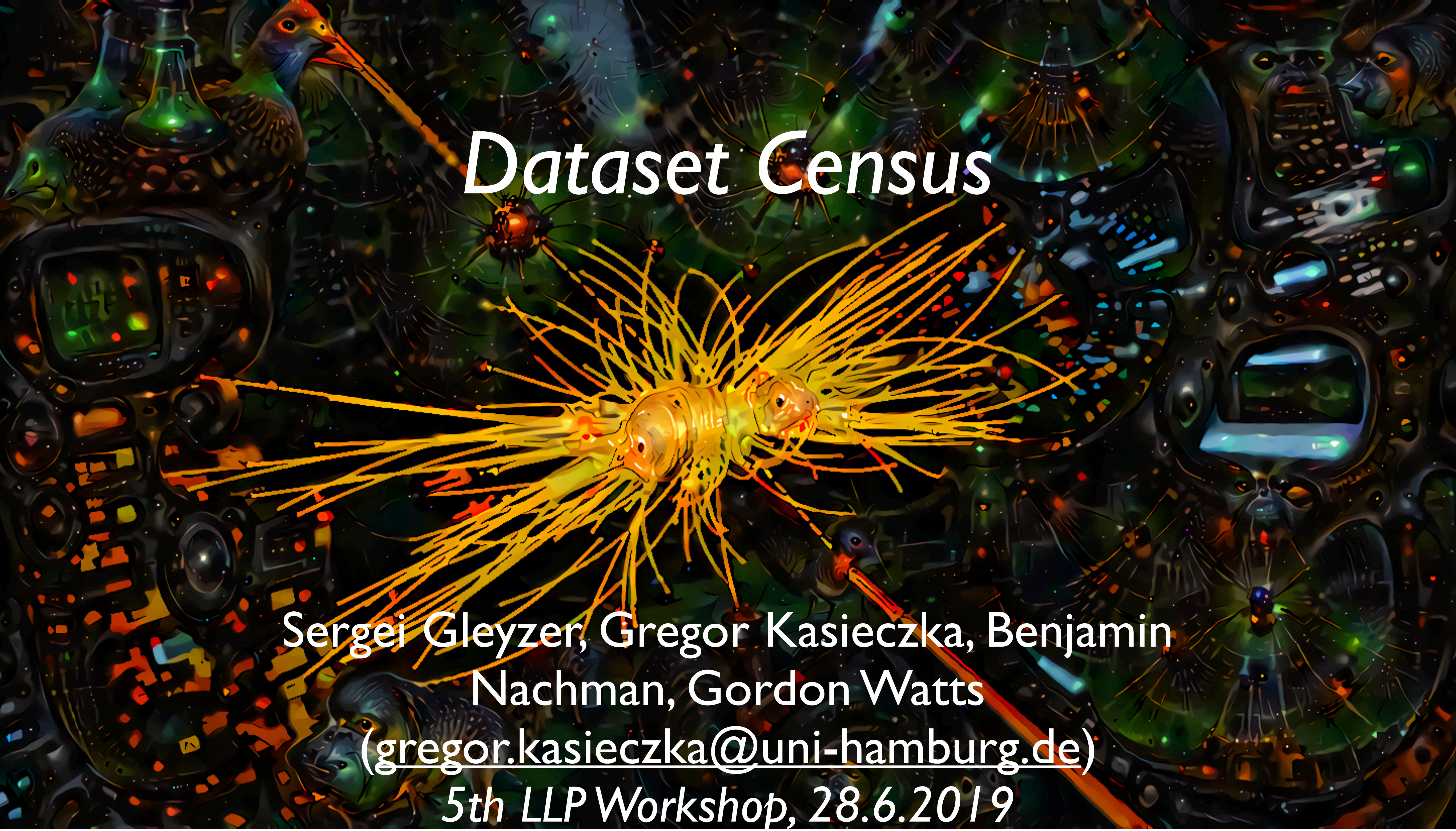


Dataset Census

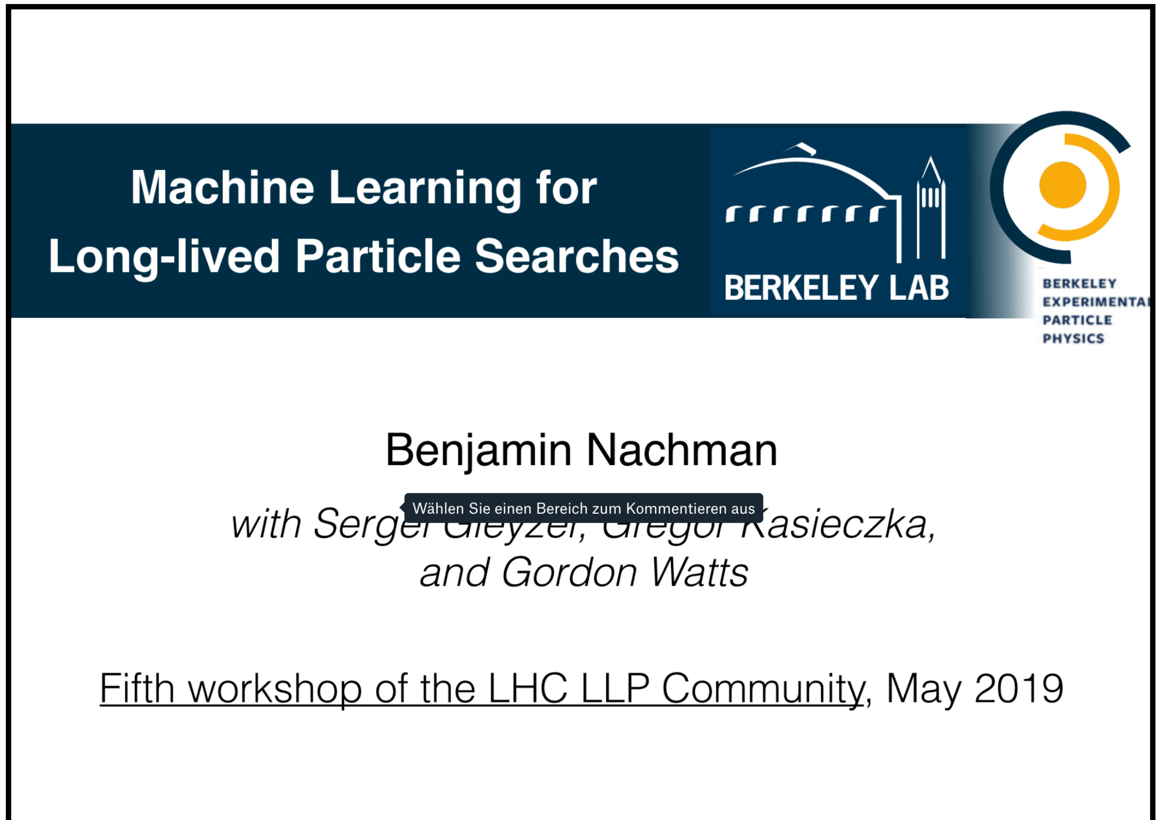
Sergei Gleyzer, Gregor Kasieczka, Benjamin Nachman, Gordon Watts
(gregor.kasieczka@uni-hamburg.de)

5th LLP Workshop, 28.6.2019

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Emmy Noether-Programm
Deutsche Forschungsgemeinschaft
DFG

Bundesministerium für Bildung und Forschung



1) Monday 13:30: Plenary introductory talks from the conveners of the groups

Overview of status quo

2) Preparation sessions on Monday

3) Parallel working sessions on Tuesday

Here we are

4) Working group reports / summary talks on Wednesday at 11:35

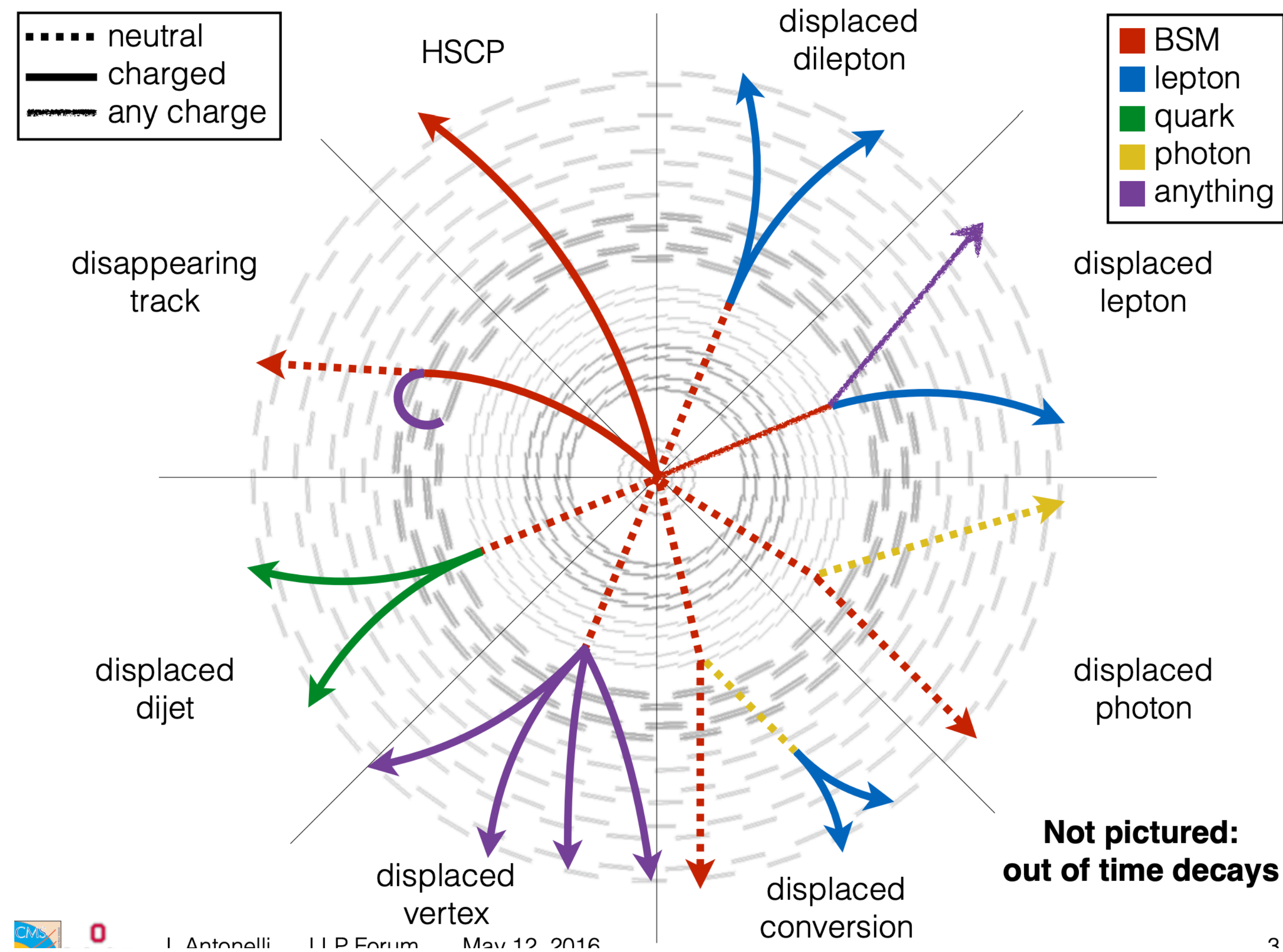

Potential Discovery
LLP Searches
ML Techniques

Topics

- (How) can we improve LLP searches using ML?
- Wide range of non-standard signatures
  - Avoid over optimisation
  - Anomaly finding?
- Goals today
  - Understand available datasets - see what is needed
  - Discuss NN architectures and see how they can be mapped to LL problems
  - (If time & interest): ML tutorial

Datasets

- No (simulated) data - no machine learning
- Need to understand what
  - ..is available
  - ..can be made available
  - ..should be produced
- Fidelity:
  - Generator- / Delphes / Geant / Data



Long-Lived Particle Datasets for ML Training

We are interested in collecting pointers to datasets that may exist that could be used for ML training. Theorists datasets, datasets internal to experiments, etc. Anything that might be made public (or already is public). We are hoping to built a list of datasets that are publicly accessible that ML practitioners can use to try out new techniques and algorithms that will end up benefiting all of us. We are asking for your email address for attribution and so we can get in touch if we have follow-up questions.

* Required

Email address *

Your email

What LLPs does the dataset contain?

Your answer

A reference for the dataset (email, url, etc.)

Your answer

SUBMIT

Never submit passwords through Google Forms.

Link

Results

Tracking down Quirks at the Large Hadron Collider *(toy)*

Simon Knapen,[1,2] Hou Keong Lou,[1,2] Michele Papucci,[1,2] and Jack Setford[3]
[1]*Department of Physics, University of California, Berkeley, California 94720, USA*
[2]*Theoretical Physics Group, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[3]*Department of Physics and Astronomy, University of Sussex, England, UK*
(Dated: November 15, 2017)

1708.02243

The Optimal Use of Silicon Pixel Charge Information for Particle Identification

Harley Patton[1] and Benjamin Nachman[2]

1803.08974

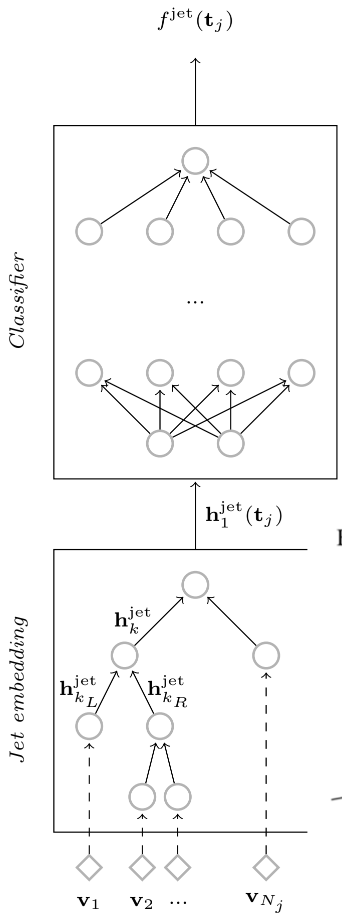
A Bottom Line for the LHC Data by Leveraging Pileup as a Zero Bias Sample

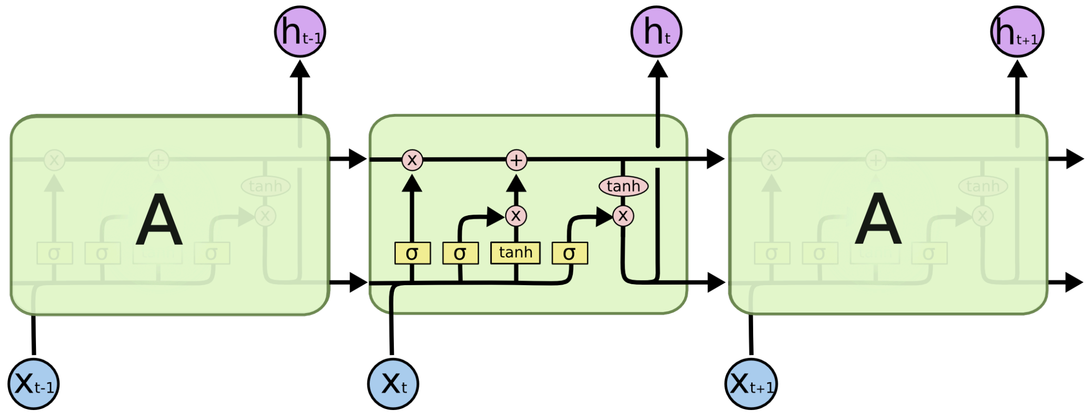
Benjamin Nachman*
Lawrence Berkeley National Laboratory

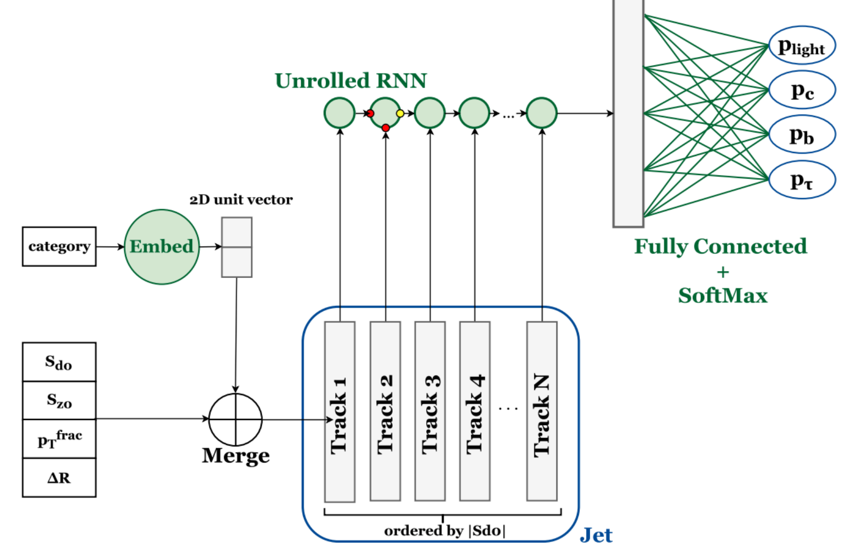
Francesco Rubbo†
SLAC National Accelerator Laboratory
(Dated: October 24, 2018)

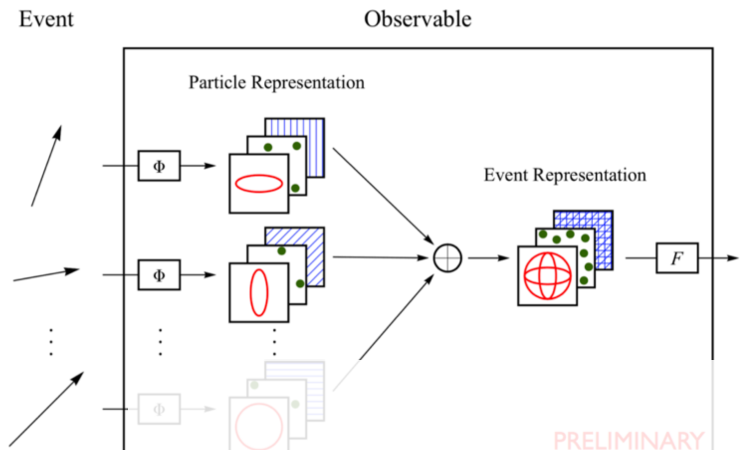
1608.06299

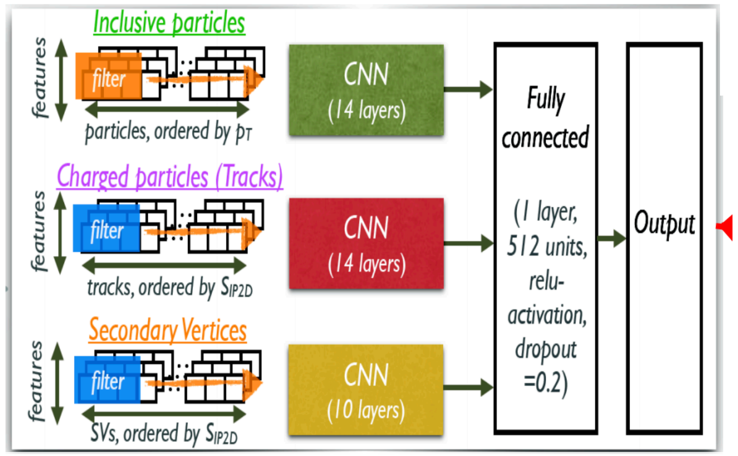
https://docs.google.com/spreadsheets/d/1tg-oOcH_HbaPX1YhrDQ6i-k8YzvIhsYq6BzzbBD-Amc/edit#gid=1678121814

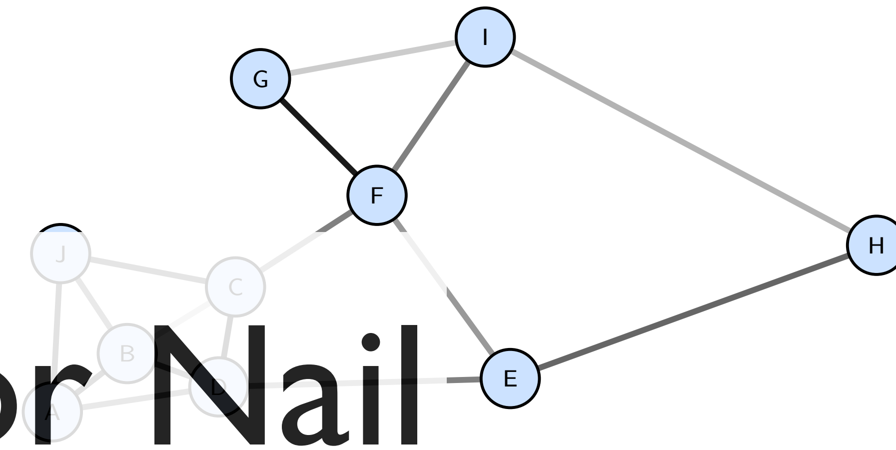




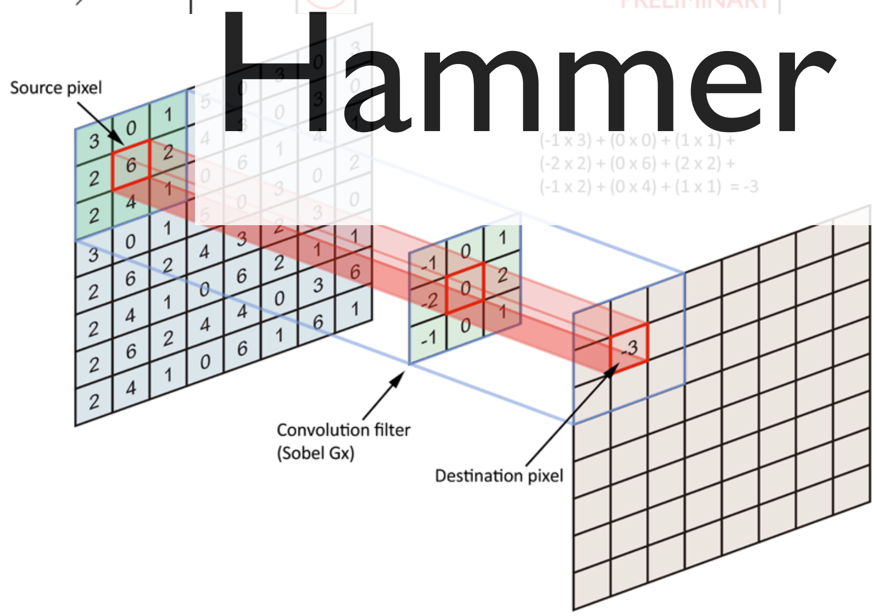


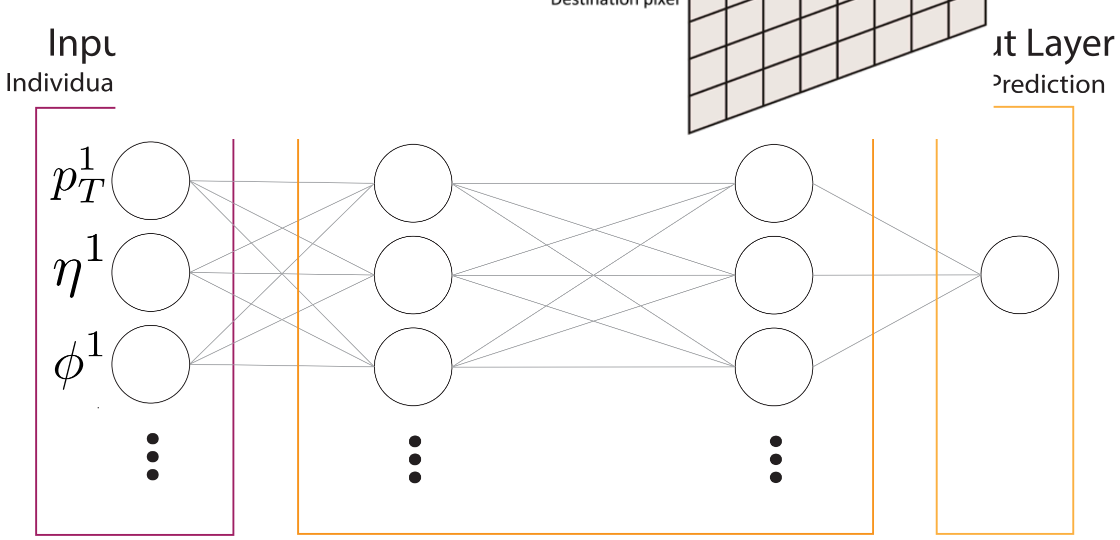






Hammer looking for Nail





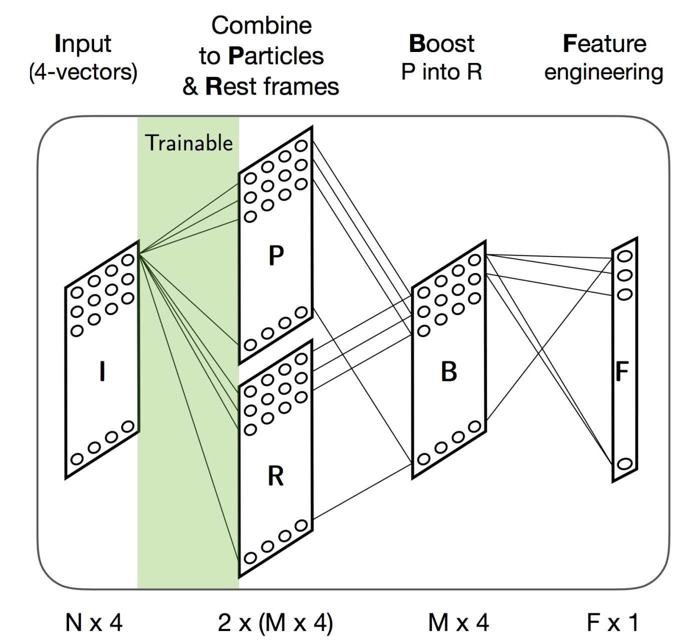
$$k_{\mu,i} = \begin{pmatrix} E_0 & E_1 & \dots & E_N \\ p_{x,0} & p_{x,1} & \dots & p_{x,N} \\ p_{y,0} & p_{y,1} & \dots & p_{y,N} \\ p_{z,0} & p_{z,1} & \dots & p_{z,N} \end{pmatrix}$$

Combination Layer (**CoLa**): create linear combinations:

$$k_{\mu,i} \xrightarrow{\text{CoLa}} \widetilde{k}_{\mu,j} = k_{\mu,i}\, C_{ij}$$

Lorentz Layer (**LoLa**): Use resulting matrix to extract physics features.
Main assumption is the Minkowski metric

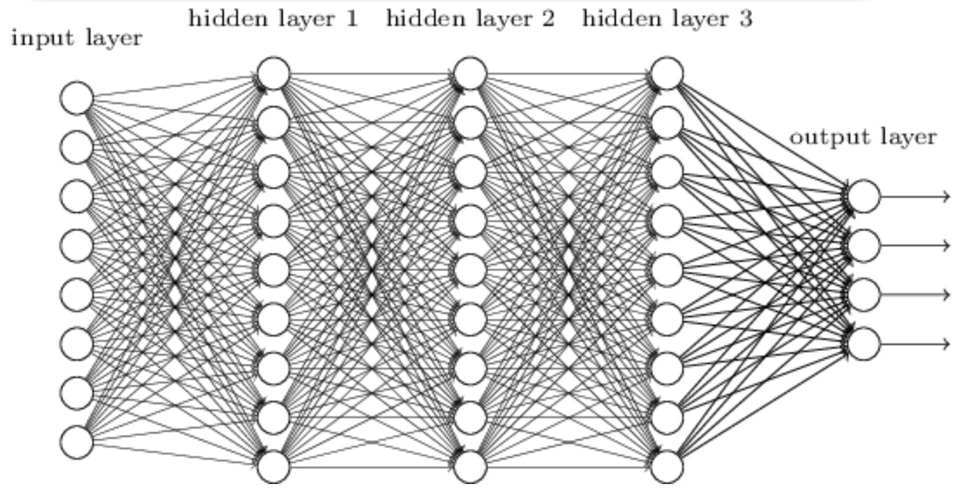
$$\widetilde{k}_{\mu,i} \to \sum_j (\widetilde{k}_i - \widetilde{k}_j)_\mu (\widetilde{k}_i - \widetilde{k}_j)_\nu \eta^{\mu\nu} B_{ij}$$



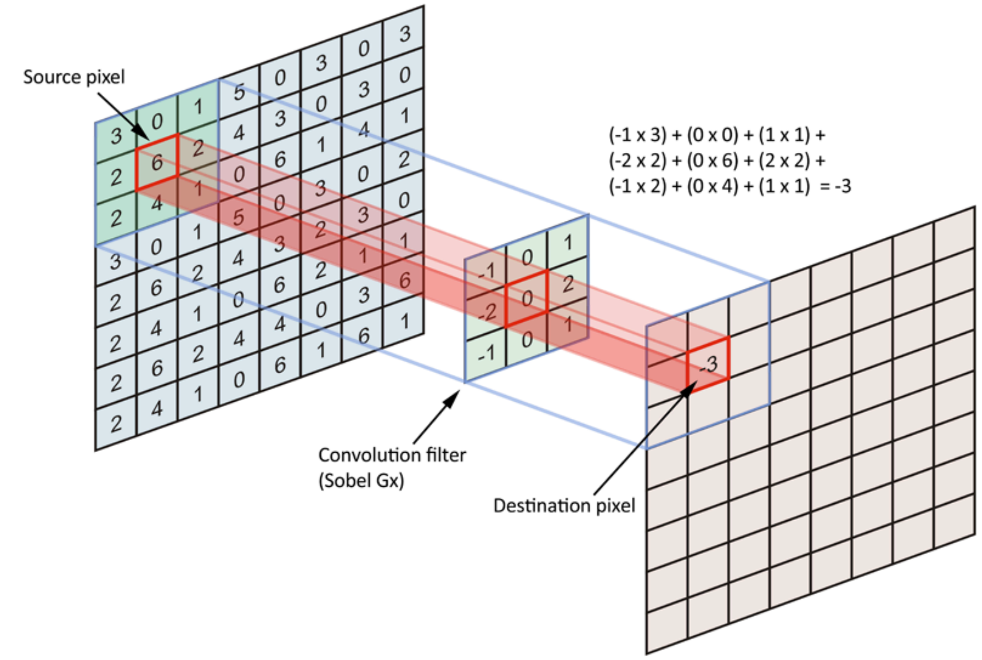
Why architectures?

- ML approach can be greatly simplified if structure of data mirrors structure of problem
- LLP problems are very complicated reconstruction questions
- Can we find architectures that match a specific problem?
  - Possible shortcuts?
- Brief overview of NN architectures

High-level

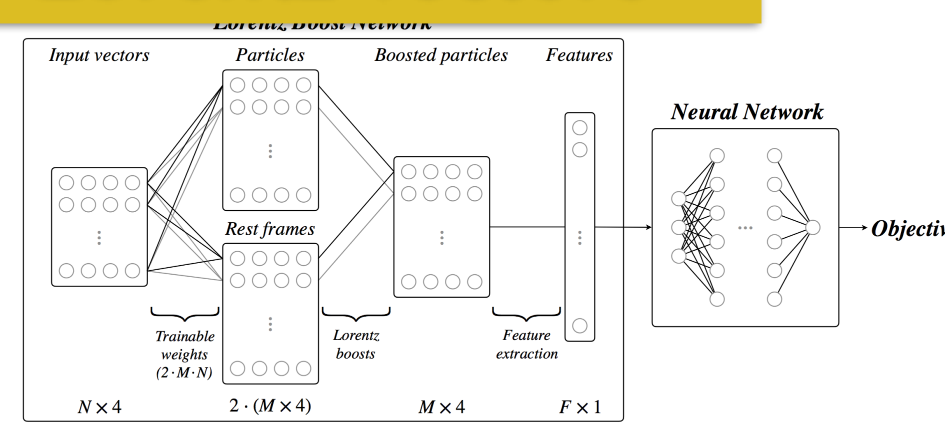


Regular grid

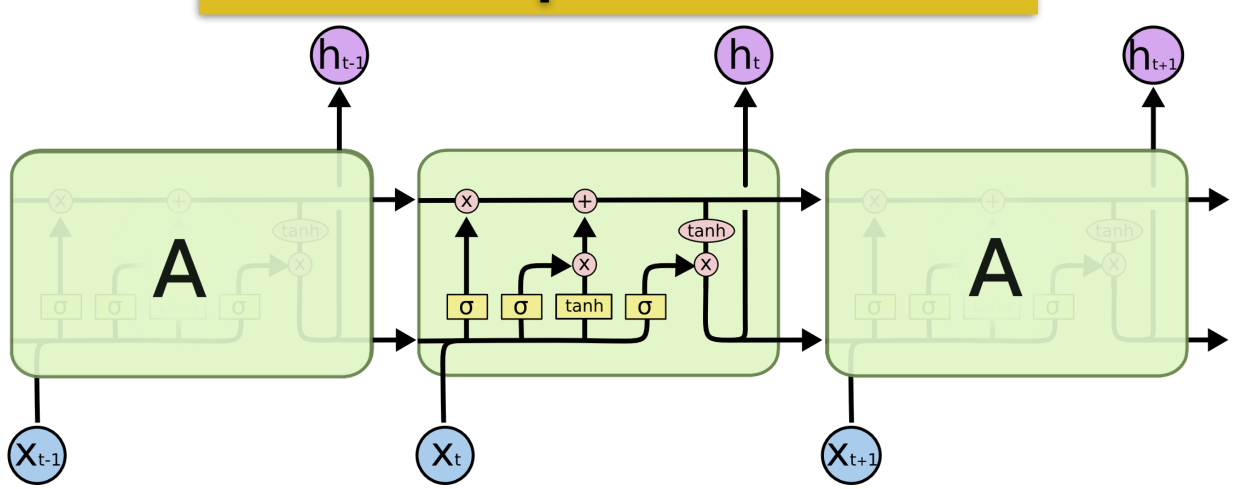


Representation

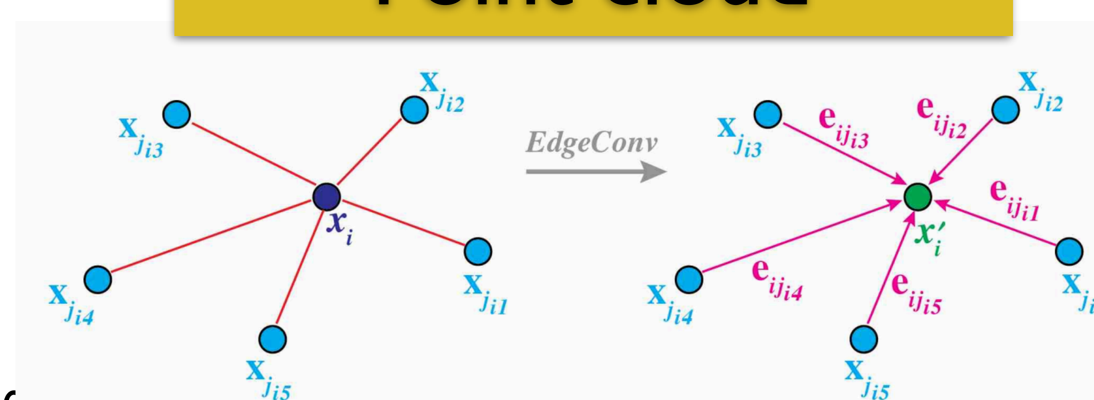
Lorentz vectors



Sequences

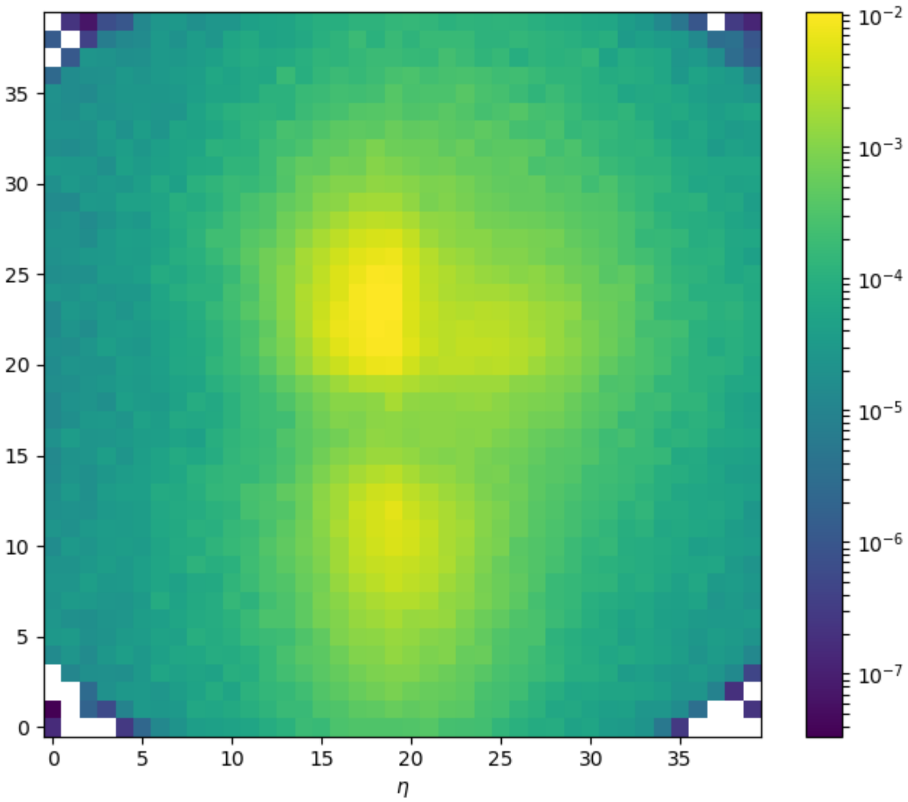


Point cloud

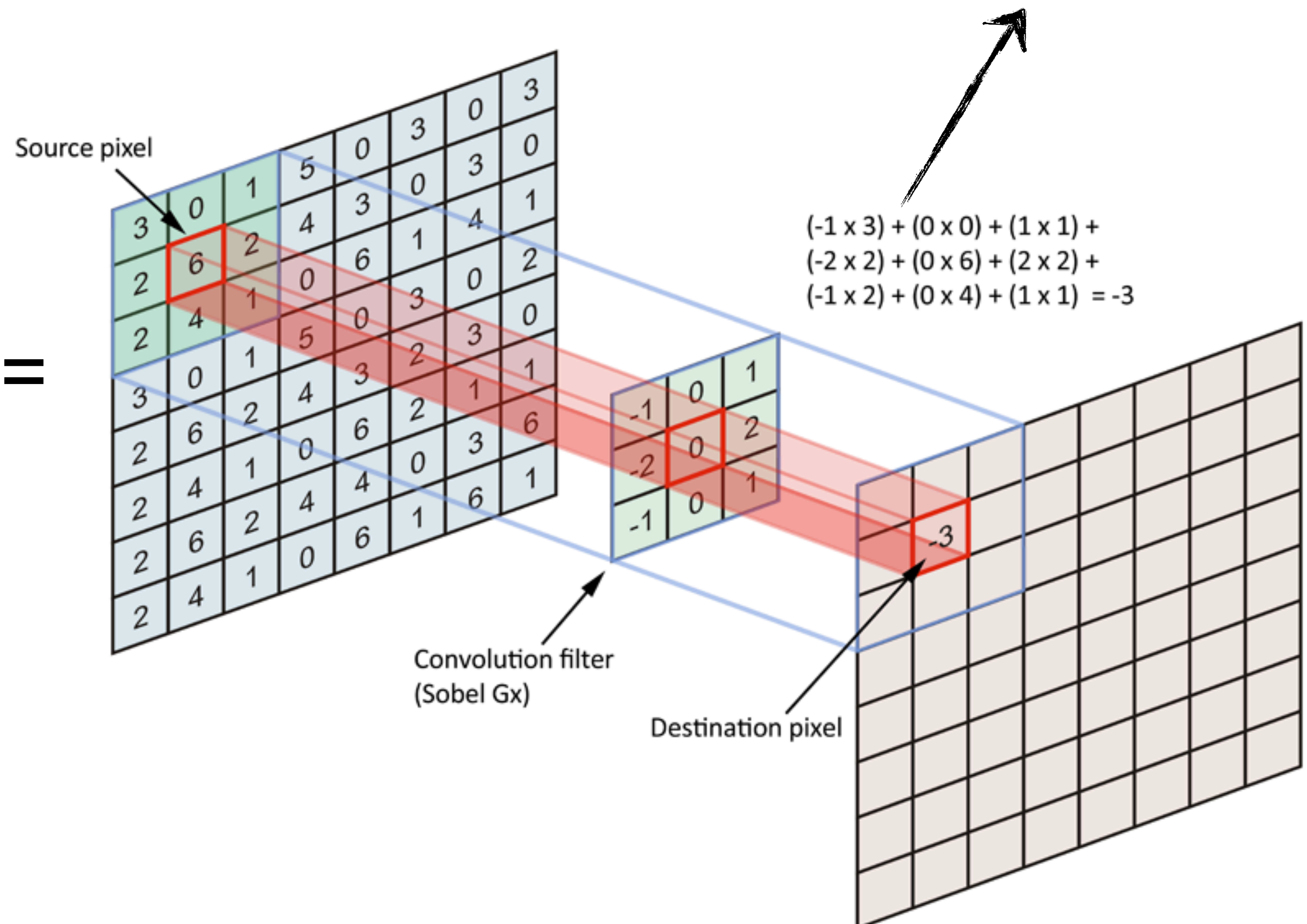


Convolution

Train these weights

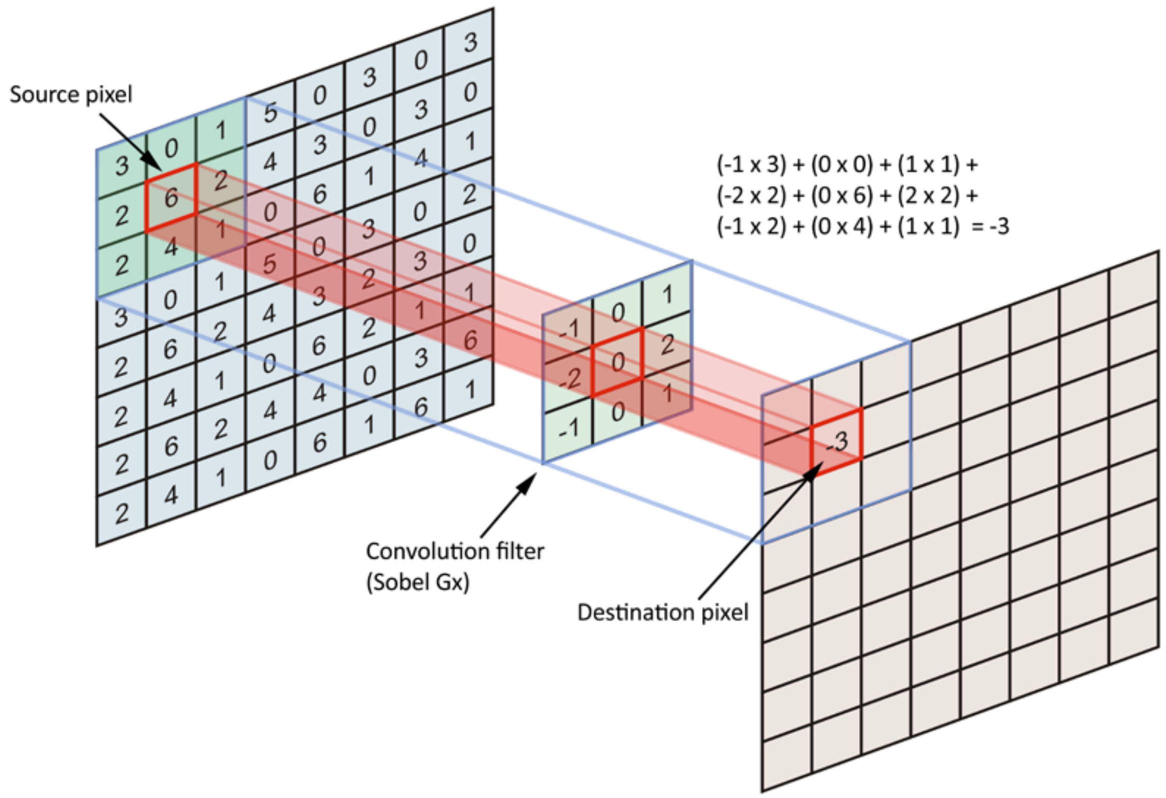


=

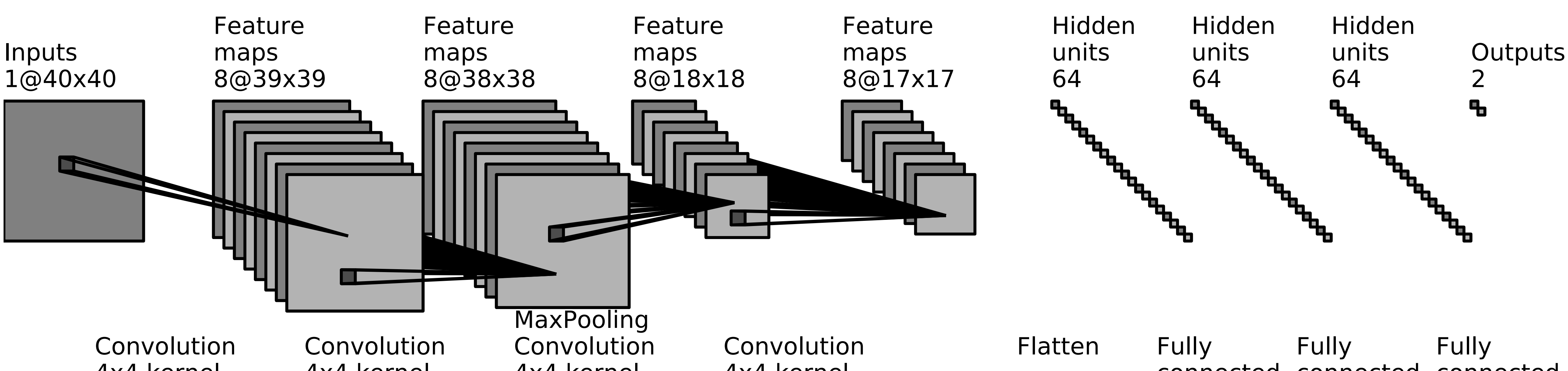


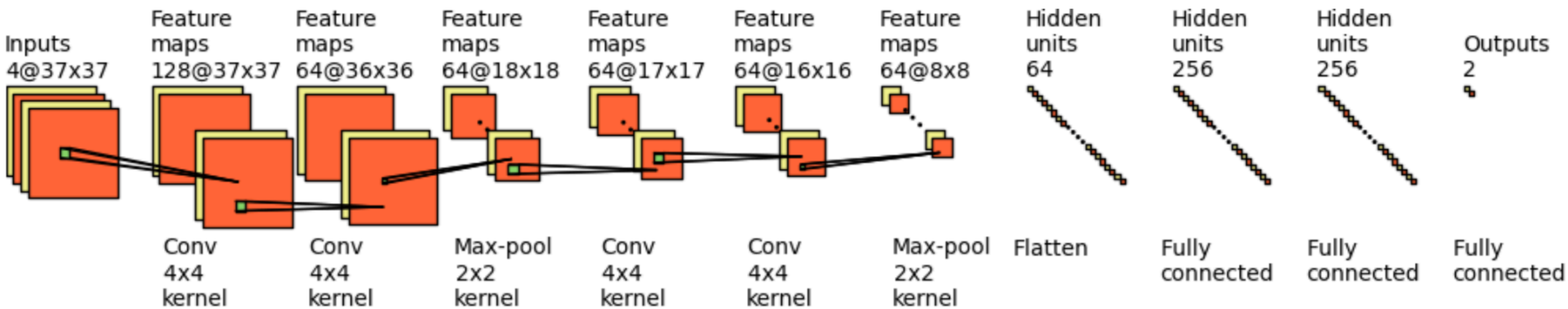
Efficient use of weights and natural encoding of translational symmetry.

Convolution network



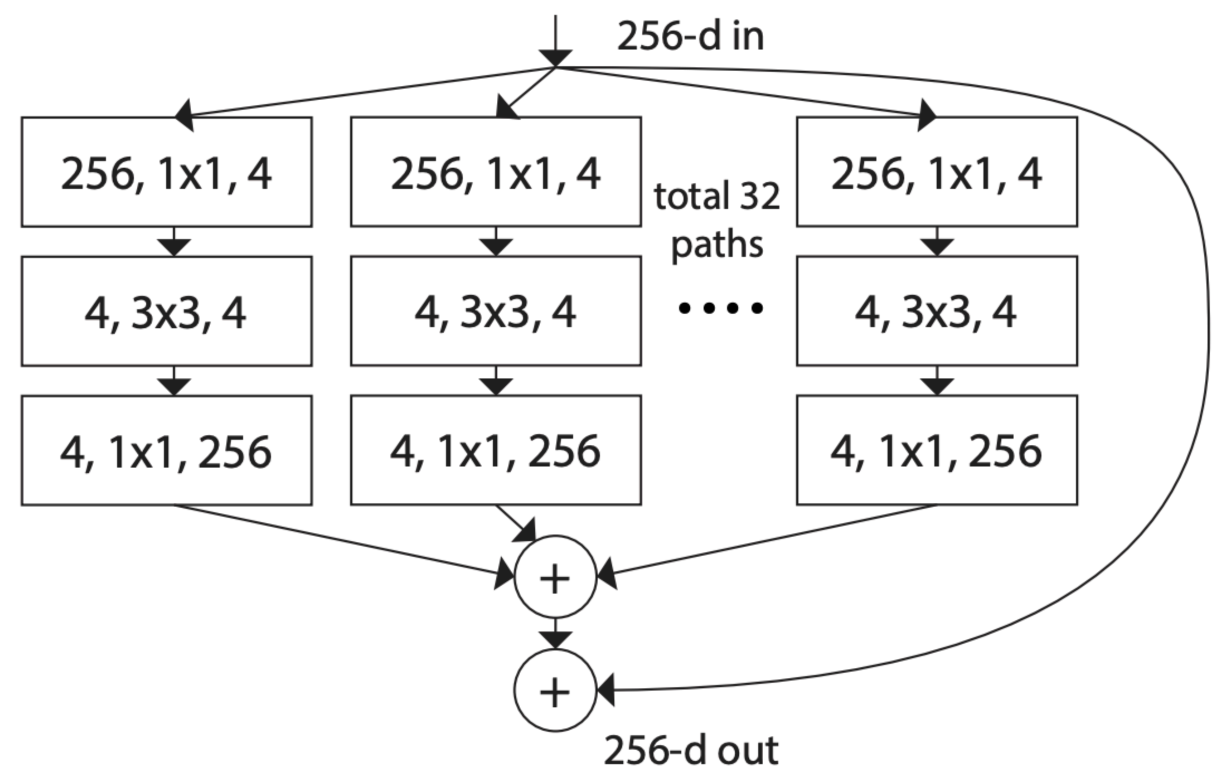
- ***How to build a convolution network?***
  - Multiple parallel and successive convolutions
  - Pooling
  - Simple network in the end
- **1D Convolutions!**
- **3D Convolutions!**





Simple CNN

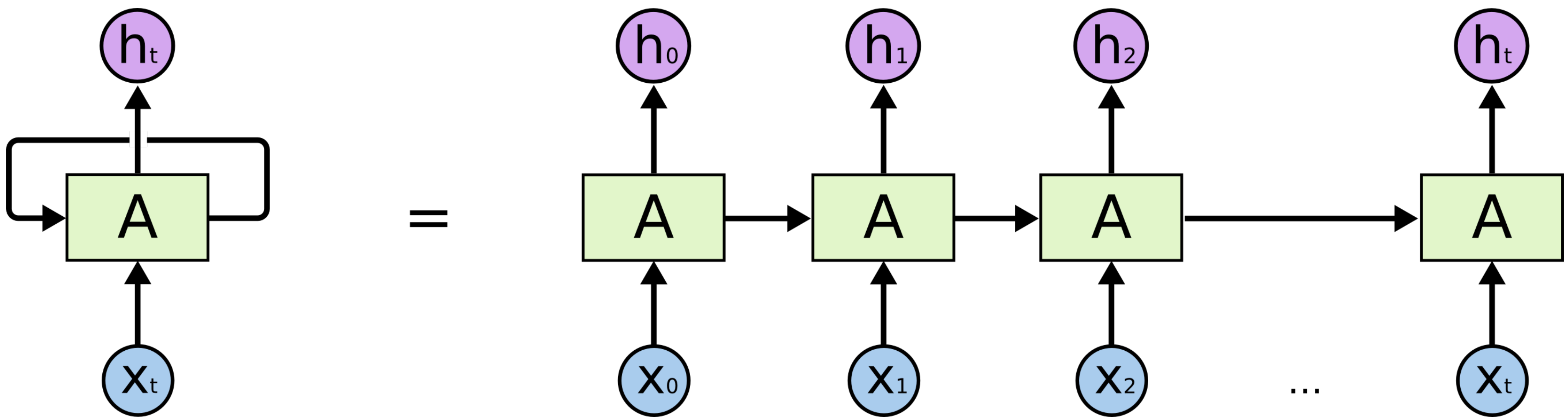
stage	output	ResNet-50	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2 [1×1, 64 3×3, 64 1×1, 256] ×3	3×3 max pool, stride 2 [1×1, 128 3×3, 128, C=32 1×1, 256] ×3
conv3	28×28	[1×1, 128 3×3, 128 1×1, 512] ×4	[1×1, 256 3×3, 256, C=32 1×1, 512] ×4
conv4	14×14	[1×1, 256 3×3, 256 1×1, 1024] ×6	[1×1, 512 3×3, 512, C=32 1×1, 1024] ×6
conv5	7×7	[1×1, 512 3×3, 512 1×1, 2048] ×3	[1×1, 1024 3×3, 1024, C=32 1×1, 2048] ×3
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params.		**25.5**$\times 10^6$	**25.0**$\times 10^6$
FLOPs		**4.1**$\times 10^9$	**4.2**$\times 10^9$



1611.05431
1803.00107

ResNeXt50
(used with 1/4 filters)

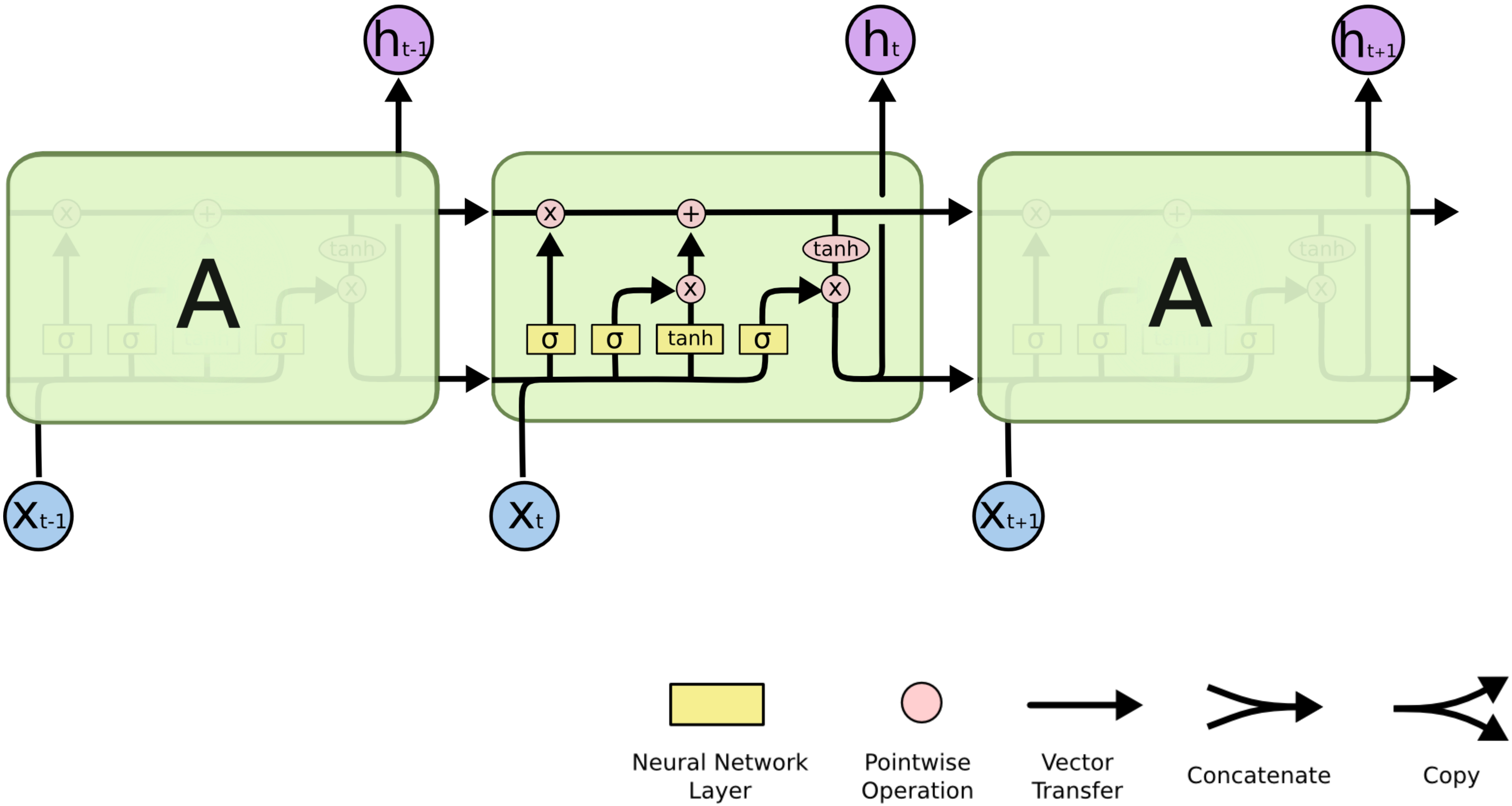
Recurrent



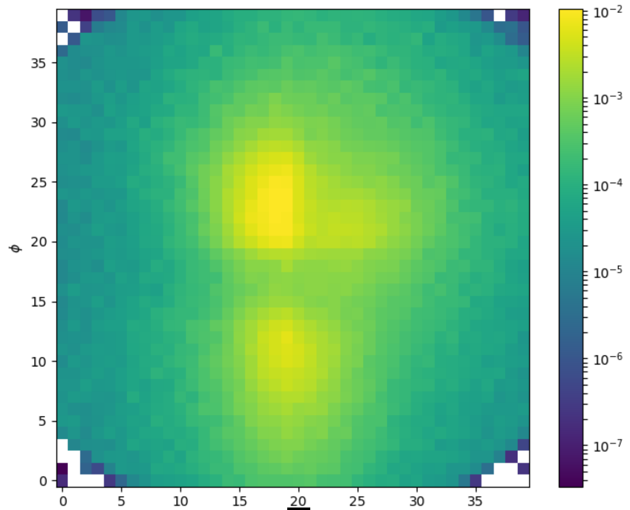
- Inspired by natural language processing
- Work with a sequence of inputs
- Inputs can change the state of the cell (*Long Short Term Memory*)
- Think of
  - One input = One jet constituent

http://colah.github.io/

LSTM



http://colah.github.io/



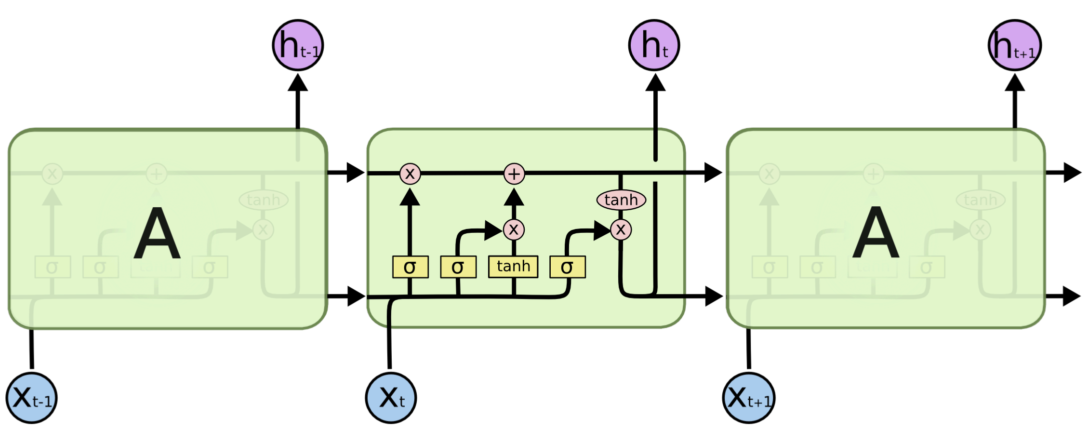
Image

- Regular 2D grid of data
    - One or more numbers/pixel
- Convolutional networks

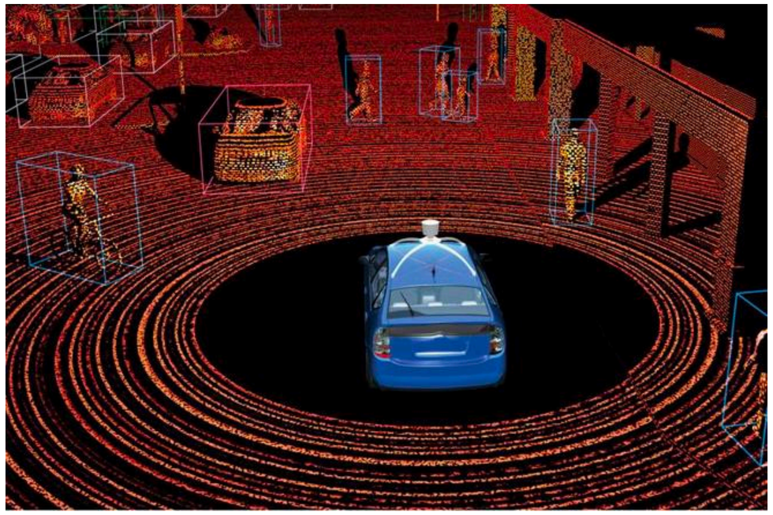
Sequence

This is a sentence.

- Ordered inputs
    - Any number of properties
- LSTM/GRU, Attention

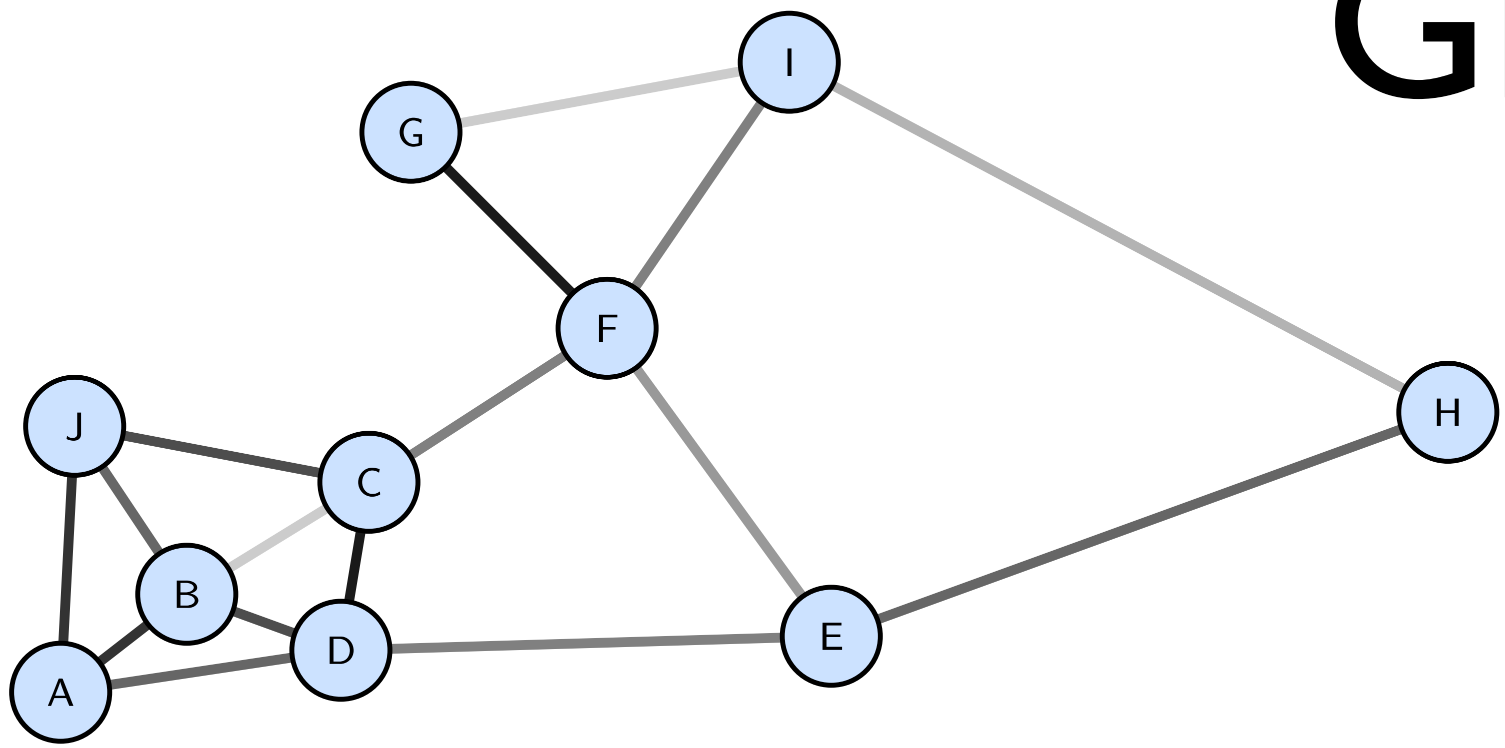


Point Cloud / Particle Cloud



- No intrinsic order
- Outside HEP:
    - 3D coordinates in xyz-space
- In HEP:
    - eg 2D coordinates in eta/phi-space
    - Additional properties
        - Energy, flavour tags, ..
- Deep Sets (later) and Graph Convolution

Graphs



Graph: *A set of vertices and edges*

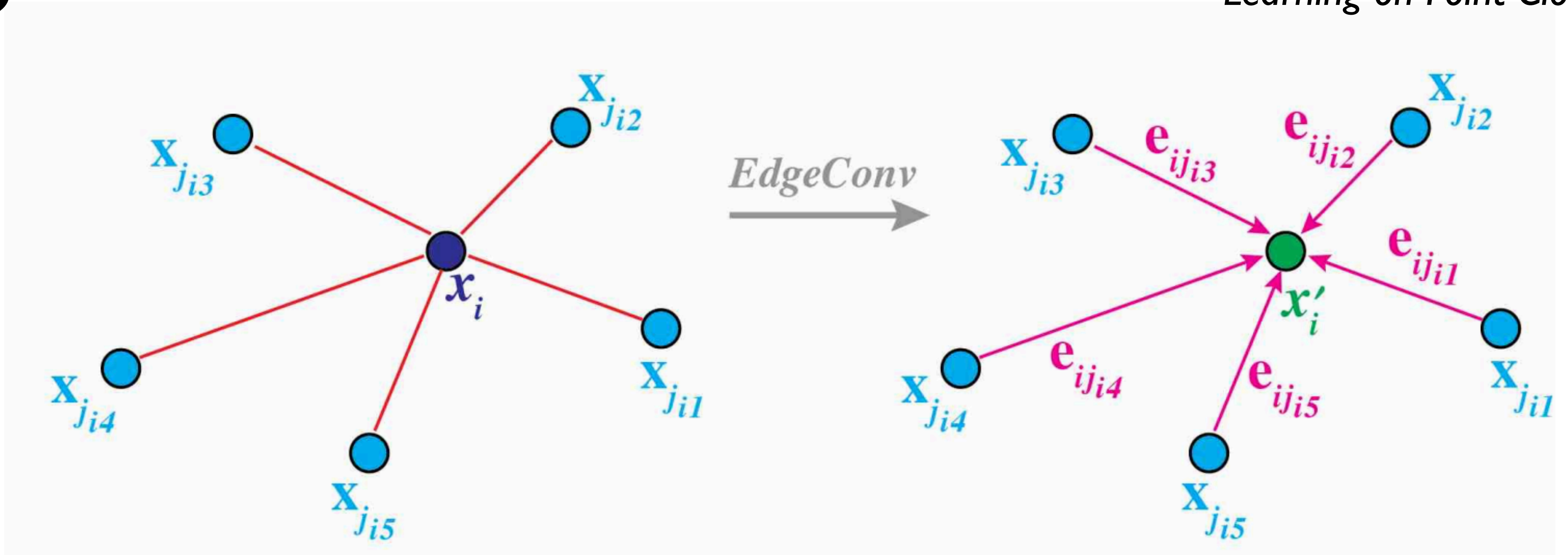
Represent as:
List of vertices *(multiple features/vertex possible)*

Adjacency matrix *(which vertices are connected and how strong)*

How to generalise convolution to graphs?

Edge Convolution

L Gouskos, H Qu https://indico.cern.ch/event/745718/contributions/3202526
Y Wang et al, *Dynamic Graph CNN for Learning on Point Clouds,* 1801.07829



- For each point:
  - Define local area as K nearest neighbours using coordinates (ie eta/phi metric)

Alternative: Neural Message Passing for Jet Physics I Henrion et al. Procs. of the Deep Learning for Physical Sciences Workshop at NIPS (2017)

- Convolution filter equivalent:

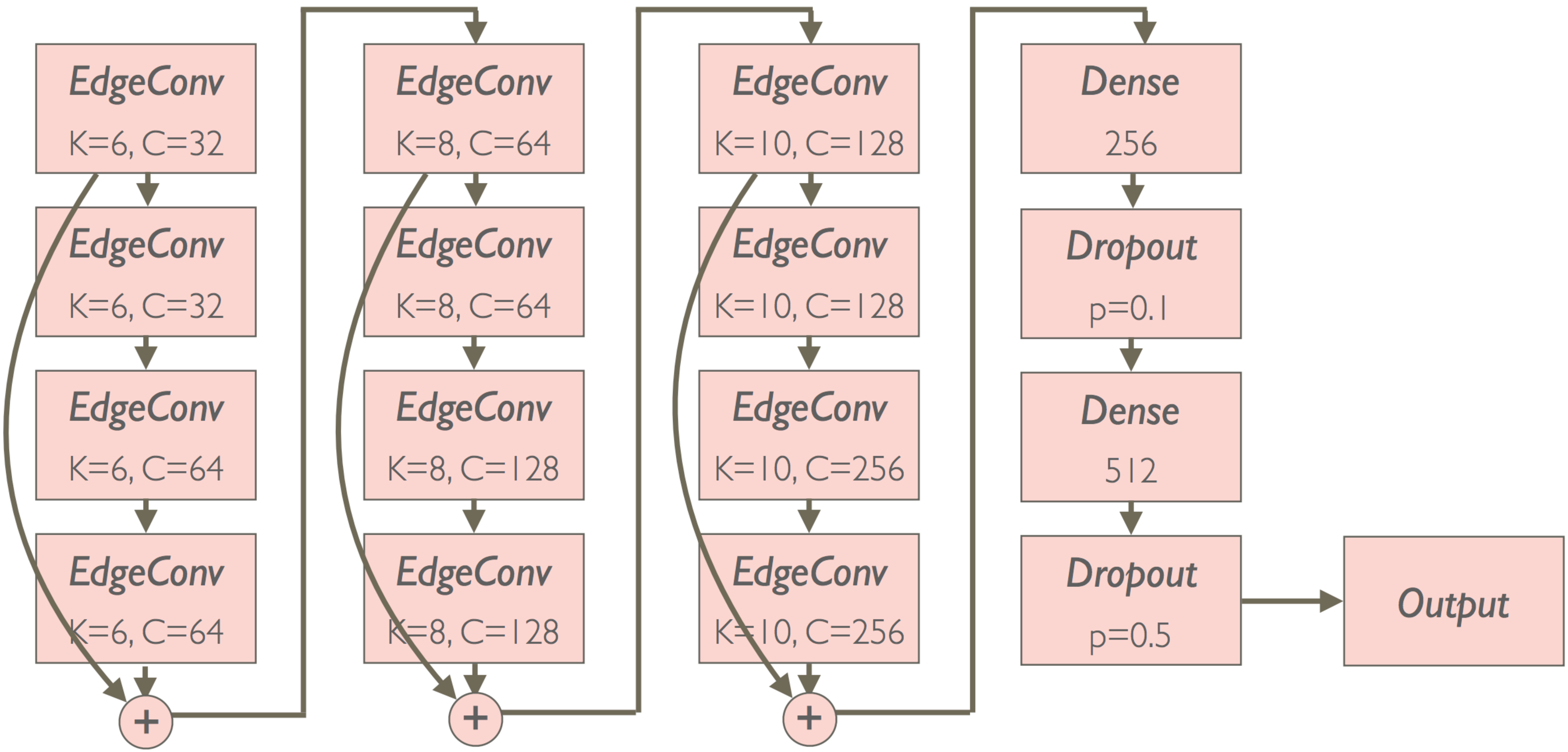
$$e_{ij} = h_\theta(x_i, x_j)$$
$$x'_i = \sum_j e_{ij}$$

Symmetric: same for all nodes and centers

- Recompute distance at each layer: Dynamic Graph CNN

Edge Convolution

L Gouskos, H Qu https://indico.cern.ch/event/745718/contributions/3202526
Y Wang et al, *Dynamic Graph CNN for Learning on Point Clouds*, 1801.07829



Another way to deal with unordered inputs

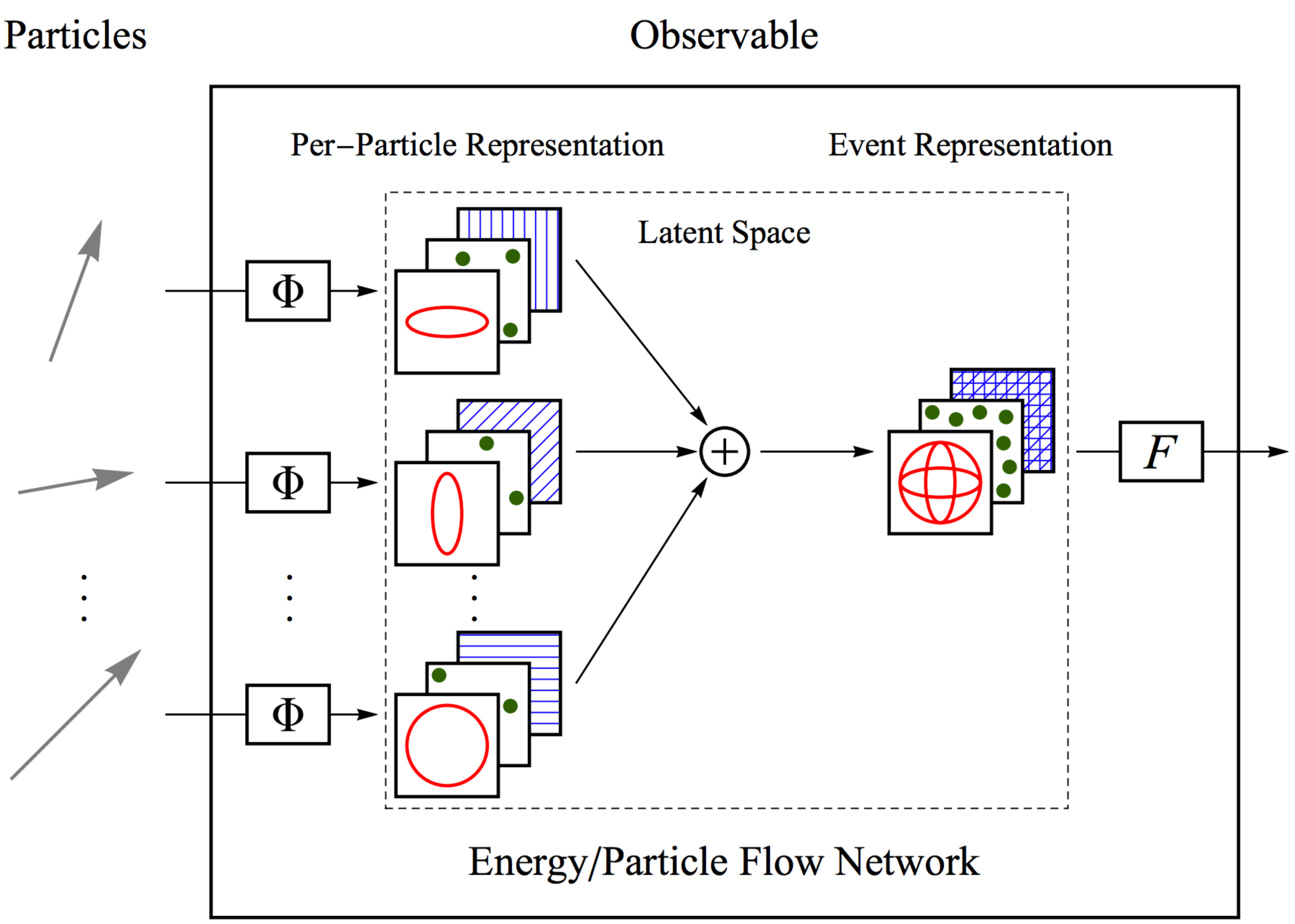
Theorem 7 *Let $f : [0,1]^M \to \mathbb{R}$ be a permutation invariant continuous function iff it has the representation*

$$f(x_1, ..., x_M) = \rho\left(\sum_{m=1}^{M} \phi(x_m)\right) \tag{18}$$

for some continuous outer and inner function $\rho : \mathbb{R}^{M+1} \to \mathbb{R}$ and $\phi : \mathbb{R} \to \mathbb{R}^{M+1}$ respectively. The inner function ϕ is independent of the function f.

Deep Sets, M Zaheer et al, 1703.06114

For physics



General :

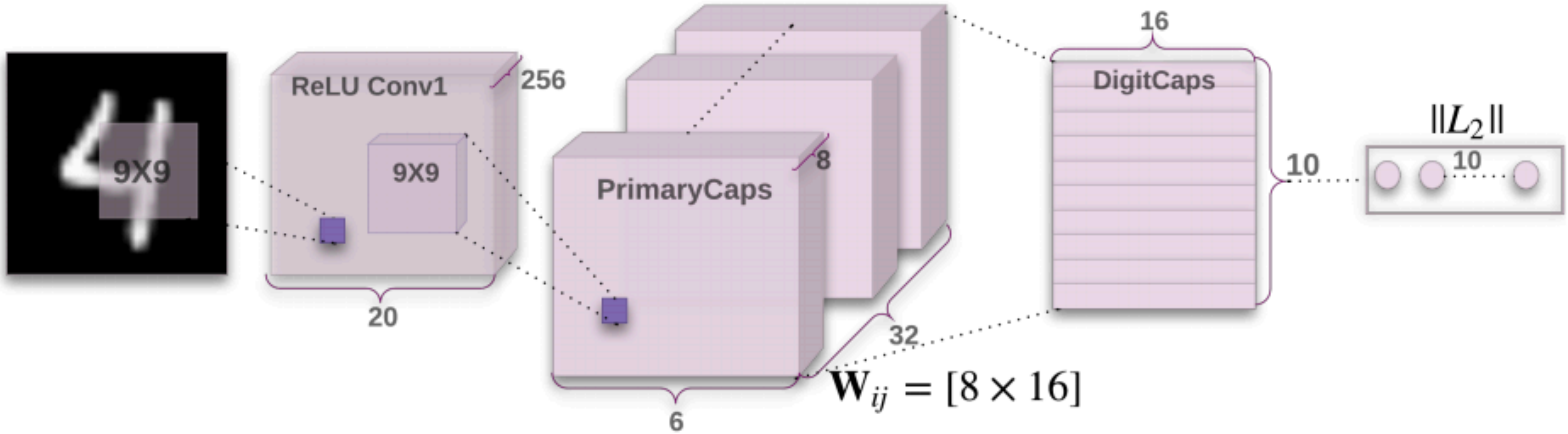
$$\text{PFN:} \quad F\left(\sum_{i=1}^{M} \Phi(p_i)\right)$$

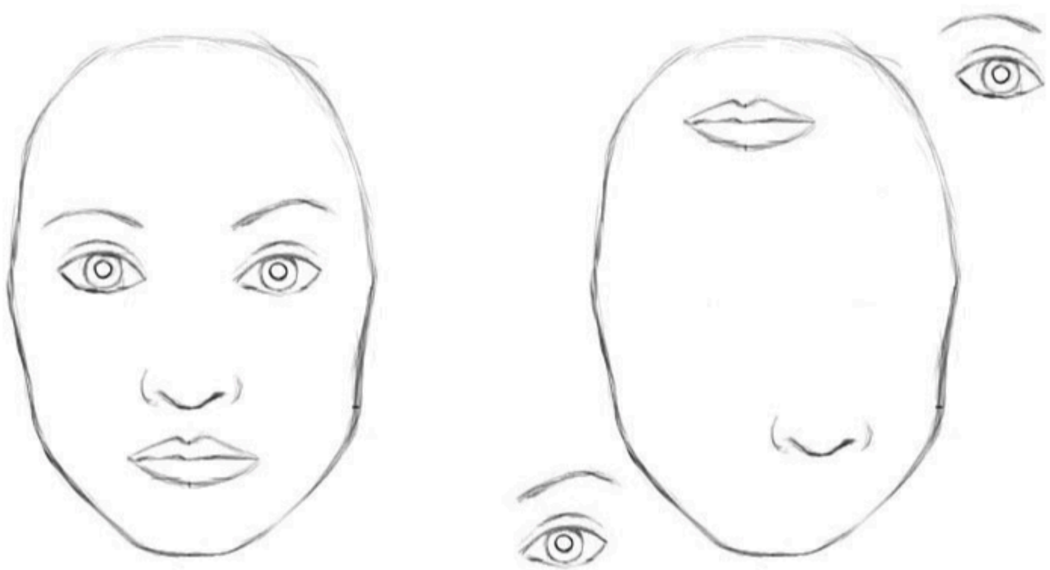
IRC safe:

$$\text{EFN:} \quad F\left(\sum_{i=1}^{M} z_i \Phi(\hat{p}_i)\right)$$

Energy Flow Networks: Deep Sets for Particle Jets, PT Komiske, EM Metodiev, J Thaler, 1810.05165

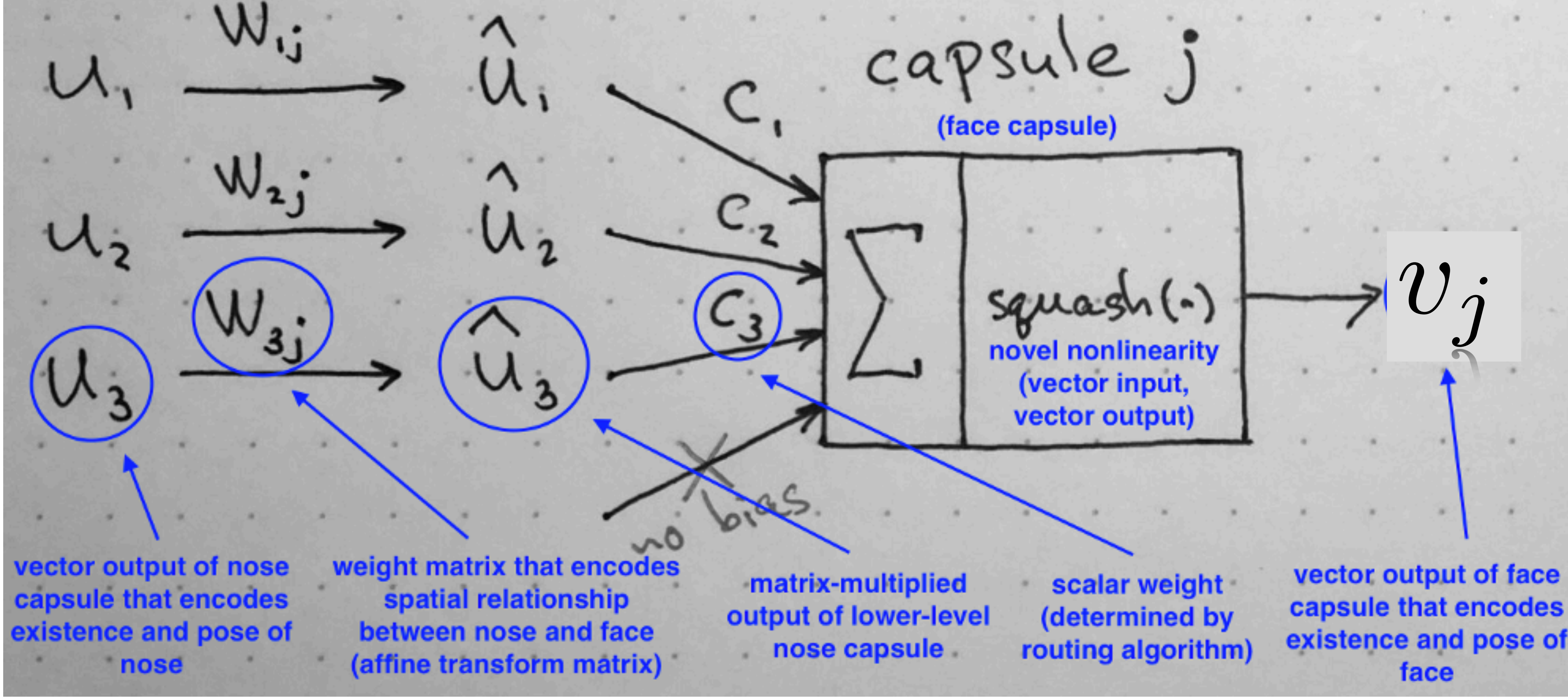
Capsule Network





- CNNs learn features, problem of spatial correlation
- Capsules are a new building block for image recognition
- Learn *instantiation vector*
- Connection by agreement (co-firing)

Dynamic Routing Between Capsules
S Sabour, N Frosst, GE Hinton
1710.09829
(medium.com)



Softmax & Routing:

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

$$b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}.\mathbf{v}_j$$

Squash:

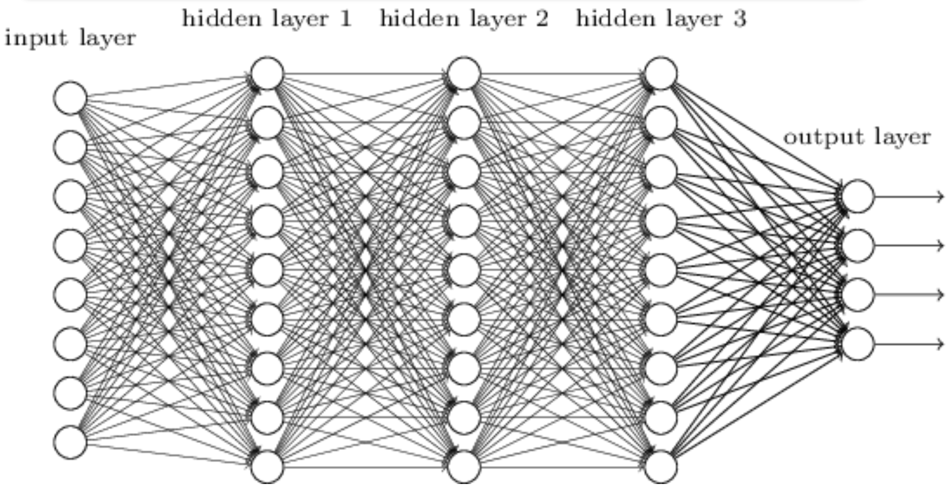
$$\mathbf{v}_j = \frac{||\mathbf{s}_j||^2}{1+||\mathbf{s}_j||^2}\frac{\mathbf{s}_j}{||\mathbf{s}_j||}$$

- Vector instead of scalar representation
  - Instantiation and relative positioning
- Routing by agreement

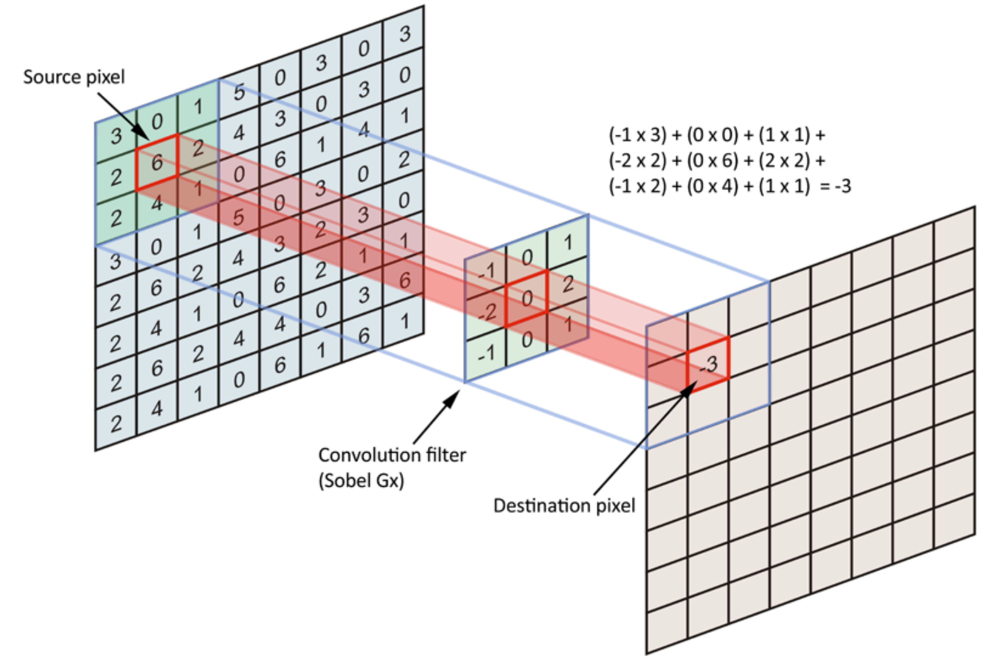
Dynamic Routing Between Capsules
S Sabour, N Frosst, GE Hinton
1710.09829
pechyonkin.me

Representation

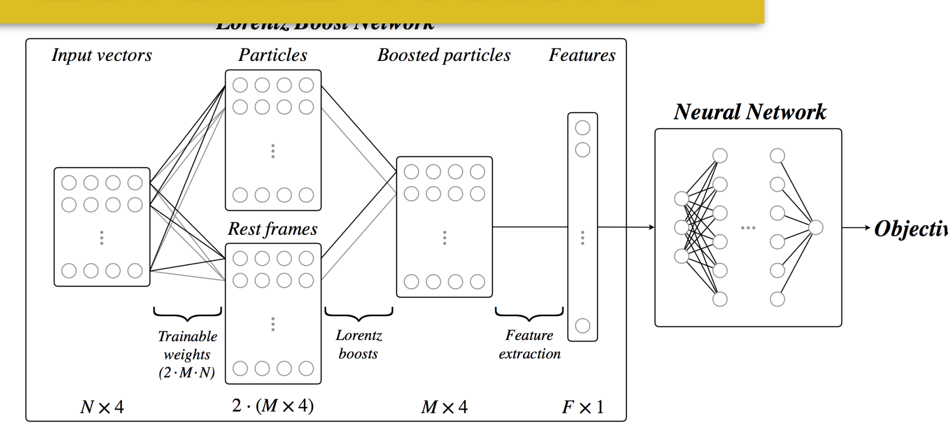
High-level



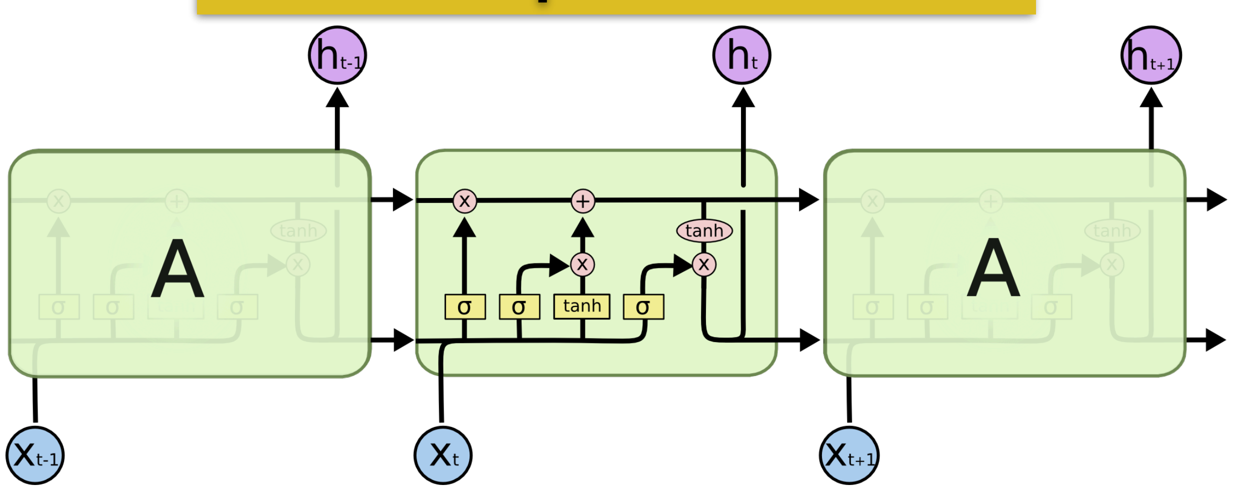
Regular grid



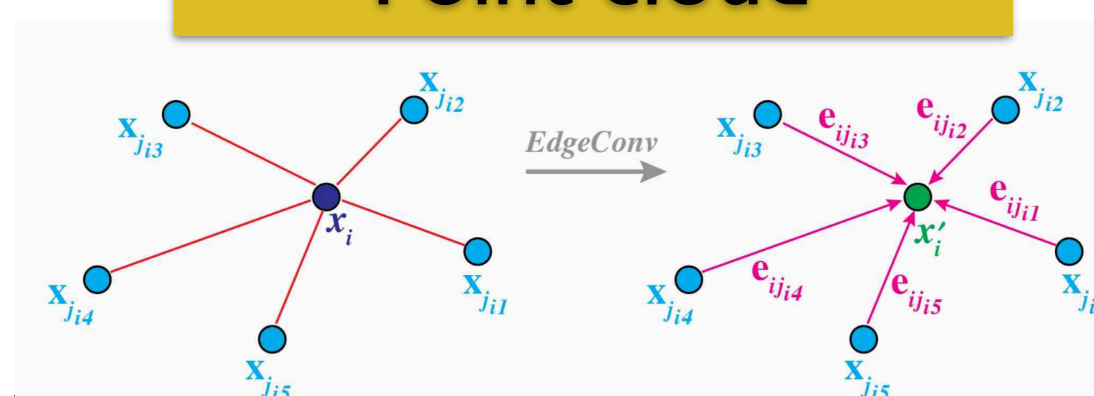
Lorentz vectors



Sequences



Point cloud



Information

- *What can we give the network to train?*

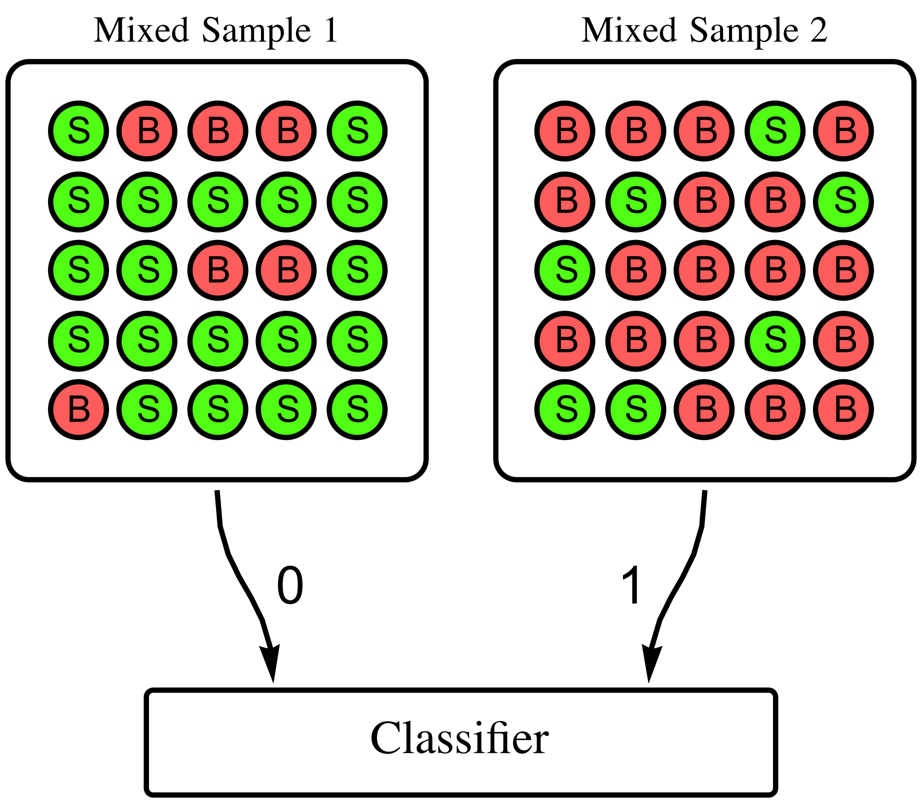
Supervised

Train
on MC simulation
or
Other source of labels
(humans)

↓

Apply to Data

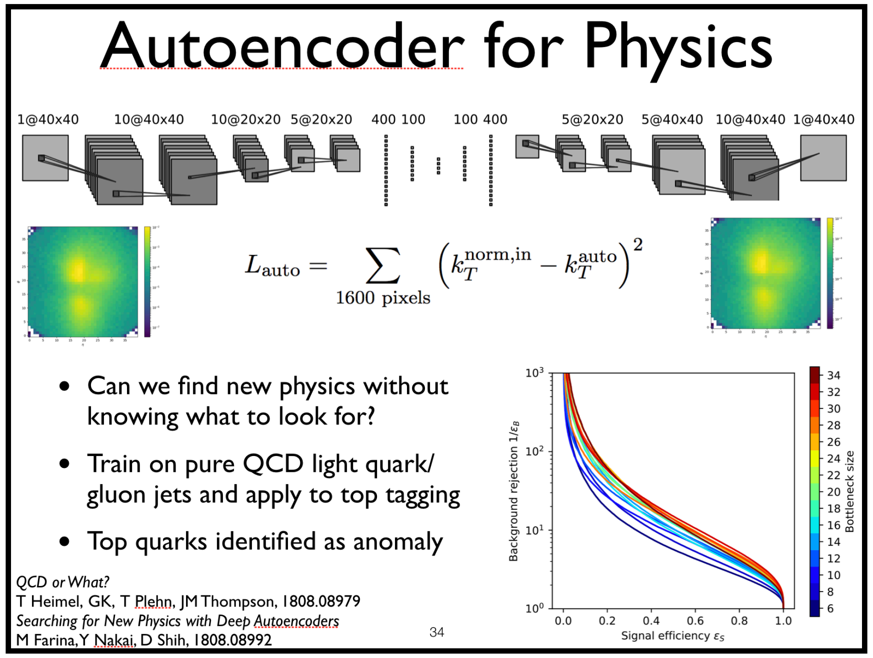
Weakly supervised



$$L_{M_1/M_2} = \frac{p_{M_1}}{p_{M_2}} = \frac{f_1\, p_S + (1-f_1)\, p_B}{f_2\, p_S + (1-f_2)\, p_B} = \frac{f_1\, L_{S/B} + (1-f_1)}{f_2\, L_{S/B} + (1-f_2)}$$

1708.02949

Unsupervised



Bonus Slides

Backup

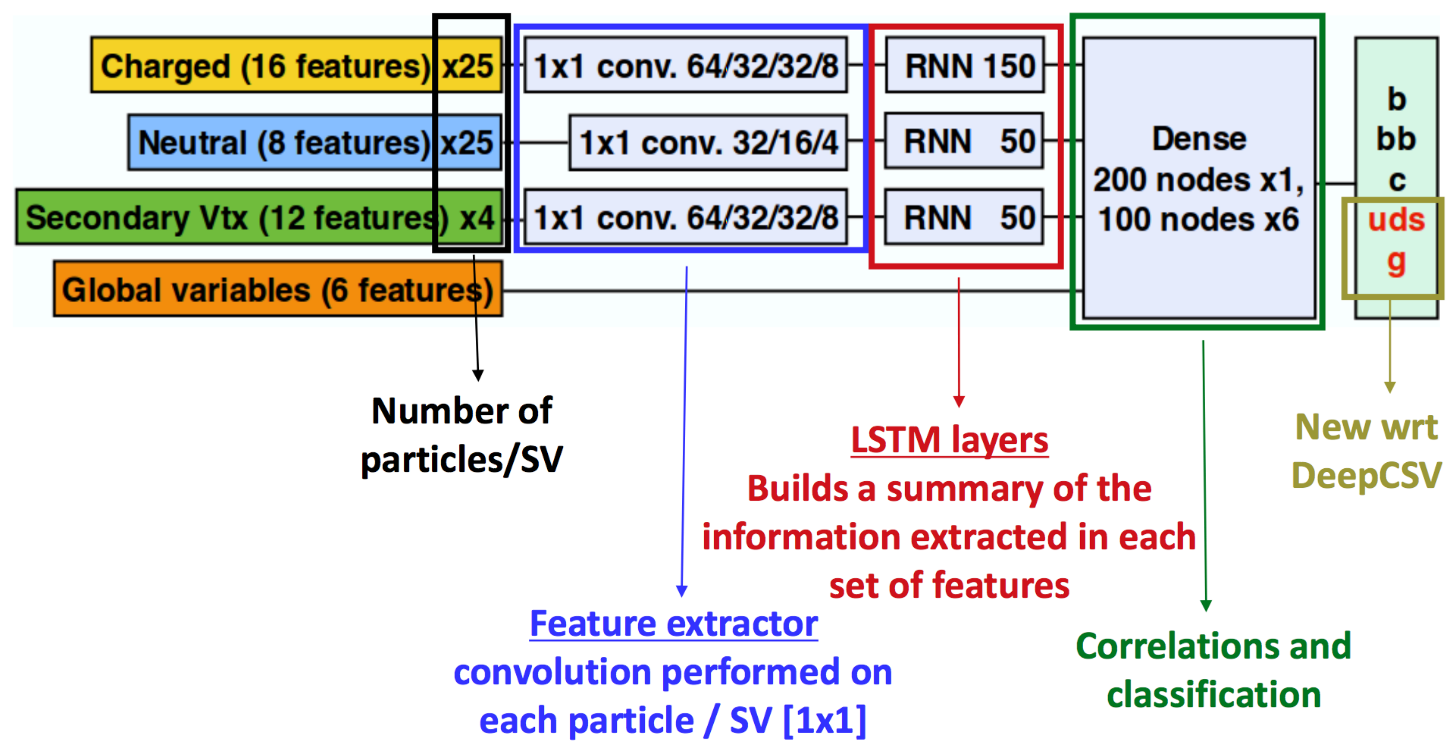
P-CNN

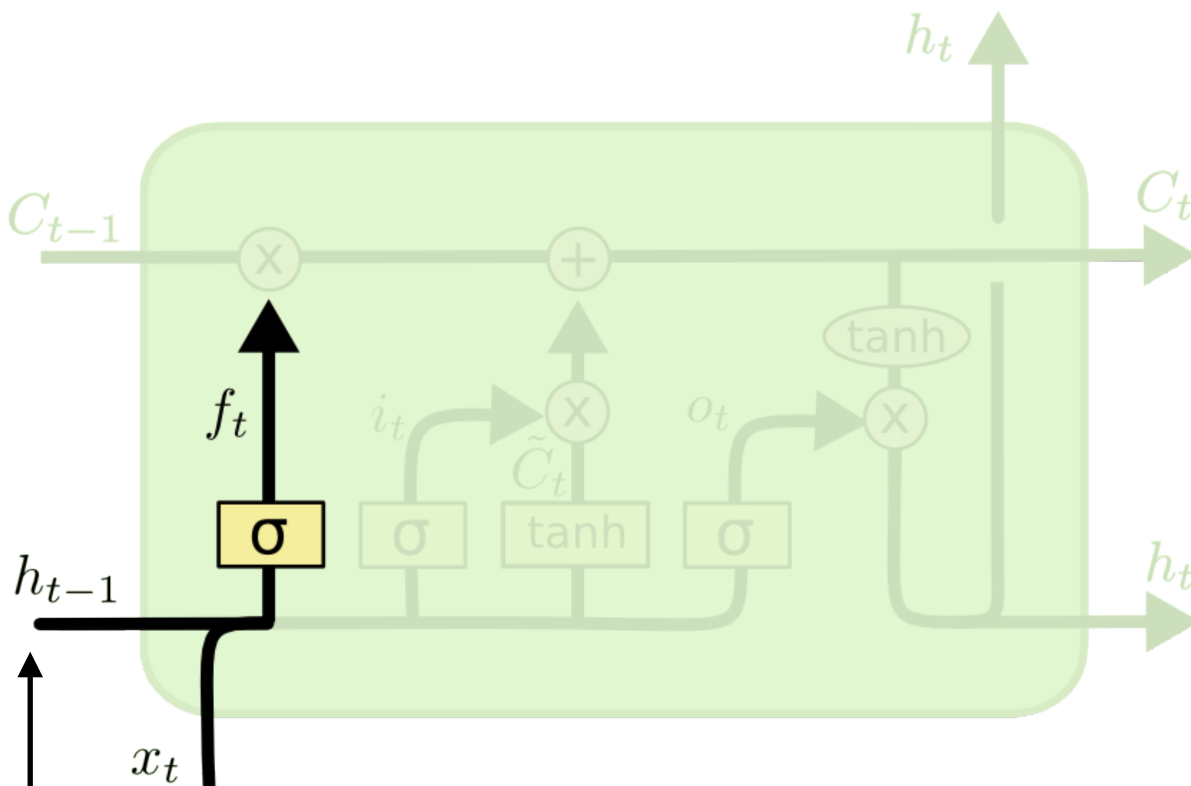
Variable	Definition
$\Delta\eta$	difference in pseudorapidity between the particle and the jet axis
$\Delta\phi$	difference in azimuthal angle between the particle and the jet axis
$\log p_T$	logarithm of the particle's p_T
$\log E$	logarithm of the particle's energy
$\log \frac{p_T}{p_T(\text{jet})}$	logarithm of the particle's p_T relative to the jet p_T
$\log \frac{E}{E(\text{jet})}$	logarithm of the particle's energy relative to the jet energy
ΔR	angular separation between the particle and the jet axis ($\sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$)

14 1D convolution layers + fully connected

Kernel size 3

(in particle space)



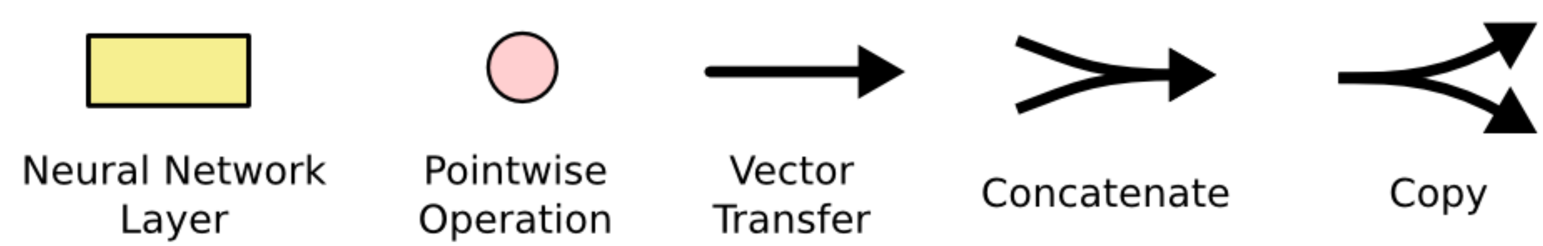


Input

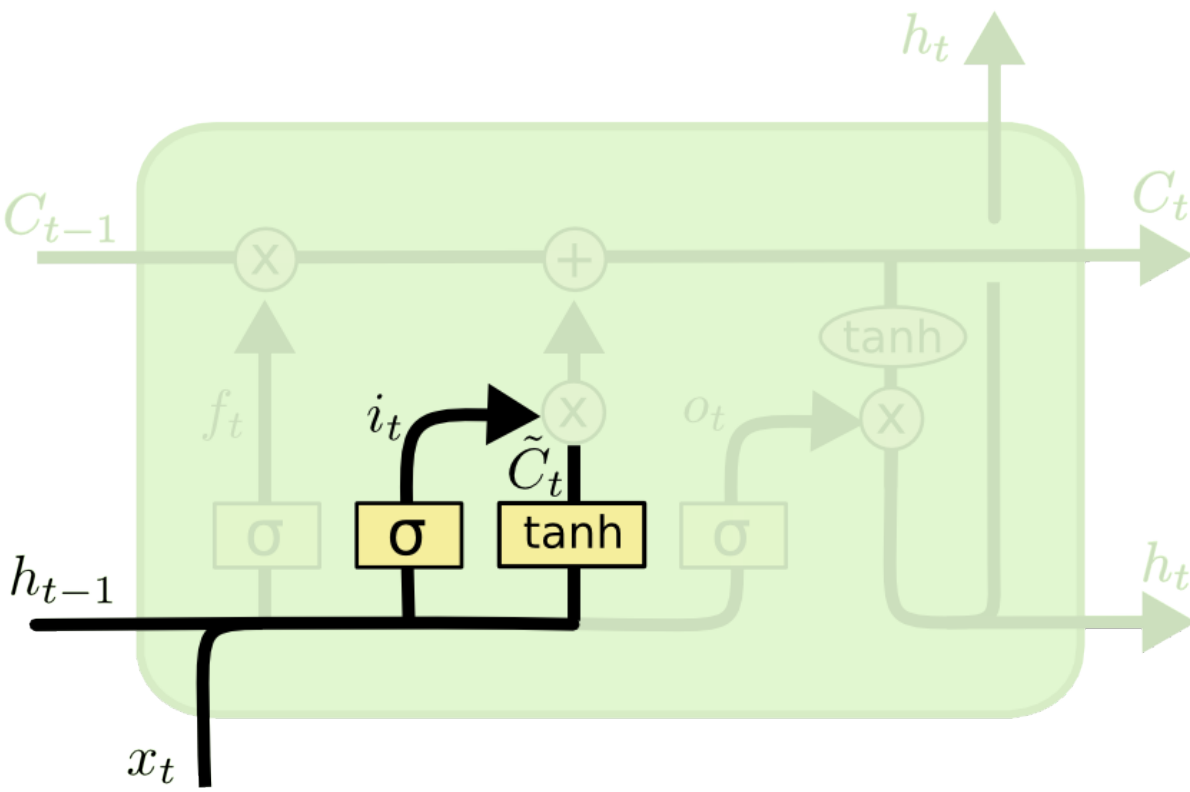
Previous hidden state

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Decide what to forget



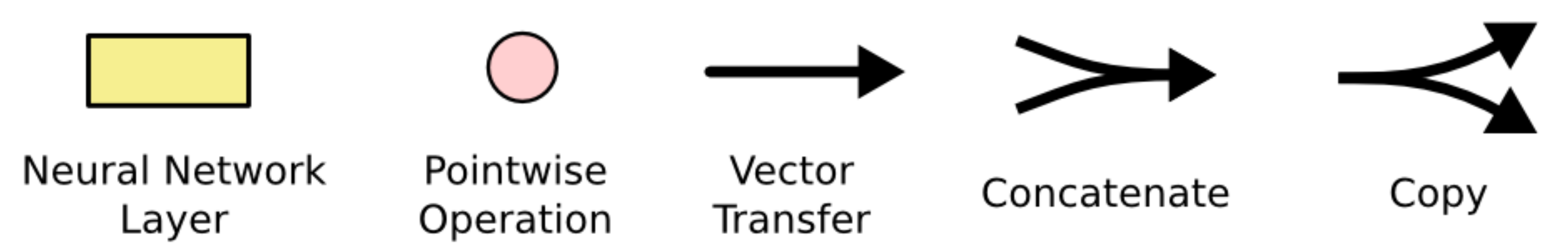
http://colah.github.io/



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

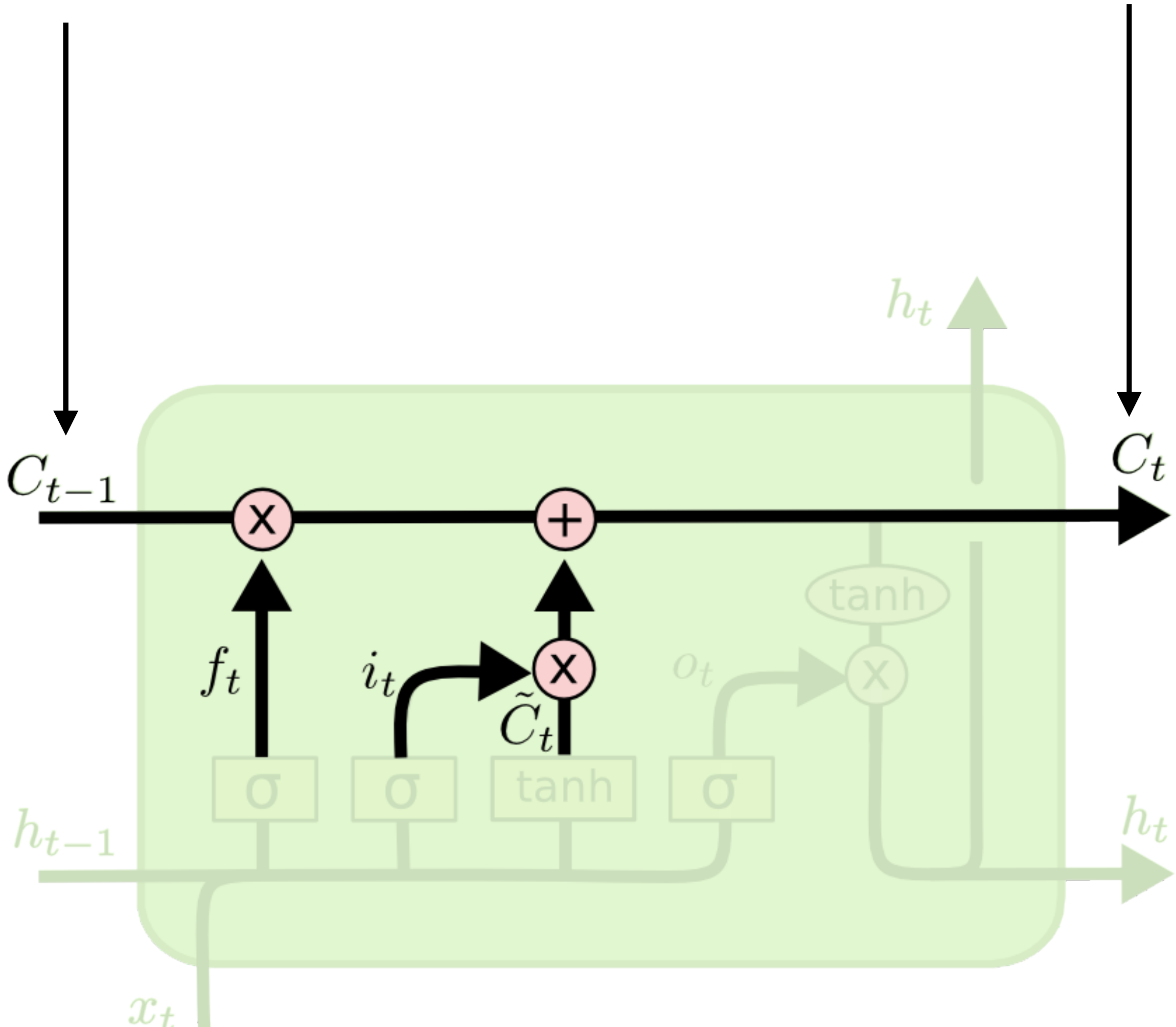
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Decide which inputs to keep?



http://colah.github.io/

Previous cell state New cell state

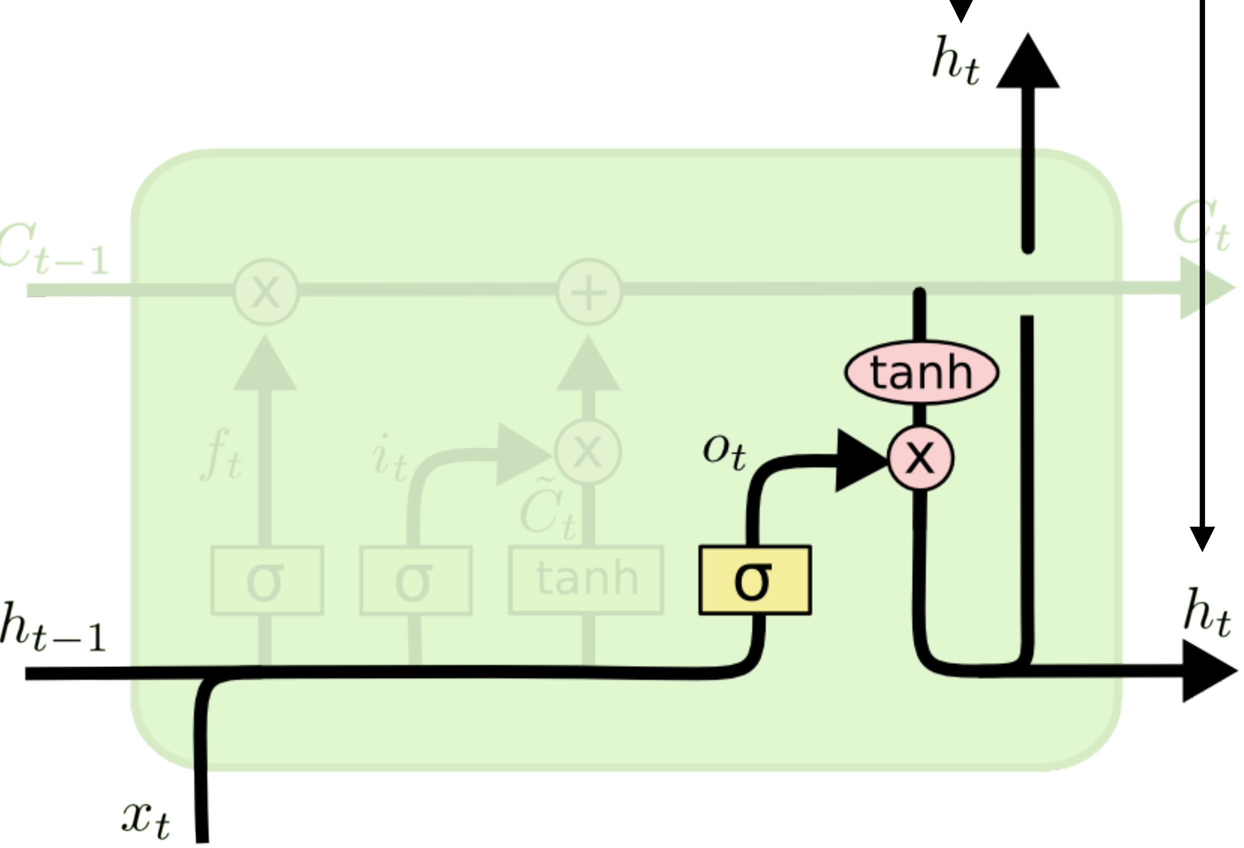


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

update cell state

http://colah.github.io/

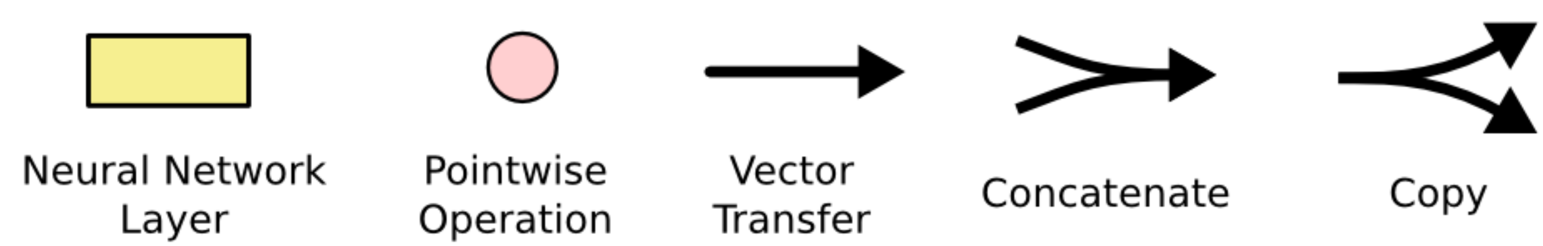
New hidden state



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

decide output

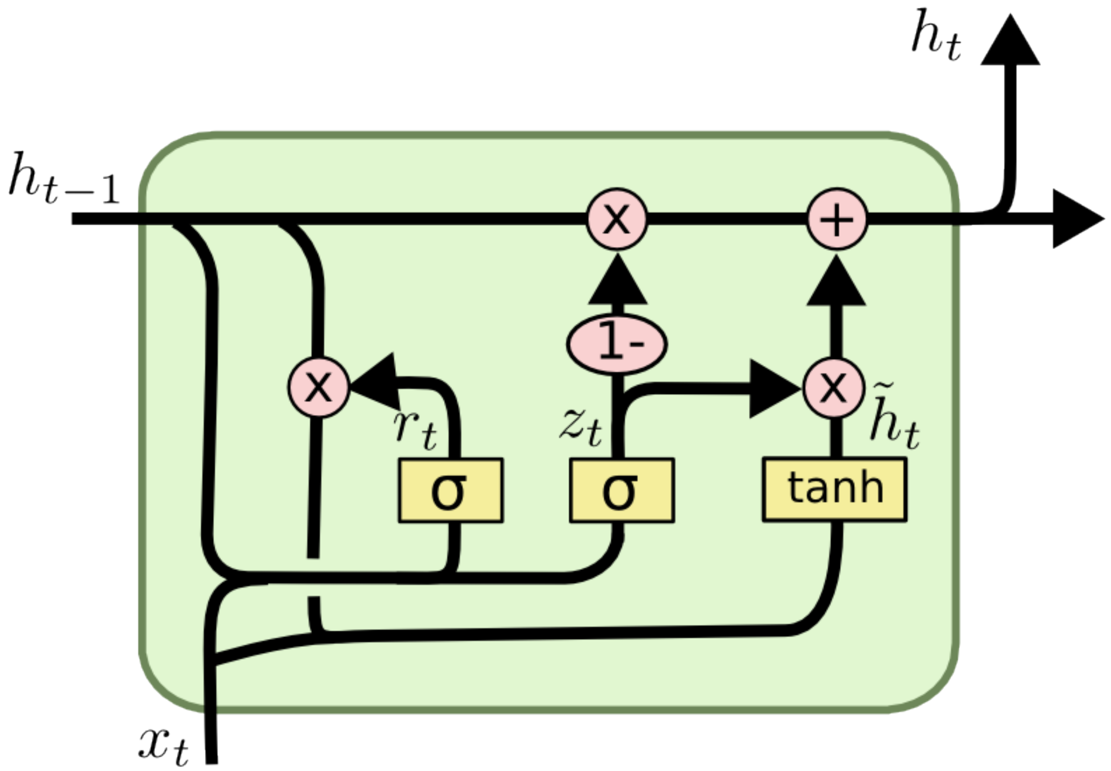


http://colah.github.io/

Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation
K Cho et al
arXiv: 1406.1078

GRU

Gated Recurrent Unit



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

- Combine forget and input gate
- Combine cell state and hidden state

http://colah.github.io/