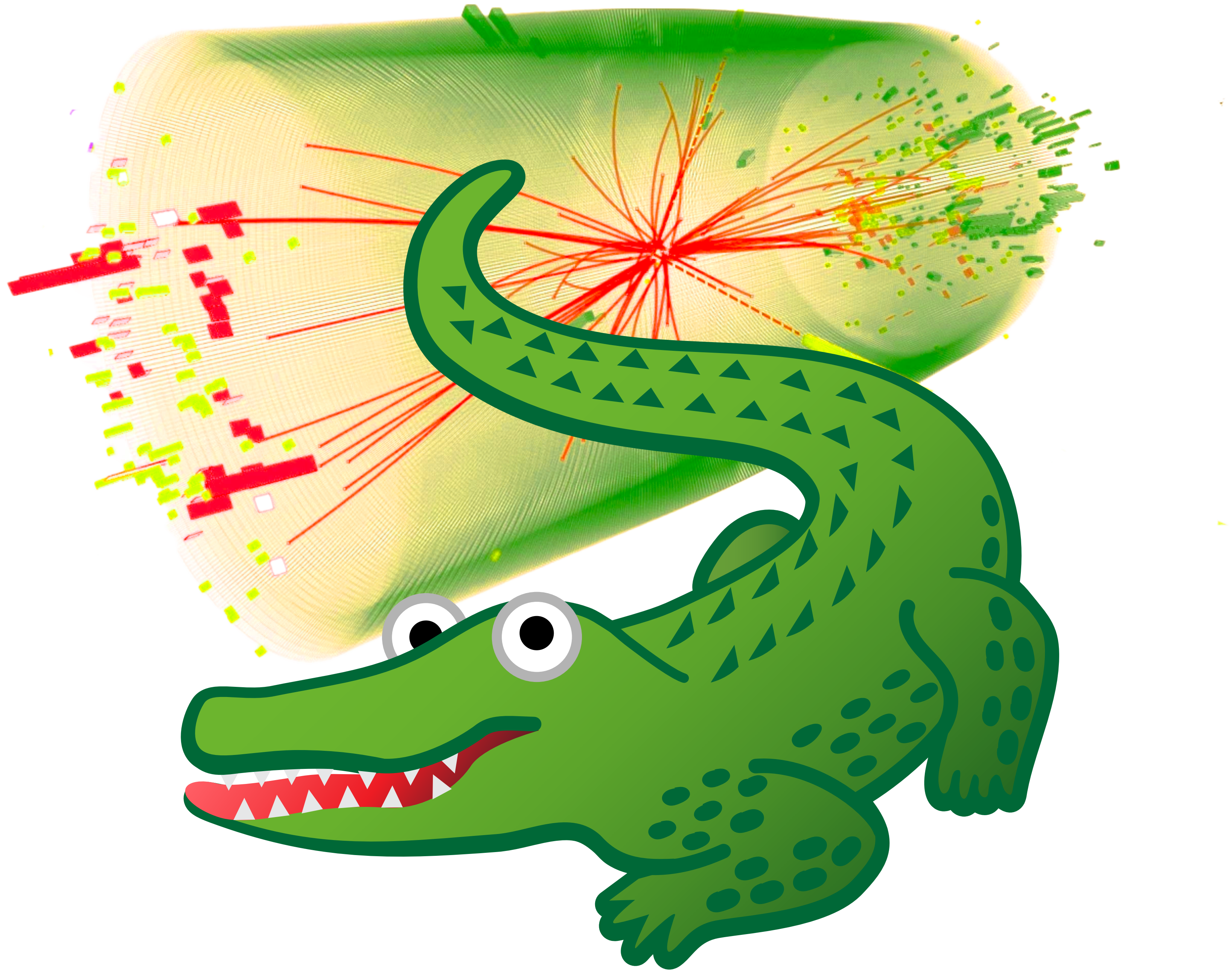
L-GATr

Lorentz-Equivariant Geometric Algebra Transformer for High-Energy Physics

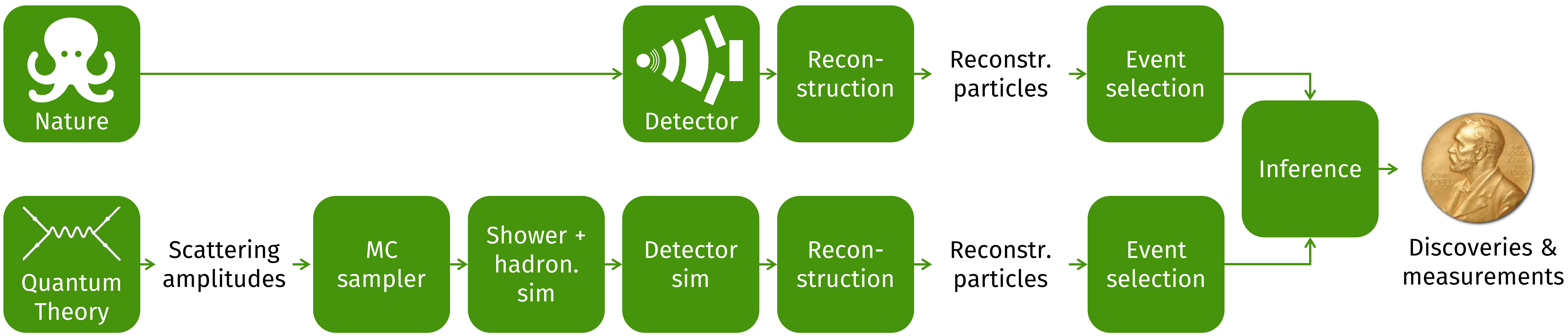


Johann Brehmer, Víctor Bresó,
Pim de Haan, Tilman Plehn, Huilin Qu,
Jonas Spinner, Jesse Thaler

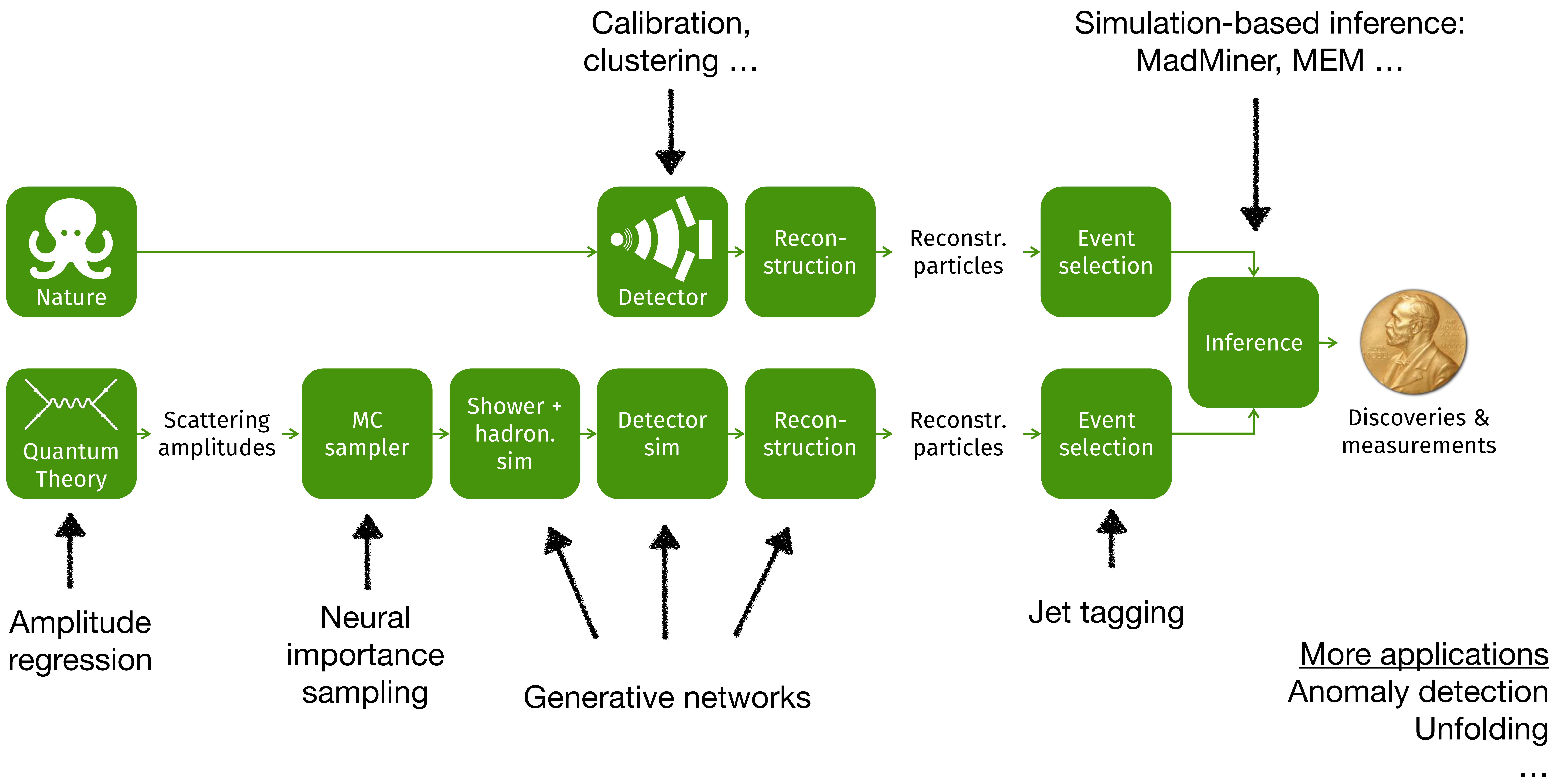
arXiv:2405.14806, arXiv:2411.00446

IRN Terascale @ IPHC Strasbourg
20.05.2025

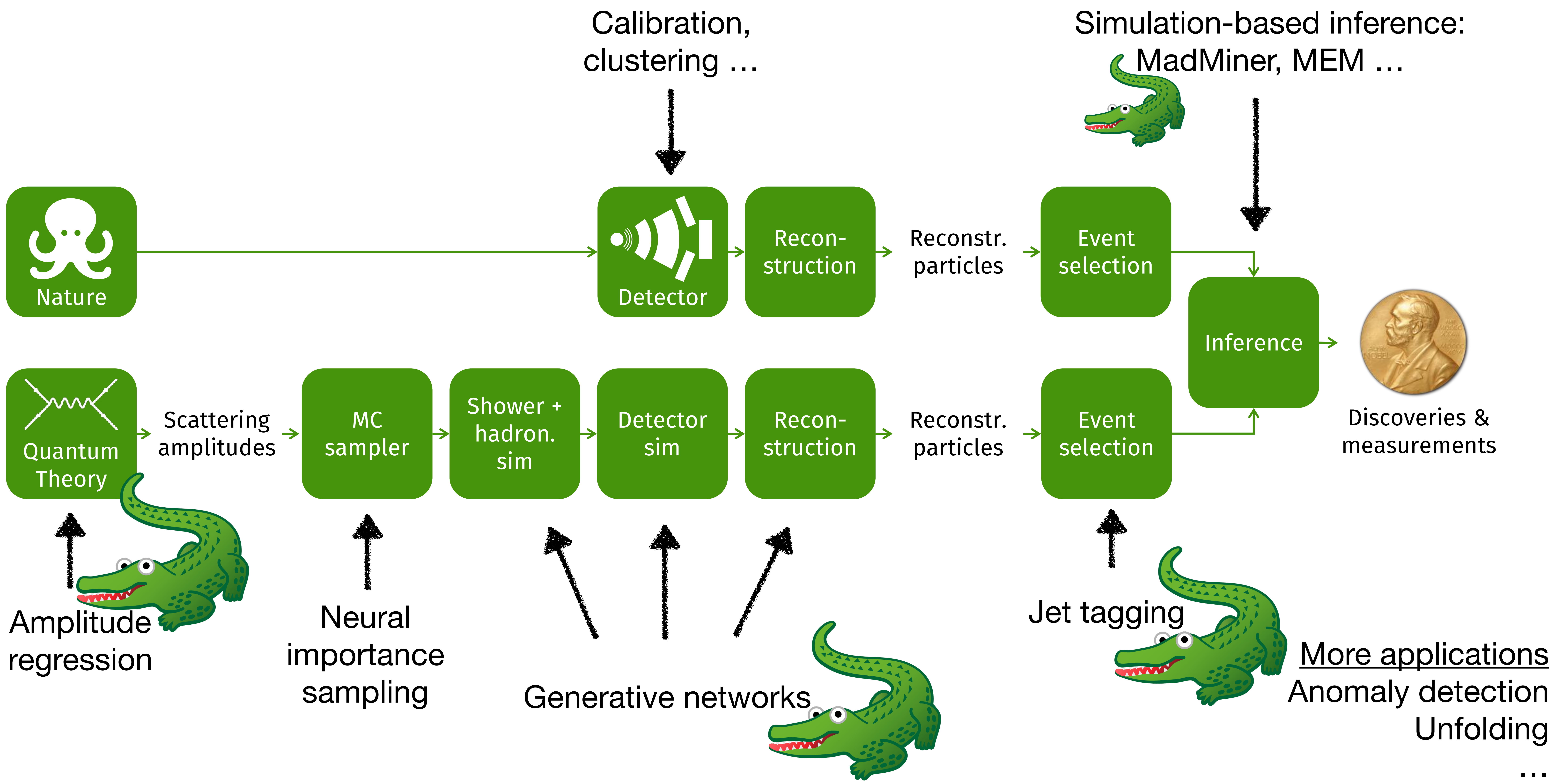
UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

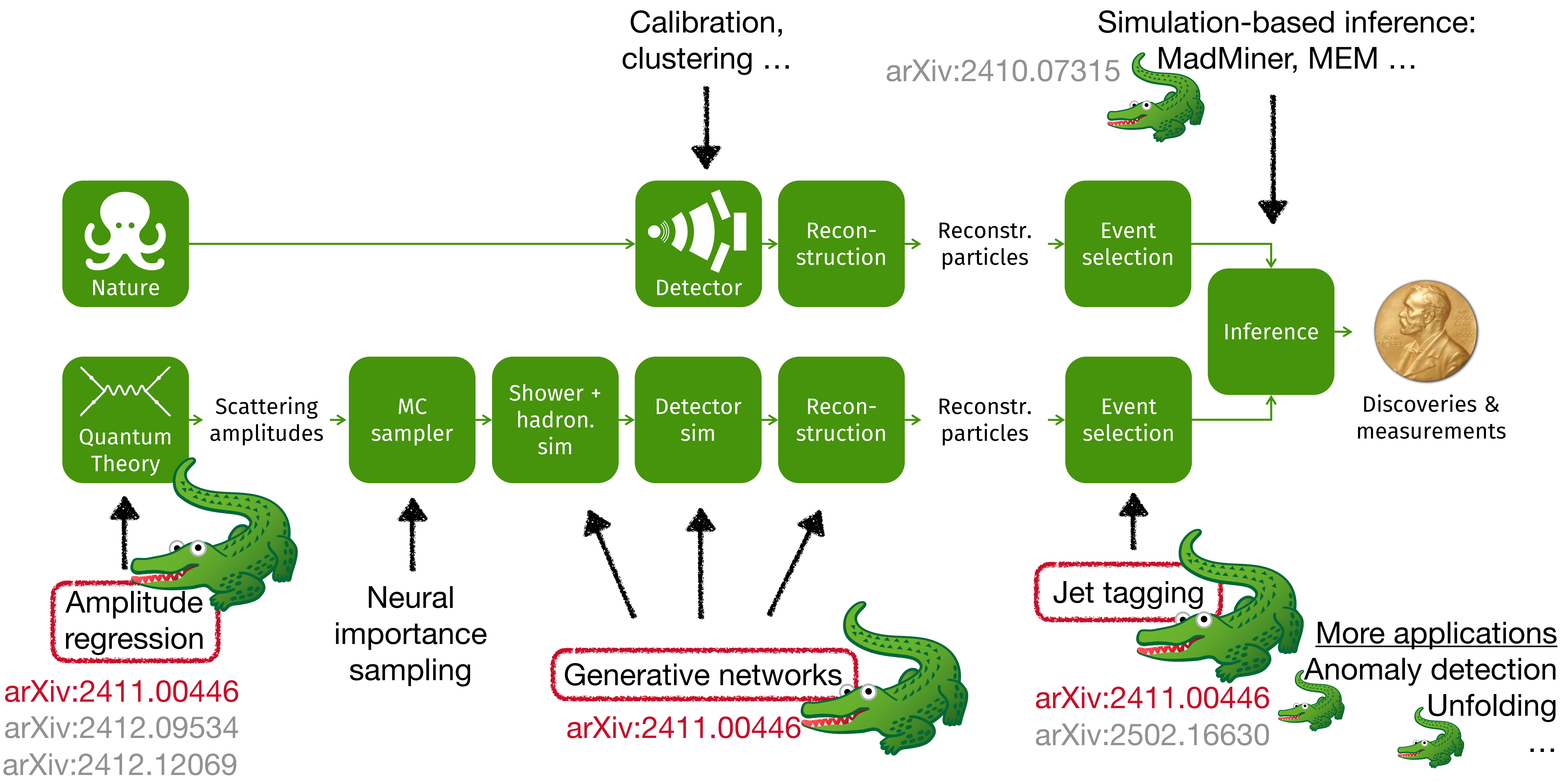


Nature
Detector
Recon-
struction
Reconstr.
particles
Event
selection
Inference
Quantum
Theory
Scattering
amplitudes
MC
sampler
Shower +
hadron.
sim
Detector
sim
Recon-
struction
Reconstr.
particles
Event
selection
Discoveries &
measurements

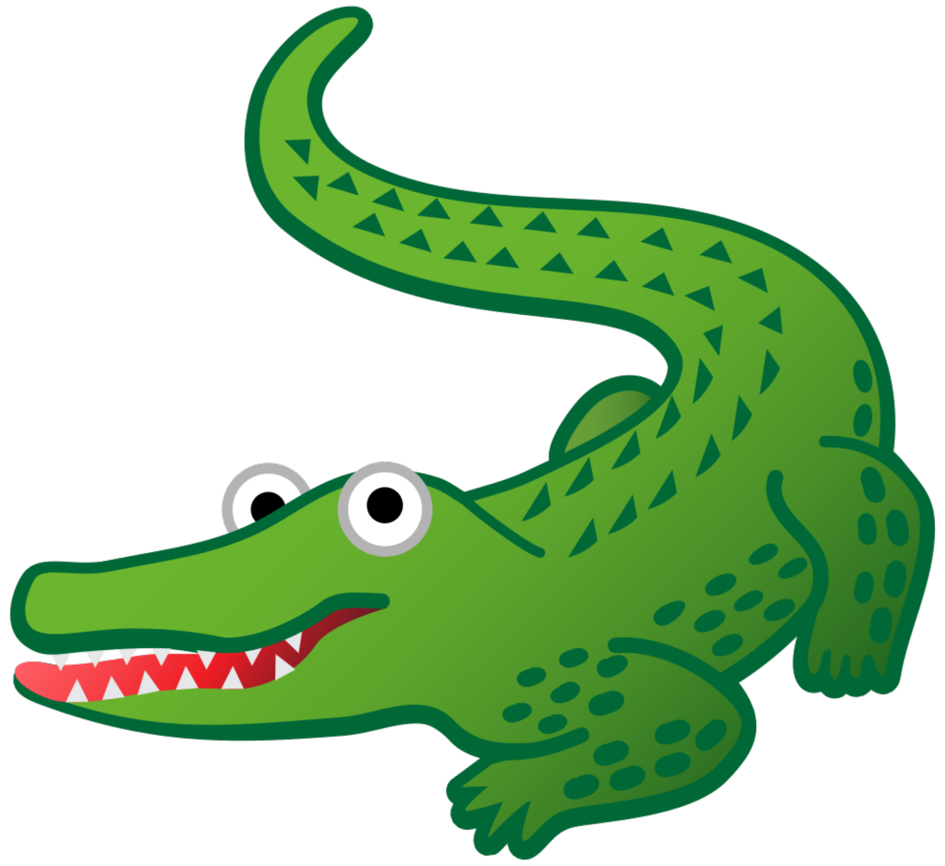


Calibration,
clustering …
Simulation-based inference:
MadMiner, MEM …
Nature
Detector
Recon-
struction
Reconstr.
particles
Event
selection
Inference
Discoveries &
measurements
Quantum
Theory
Scattering
amplitudes
MC
sampler
Shower +
hadron.
sim
Detector
sim
Recon-
struction
Reconstr.
particles
Event
selection
Amplitude
regression
Neural
importance
sampling
Generative networks
Jet tagging
More applications
Anomaly detection
Unfolding
…


Calibration,
clustering …
Simulation-based inference:
MadMiner, MEM …
Nature
Detector
Recon-
struction
Reconstr.
particles
Event
selection
Inference
Discoveries &
measurements
Quantum
Theory
Scattering
amplitudes
MC
sampler
Shower +
hadron.
sim
Detector
sim
Recon-
struction
Reconstr.
particles
Event
selection
Amplitude
regression
Neural
importance
sampling
Generative networks
Jet tagging
More applications
Anomaly detection
Unfolding
…



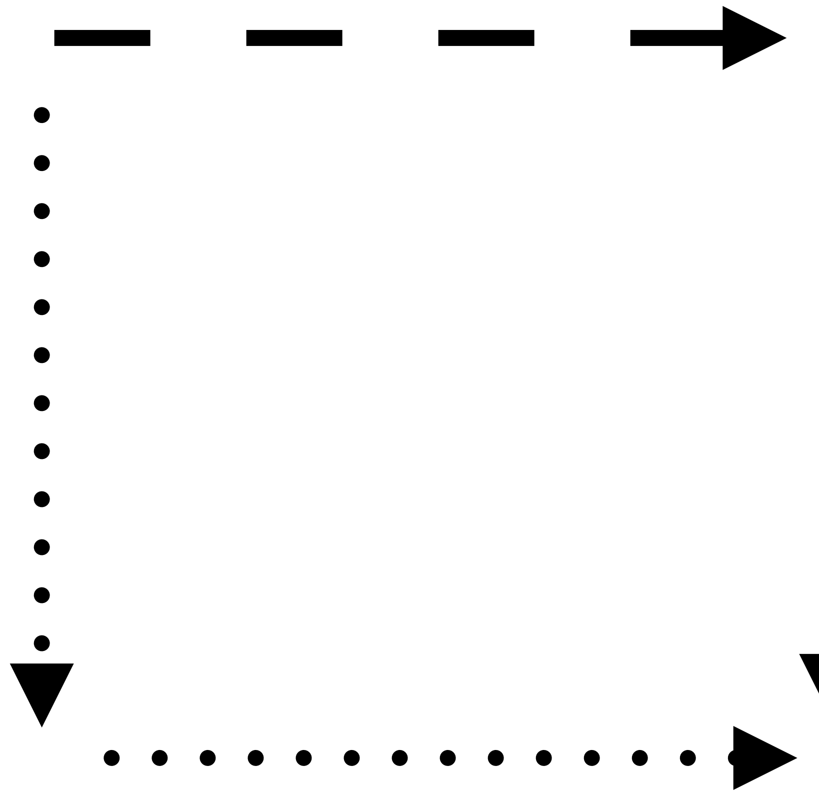
Calibration,
clustering …
Simulation-based inference:
MadMiner, MEM …
arXiv:2410.07315
Nature
Detector
Recon-
struction
Reconstr.
particles
Event
selection
Inference
Discoveries &
measurements
Quantum
Theory
Scattering
amplitudes
MC
sampler
Shower +
hadron.
sim
Detector
sim
Recon-
struction
Reconstr.
particles
Event
selection
Amplitude
regression
arXiv:2411.00446
arXiv:2412.09534
arXiv:2412.12069
Neural
importance
sampling
Generative networks
arXiv:2411.00446
Jet tagging
arXiv:2411.00446
arXiv:2502.16630
More applications
Anomaly detection
Unfolding
…



Lorentz-Equivariant
Geometric **A**lgebra
Transformer

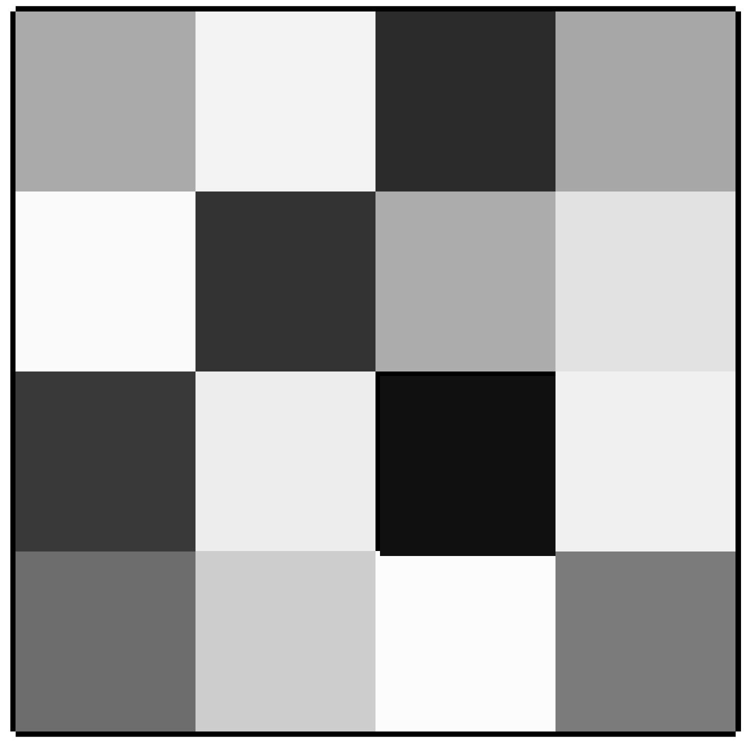
$$= \begin{pmatrix} 1 \\ \gamma^\mu \\ \sigma^{\mu\nu} \\ \gamma^\mu \gamma_5 \\ \gamma_5 \end{pmatrix} +$$

Geometric algebra
representations



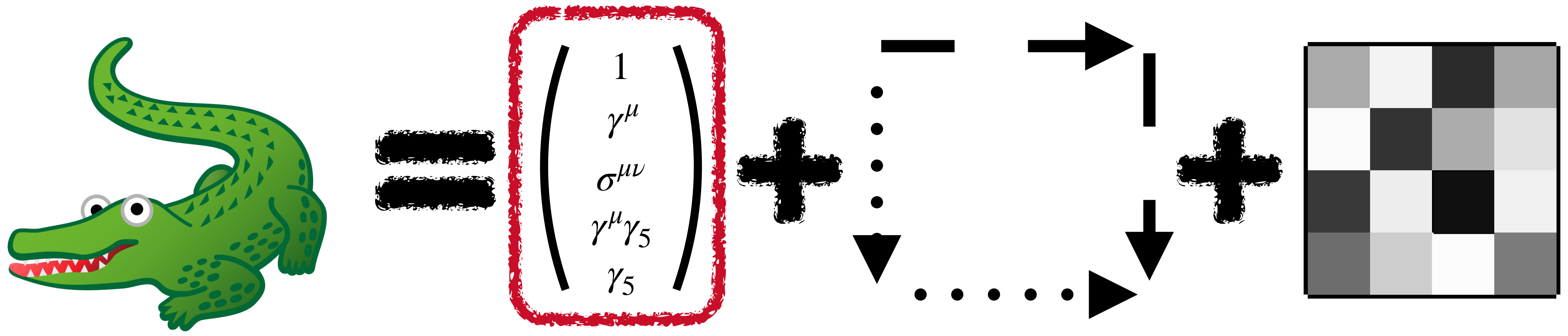
$+$

Lorentz-Equivariant
layers



Transformer
architecture

GATr was originally
developed for E(3)
arXiv:2305.18415

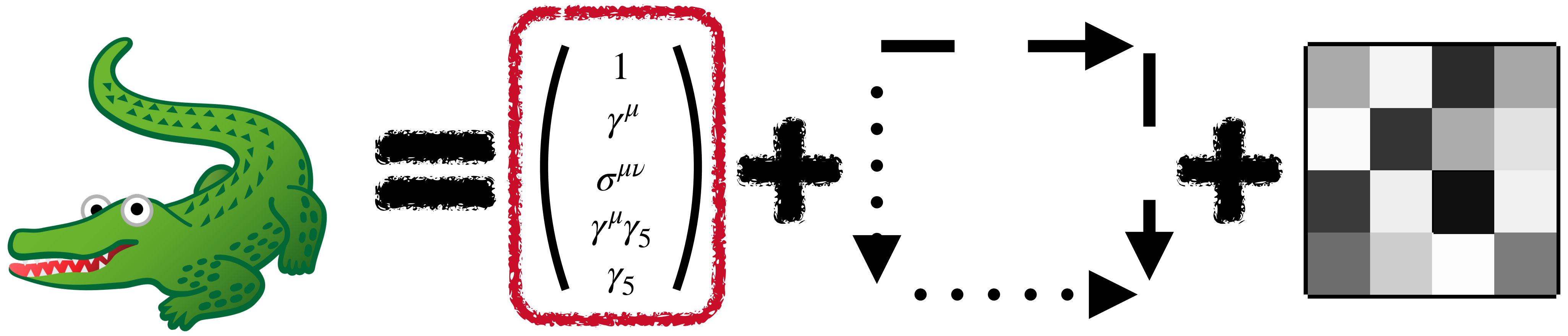


L-GATr

Geometric algebra = Clifford algebra

Geometric algebra = Vector space + geometric product $xy = \frac{\{x, y\}}{2} + \frac{[x, y]}{2}$

L-GATr



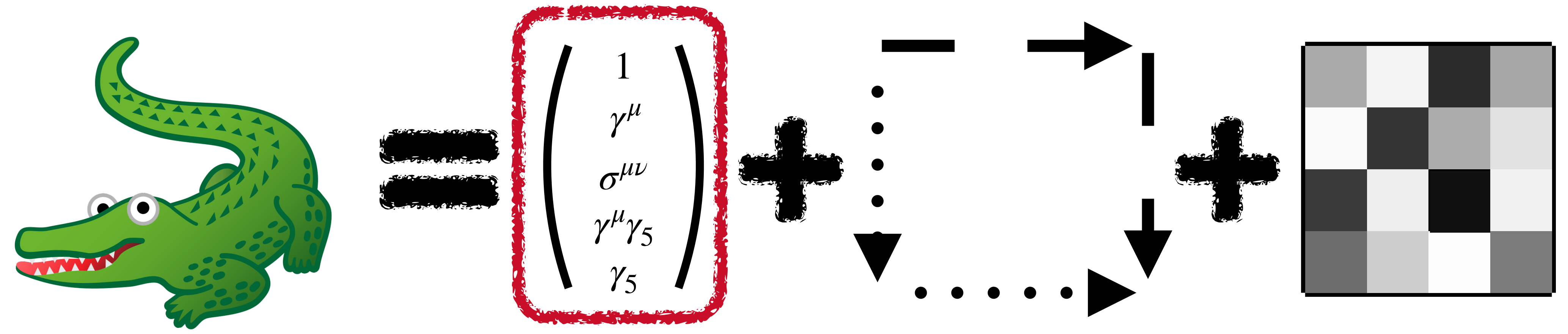
Geometric algebra = Clifford algebra

Geometric algebra = Vector space + geometric product $xy = \frac{\{x, y\}}{2} + \frac{[x, y]}{2}$

Spacetime geometric algebra: Geometric algebra over vector space $\mathbb{R}^4$ with Minkowski metric $g = \mathrm{diag}(1, -1, -1, -1)$

- Basis elements γ^μ are orthonormal: $\{\gamma^\mu, \gamma^\nu\} = 2g^{\mu\nu}$
- Dirac algebra is the same up to $\mathbb{R} \to \mathbb{C}$

L-GATr



Building multivectors

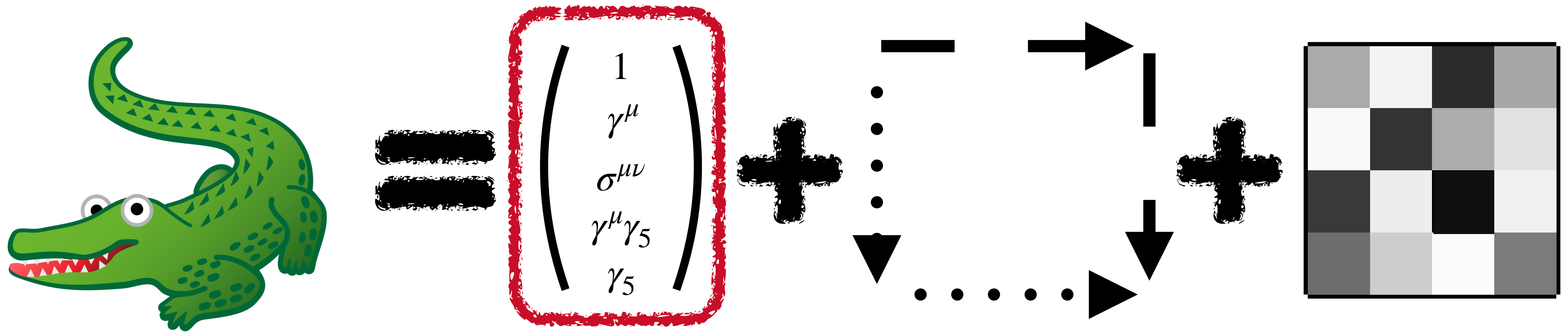
- Scalar and vectors $1,\ \gamma^\mu$ (1+4 objects)
- Product of two vectors: $\gamma^\mu\gamma^\nu = \dfrac{\{\gamma^\mu,\gamma^\nu\}}{2} + \dfrac{[\gamma^\mu,\gamma^\nu]}{2} = g^{\mu\nu} + \sigma^{\mu\nu}$ (6 new objects)
- Axial vector: $\epsilon_{\mu\nu\rho\sigma}\gamma^\nu\gamma^\rho\gamma^\sigma \propto \gamma_\mu\gamma^5$ (4 new objects)
- Pseudoscalar: $\gamma^5 = \gamma^0\gamma^1\gamma^2\gamma^3 = \dfrac{1}{4!}\epsilon_{\mu\nu\rho\sigma}\gamma^\mu\gamma^\nu\gamma^\rho\gamma^\sigma$ (1 new object)

Multivector: $x = x^S 1 + x^V_\mu\gamma^\mu + x^B_{\mu\nu}\sigma^{\mu\nu} + x^A_\mu\gamma^\mu\gamma^5 + x^P\gamma^5$ with $(x^S, x^V_\mu, x^B_{\mu\nu}, x^A_\mu, x^P) \in \mathbb{R}^{16}$

Particle: $x^V = (E, p_x, p_y, p_z), \quad x^S = \mathrm{PID}$

L-GATr

Geometric algebra representations



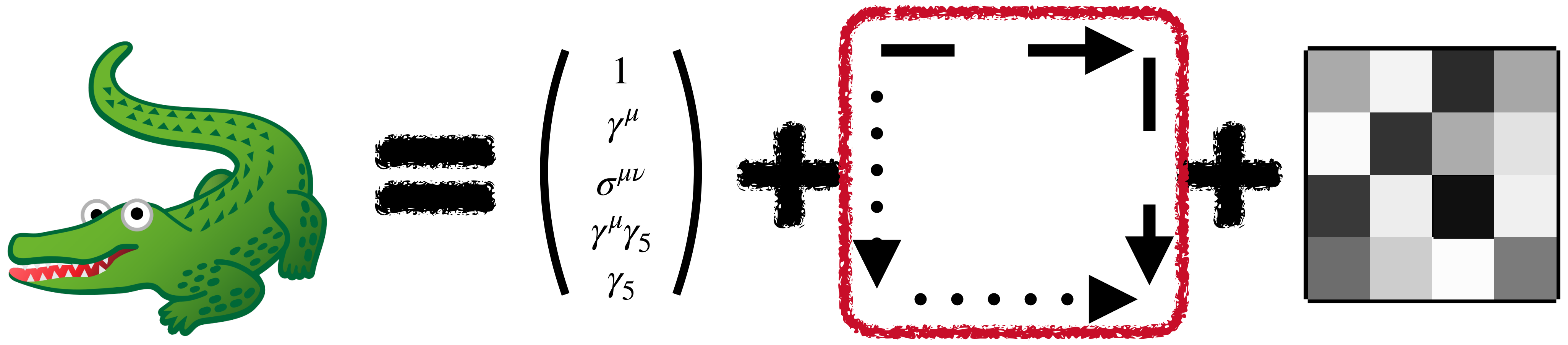
$$x^S \in \mathbb{R} \quad \Rightarrow \quad \begin{pmatrix} x^S \\ x^V_\mu \\ x^B_{\mu\nu} \\ x^A_\mu \\ x^P \end{pmatrix} \in \mathbb{R}^{16}$$

scalar channels

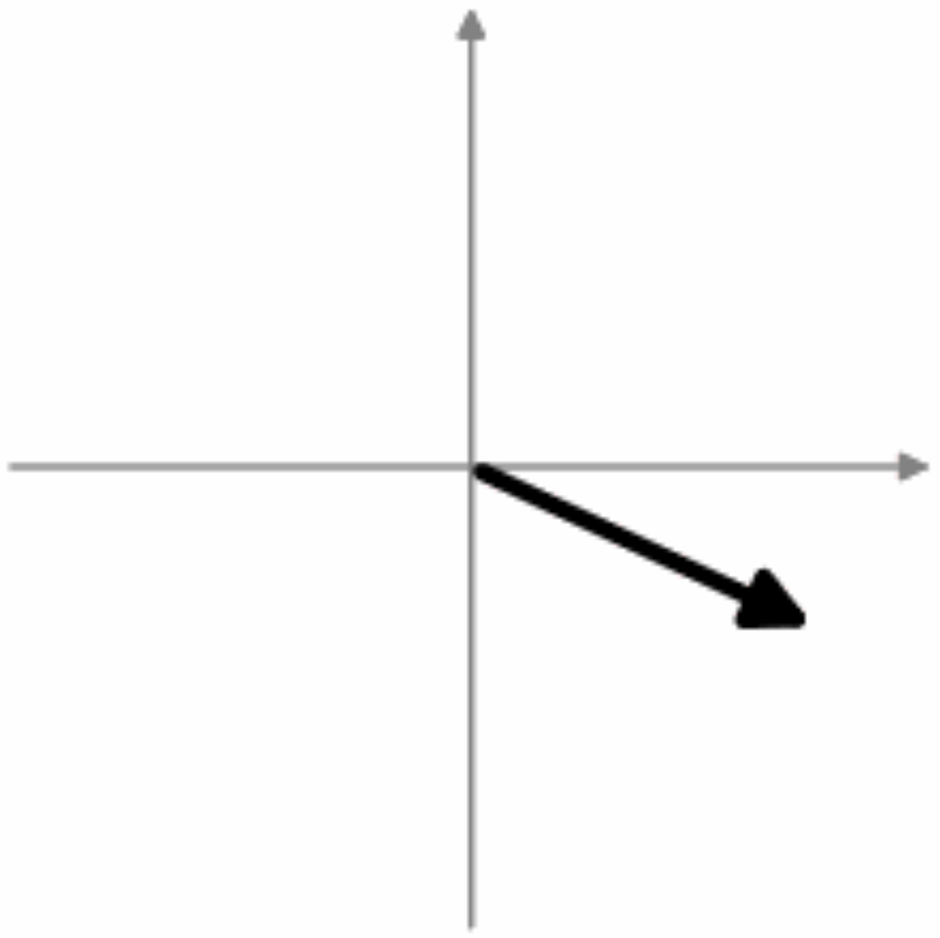
multivector channels

L-GATr

Invariance



Input



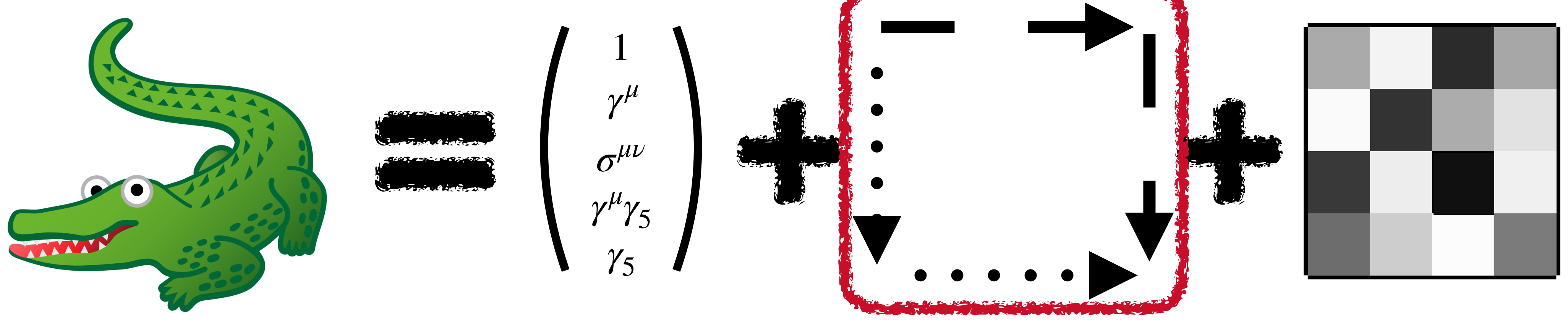
Output
(not invariant)

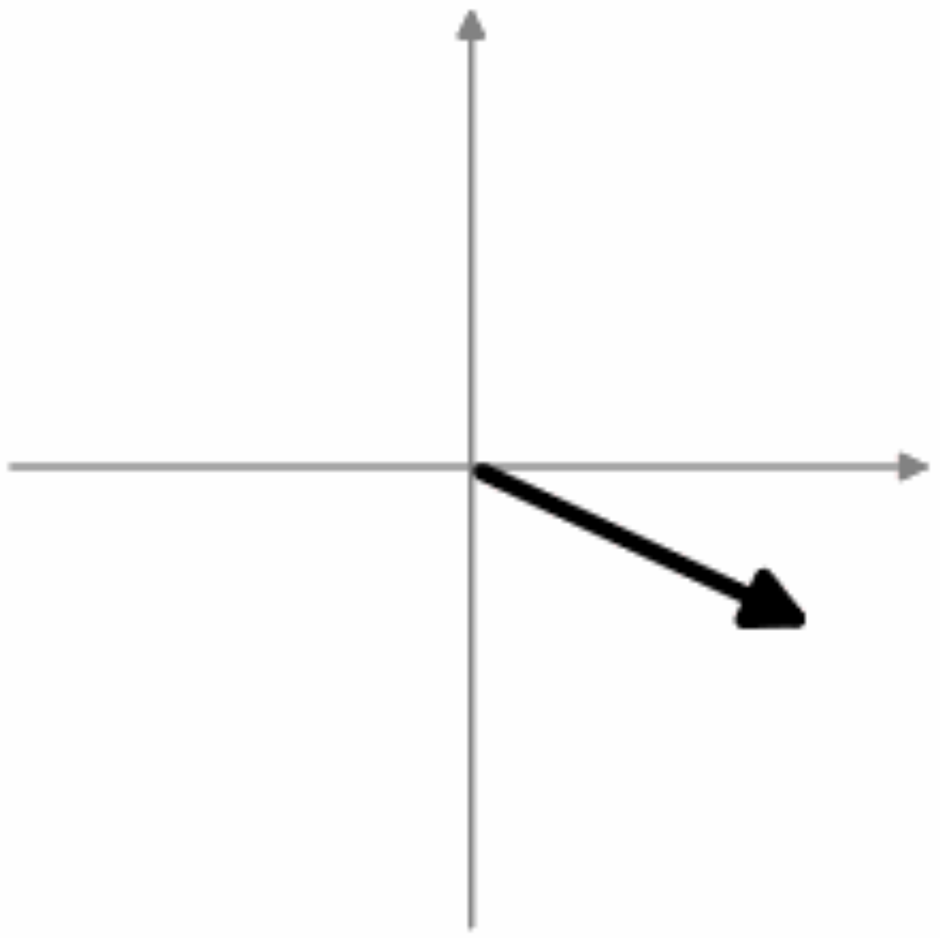
0.8

Output
(invariant)

0.8

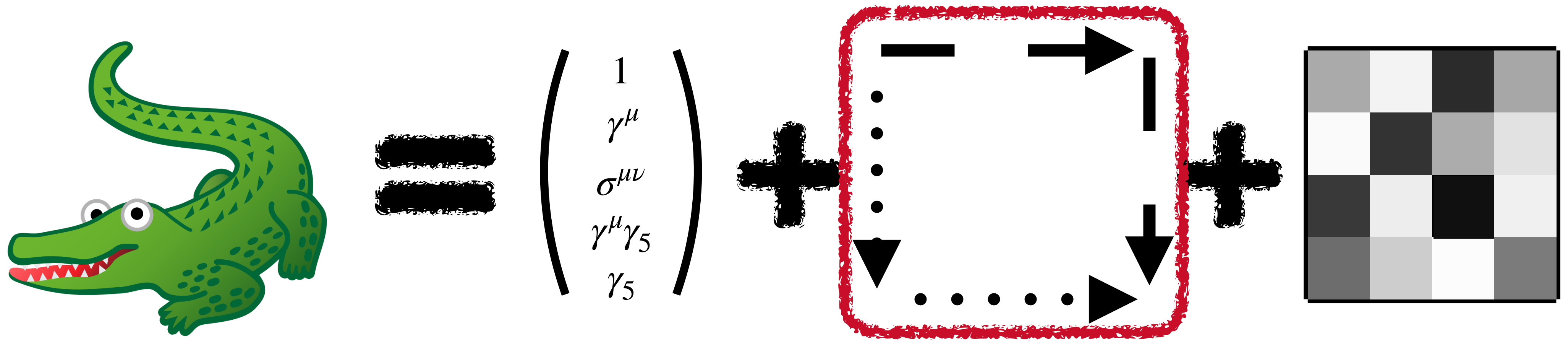
L-GATr
Invariance



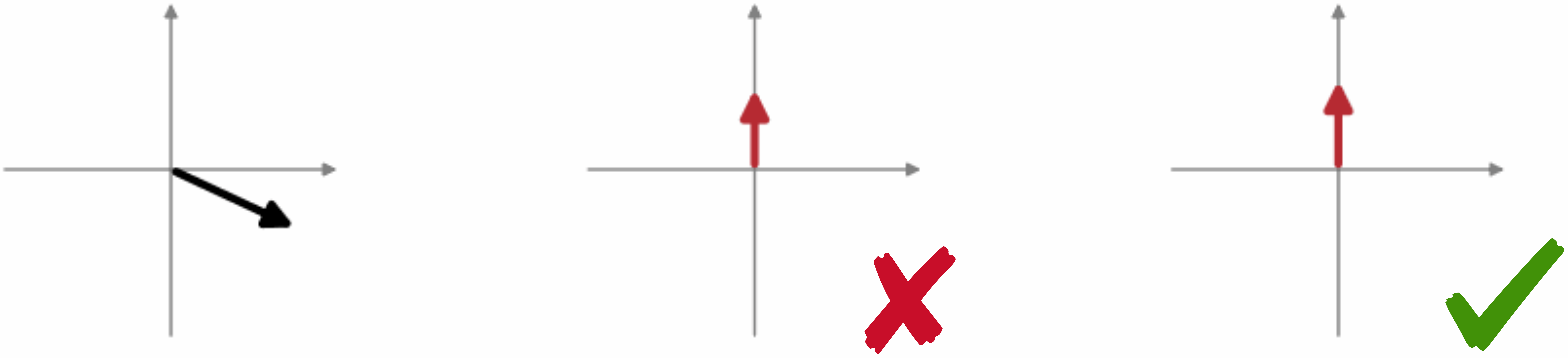
Input	Output (not invariant)	Output (invariant)
	0.8	0.8

L-GATr

Equivariance = Covariance

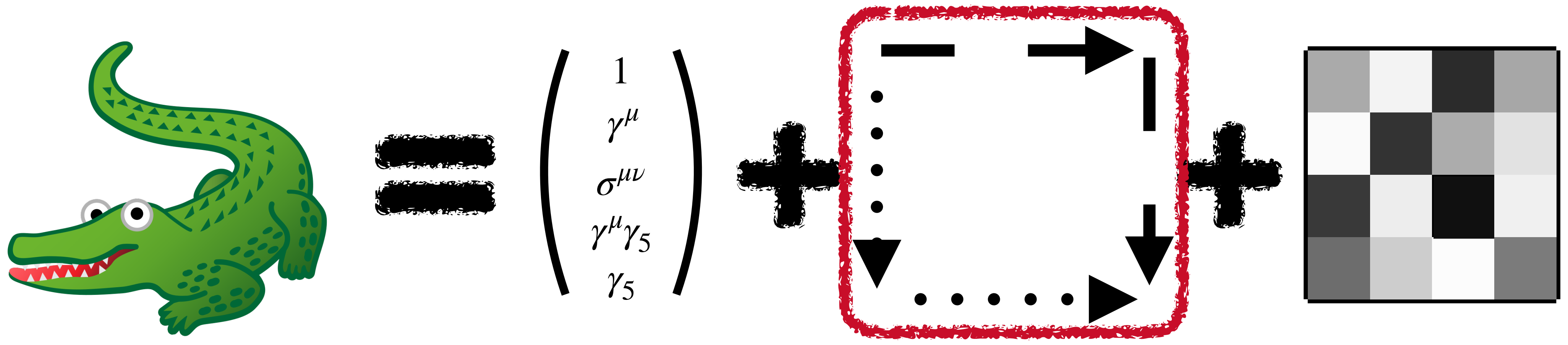


Input | Output (not equivariant) | Output (equivariant)



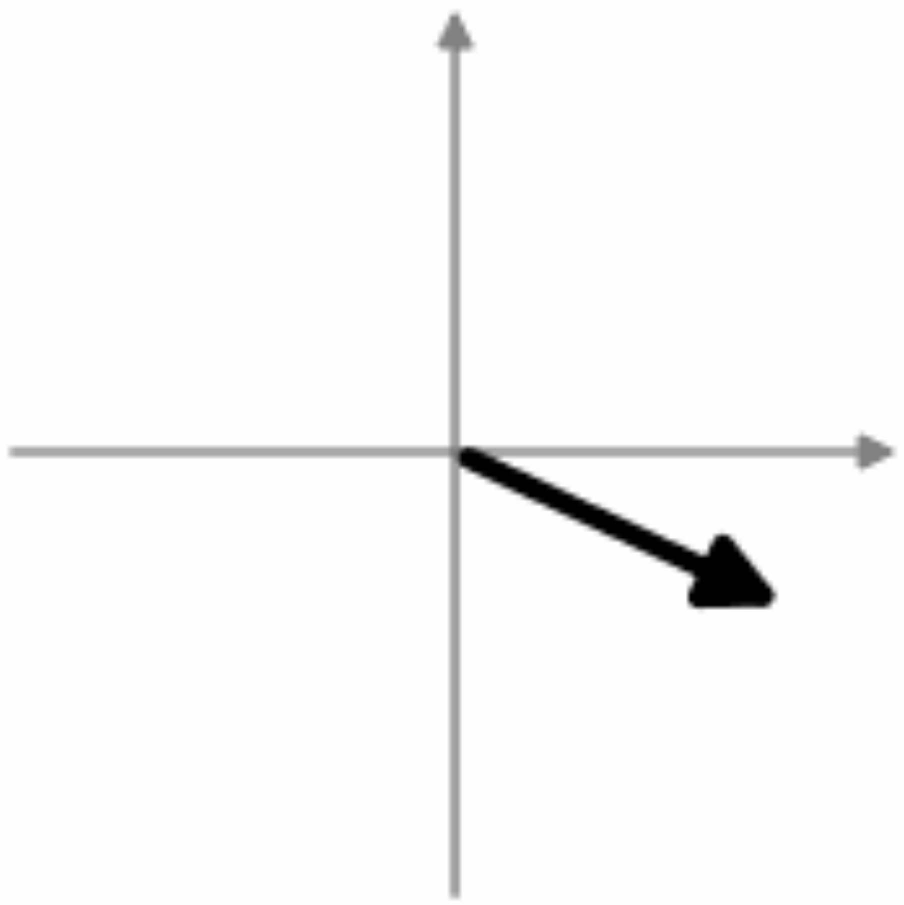
Invariance = Equivariance with scalar output

L-GATr

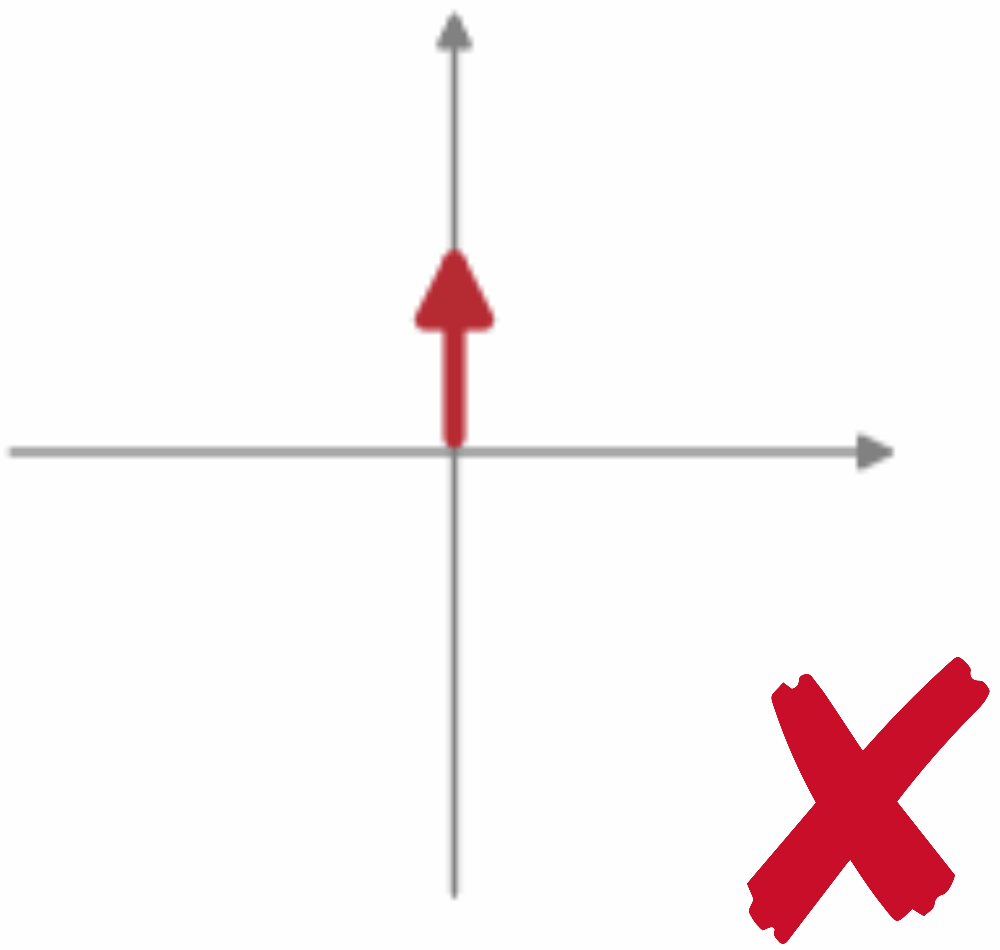


Equivariance = Covariance

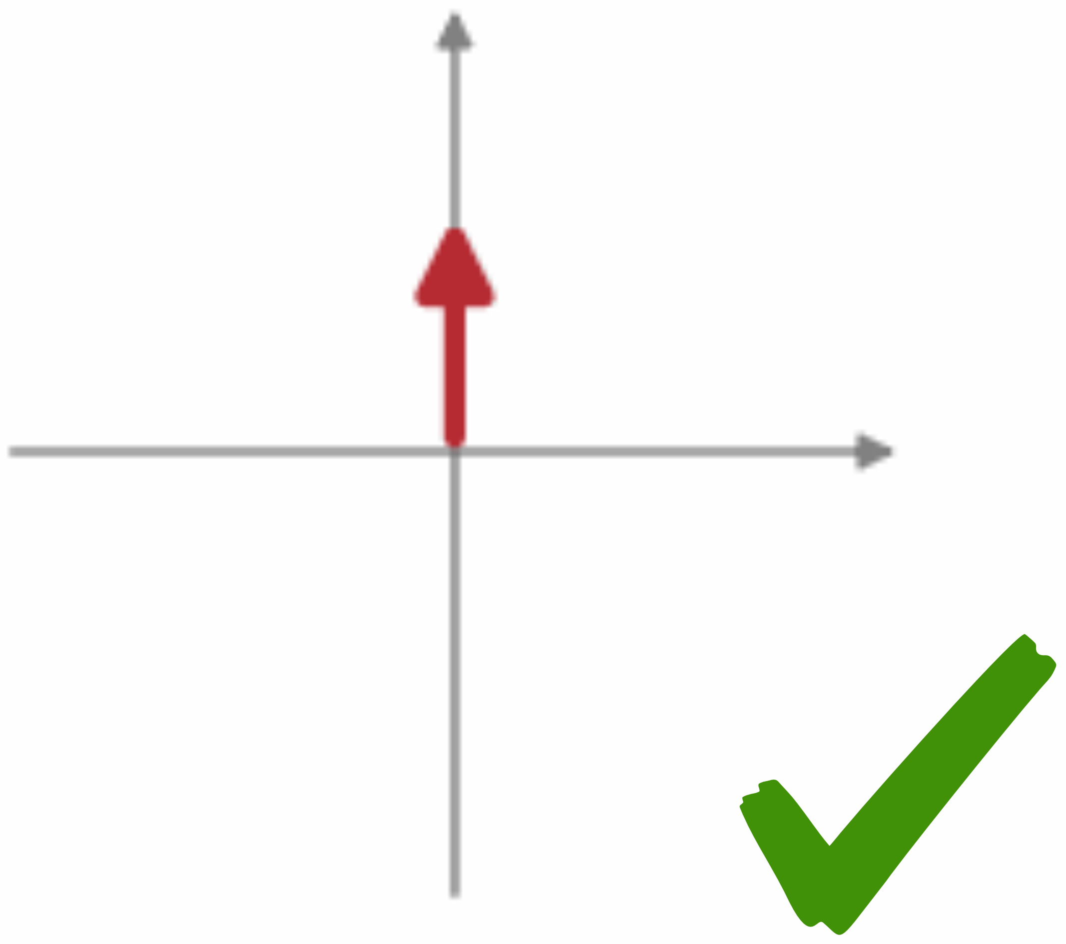
Input



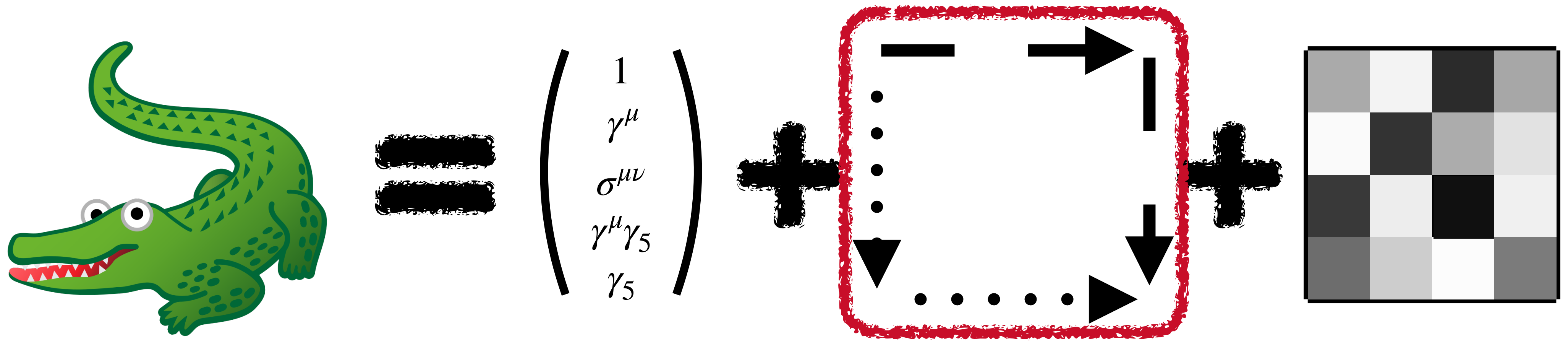
Output
(not equivariant)



Output
(equivariant)



Invariance = Equivariance with scalar output



L-GATr

Equivariance - formalised

Lorentz group transformation $\mathscr{G}$

neural network transformation $\mathscr{N}$

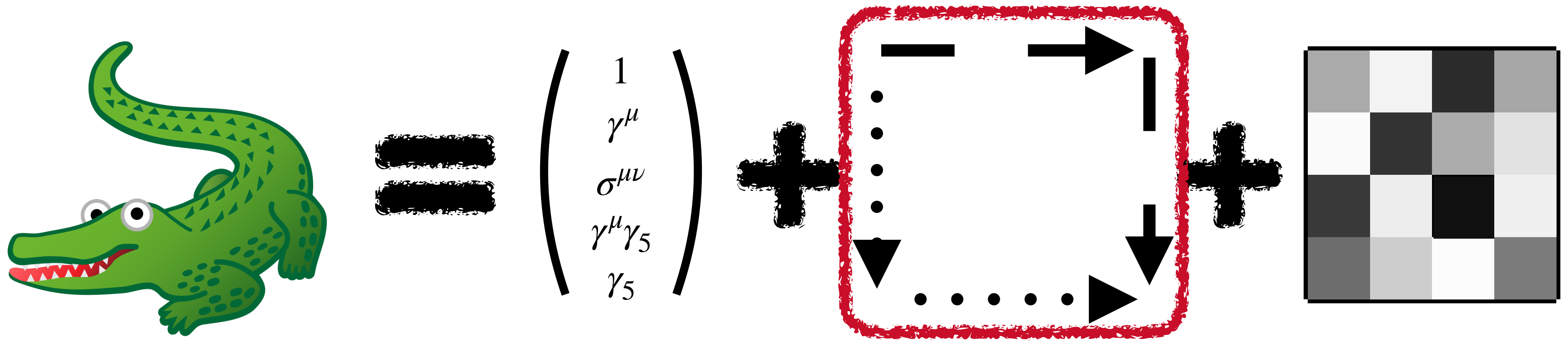
$$\mathscr{G}(\mathscr{N}(x)) = \mathscr{N}(\mathscr{G}(x))$$

$\mathscr{N}$

$\mathscr{G}$

L-GATr

Equivariance - formalised



Lorentz group transformation $\mathscr{G}$

neural network transformation $\mathscr{N}$

$$\mathscr{G}(\mathscr{N}(x)) = \mathscr{N}(\mathscr{G}(x))$$

$\mathscr{N}$

$\mathscr{G}$

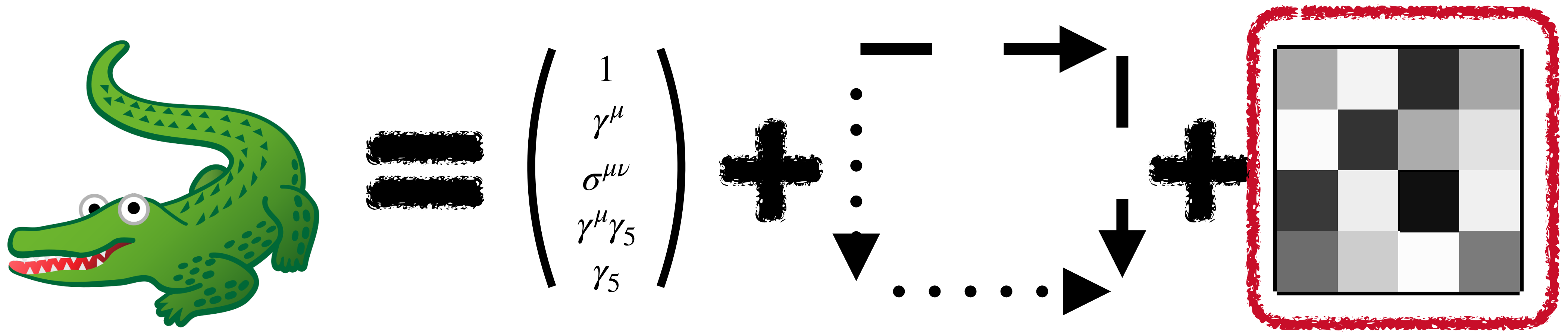
Each L-GATr operation $\mathscr{N}$ transforms all components within a given grade equally

$$\begin{pmatrix} x^S \\ x^V_\mu \\ x^B_{\mu\nu} \\ x^A_\mu \\ x^P \end{pmatrix} \xrightarrow{\mathscr{N}} \begin{pmatrix} w^S x^S \\ w^V x^V_\mu \\ w^B x^B_{\mu\nu} \\ w^A x^A_\mu \\ w^P x^P \end{pmatrix}$$

This turns L-GATr equivariant, because grades form sub-representations of $\mathscr{G}$

L-GATr

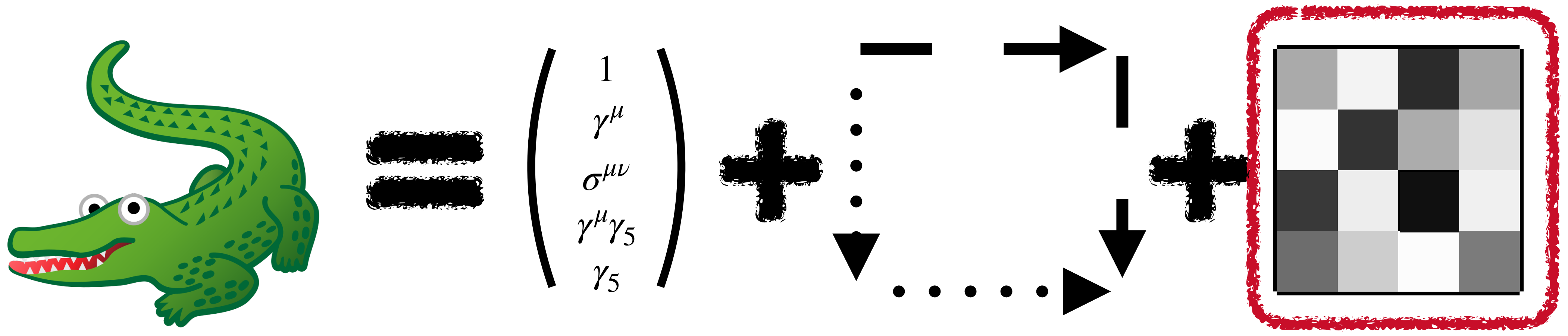
GATr-ing all transformer layers



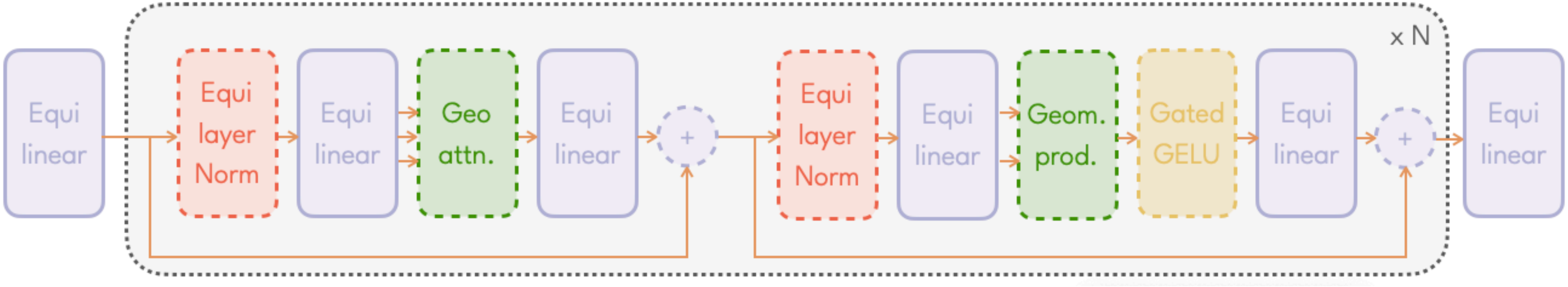
Layer type	Transformer	L-GATr
Linear(x)	$vx + w$	$\sum_{k=0}^{4} v_k \langle x \rangle_k \left(+ \sum_{k=0}^{4} w_k \gamma^5 \langle x \rangle_k \right)$
Attention$(q,k,v)_{ic}$	$\sum_{j=1}^{n_t} \text{Softmax}_j \left(\sum_{c'=1}^{n_c} \frac{q_{ic'} k_{jc'}}{\sqrt{n_c}} \right) v_{jc}$	$\sum_{j=1}^{n_t} \text{Softmax}_j \left(\sum_{c'=1}^{n_c} \frac{\langle q_{ic'}, k_{jc'} \rangle}{\sqrt{16 n_c}} \right) v_{jc}$
LayerNorm(x)	$x \left[\frac{1}{n_c} \sum_{c=1}^{n_c} x_c^2 + \epsilon \right]^{-1/2}$	$x \left[\frac{1}{n_c} \sum_{c=1}^{n_c} \sum_{k=0}^{4} \left\lvert \left\langle \langle x_c \rangle_k, \langle x_c \rangle_k \right\rangle \right\rvert + \epsilon \right]^{-1/2}$
Activation(x)	GELU(x)	GELU$(\langle x \rangle_0) x$
GP(x,y)	–	xy

L-GATr

Full architecture



GATr blocks
can be stacked
to large depth

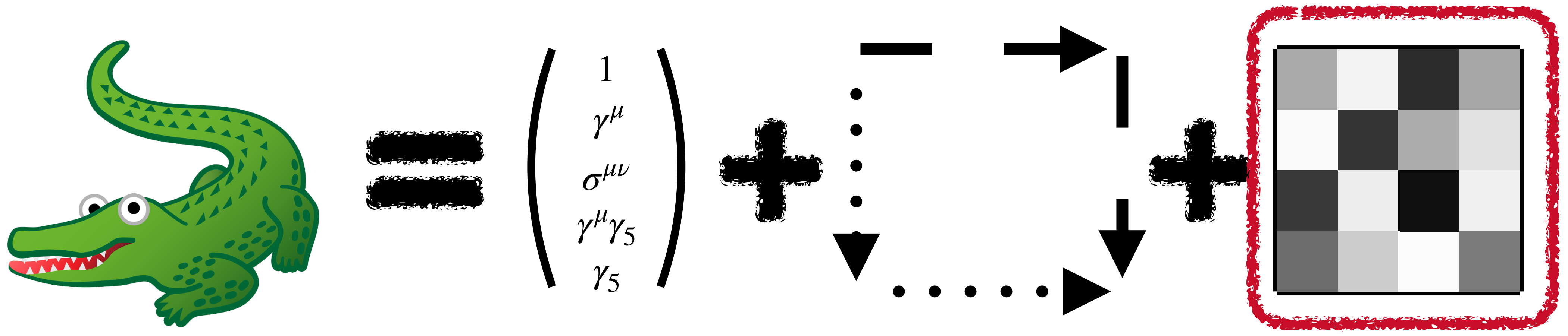


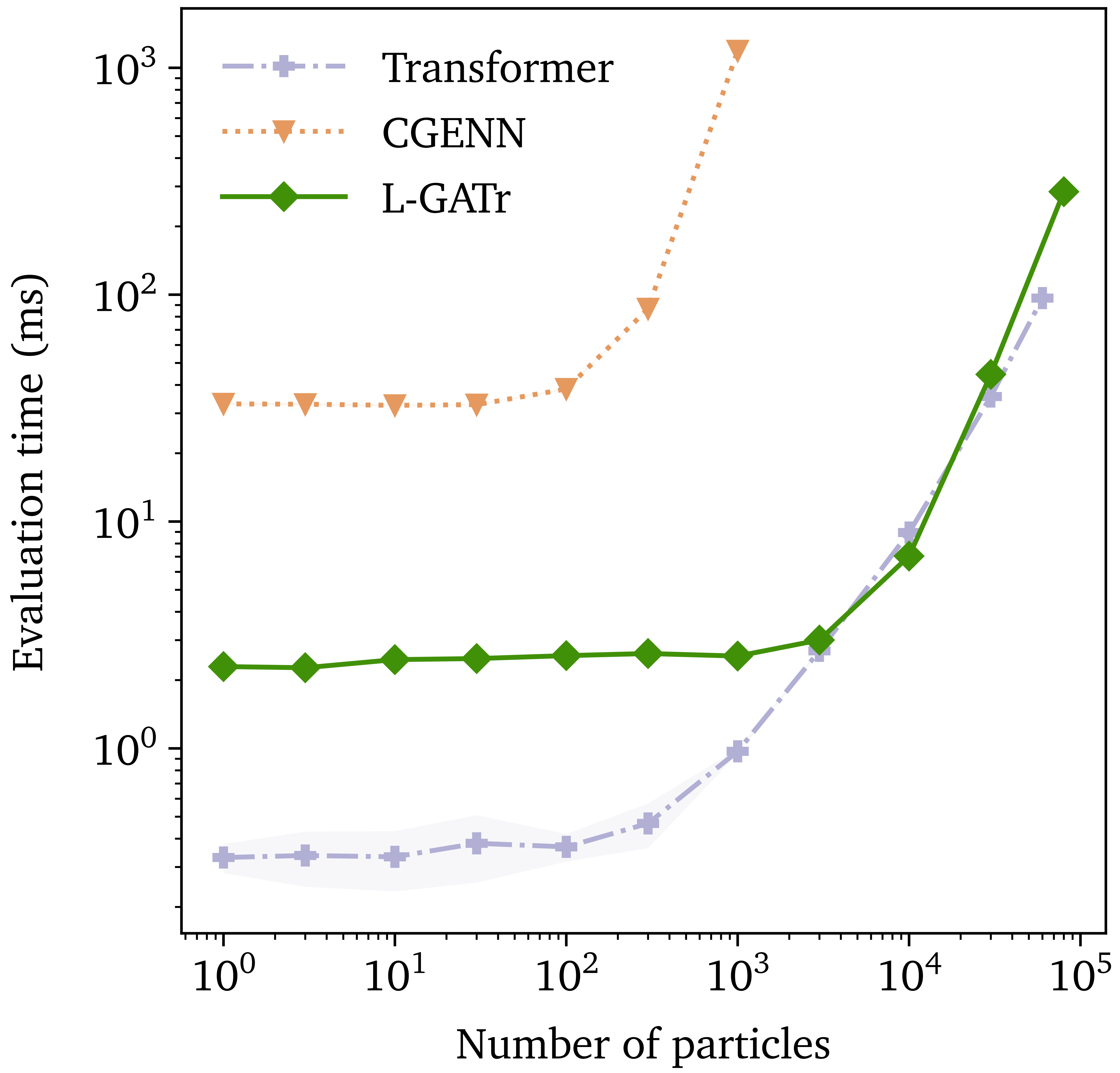
Equivariant **Self-Attention**

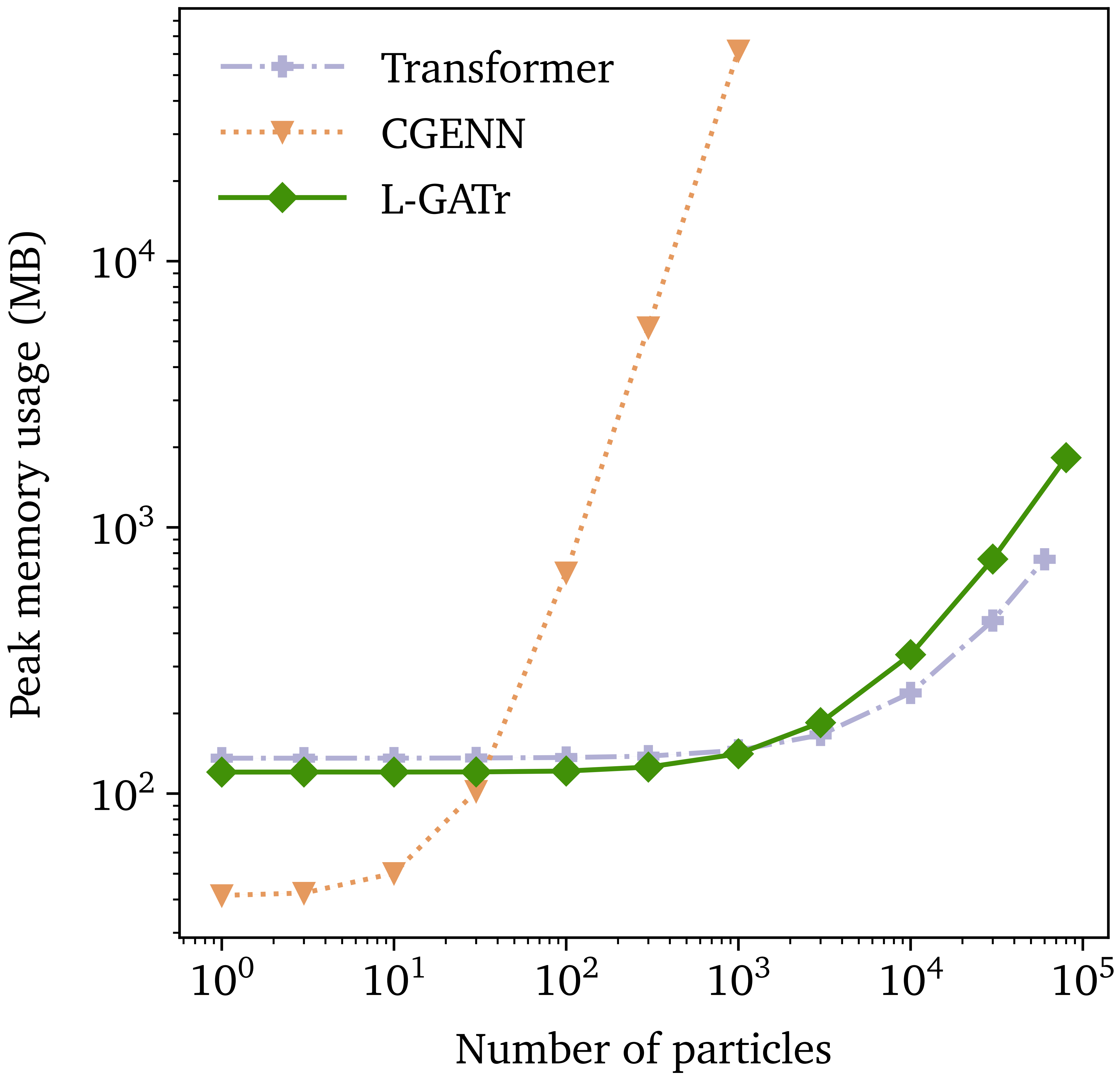
Equivariant **MLP**

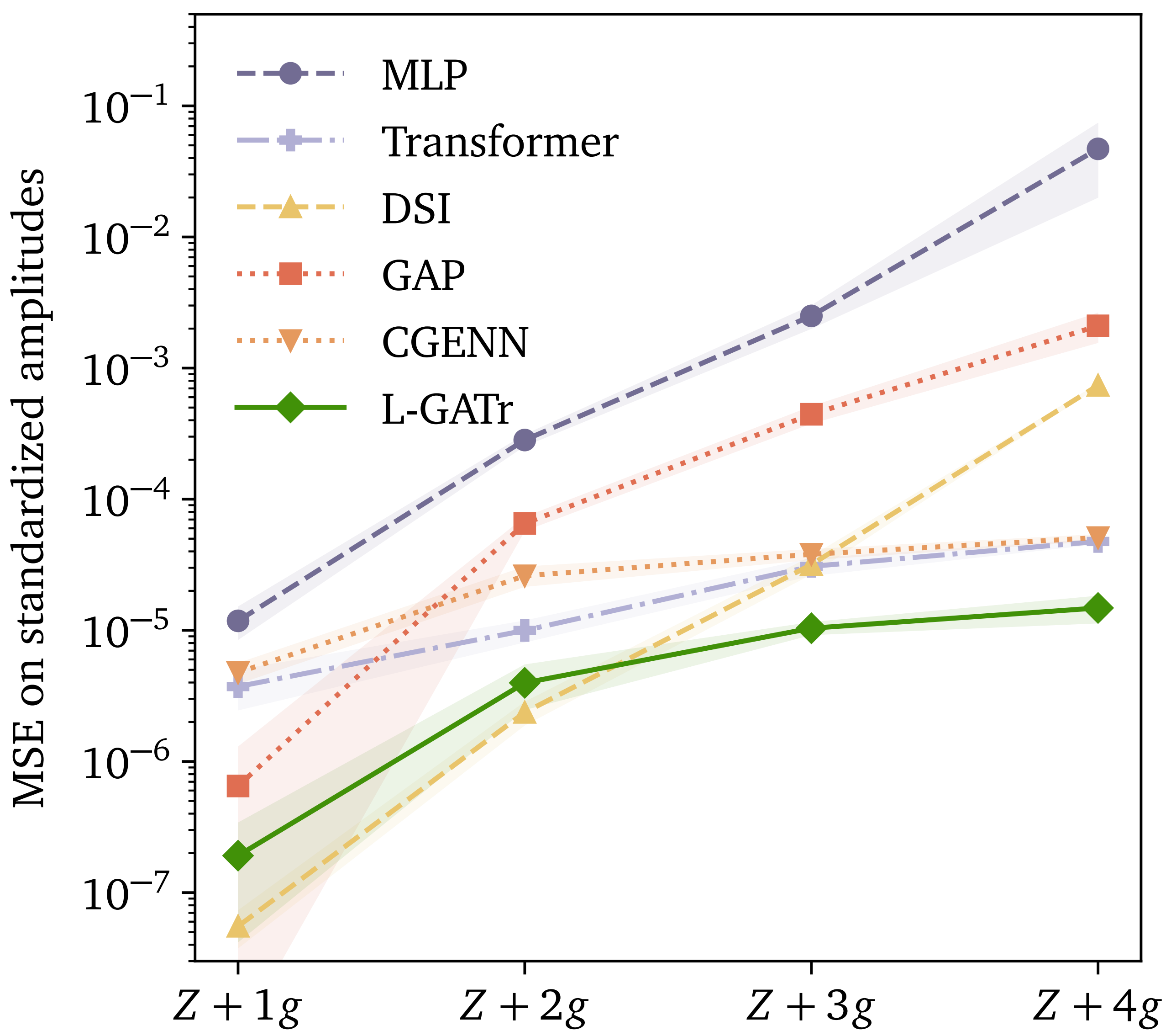
L-GATr

Transformers scale better than graphs

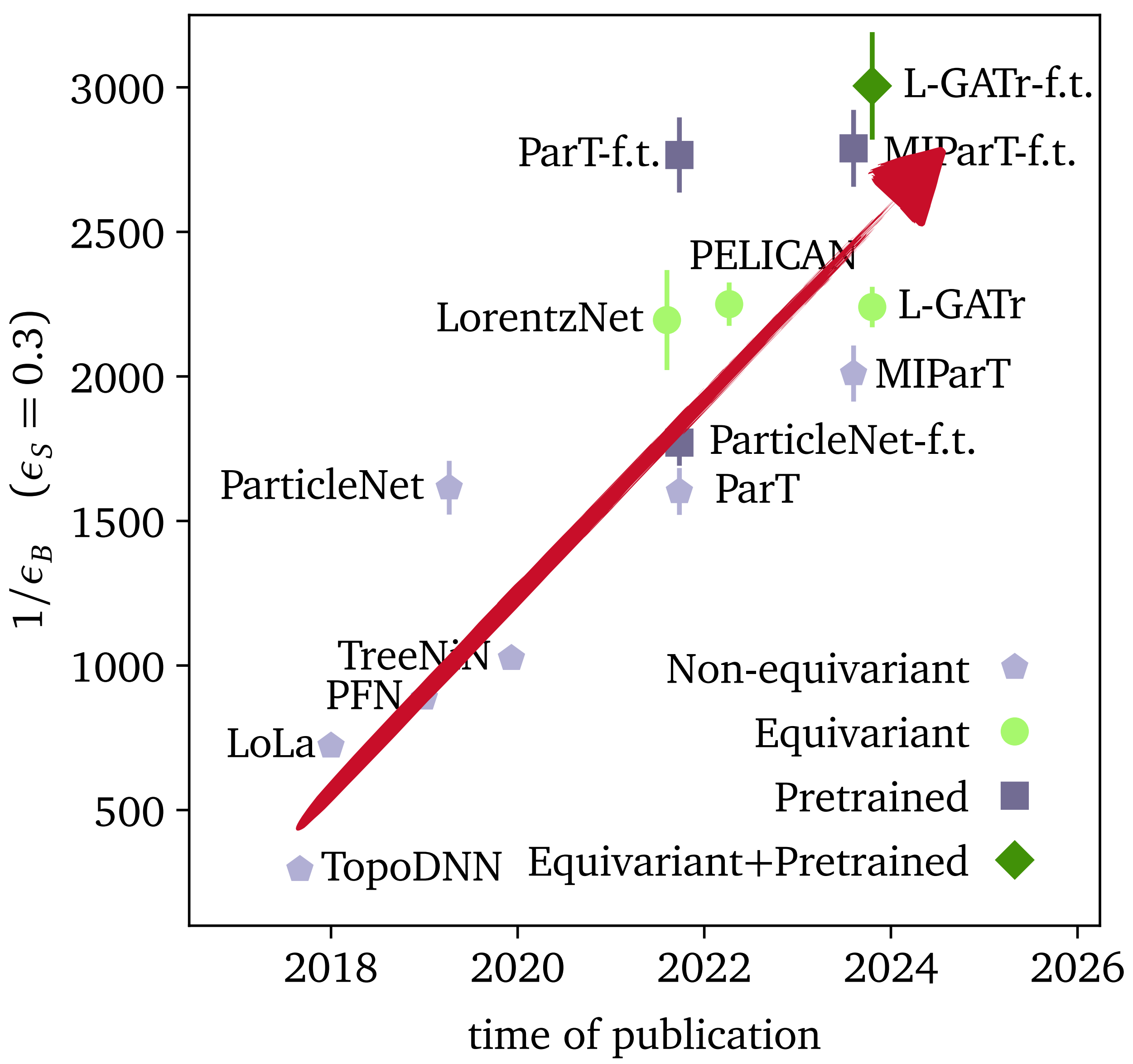




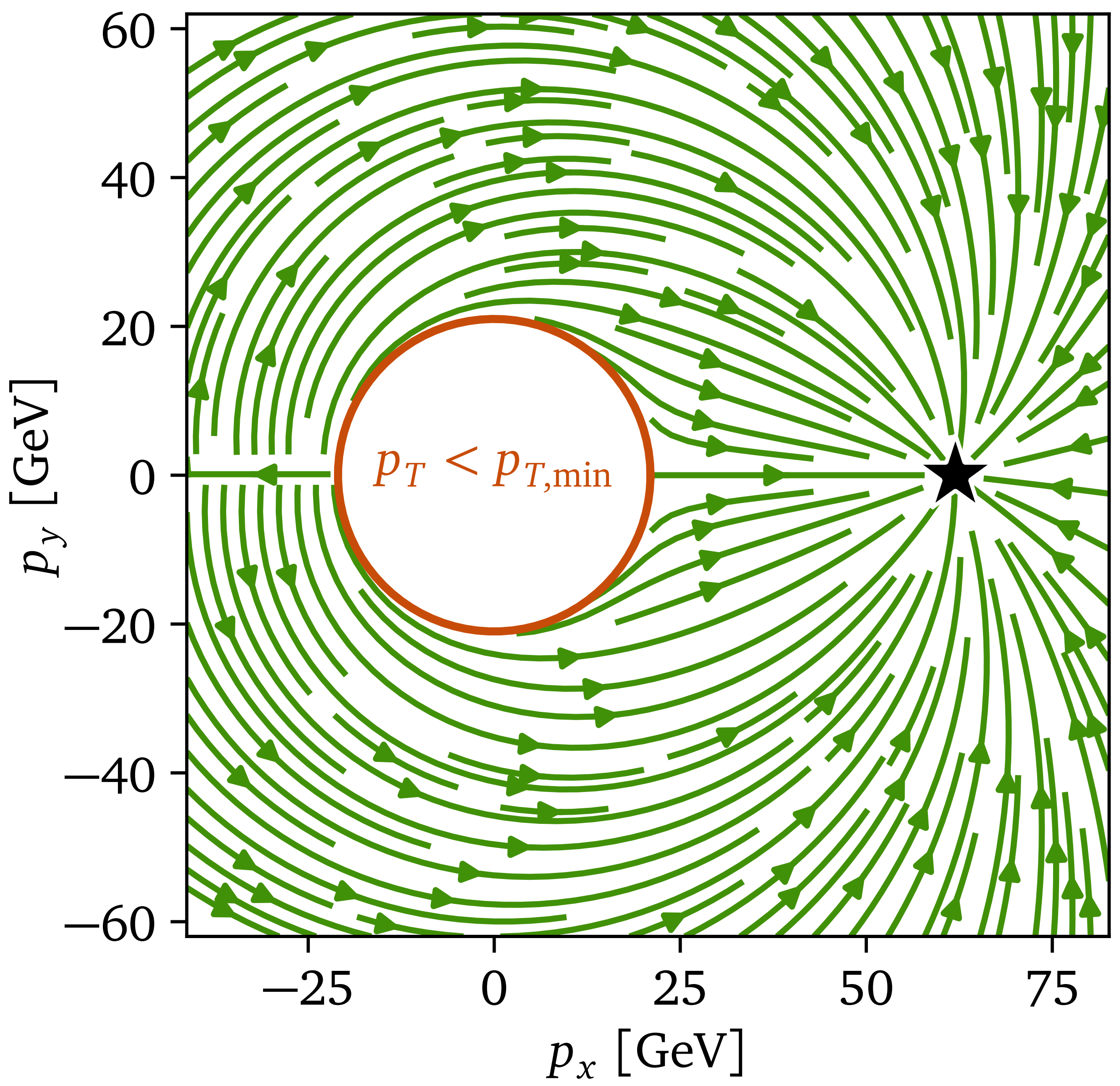




Regression
QFT amplitudes

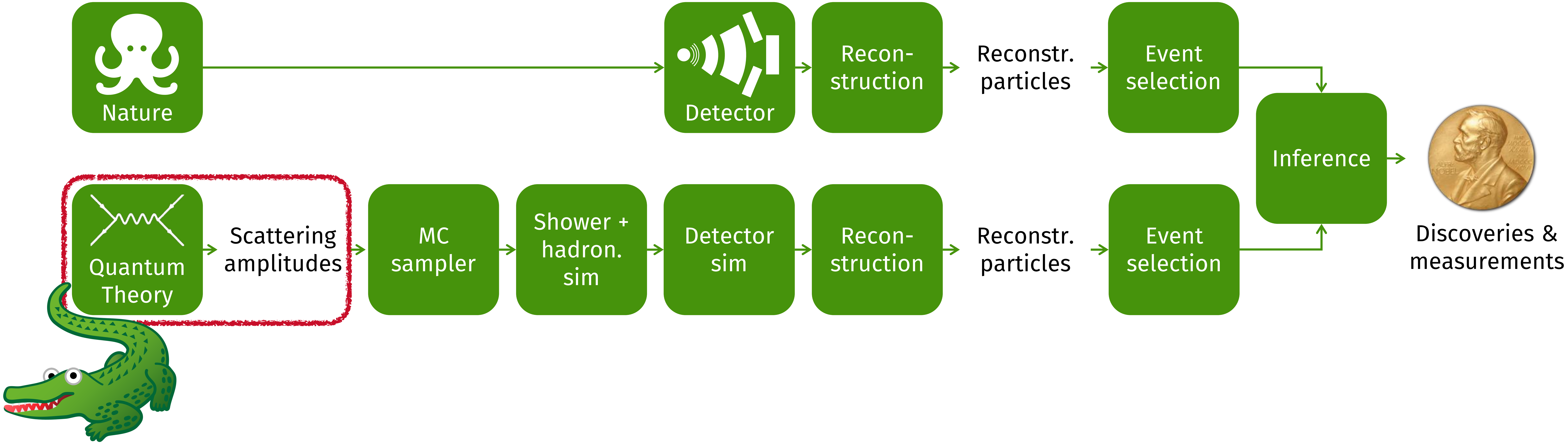


Classification
Jet tagging



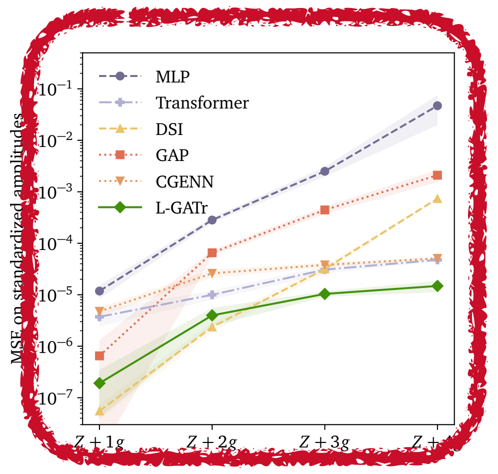
Generation
Reconstructed events

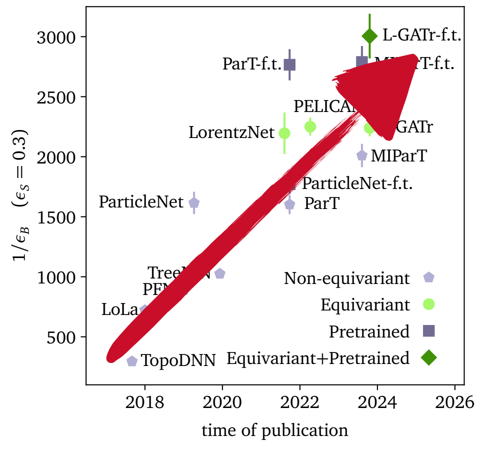
QFT amplitude regression

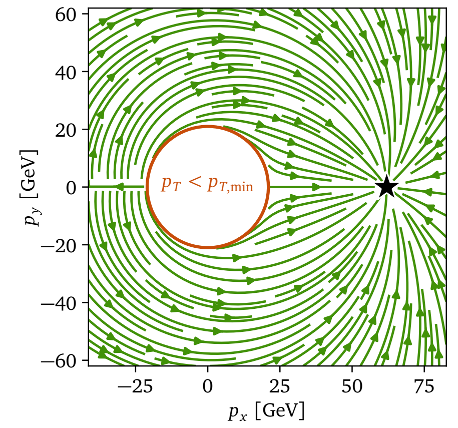


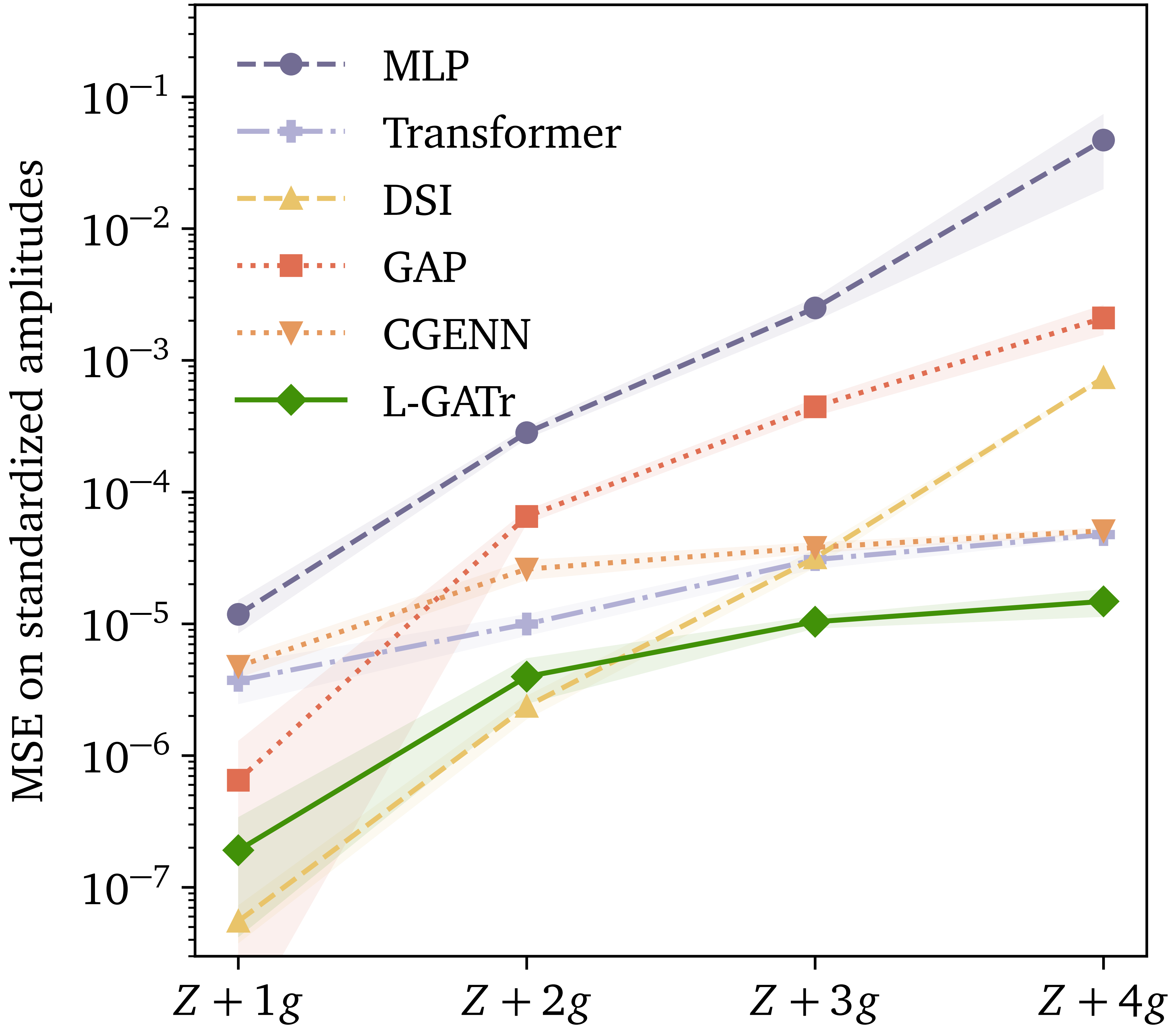
QFT amplitude regression

Permutation + Lorentz equivariance





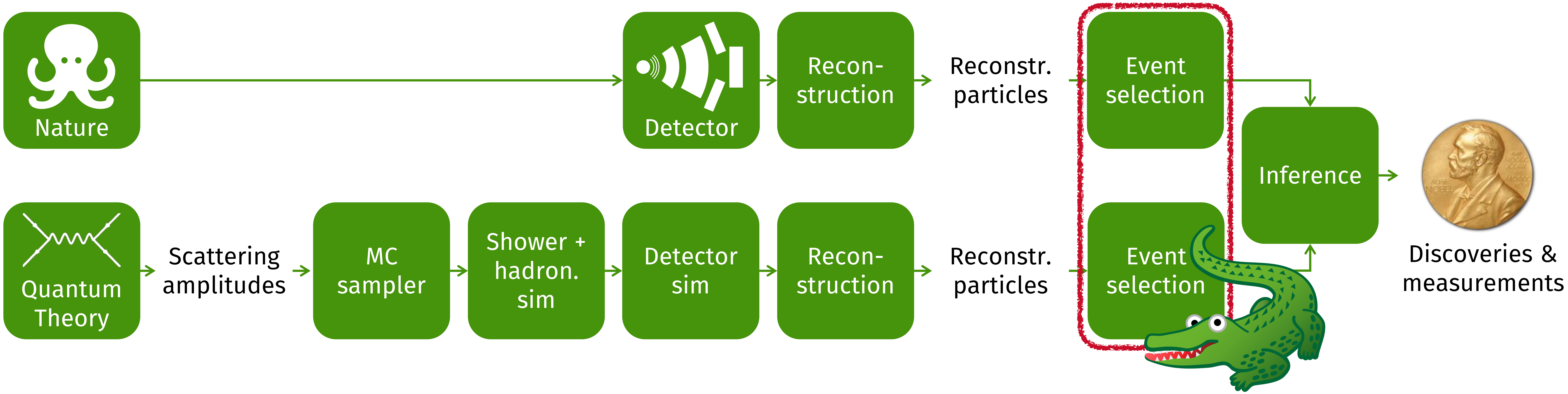




Permutation equivariance improves the scaling to high multiplicity

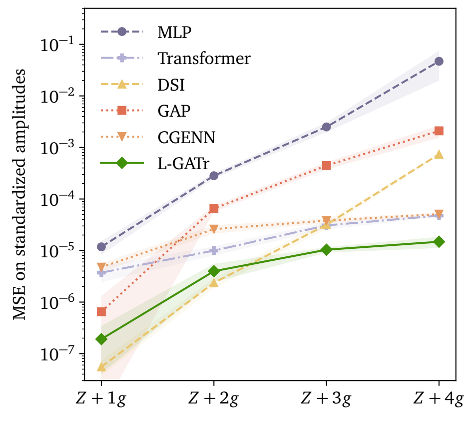
Lorentz equivariance gives a constant improvement across multiplicities

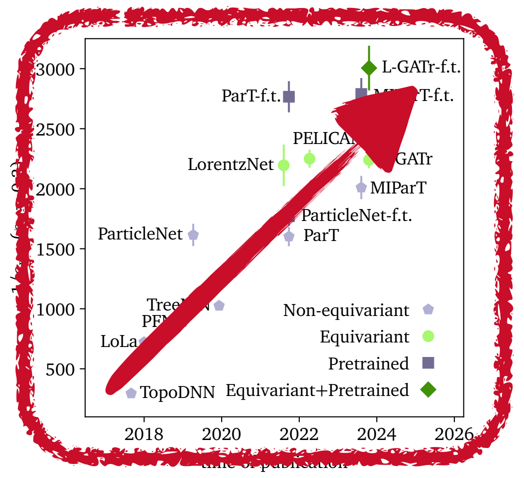
Jet Tagging

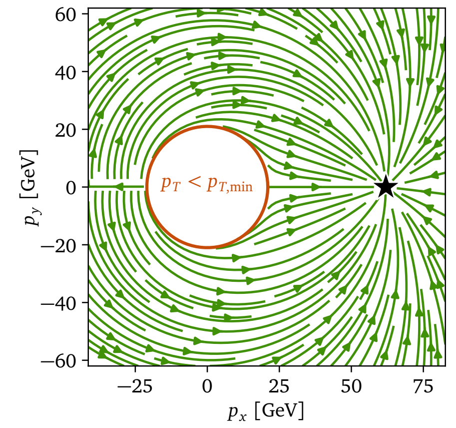

Nature
Detector
Recon-
struction
Reconstr.
particles
Event
selection
Inference
Quantum
Theory
Scattering
amplitudes
MC
sampler
Shower +
hadron.
sim
Detector
sim
Recon-
struction
Reconstr.
particles
Event
selection
Discoveries &
measurements

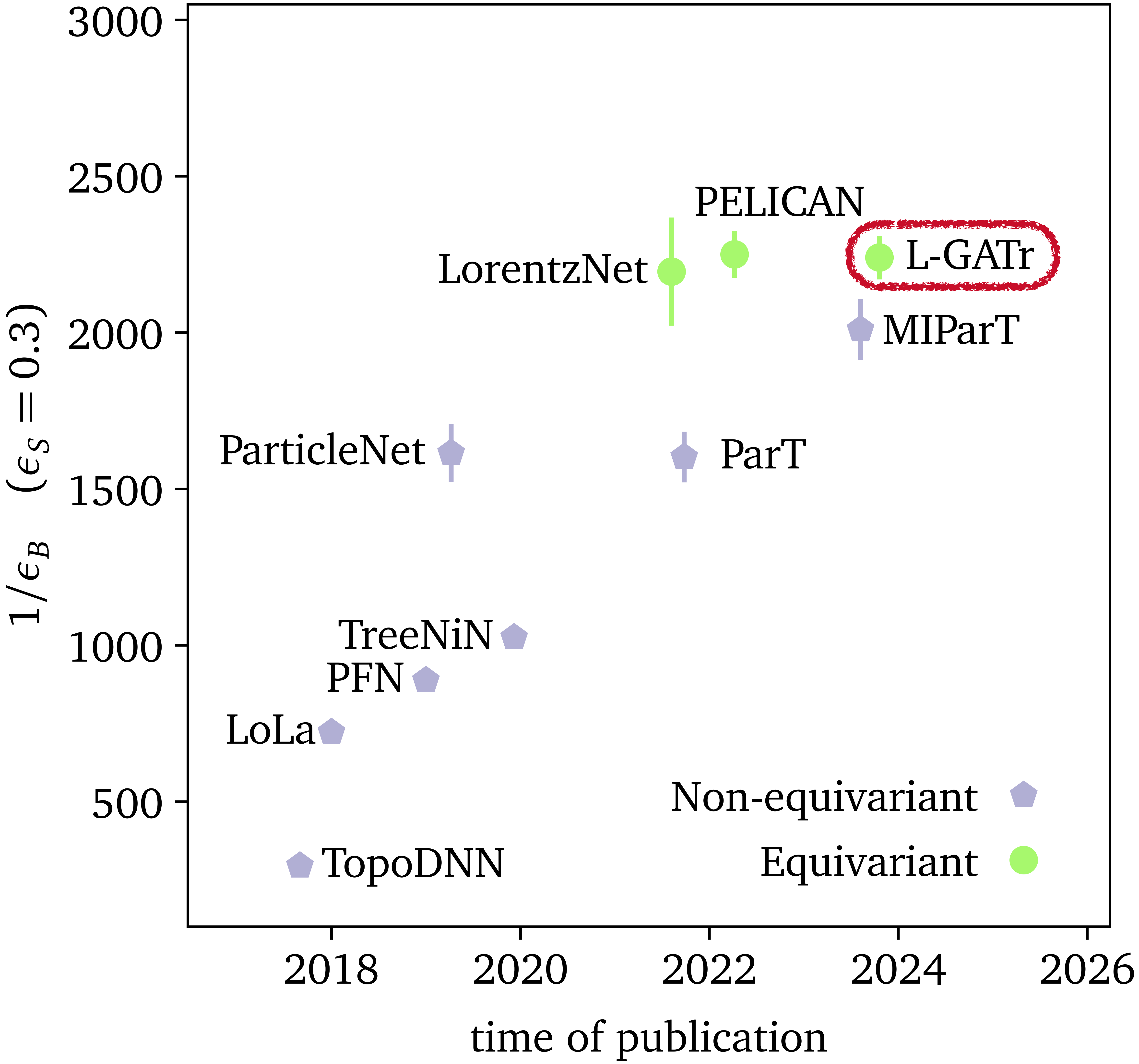
Jet Tagging

The history of top tagging

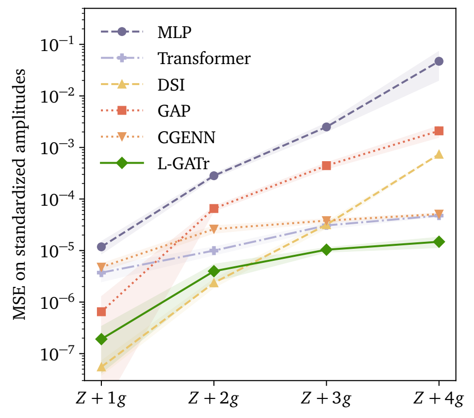


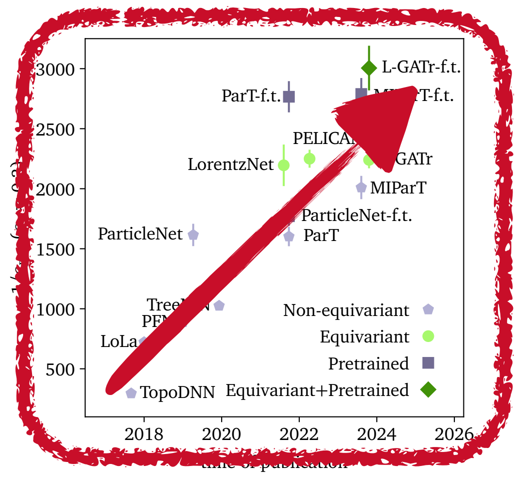


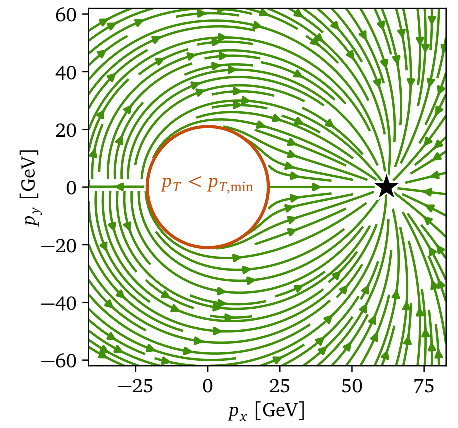




All top-performing taggers
are Lorentz-equivariant

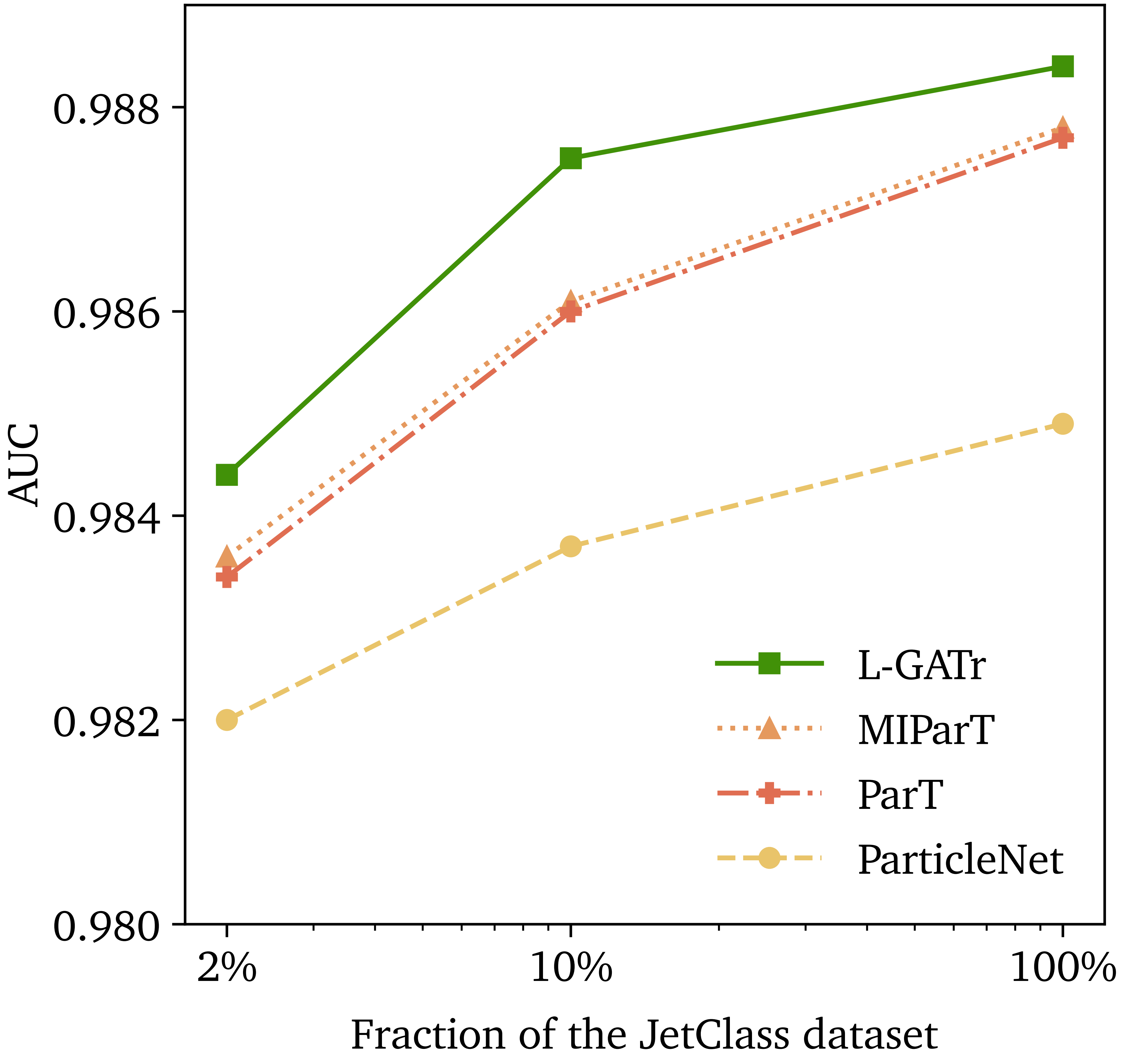






Jet Tagging

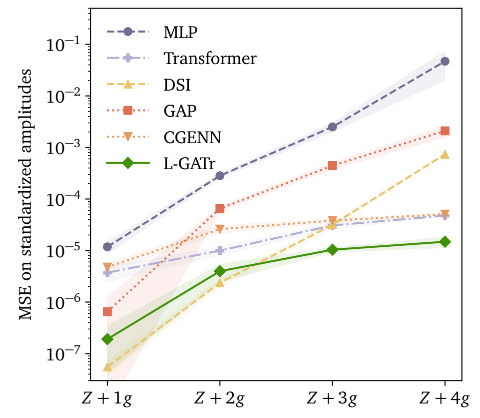
Training on 100M jets

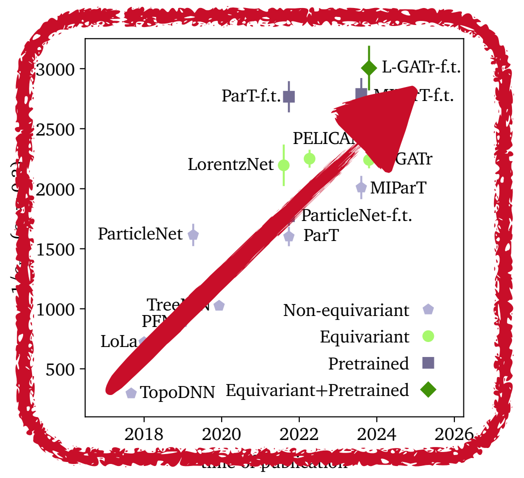


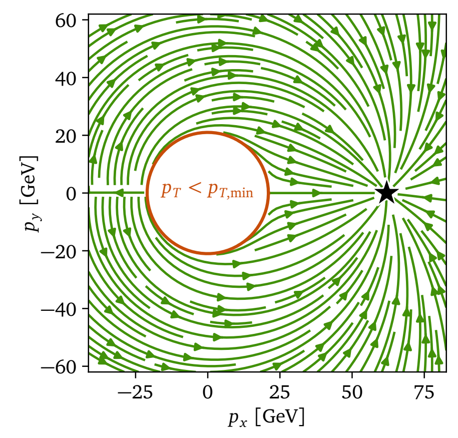
Transformers outperform
graph networks
on large datasets

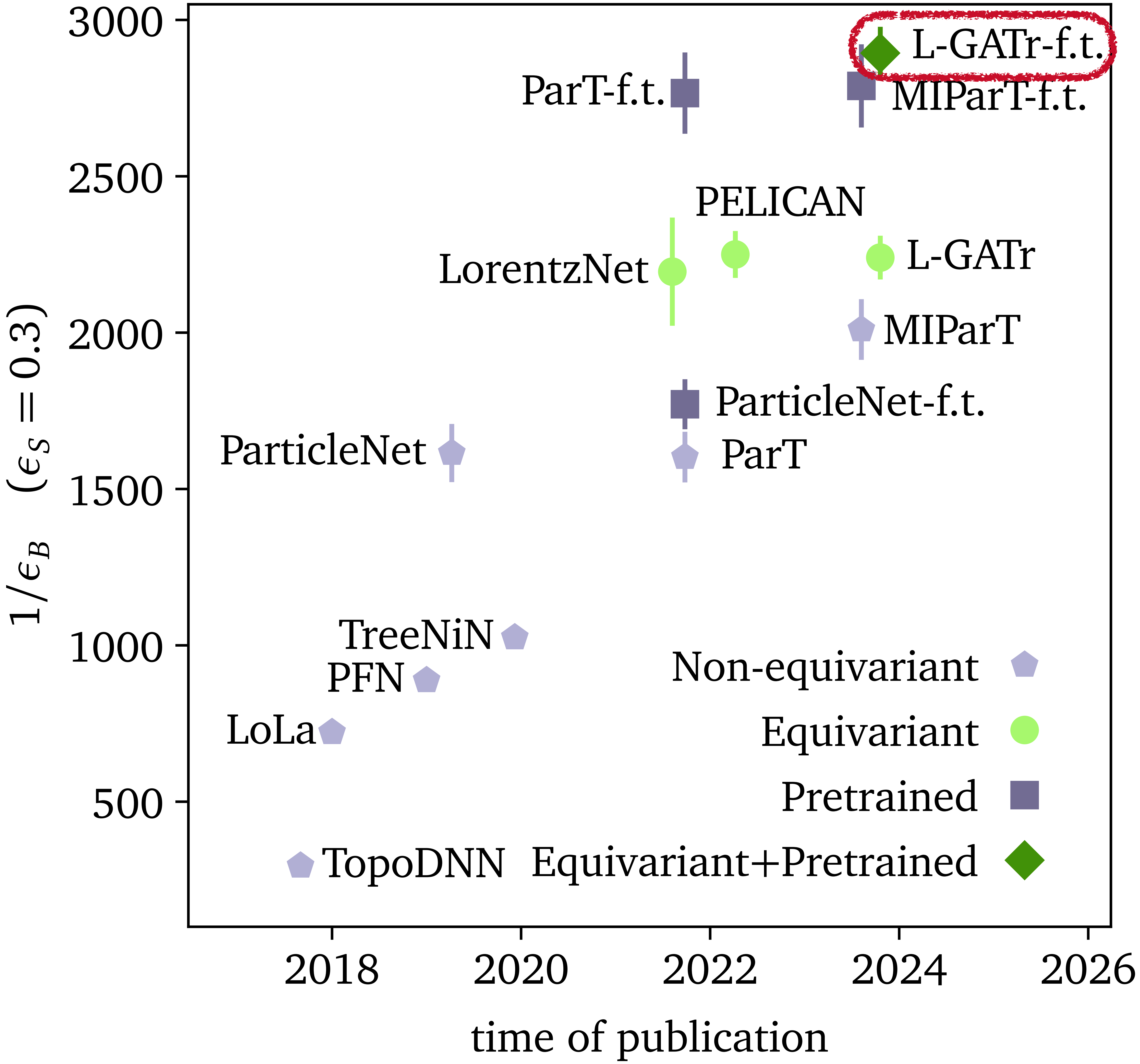
Jet Tagging

Transfer learning for top tagging



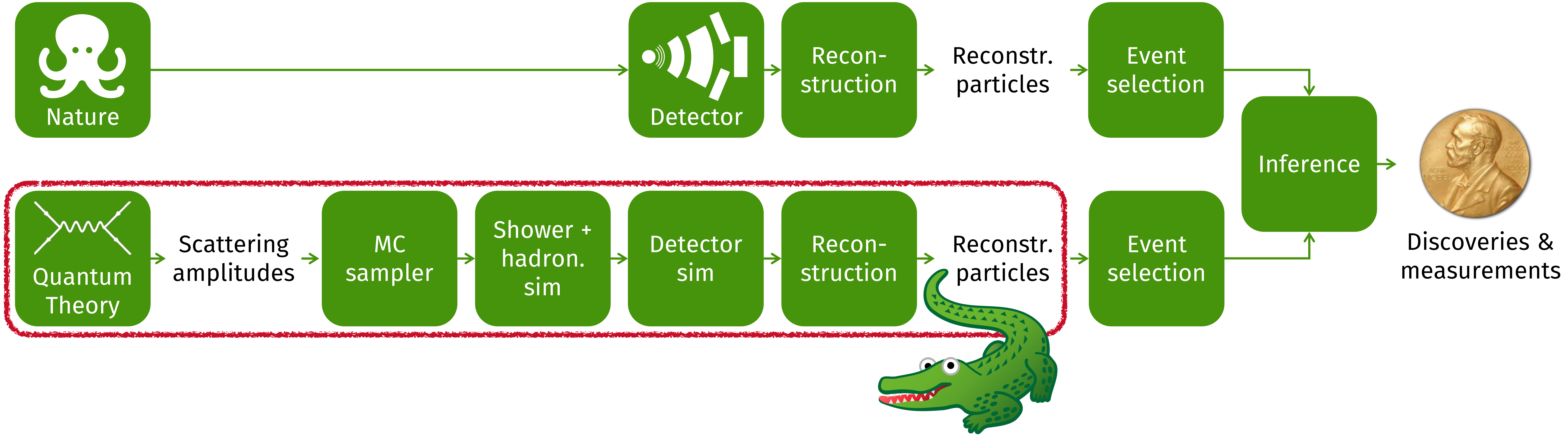






L-GATr combines the benefits of pretraining and Lorentz equivariance

Event generation

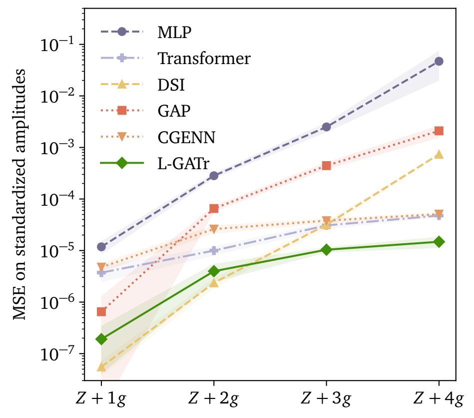


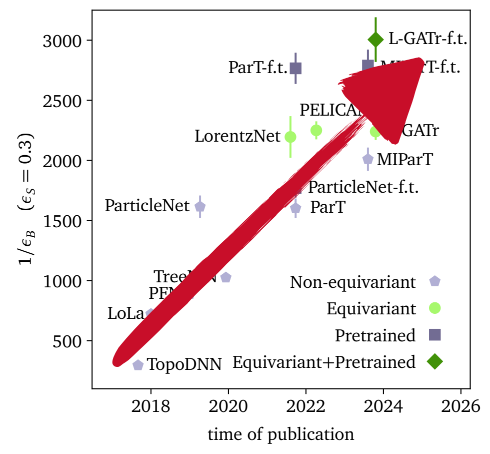
Event generation

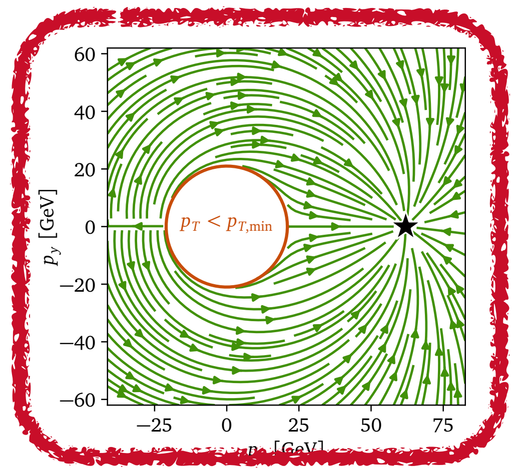
Task

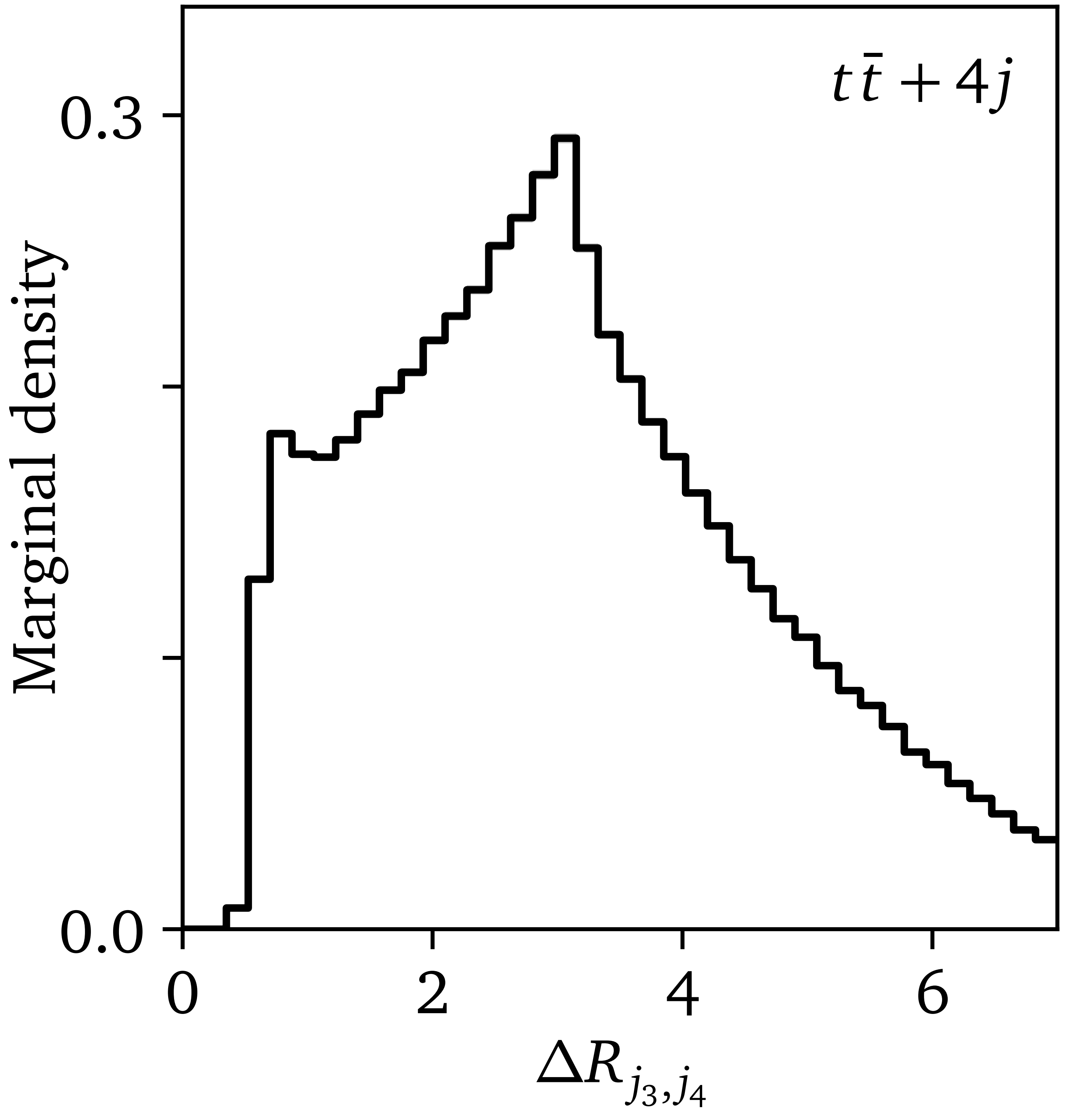
Dataset: $pp \to t_h \bar{t}_h + n\,j,\;\; n = 0\ldots4$

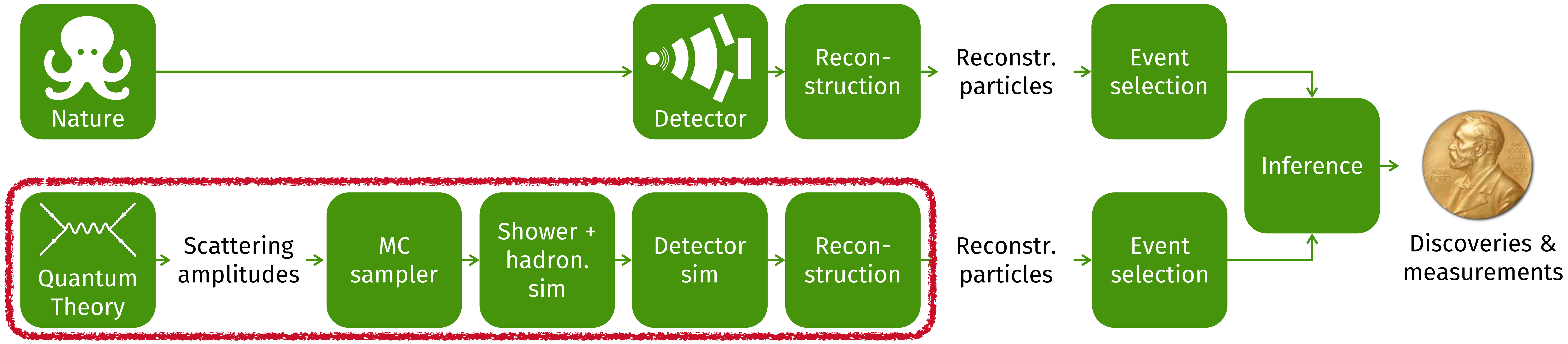
- MadGraph + Pythia + Delphes + Reconstruction
- Challenging features: $m_t, m_W, \Delta R_{jj} > 0.5$





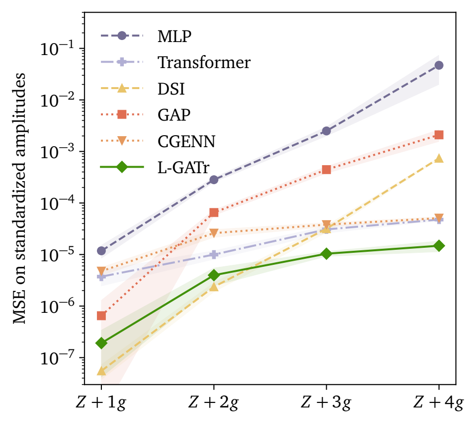


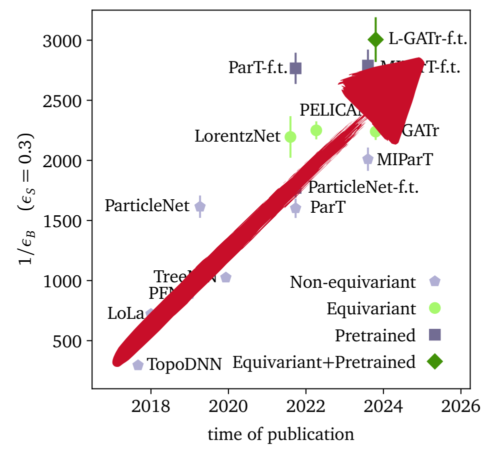


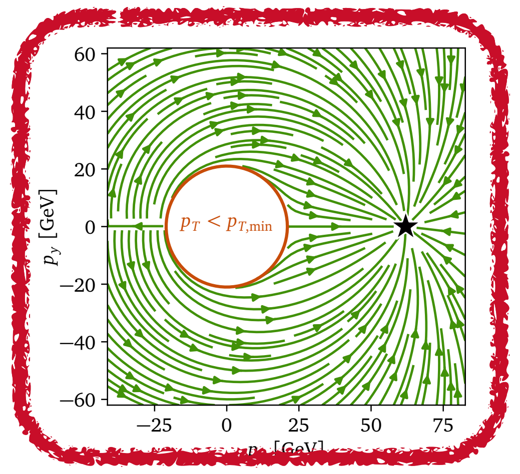


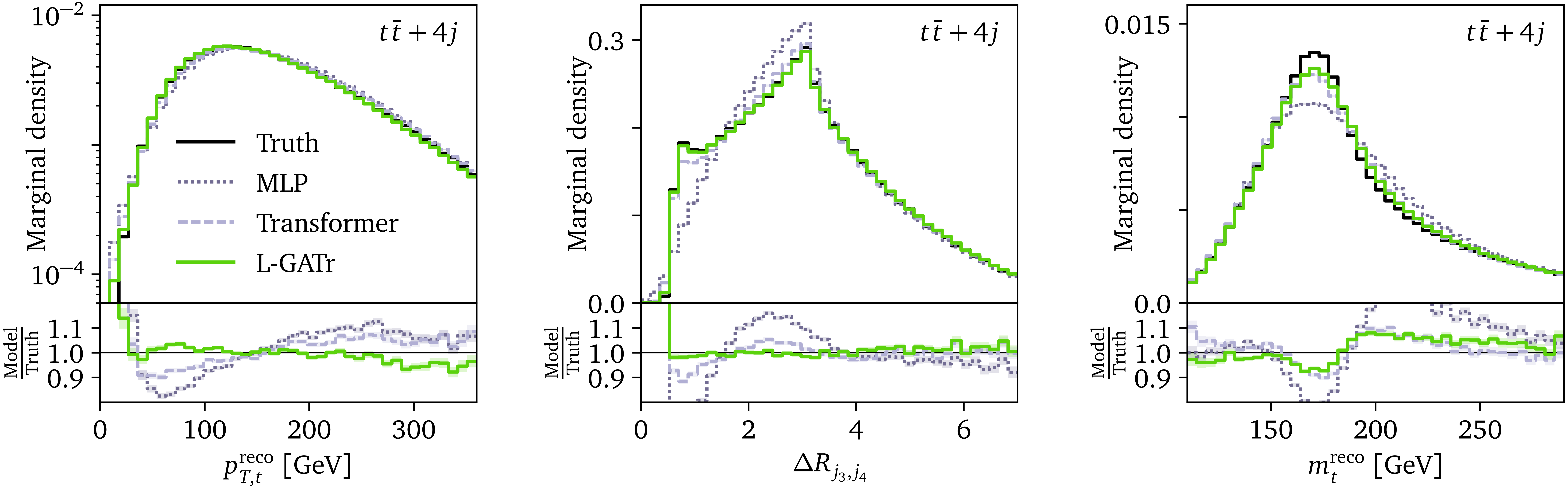
Event generation

Kinematic distributions





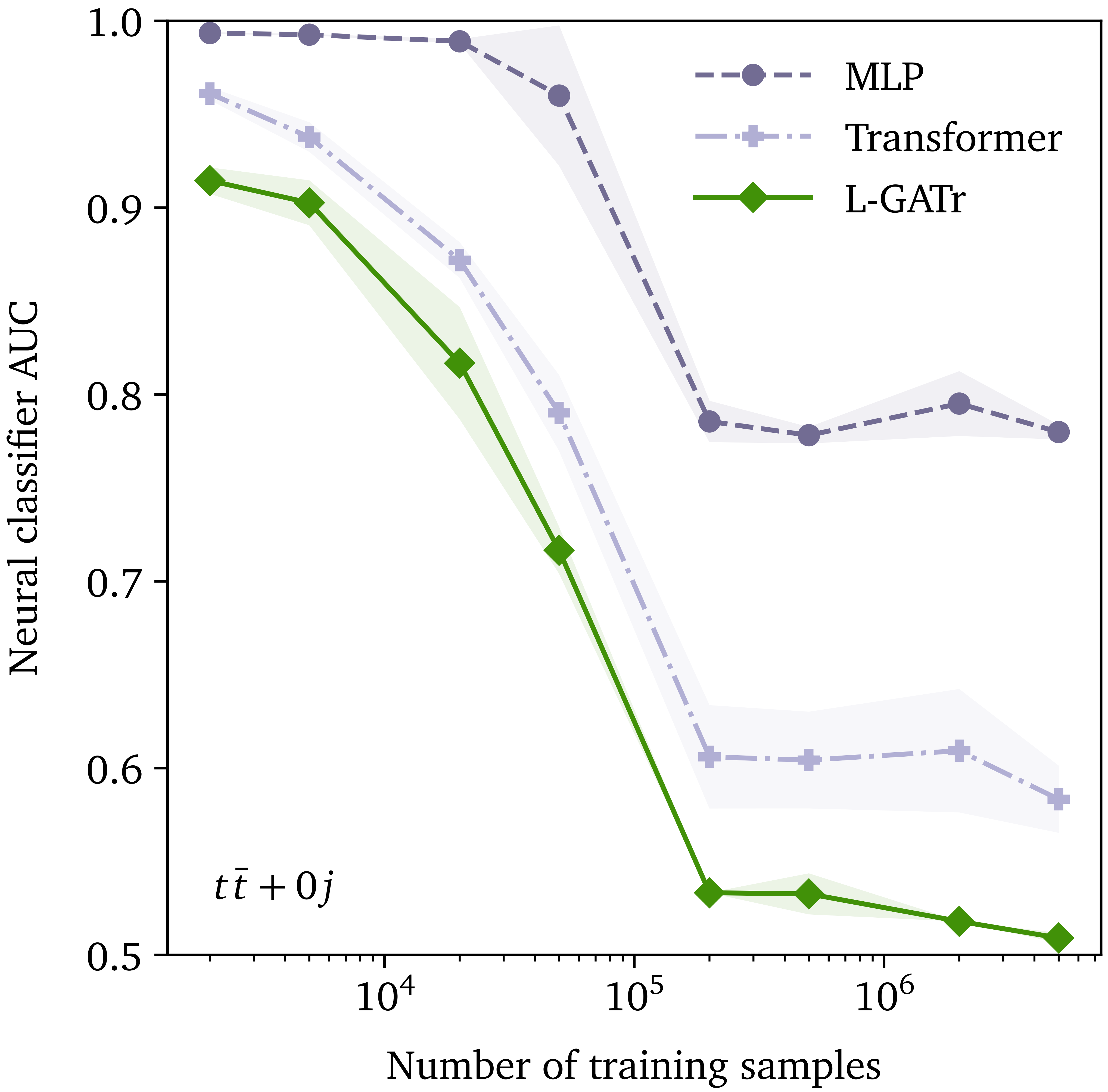


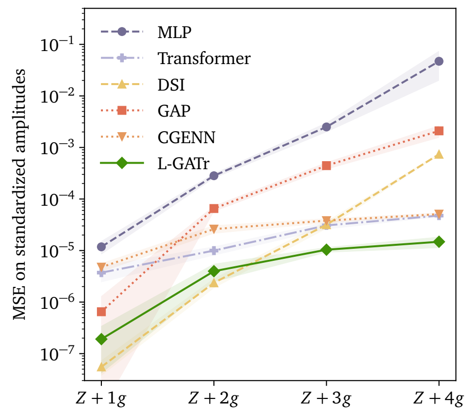


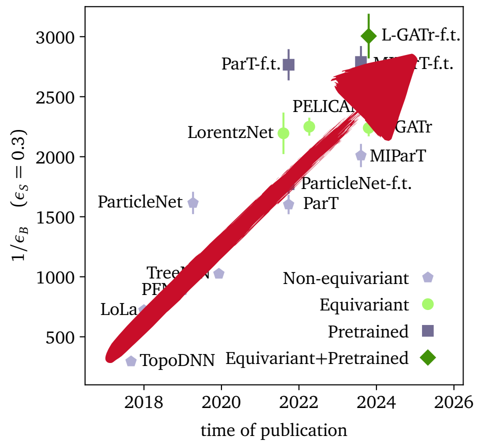
Equivariance helps,
especially for angular correlations

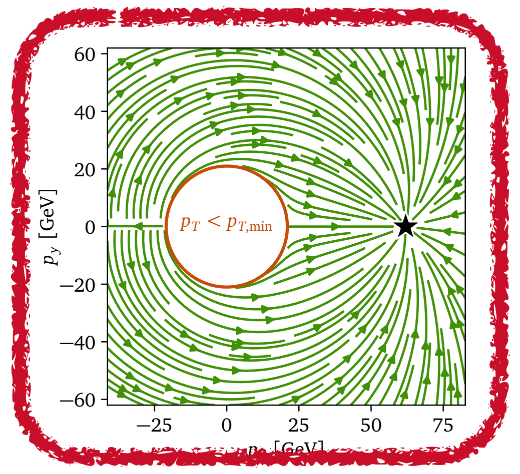
Event generation

Neural classifier metric





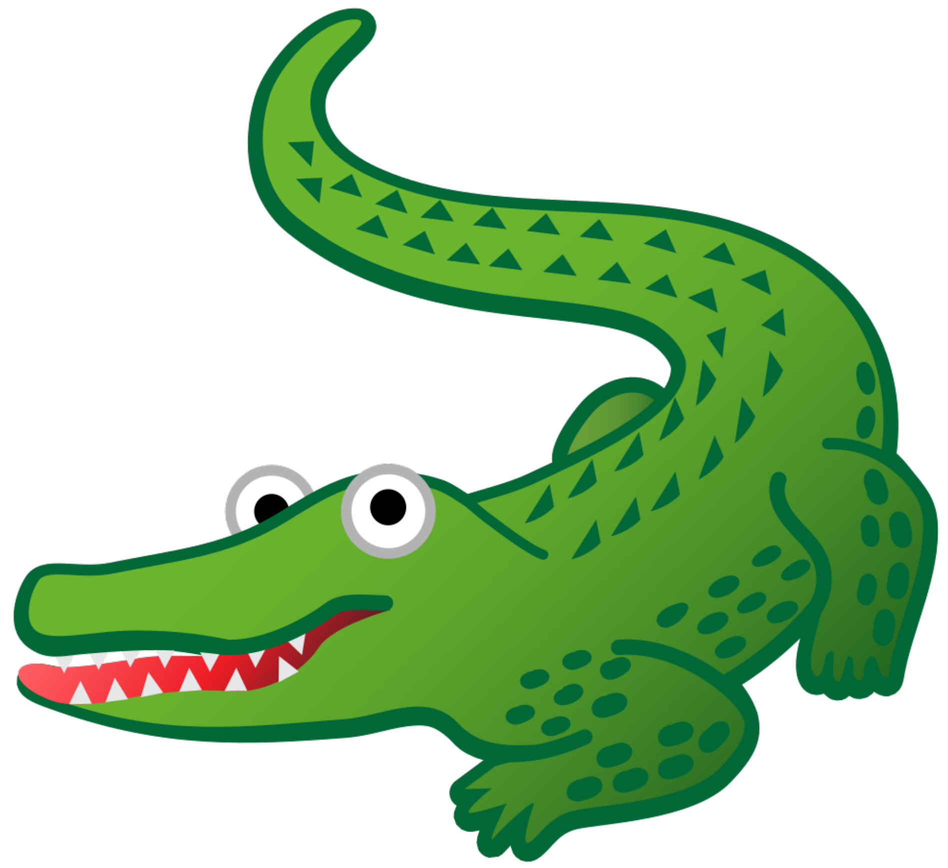




L-GATr generates samples that a classifier can barely distinguish from the ground truth

Messages

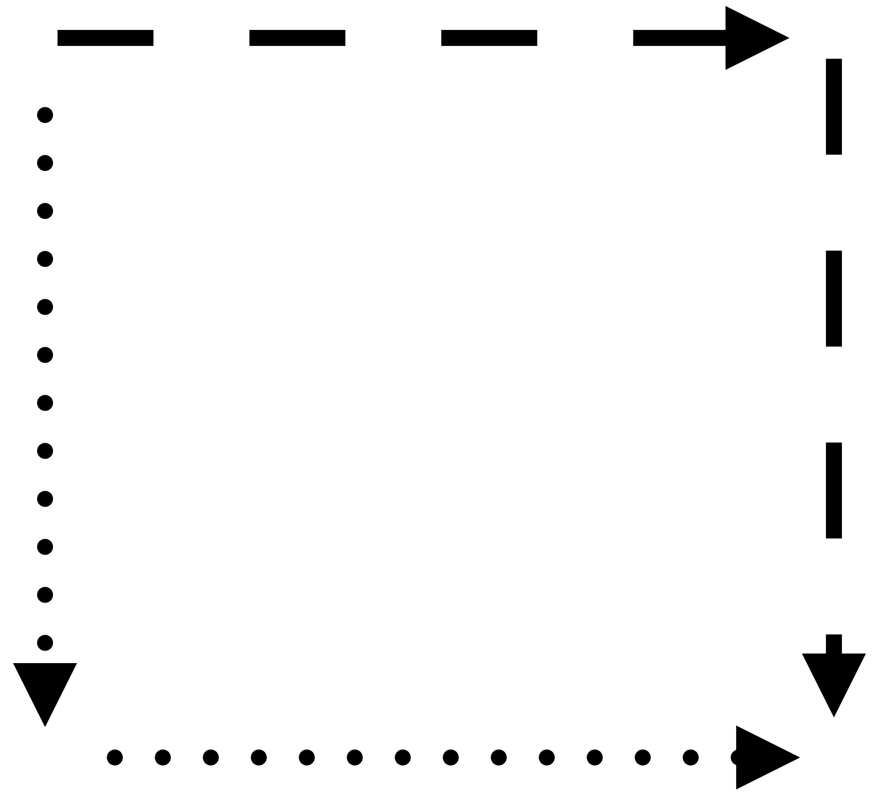
- L-GATr is a multi-purpose architecture for LHC physics
- In jet tagging, L-GATr combines the benefits from Lorentz equivariance and pretraining
- Lorentz-equivariant generators are tricky but great



Lorentz-Equivariant **G**eometric **A**lgebra **Tr**ansformer

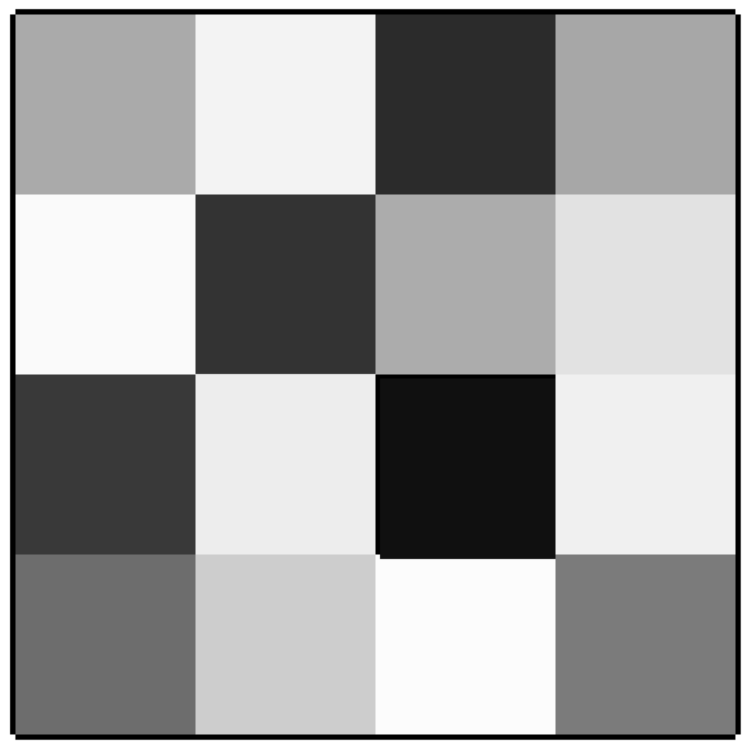
$$= \begin{pmatrix} 1 \\ \gamma^\mu \\ \sigma^{\mu\nu} \\ \gamma^\mu \gamma_5 \\ \gamma_5 \end{pmatrix} +$$

Geometric algebra representations



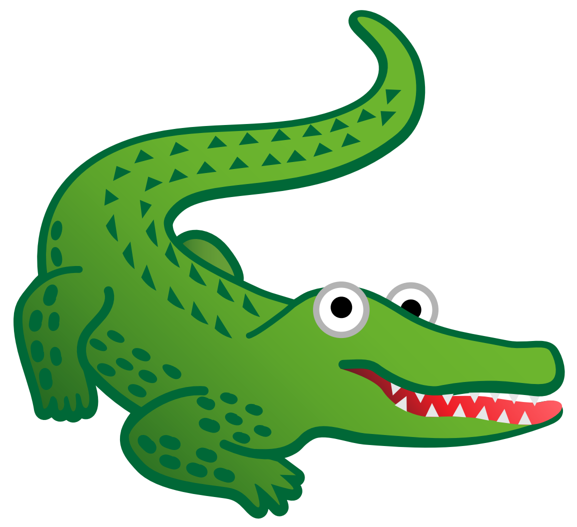
$+$

Lorentz-Equivariant layers



Transformer architecture

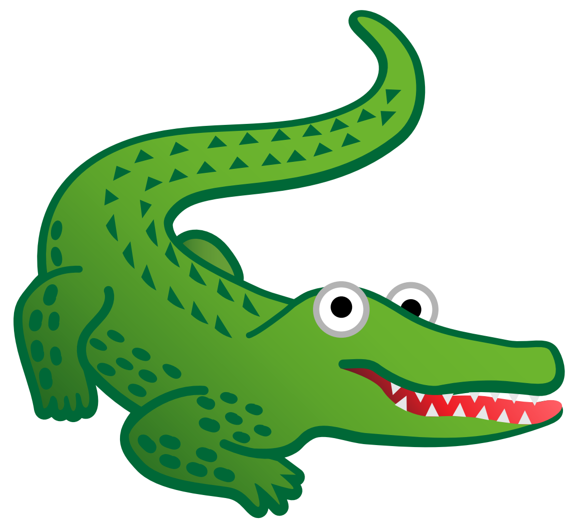
L-GATr is easy to use



```python
from gatr import GATr, SelfAttentionConfig, MLPConfig
from gatr.interface import embed_vector, extract_scalar, embed_spurions
import torch


class ExampleWrapper(torch.nn.Module):
    """Example wrapper around a L-GATr model.

    Parameters
    ----------
    num_blocks : int
        Number of transformer blocks
    hidden_mv_channels : int
        Number of hidden multivector channels
    hidden_s_channels : int
        Number of hidden scalar channels
    """

    def __init__(self, blocks=6, hidden_mv_channels=16, hidden_s_channels=32):
        super().__init__()
        self.gatr = GATr(
            in_mv_channels=3,
            out_mv_channels=1,
            hidden_mv_channels=hidden_mv_channels,
            in_s_channels=None,
            out_s_channels=None,
            hidden_s_channels=hidden_s_channels,
            num_blocks=num_blocks,
            attention=SelfAttentionConfig(),  # Use default parameters for attention
            mlp=MLPConfig(),  # Use default parameters for MLP
        )
```

```python
def forward(self, fourmomenta):
    """Forward pass.

    Parameters
    ----------
    fourmomenta : torch.Tensor with shape (batchsize, num_points, 4)
        fourmomentum point cloud input data

    Returns
    -------
    outputs : torch.Tensor with shape (batchsize, 1)
        Model prediction: a single scalar for the whole point cloud.
    """
    batchsize, num_points, _ = fourmomenta.shape

    # Embed fourmomentum point cloud inputs in GA
    multivectors = embed_vector(fourmomenta).unsqueeze(-2)  # (batchsize, num_points, 1, 1

    # Append spurions for symmetry breaking (optional)
    spurions = embed_spurions(beam_reference="xyplane", add_time_reference=True, device=fo
    spurions = spurions[None, None, ...].repeat(batchsize, num_points, 1, 1)  # (batchsize
    multivectors = torch.cat((multivectors, spurions), dim=-2)  # (batchsize, num_points,

    # Pass data through GATr
    multivector_outputs, _ = self.gatr(multivectors, scalars=None)  # (batchsize, num_poin

    # Extract scalar outputs
    outputs = extract_scalar(multivector)  # (batchsize, num_points, 1)

    # Mean aggregation to extract a single scalar for the whole point cloud
    score = outputs.mean(dim=1)

    return score
```

L-GATr is easy to use



github.com/heidelberg-hepml/lgatr

```python
from gatr import GATr, SelfAttentionConfig, MLPConfig
from gatr.interface import embed_vector, extract_scalar, embed_spurions
import torch


class ExampleWrapper(torch.nn.Module):
    """Example wrapper around a L-GATr model.

    Parameters
    ----------
    num_blocks : int
        Number of transformer blocks
    hidden_mv_channels : int
        Number of hidden multivector channels
    hidden_s_channels : int
        Number of hidden scalar channels
    """

    def __init__(self, blocks=6, hidden_mv_channels=16, hidden_s_channels=32):
        super().__init__()
        self.gatr = GATr(
            in_mv_channels=3,
            out_mv_channels=1,
            hidden_mv_channels=hidden_mv_channels,
            in_s_channels=None,
            out_s_channels=None,
            hidden_s_channels=hidden_s_channels,
            num_blocks=num_blocks,
            attention=SelfAttentionConfig(),  # Use default parameters for attention
            mlp=MLPConfig(),  # Use default parameters for MLP
        )
```

```python
def forward(self, fourmomenta):
    """Forward pass.

    Parameters
    ----------
    fourmomenta : torch.Tensor with shape (batchsize, num_points, 4)
        fourmomentum point cloud input data

    Returns
    -------
    outputs : torch.Tensor with shape (batchsize, 1)
        Model prediction: a single scalar for the whole point cloud.
    """
    batchsize, num_points, _ = fourmomenta.shape

    # Embed fourmomentum point cloud inputs in GA
    multivectors = embed_vector(fourmomenta).unsqueeze(-2)  # (batchsize, num_points, 1, 1

    # Append spurions for symmetry breaking (optional)
    spurions = embed_spurions(beam_reference="xyplane", add_time_reference=True, device=fo
    spurions = spurions[None, None, ...].repeat(batchsize, num_points, 1, 1)  # (batchsize
    multivectors = torch.cat((multivectors, spurions), dim=-2)  # (batchsize, num_points,

    # Pass data through GATr
    multivector_outputs, _ = self.gatr(multivectors, scalars=None)  # (batchsize, num_poin

    # Extract scalar outputs
    outputs = extract_scalar(multivector)  # (batchsize, num_points, 1)

    # Mean aggregation to extract a single scalar for the whole point cloud
    score = outputs.mean(dim=1)

    return score
```

Victor Bresó

Pim de Haan

Tilman Plehn

Huilin Qu

Jesse Thaler

Johann Brehmer

Lorentz-Equivariant Geometric Algebra Transformer for High-Energy Physics

Jonas Spinner*, Victor Breso*, Pim de Haan,
Tilman Plehn, Jesse Thaler, Johann Brehmer
NeurIPS 2024, arXiv:2405.14806

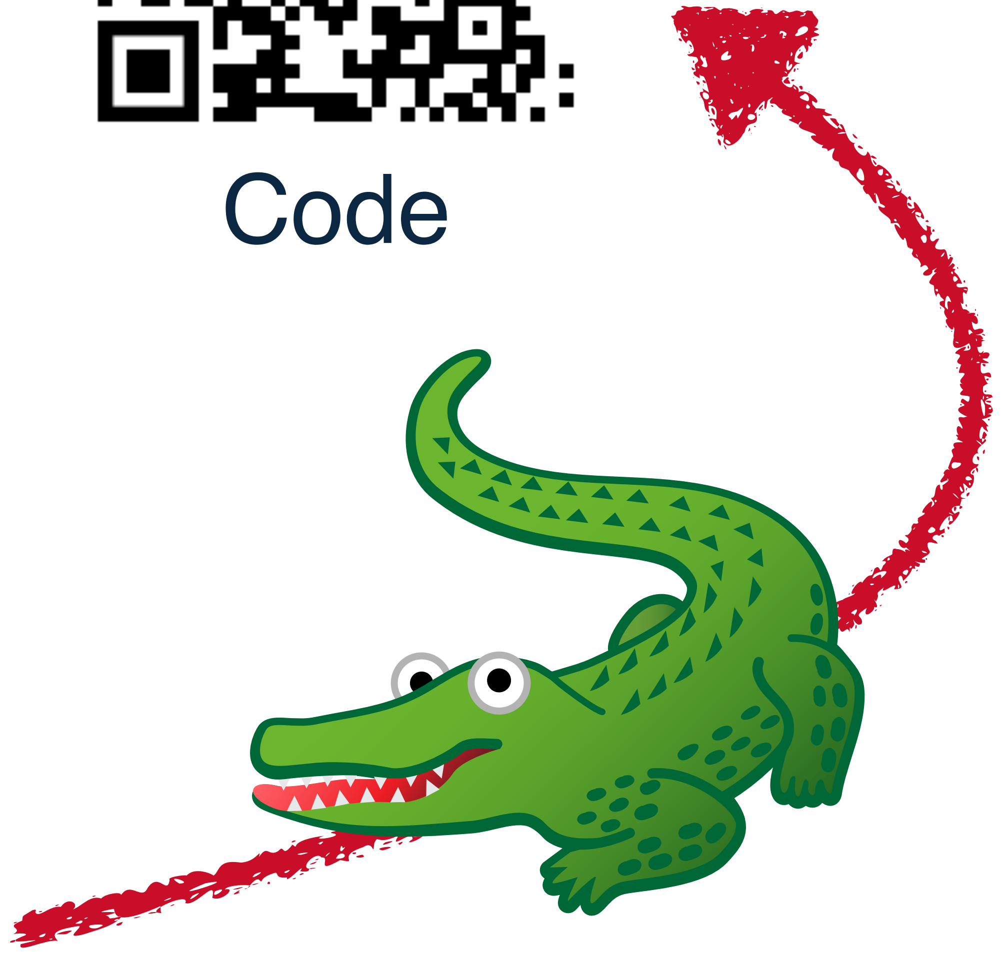
CS paper

A Lorentz-Equivariant Transformer for all of the LHC

Johann Brehmer, Víctor Bresó,
Pim de Haan, Tilman Plehn,
Huilin Qu, Jonas Spinner, Jesse Thaler
arXiv:2411.00446

HEP paper

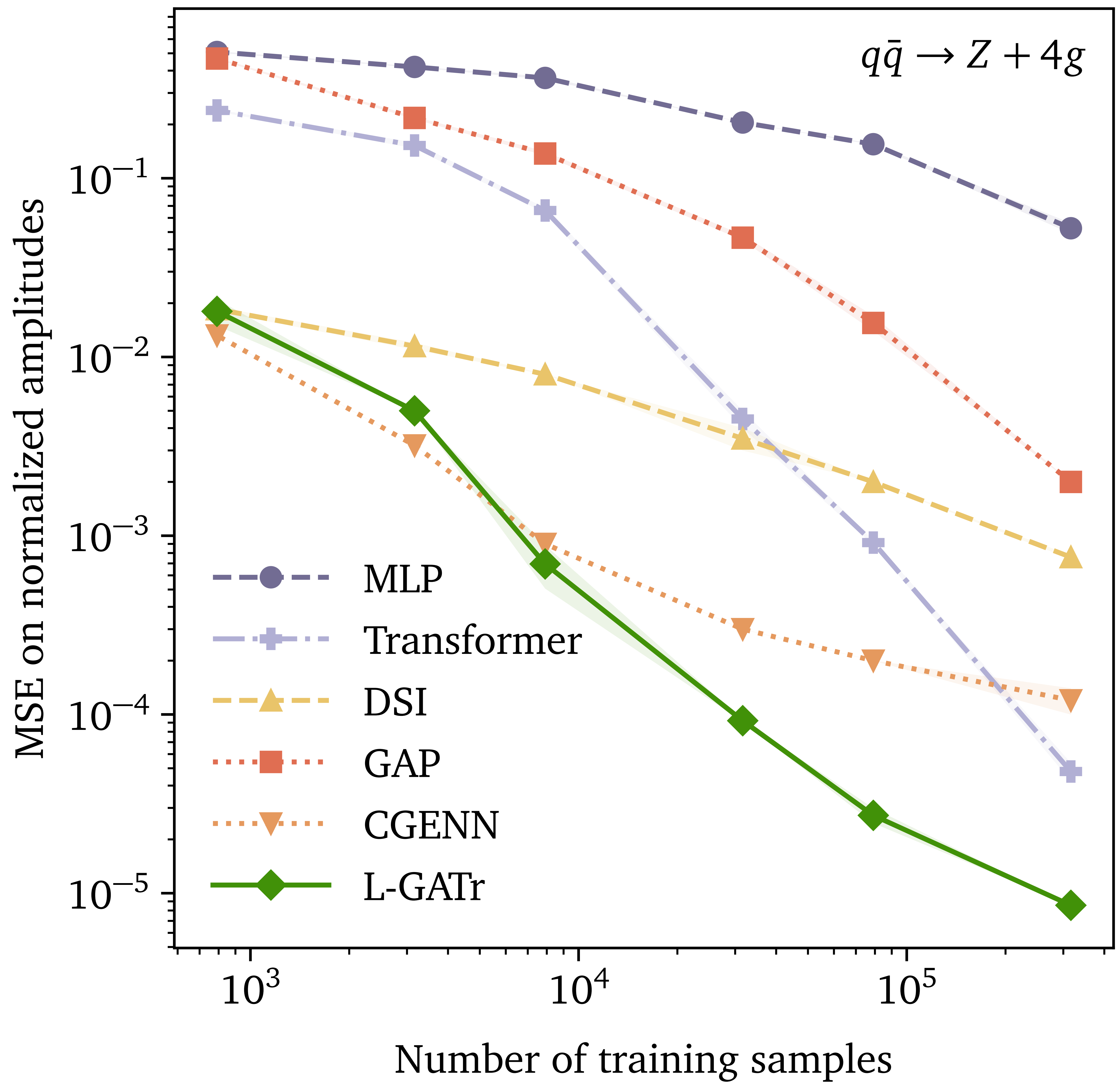
Code

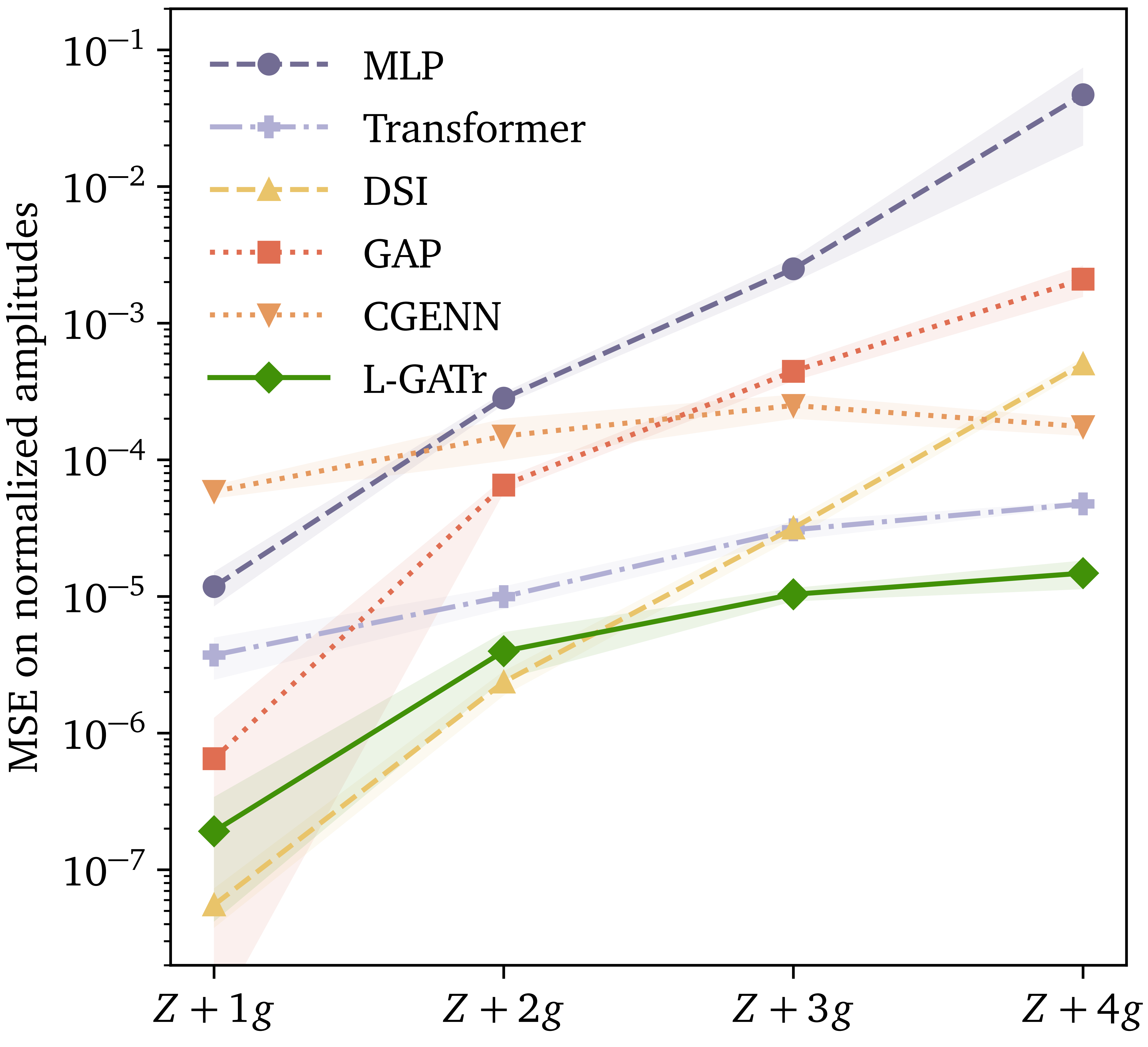


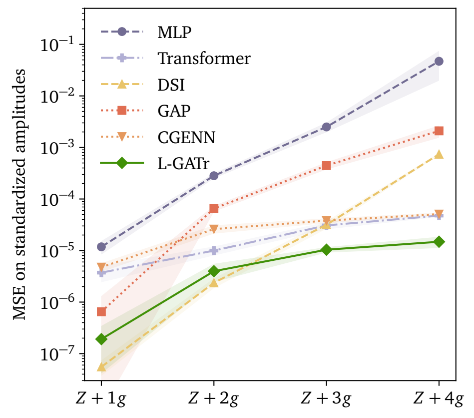
For what will **you** use L-GATr?

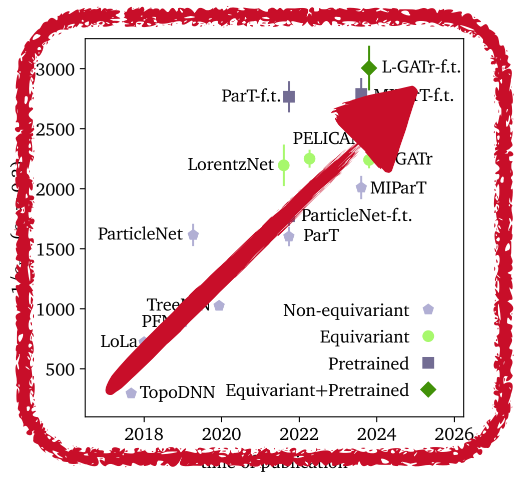
Bonus material

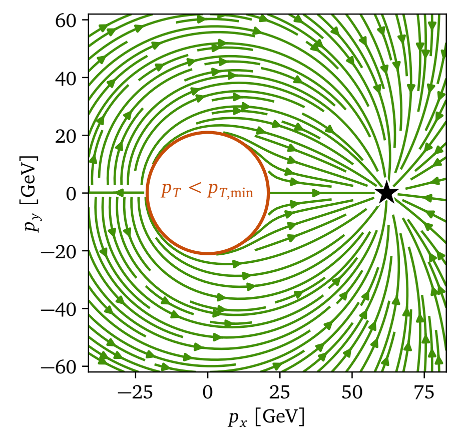
Amplitude regression











Jet Tagging

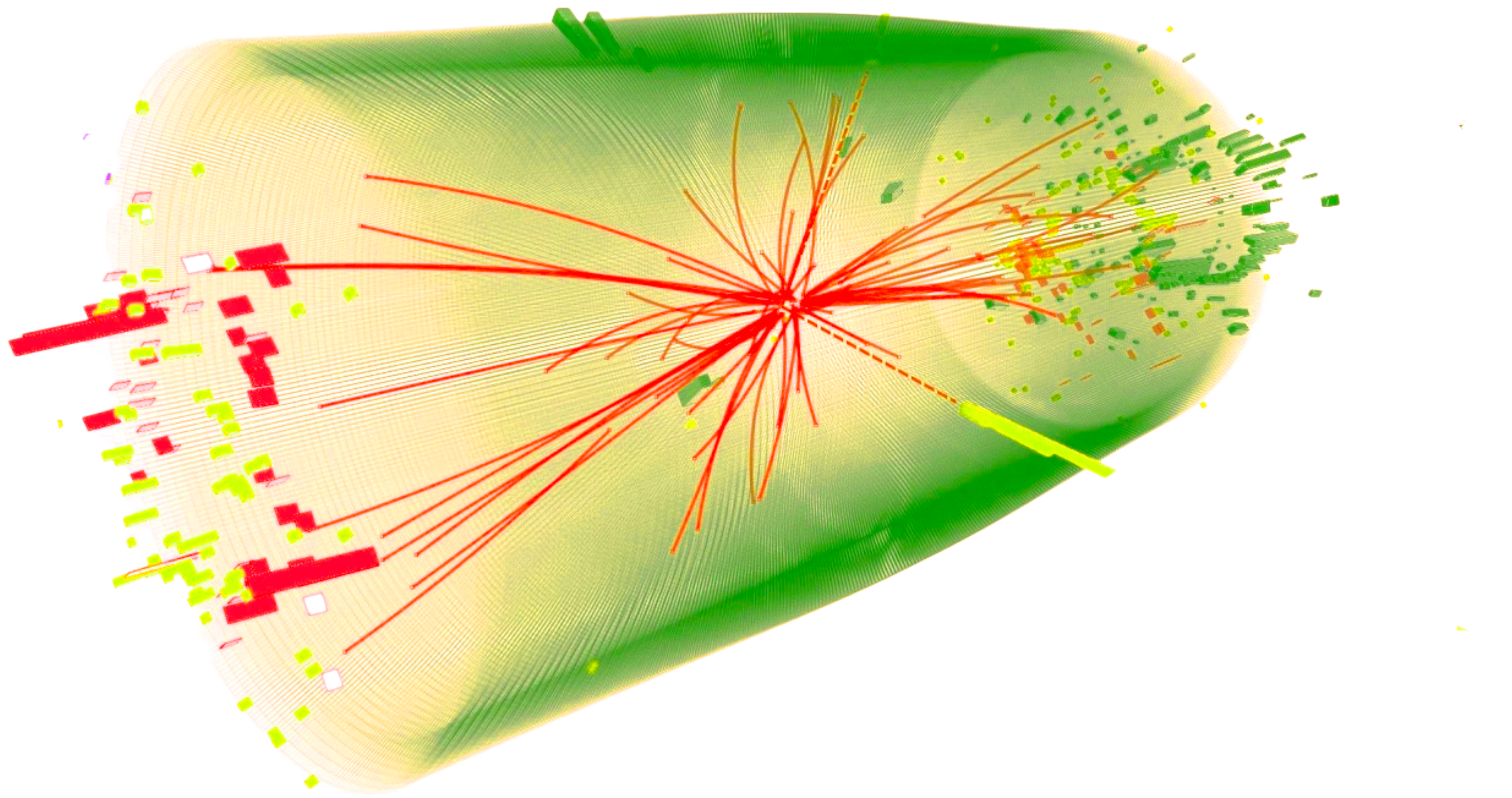
Symmetry breaking with multivectors

The jet tagging score is not Lorentz-equivariant

- Beam direction breaks invariance under rotations around the x- and y-axis
- Detector breaks boost invariance

Add reference multivectors as extra particles to break Lorentz symmetry

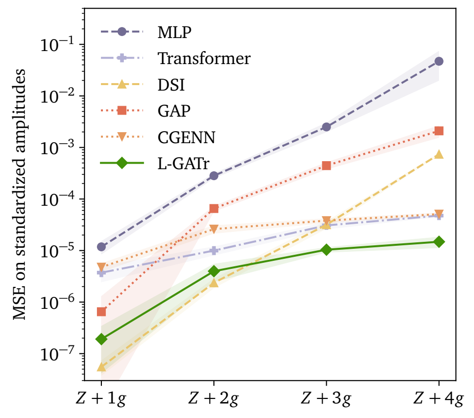
- Beam reference multivector: $x^V = (0,0,0,\pm 1)$ or $x^B_{12} = 1$
- Time reference multivector: $x^V = (1,0,0,0)$

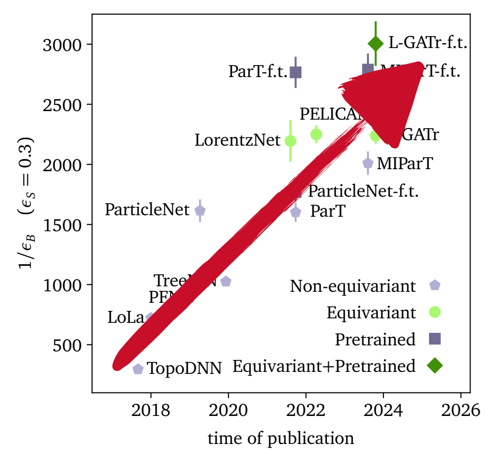


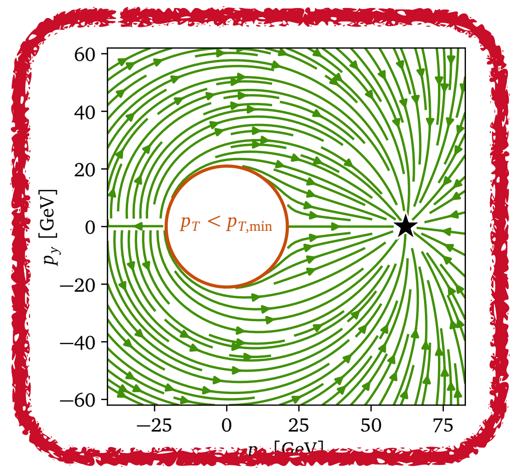
All Lorentz-equivariant taggers use symmetry-breaking inputs

Network	Accuracy	AUC	$1/\epsilon_B$ ($\epsilon_S = 0.5$)	$1/\epsilon_B$ ($\epsilon_S = 0.3$)
TopoDNN [54]	0.916	0.972	–	295 ± 5
LoLa [9]	0.929	0.980	–	722 ± 17
N-subjettiness [55]	0.929	0.981	–	867 ± 15
PFN [56]	0.932	0.9819	247 ± 3	888 ± 17
TreeNiN [57]	0.933	0.982	–	1025 ± 11
ParticleNet [58]	0.940	0.9858	397 ± 7	1615 ± 93
ParT [59]	0.940	0.9858	413 ± 16	1602 ± 81
MIParT [60]	0.942	0.9868	505 ± 8	2010 ± 97
LorentzNet* [10]	0.942	0.9868	498 ± 18	2195 ± 173
CGENN* [14]	0.942	0.9869	500	2172
PELICAN* [42]	0.9426 ± 0.0002	0.9870 ± 0.0001	–	2250 ± 75
L-GATr* [35]	0.9423 ± 0.0002	0.9870 ± 0.0001	540 ± 20	2240 ± 70
ParticleNet-f.t. [60]	0.942	0.9866	487 ± 9	1771 ± 80
ParT-f.t. [60]	0.944	0.9877	691 ± 15	2766 ± 130
MIParT-f.t. [60]	0.944	0.9878	640 ± 10	2789 ± 133
L-GATr-f.t.* (new)	0.9442 ± 0.0002	0.98792 ± 0.00004	661 ± 24	3005 ± 186

	All classes		$H \to b\bar{b}$	$H \to c\bar{c}$	$H \to gg$	$H \to 4q$	$H \to l\nu q\bar{q}'$	$t \to bq\bar{q}'$	$t \to bl\nu$	$W \to q\bar{q}'$	$Z \to q\bar{q}$
	Accuracy	AUC	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{99\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{99.5\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$
ParticleNet [58]	0.844	0.9849	7634	2475	104	954	3339	10526	11173	347	283
ParT [59]	0.861	0.9877	10638	4149	123	1864	5479	32787	15873	543	402
MIParT [60]	0.861	0.9878	10753	4202	123	1927	5450	31250	16807	542	402
L-GATr	0.865	0.9884	12195	4819	128	2304	5764	37736	19231	580	427







Event generation

Conditional Flow Matching

Continuous normalising flow (CNF)
connect a simple base density
to a complex target density
through a neural differential equation

$$\frac{d}{dt}x = v_t(x)$$

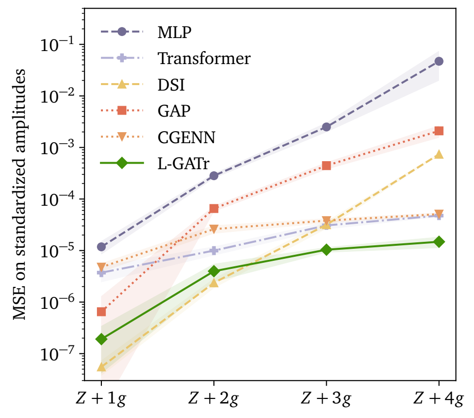
arXiv:1806.07366

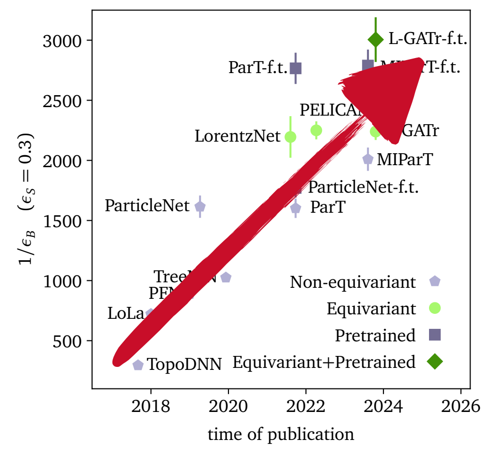
Conditional flow matching (CFM)
is a simple way to train CNFs
by comparing the learned velocity $v_t(x)$
to a conditional target velocity $u_t(x\,|\,x_1)$

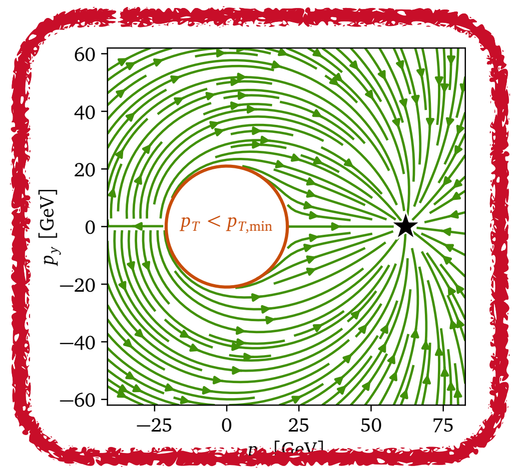
$$\mathscr{L} = \left\langle (v_t(x) - u_t(x\,|\,x_1))^2 \right\rangle$$

arXiv:2210.02747

How to pick the target velocity $u_t(x\,|\,x_1)$?





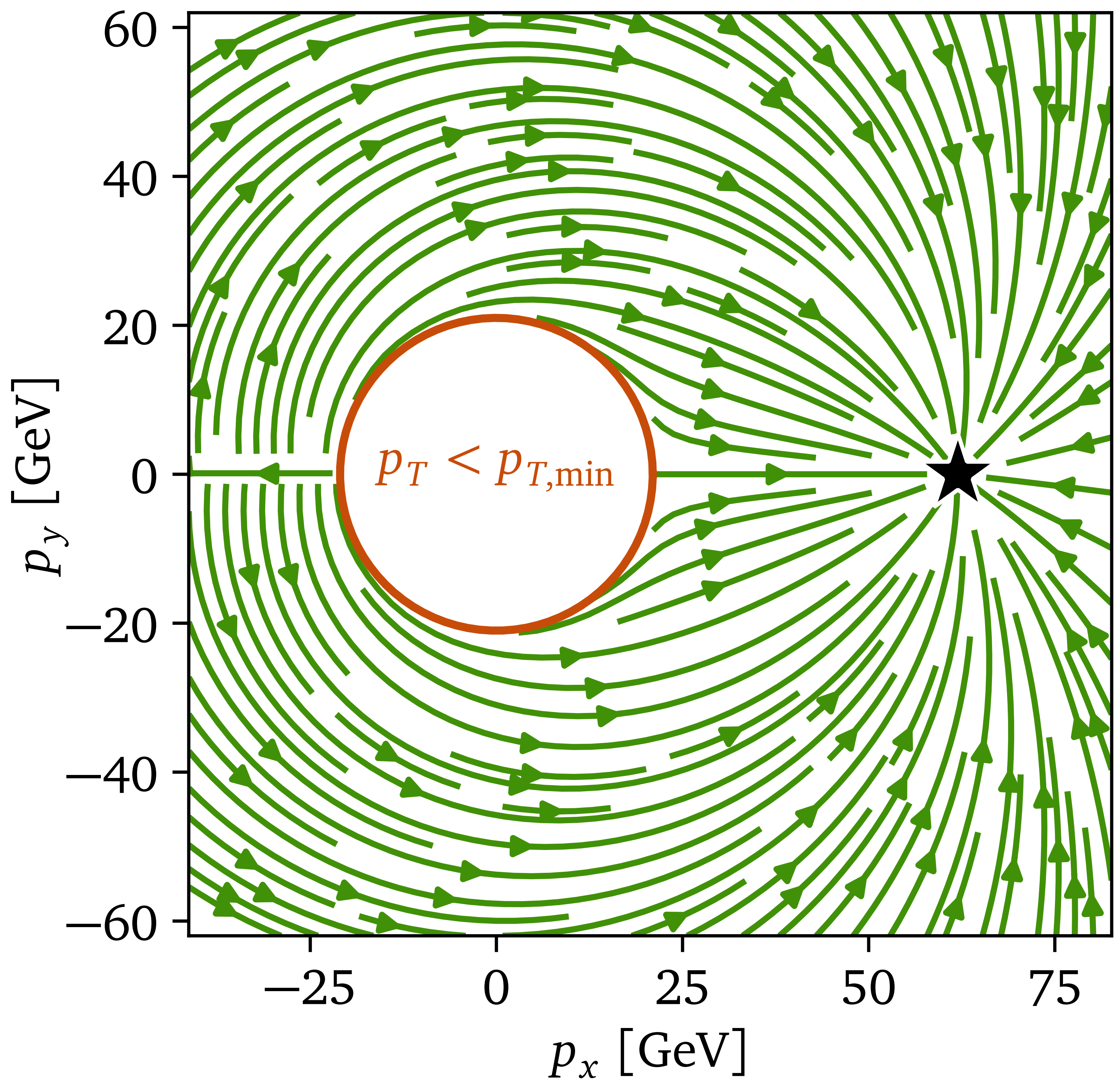


Event generation

Physics-inspired target trajectories

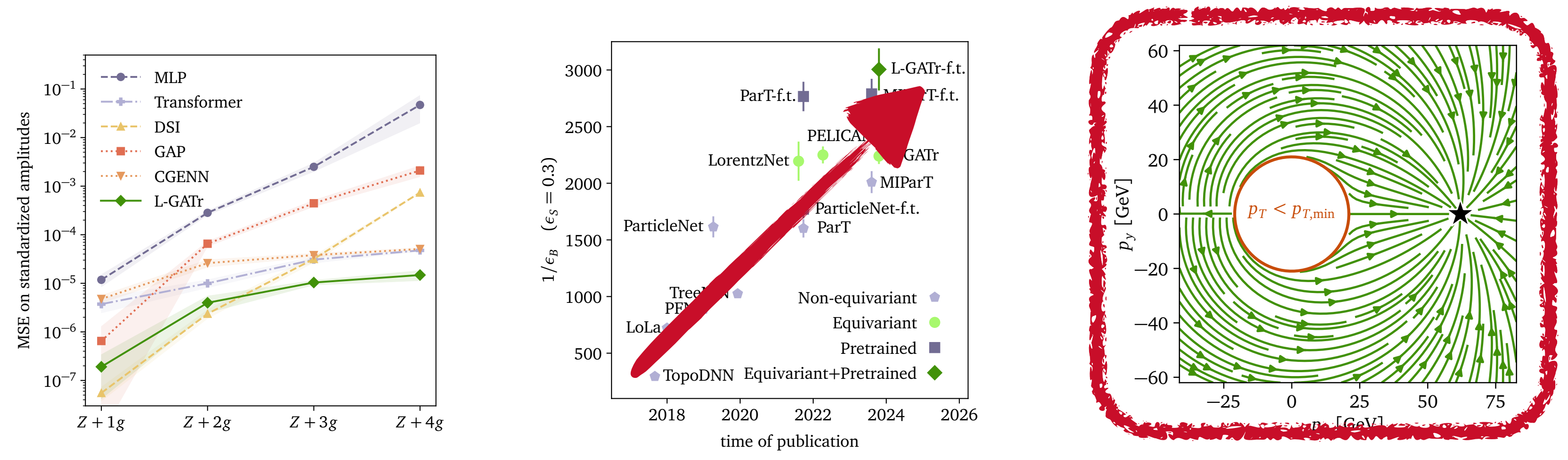
Straight trajectories in ‘modified jet momenta’ x:

$$p = \begin{pmatrix} E \\ p_x \\ p_y \\ p_z \end{pmatrix} \quad \to \quad f^{-1}(p) = x = \begin{pmatrix} x_p \\ x_m \\ x_\eta \\ x_\phi \end{pmatrix} \equiv \begin{pmatrix} \log(p_T - p_{T,\mathrm{min}}) \\ \log m^2 \\ \eta \\ \phi \end{pmatrix}$$



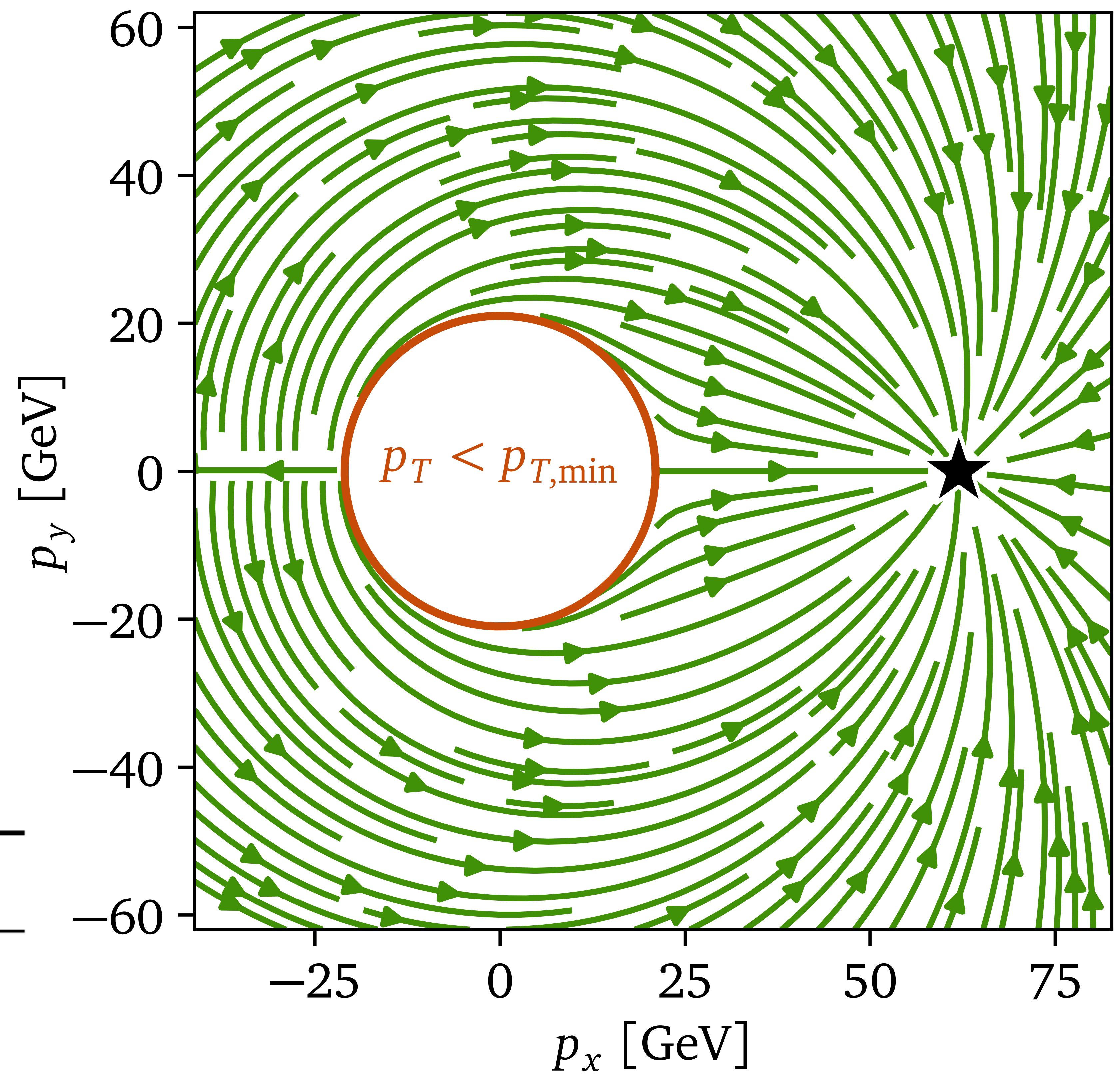
Event generation

Physics-inspired target trajectories



Straight trajectories in ‘modified jet momenta’ x:

$$p = \begin{pmatrix} E \\ p_x \\ p_y \\ p_z \end{pmatrix} \quad \rightarrow \quad f^{-1}(p) = x = \begin{pmatrix} x_p \\ x_m \\ x_\eta \\ x_\phi \end{pmatrix} \equiv \begin{pmatrix} \log(p_T - p_{T,\mathrm{min}}) \\ \log m^2 \\ \eta \\ \phi \end{pmatrix}$$

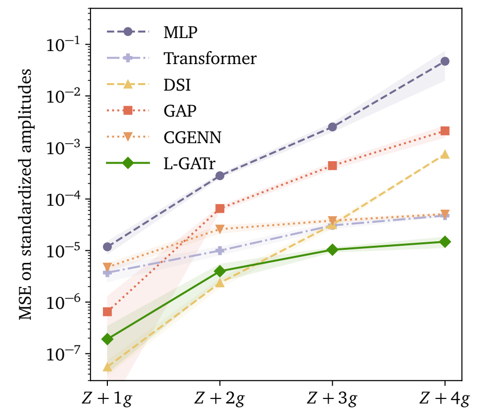


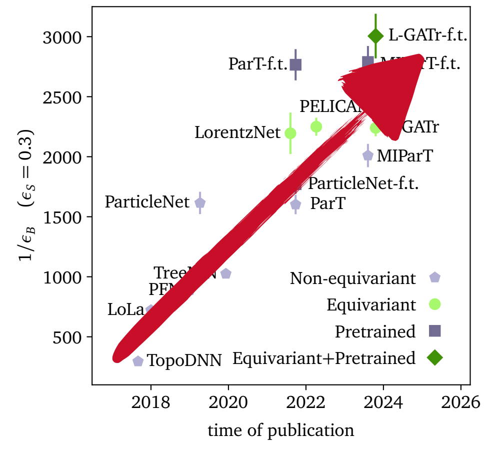
Data	Architecture	Base distribution	Periodic	Neg. log-likelihood	AUC
p	L-GATr	rejection sampling	✓	-30.80 ± 0.17	0.945 ± 0.004
x	MLP	rejection sampling	✓	-32.13 ± 0.05	0.780 ± 0.003
x	L-GATr	rejection sampling	✗	-32.57 ± 0.05	0.530 ± 0.017
x	L-GATr	no rejection sampling	✓	-32.58 ± 0.04	0.523 ± 0.014
x	L-GATr	rejection sampling	✓	-32.65 ± 0.04	0.515 ± 0.009

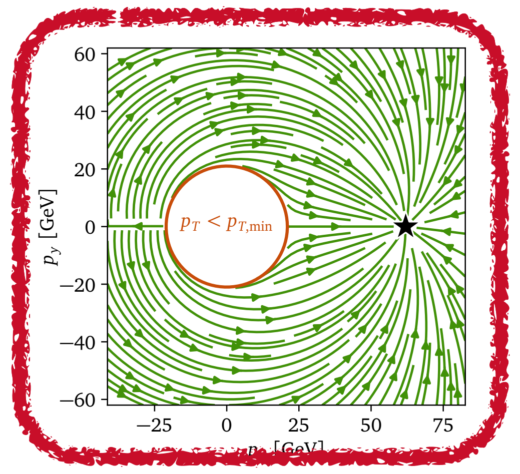
default

Event generation

How to build an equivariant CFM field $v_x(x)$?

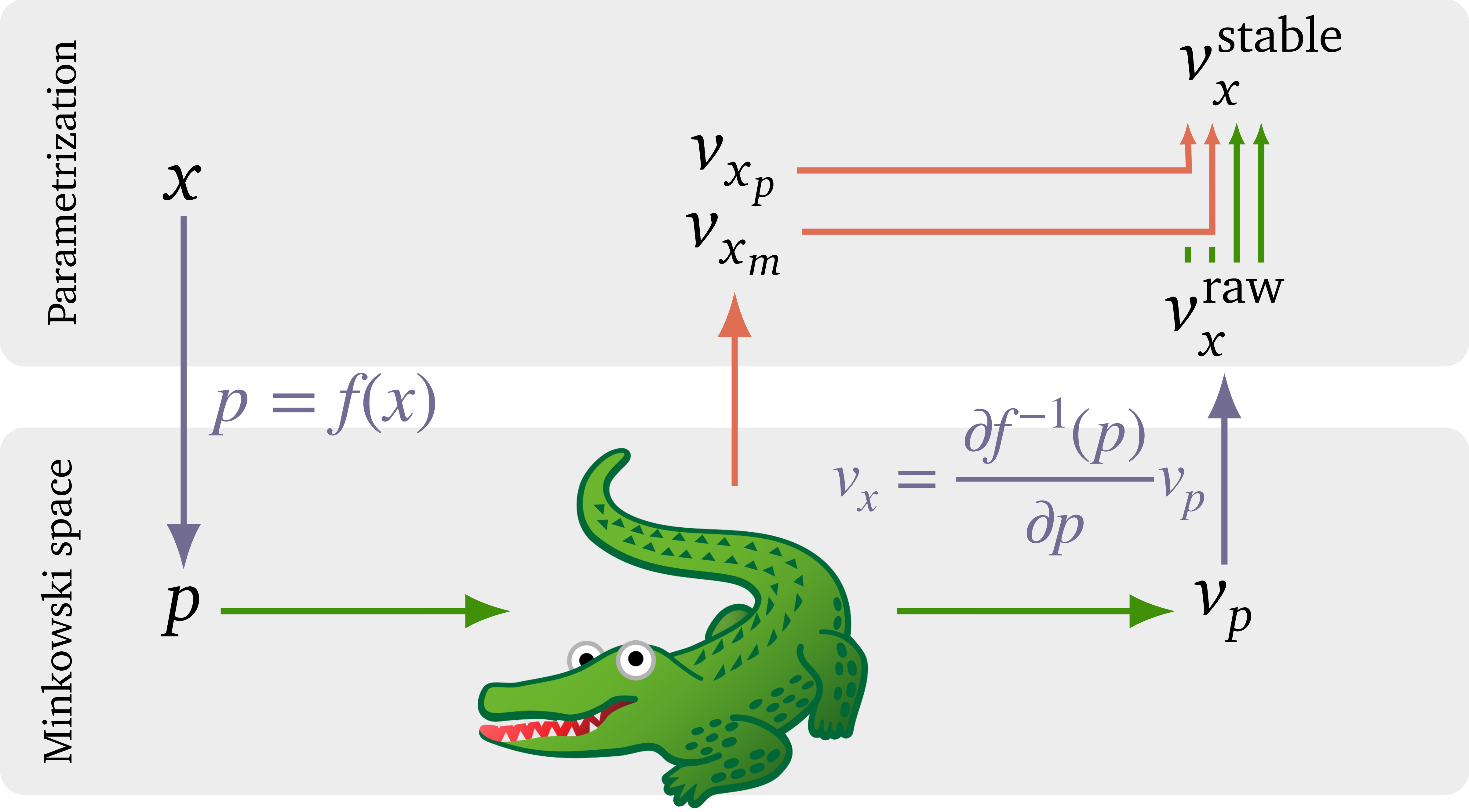






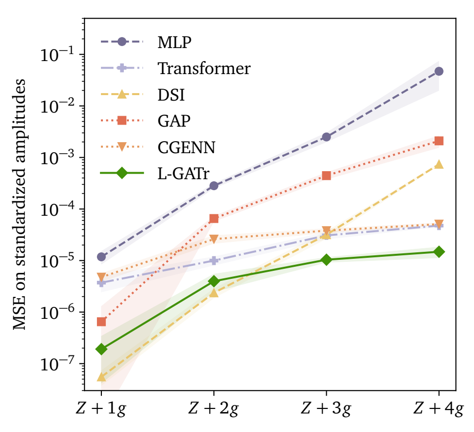
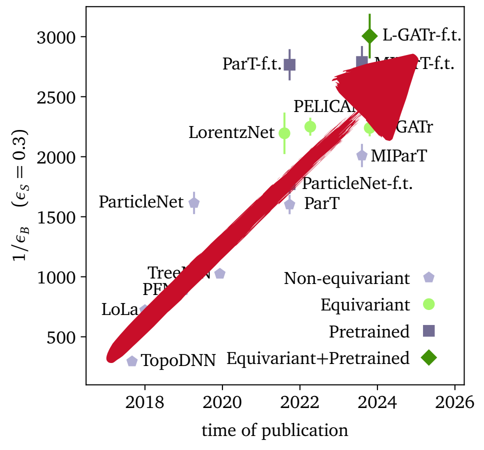
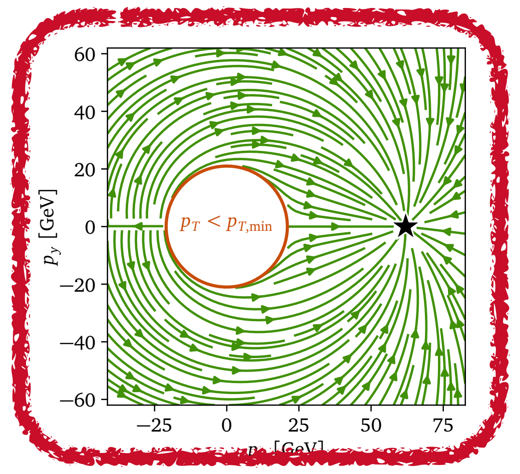
Extend standard CFM workflow with L-GATr:

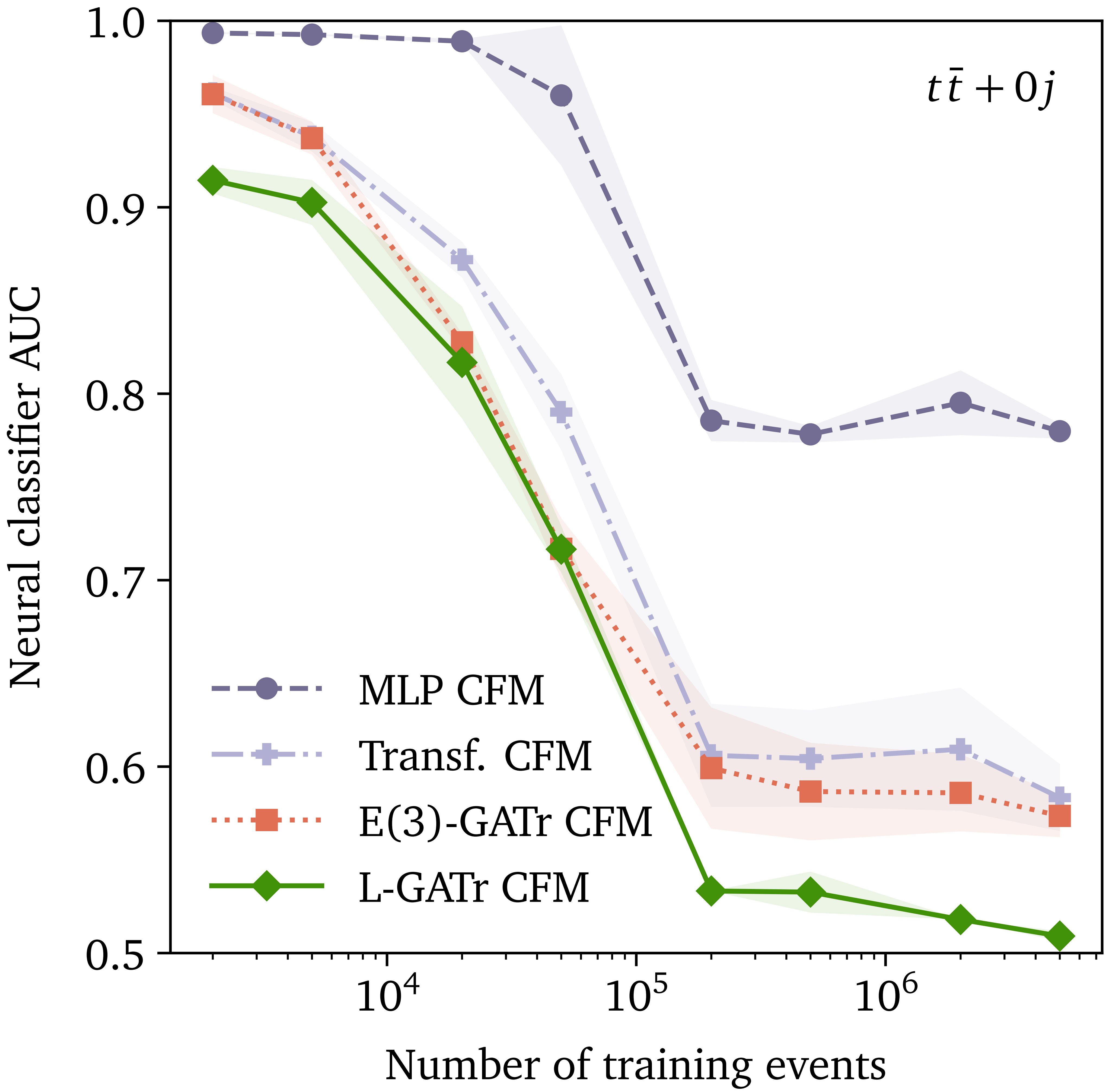
- Transformations $f(x)$ between Minkowski space p and the parametrization x
- Equivariant L-GATr operations using multivectors
- Symmetry-breaking operations using scalars (required for numerical stability)



Event generation

Boost equivariance lets L-GATr shine

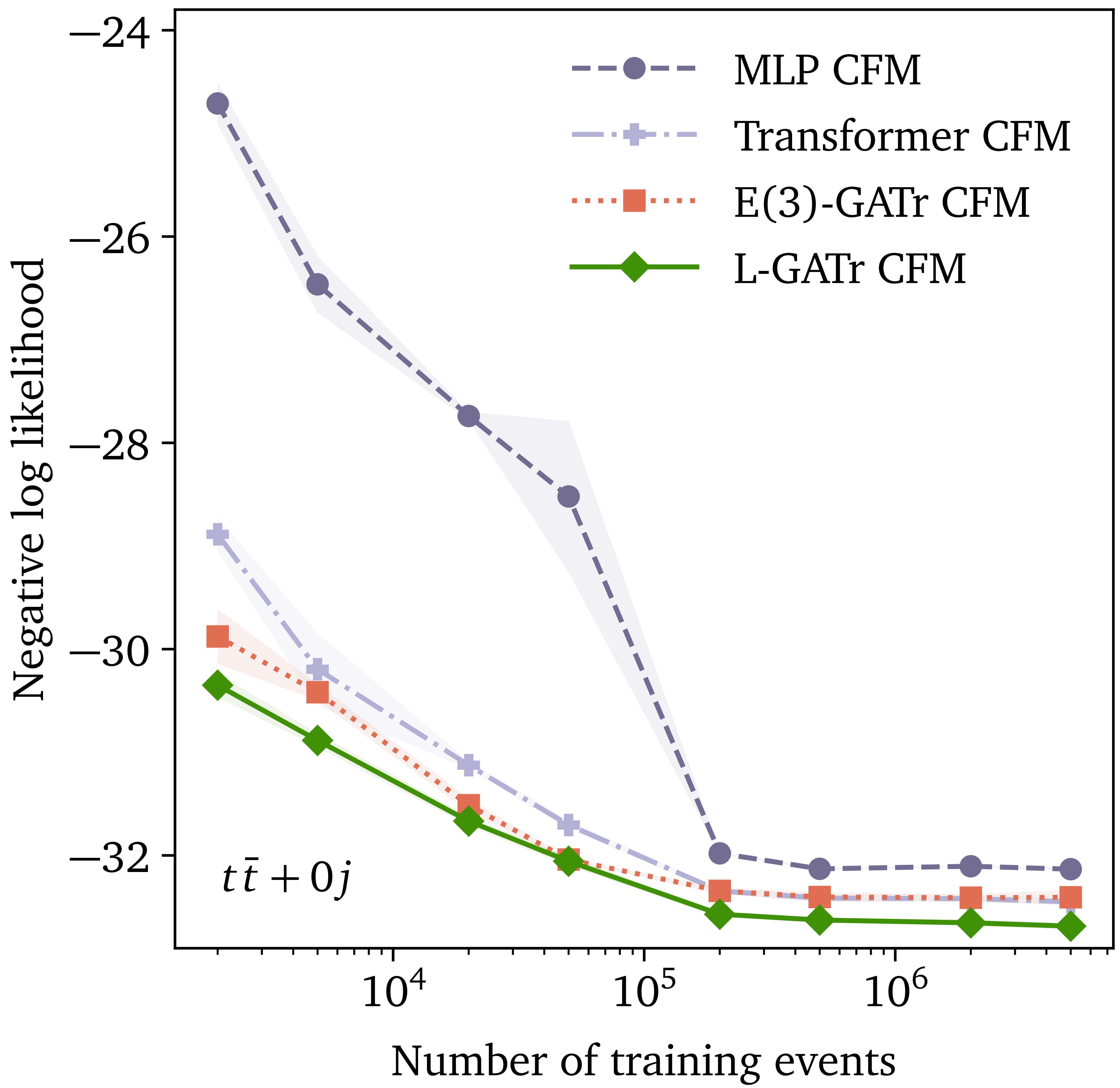


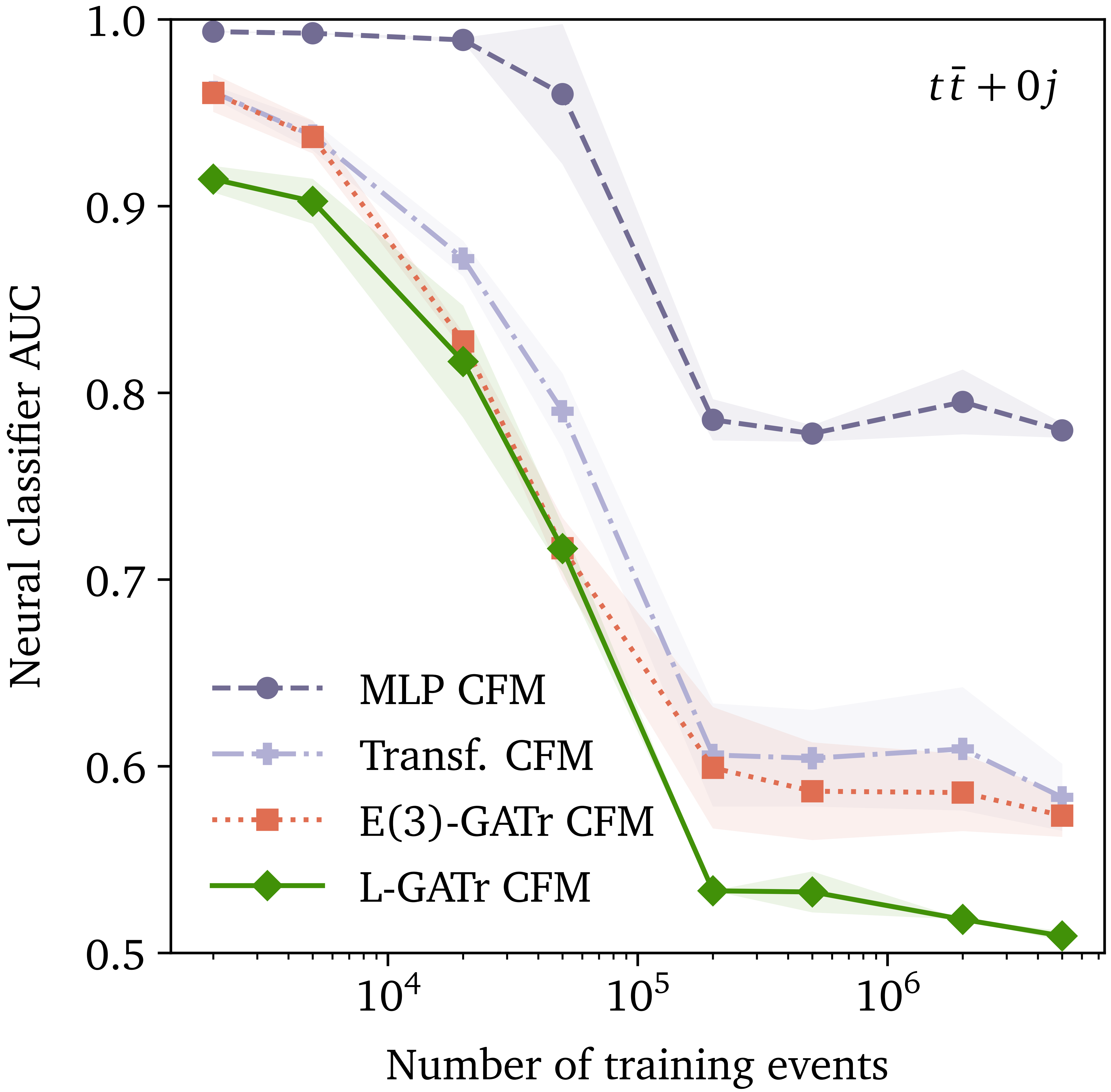
L-GATr and E(3)-GATr are both not boost-equivariant:

- E(3)-GATr: No boost-equivariance at all
- L-GATr: Architecture is boost-equivariant, but network inputs are not

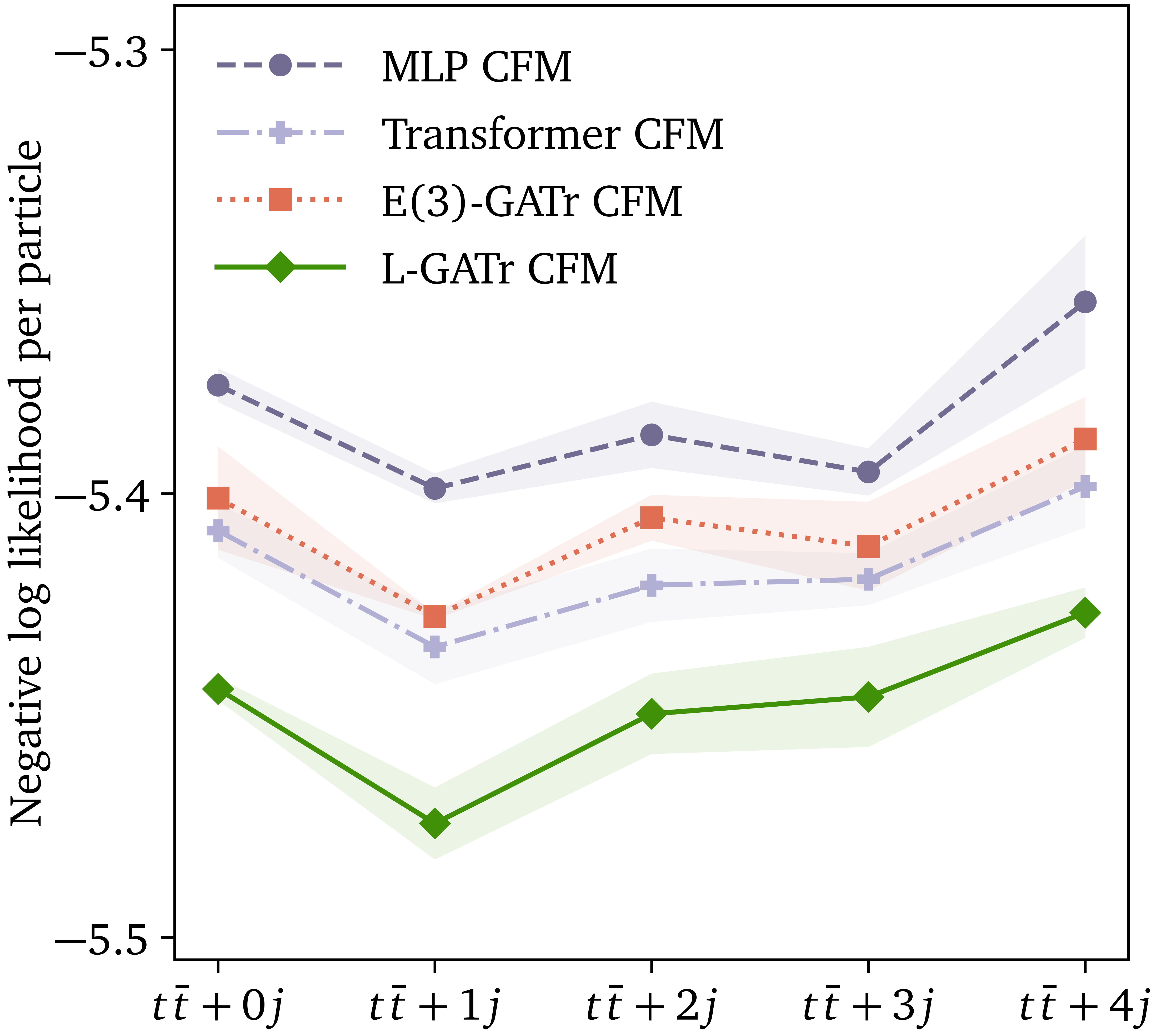
Its all about how we break the symmetry!

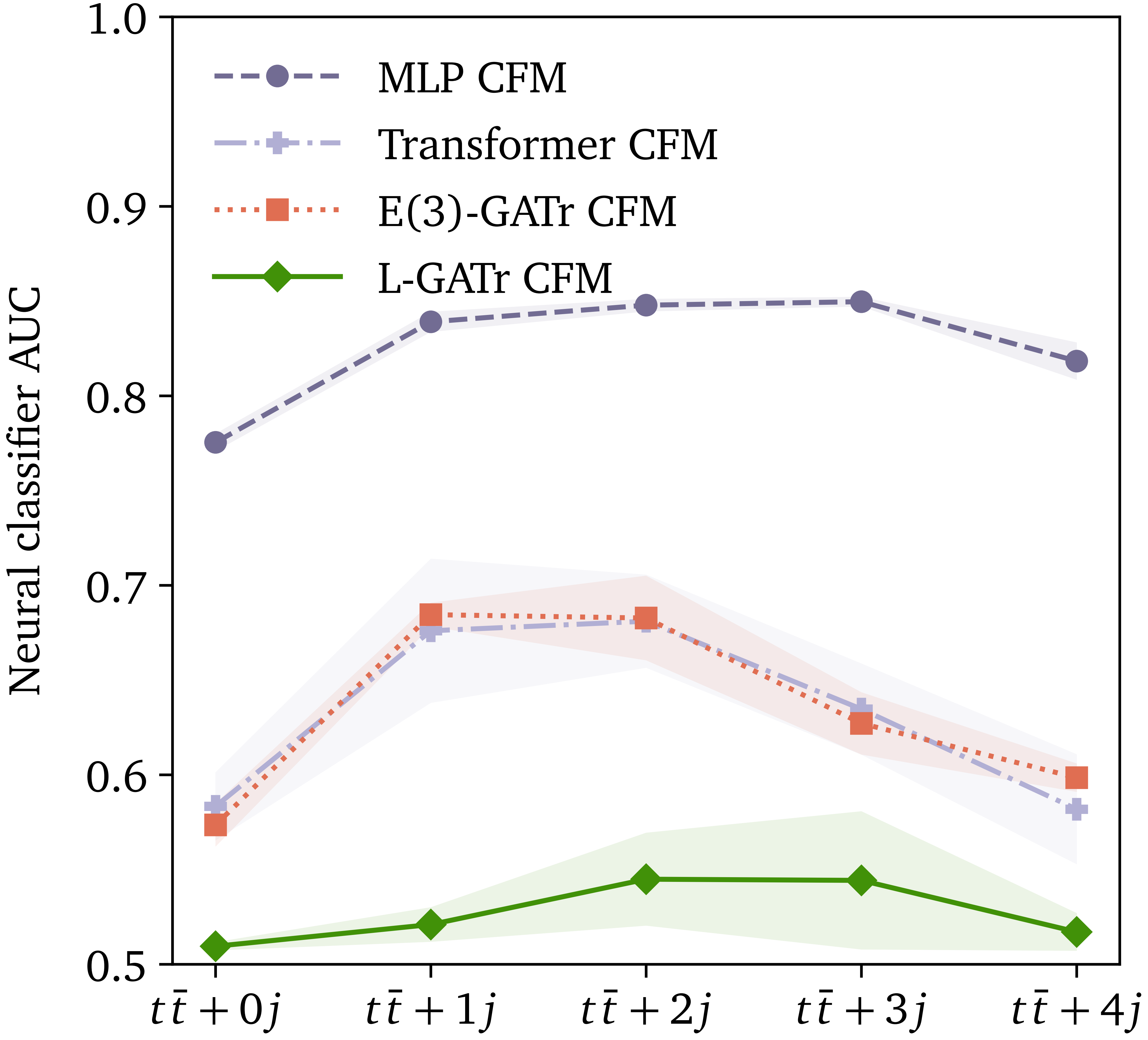
Event generation



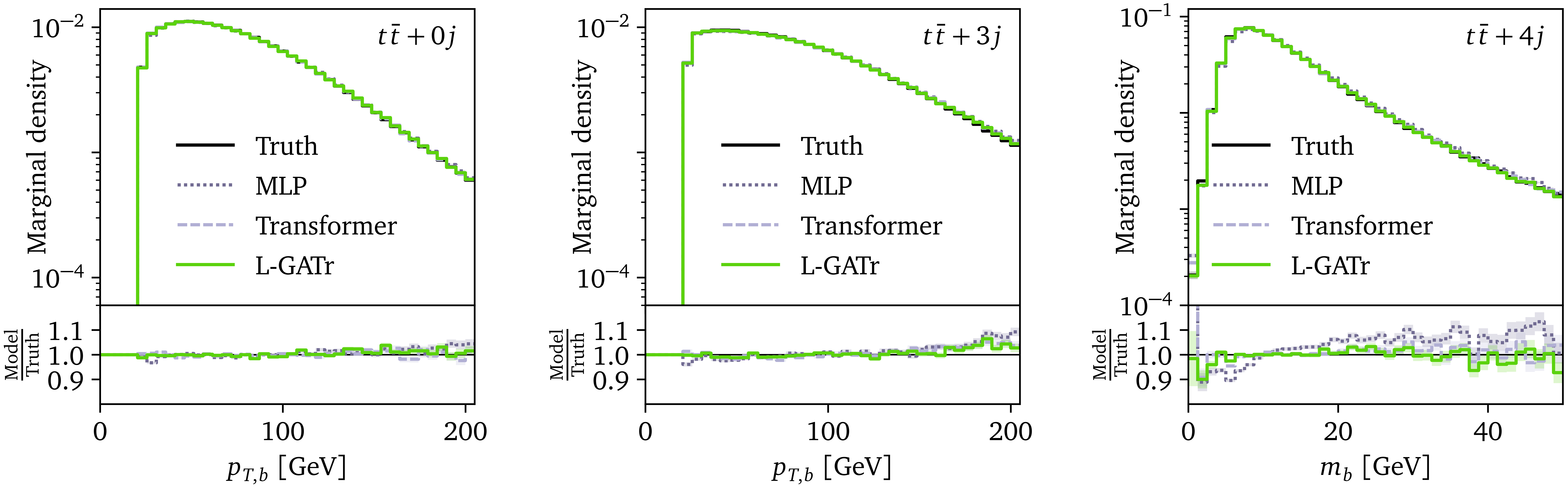


Event generation

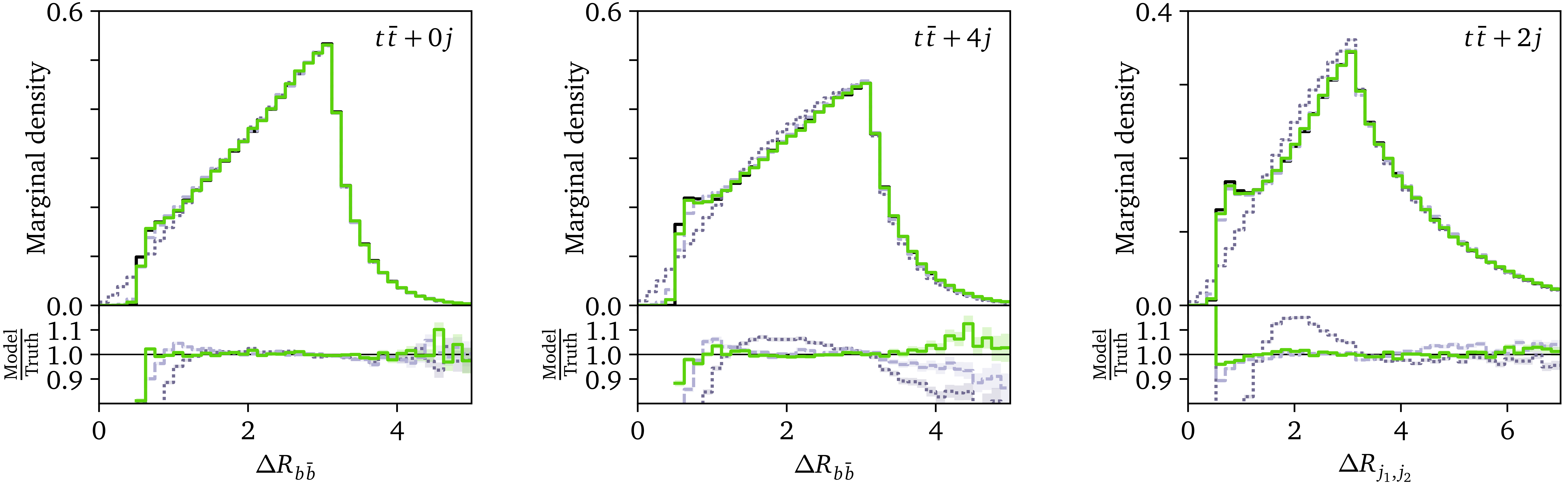




Event generation



Event generation



Event generation