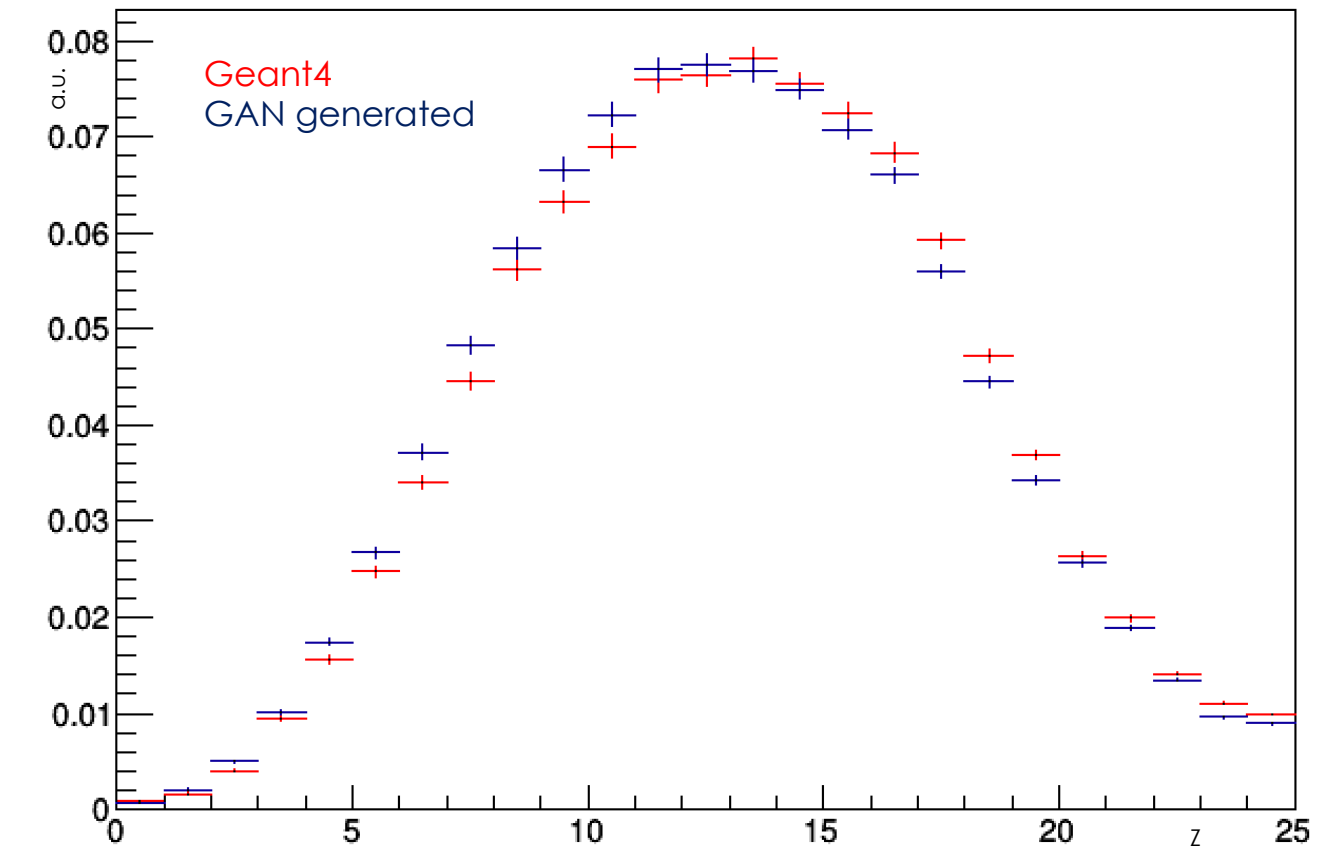


Particle shower generation

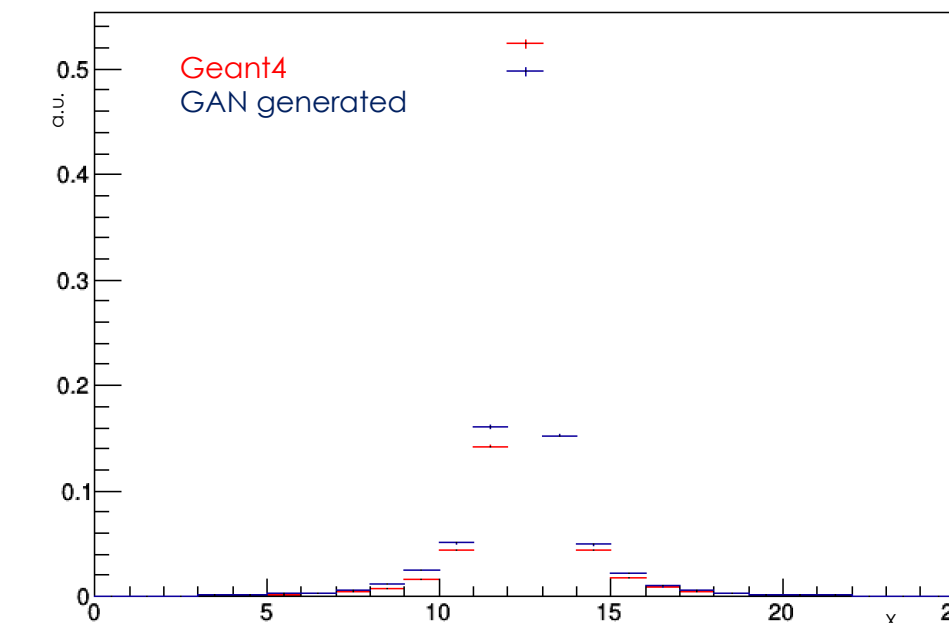
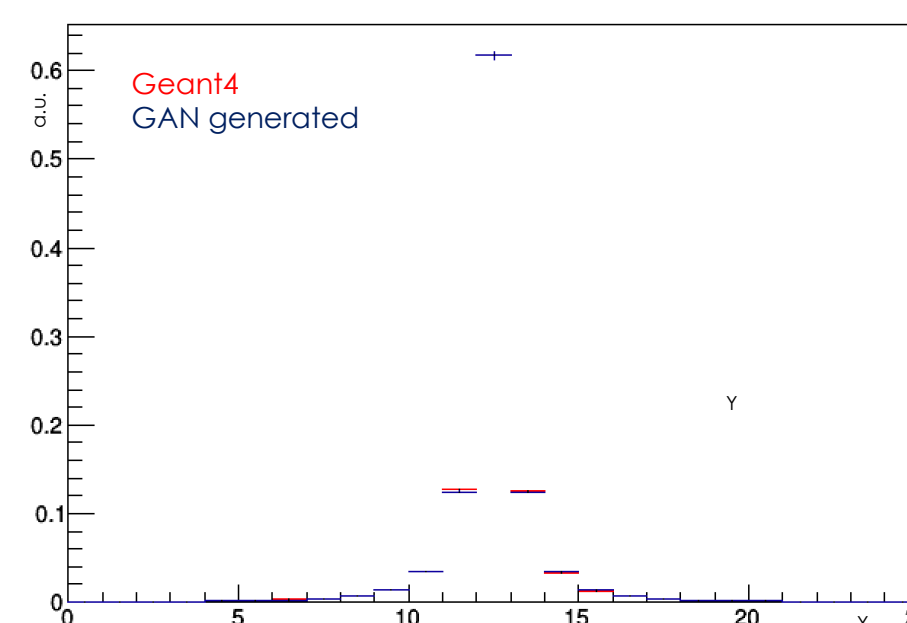
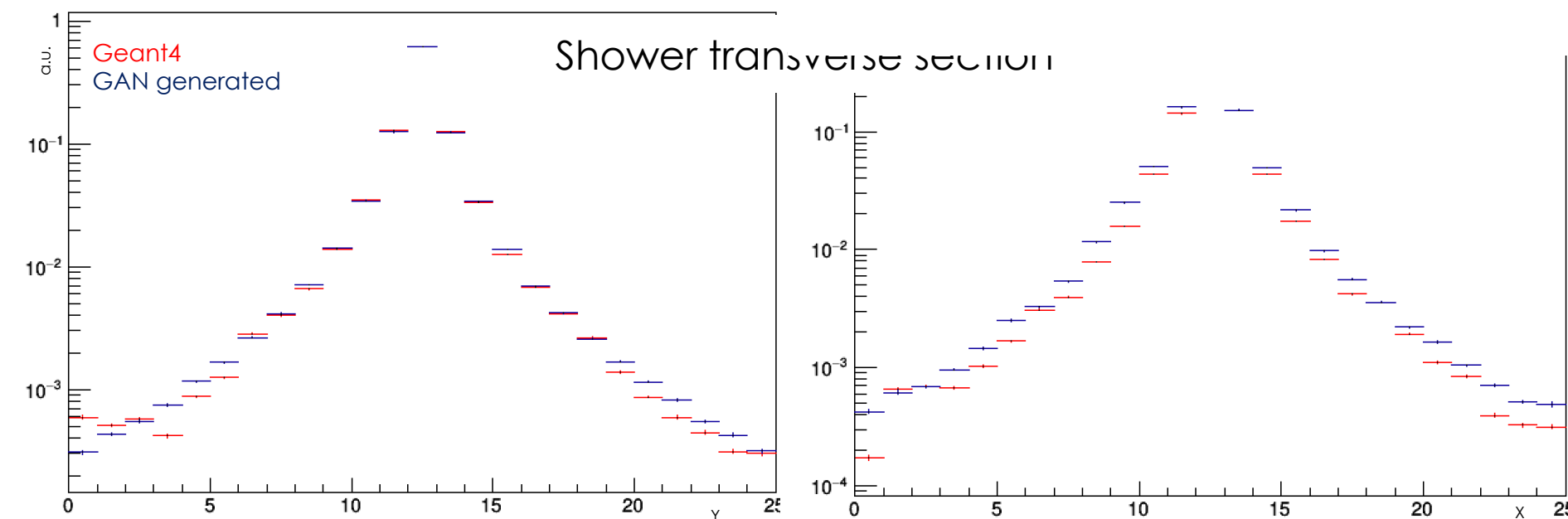
See contribution to NIPS workshop

- Start from random noise
- Works very well with images
- Applied to electron showers in digital calorimeters as a replacement of GEANT

Shower longitudinal section



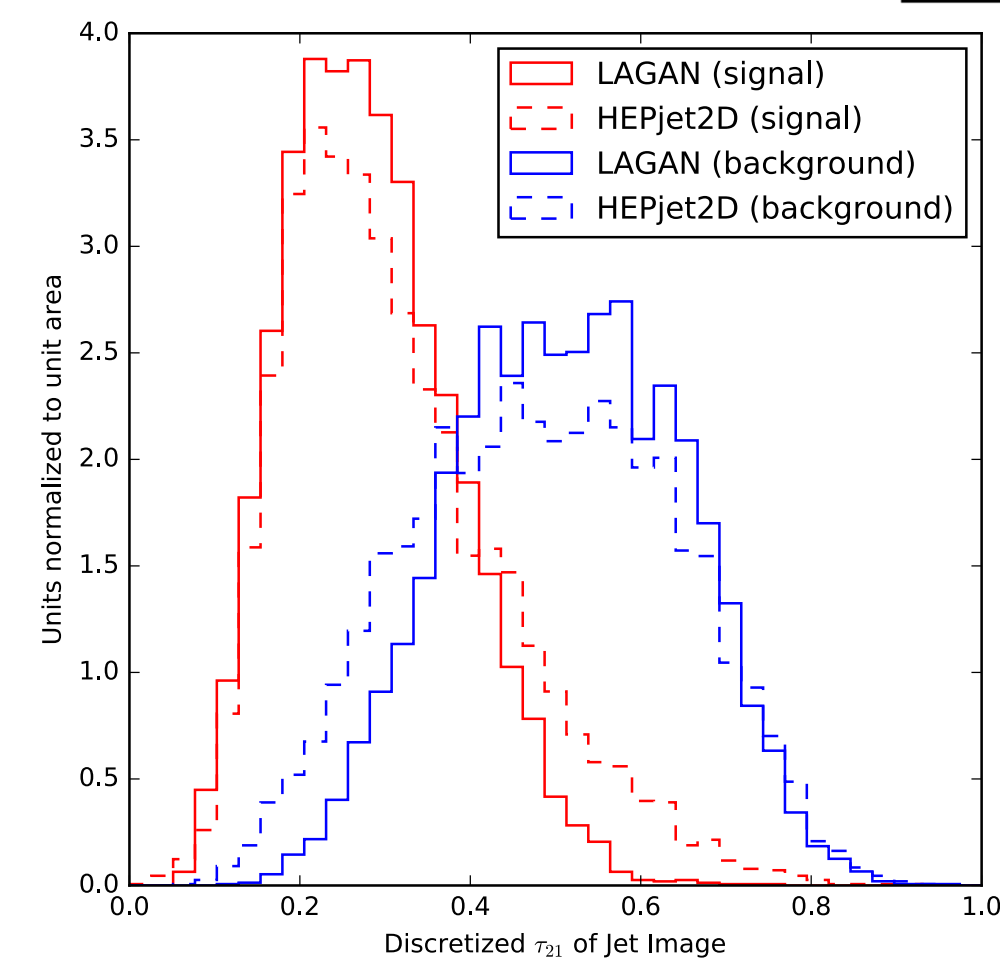
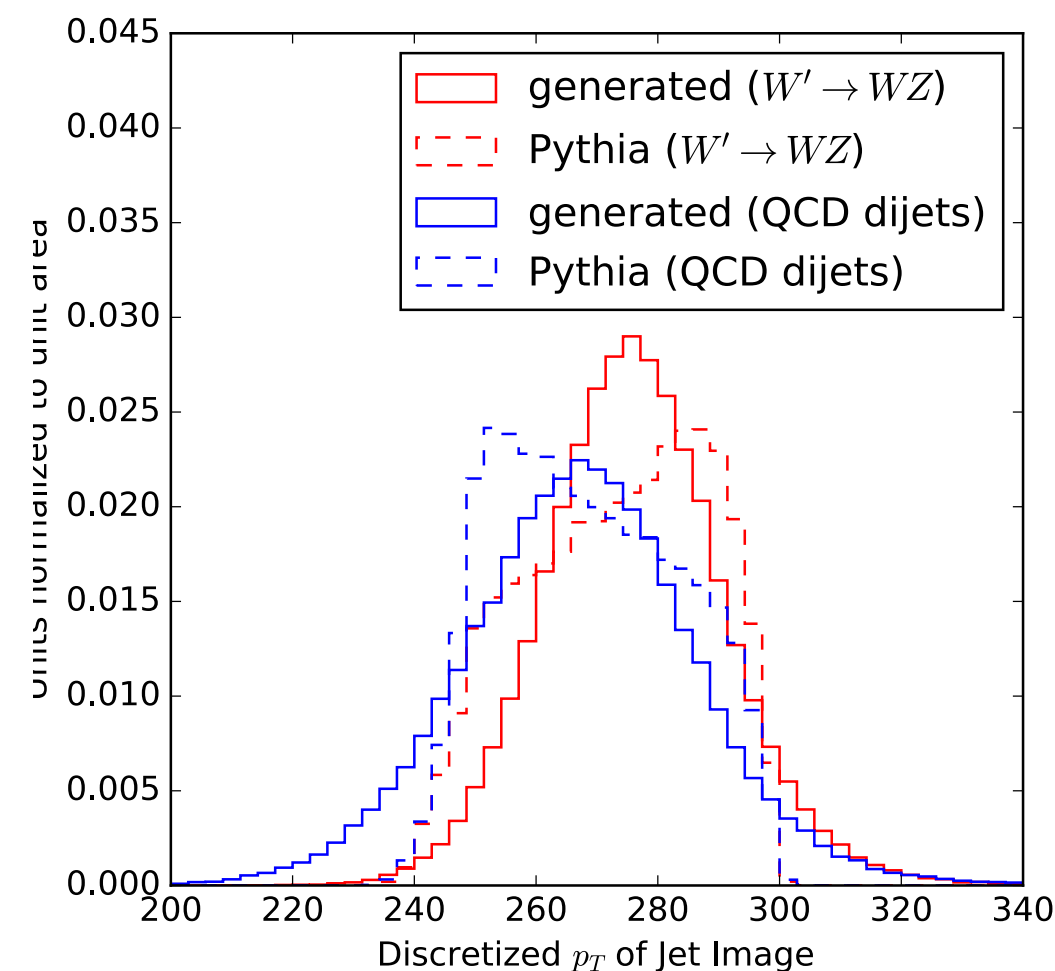
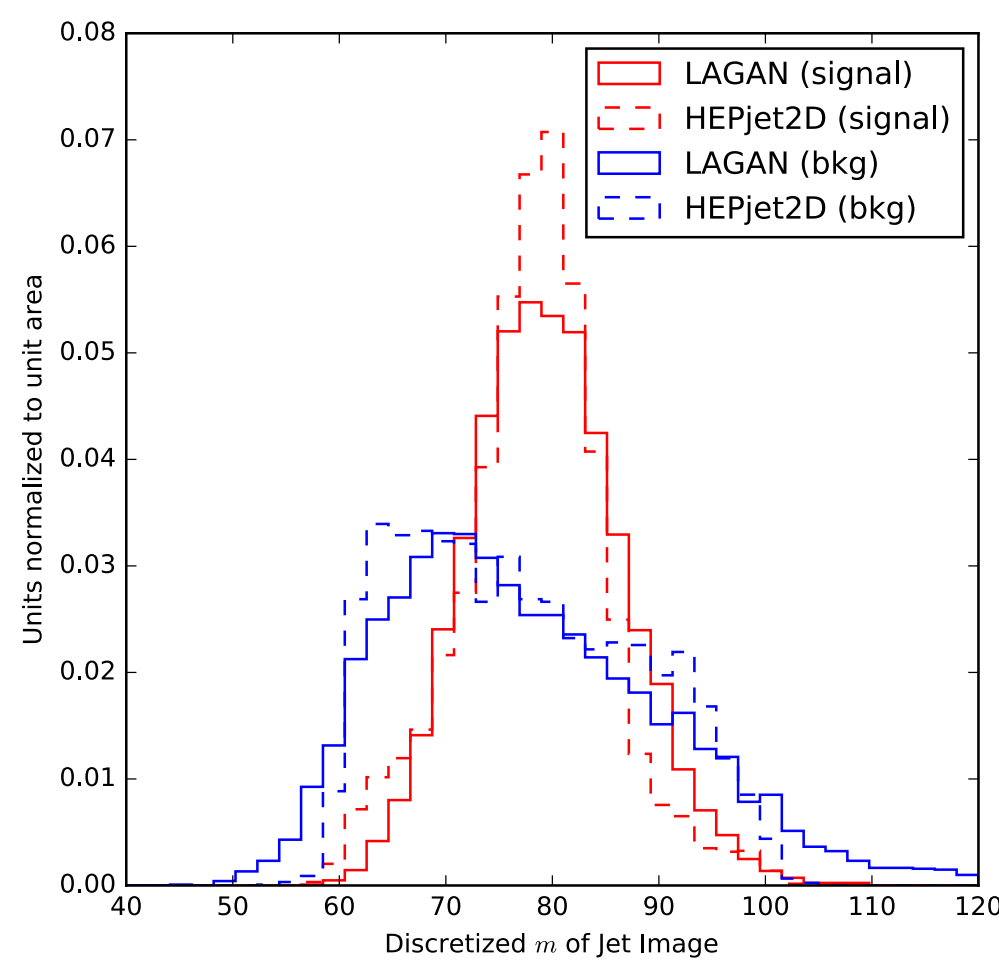
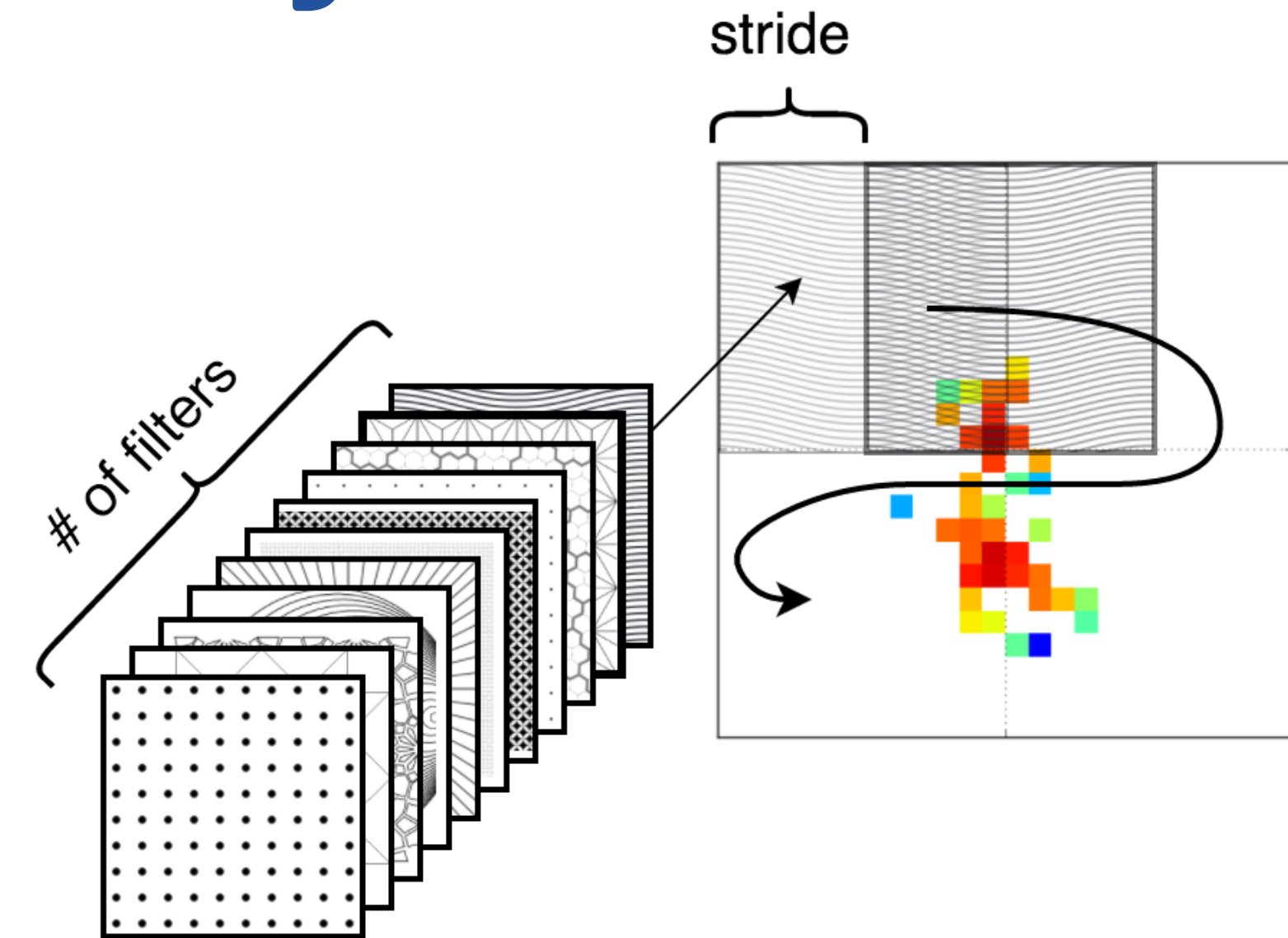
Shower transverse section



see also de Olivera, Paganini, and Nachman
<https://arxiv.org/abs/1712.10321>

Generating full jets

- Start from random noise
- Works very well with images
- Applied to electron showers in digital calorimeters as a replacement of GEANT



de Olivera, Paganini, and Nachman
<https://arxiv.org/pdf/1701.05927.pdf>

Figure 6: The distributions of image mass $m(I)$, transverse momentum $p_T(I)$, and n -subjettiness $\tau_{21}(I)$. See the text for definitions.