Transformers, Attention, and Symmetries for Inverse Problems in High Energy Physics

Alexander Shmakov
Jul 28, 2023

Based on work performed for

SPANet: Generalized Permutationless Set Assignment for Particle Physics using Symmetry Preserving Attention
Alexander Shmakov, Michael James Fenton, Ta-Wei Ho, Shih-Chieh Hsu, Daniel Whiteson, Pierre Baldi
SciPost Phys. 12, 178 (2022)

https://github.com/Alexanders101/SPANet

Problem Overview Reconstruction

- The Large Hadron Collider (LHC) at CERN produces observations from decays of high energy particle collisions.
- The LHC reproduces the conditions present in the early universe, shortly after the Big Bang.
- We work with the ATLAS detector, which provides calorimetric momentum measurements of decay products

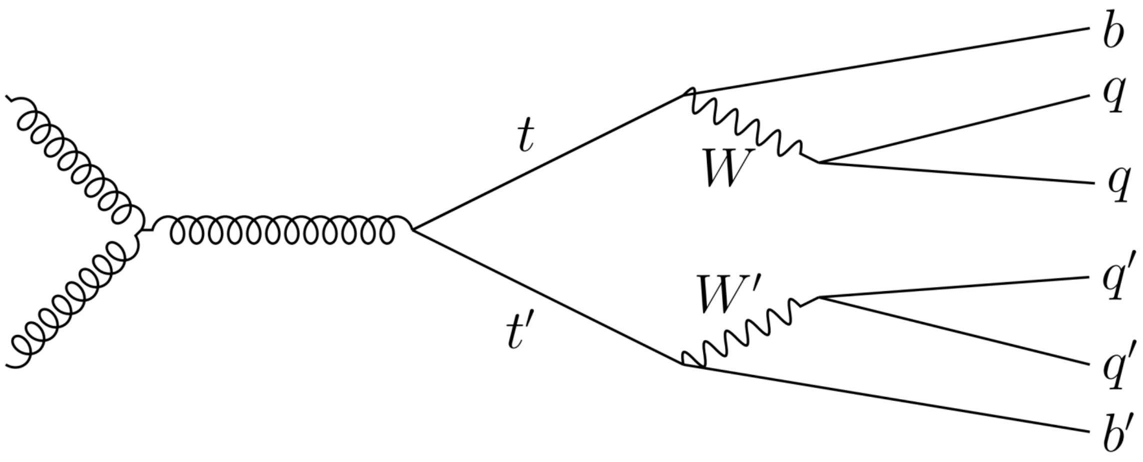
Reconstruction

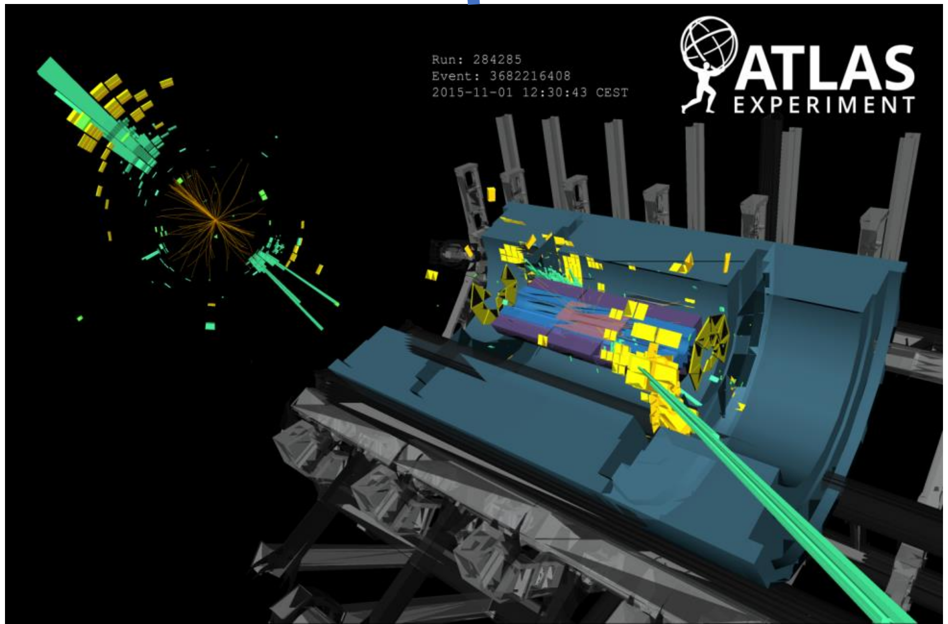
Theory

Partons

Reality

Jets

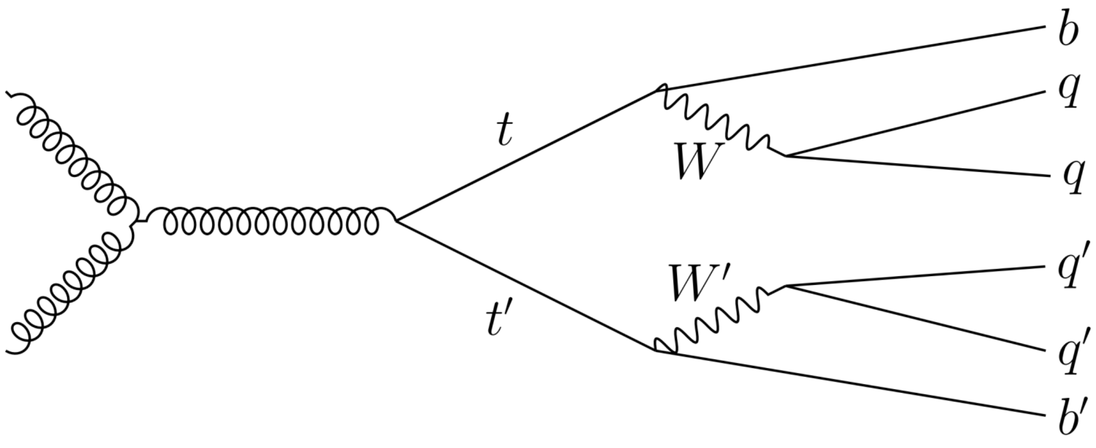




Problem Overview Jet-Parton Matching

The simplest initial step to reconstruction: **What are each of these jets?**

- Could be many simultaneous decays in each event.
- Typically many more jets than partons.
- Initial cuts and requirements eliminate most of the garbage jets, but most events have at least 2 extra jets.
- Two additional complications.
  - Inputs are unordered collections (Sets) of jets.
  - Outputs are not unique! Notice that every top produces two q partons, producing four symmetric answers.



Set of Jets

$$\{j_1, j_2, j_3, j_4, j_5, j_6, j_7, j_8\}$$

Match Jets to Parton Labels

$$\{b, q', \emptyset, q', b', \emptyset, q, q\}$$

Targets

Garbage Jets

Problem Overview Set Assignment

This modeling task reduces to a unique set assignment problem.

Input is a set of size N

$$\{j_1, j_2, \dots, j_N\}$$

Possible Targets are a set of size $C \leq N$ and a special null target $\emptyset$

$$\{\emptyset, t_1, t_2, \dots, t_C\}$$

Output is another set of size N

with each $p \in \{\emptyset, t_1, t_2, \dots, j_C\}$ $s.t. p_i \neq p_j$ or $p_i = \emptyset$

$$\{p_1, p_2, \dots, p_N\}$$

Set Assignment Itemized Approach

The simplest approach to unique set classification would be **independent classification.**

Train a jet classifier f_θ which treats each jet as a separate object.

Postprocess your predictions and select the highest probability assignment.

$$\{j_1, j_2, \ldots, j_N\}$$

$$\downarrow f_\theta$$

$$\{\vec{d}_1, \vec{d}_2, \ldots, \vec{d}_N\}$$

$$\downarrow max$$

$$\{p_1, p_2, \ldots, p_N\}$$

Big Problems!

- How to prevent two identical targets being predicted? Maybe removing elements?
- How to pick order to go through targets? Different ordering could change the prediction!
- The network has no information about the uniqueness. No context for each input!

Set Assignment Permutation Approach

A more invariant approach to this would be the **permutation score function**.

Generate every C-permutation of your set.

Score each permutation with DNN $f_\theta(S) \in \mathbb{R}$.

Predict the highest scoring permutation.

$$\{j_1, j_2, \ldots, j_k\}$$

↓ **Permute**

$$\{S_1, S_2, \ldots, S_{k\mathbf{P}C}\}$$

↓ **Score**

$$\{f_\theta(S_1), f_\theta(S_2), \ldots, f_\theta(S_{k\mathbf{P}T})\}$$

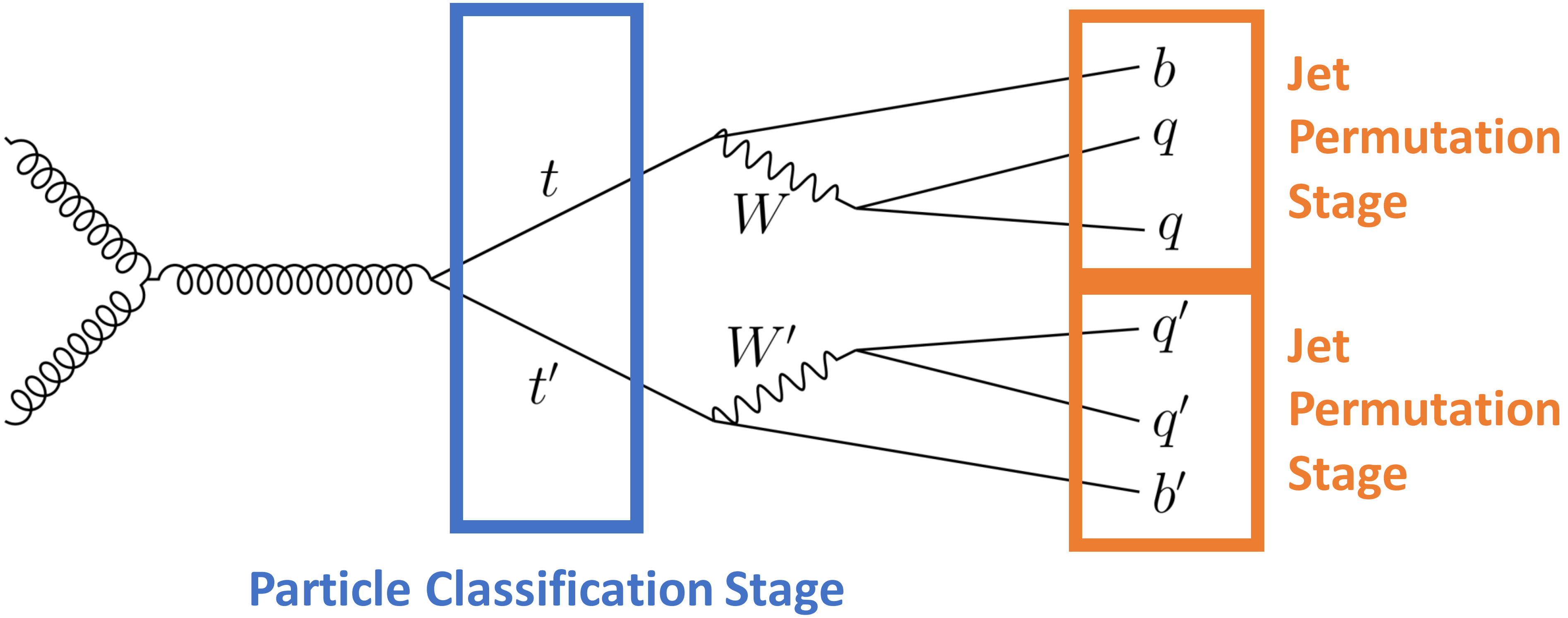
Good approach but terrible run-time

- Currently used in many baseline methods
- Need to generate every permutation! Runtime is $O(N^C)$.

SPANet A Combined Approach

Merge these two approaches to get the best of both **plus symmetry**!

Output **independent sub-permutations scores** for each top-level particle and learn to **differentiate sub-permutations with classification**.



Symmetry Input Permutation Equivariance

- Input has not inherent order, just a collection of observations.
- Outputs match the order of input.
- Any approach must work for **any** initial ordering inputs.

$$\{j_1, j_2, j_3, j_4, j_5, j_6, j_7, j_8\} \cong \{j_3, j_7, j_1, j_2, j_8, j_4, j_6, j_5\}$$
$$\{b, q', \emptyset, q', b', \emptyset, q, q\} \cong \{\emptyset, q, b, q', q, q', \emptyset, b'\}$$

- *Enforce* an arbitrary consistent ordering? **Hard to justify.**
- Feed in all the permutations and average? **Very expensive**.
- Use a permutation equivariant architecture from the start!

Attention

Attention Overview

Vectors

Continuous, differentiable
key-value database

$$Q = \{q_1, q_2, \ldots, q_m\} \quad \textsc{Queries}$$
$$K = \{k_1, k_2, \ldots, k_n\} \quad \textsc{Keys}$$
$$V = \{v_1, v_2, \ldots, v_n\} \quad \textsc{Values}$$

Pick a **SIMILARITY** function. Compute and normalize similarity between query-key pairs.

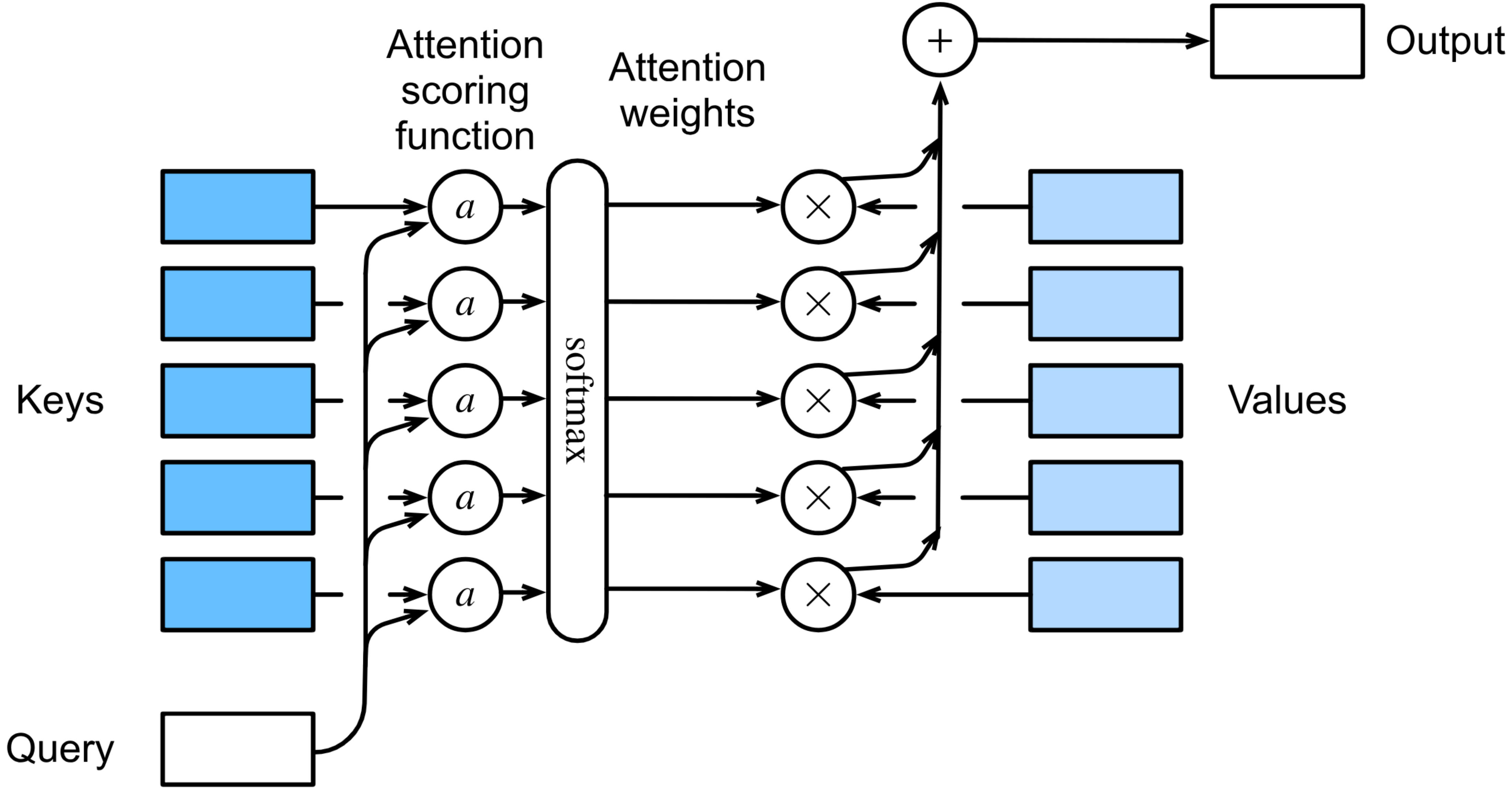
$$S_{ij} = \textsc{Similarity}(q_i, k_j)$$

$$A_{ij} = \textsc{Normalize}(S_{ij}) = \frac{e^{S_{ij}}}{\sum_{l=1}^{n} e^{S_{il}}}$$

Output a weighted average of all values based on similarity.

$$O_i = A_{ij} V^j$$

Attention Overview



Attention Self-Attention

Special Case of attention where we use the same input stream, X, as the queries, keys, and values.

Used to add **context** to collections of objects. Make every element aware of the other elements and learn the relationship between them.

$$Q = f_\theta^Q(X)$$

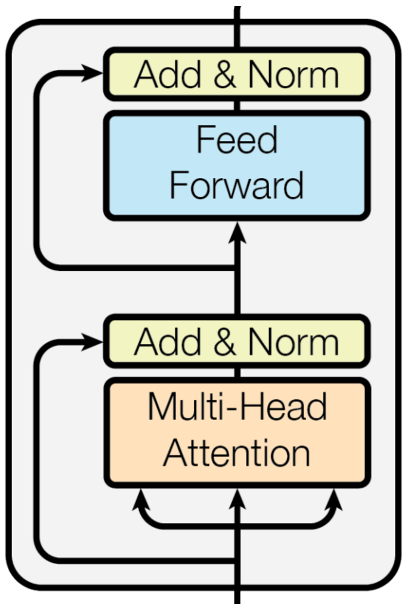
$$K = f_\theta^K(X)$$

$$V = f_\theta^V(X)$$

Attention Transformers

$$\textsc{Similarity}(q_i, k_j) = \frac{q_i \cdot k_j}{\sqrt{D}}$$

Self-attention[1] with ***scaled dot-product*** as the similarity measure.



The **transformer encoder**[2] combines

- Scaled dot-product attention
- Skip-connections
- Layer Normalization
- Position-independent feed-forward layers.

1. The full transformer uses "multi-head" self-attention, but this is conceptually equivalent for our purposes.
2. Vaswani, Ashish, et al. "Attention Is All You Need." Dec. 2017.

Attention Permutation Equivariance

Scaled dot-product is permutation equivariant!

$$P_\pi \in \mathbb{P}^{N \times N}$$

$$\begin{aligned} A_\pi &= \textsc{Normalize}\left(\frac{(P_\pi Q)(P_\pi K)^T}{\sqrt{D}}\right) \\ &= \textsc{Normalize}\left(\frac{P_\pi (QK^T) P_\pi^T}{\sqrt{D}}\right) \\ &= P_\pi \textsc{Normalize}\left(\frac{(QK^T)}{\sqrt{D}}\right) P_\pi^T \\ &= P_\pi A P_\pi^T \end{aligned}$$

$$O_\pi = A_\pi (P_\pi V) = P_\pi A P_\pi^T P_\pi V = P_\pi O$$

Symmetry Target Symmetries

One very interesting property of Feynman Diagram matching is the presence of symmetries.
The following target sets are equivalent due to charge symmetry.

$$\boldsymbol{q_1 q_2} b q_1' q_2' b' \leftrightarrow \boldsymbol{q_2 q_1} b q_1' q_2' b'$$

$$q_1 q_2 b \boldsymbol{q_1' q_2'} b' \leftrightarrow q_1 q_2 b \boldsymbol{q_2' q_1'} b'$$

We call these **jet symmetries.**
We will handle this with **attention**.

$$\underbrace{q_1 q_2 b}_{\mathcal{T}_1} \underbrace{\boldsymbol{q_1' q_2' b'}}_{\mathcal{T}_2} \leftrightarrow \underbrace{\boldsymbol{q_1' q_2' b'}}_{\mathcal{T}_2} \underbrace{q_1 q_2 b}_{\mathcal{T}_1}$$

We call this **particle symmetries.**
We will handle this with a **special loss function**.

Note: this is not the same as allowing duplicate targets because the target groupings **must remain together**.

$$\boldsymbol{q_1 q_2 b q_1'} q_2' b' \neq \boldsymbol{q_1' q_2 b q_1} q_2' b'$$

Tensor Attention Overview

We can also use attention to produce **joint distributions over N dimensions**.
Generalization of dot-product attention: **Tensor Attention**

Suppose $\boldsymbol{X}$ is our list of vectors. This can be viewed as a $(1,1)$-tensor with ranks (N, D).

Suppose $\boldsymbol{\Theta}$ is a $(0, K)$-tensor of **learnable weights** with rank $(D, D, \ldots, D)$.

1. Perform generalized dot-product self-attention on $\boldsymbol{X}$ with the mixing weights $\boldsymbol{\Theta}$.

$$O^{j_1 j_2 \cdots j_N} = X_{n_1}^{j_1} X_{n_2}^{j_2} \ldots X_{n_N}^{j_N} \Theta^{n_1 n_2 \ldots n_N}$$

2. Create a $\boldsymbol{K}$-joint distribution $\boldsymbol{P}$ by normalizing $\boldsymbol{O}$.

$$\mathcal{P}^{j_1 j_2 \cdots j_N} = \frac{\exp O^{j_1 j_2 \cdots j_N}}{\sum \exp O}$$

Tensor Attention Symmetric Attention

Suppose we want our joint distribution to obey **permutation symmetries**.
For example: $p(j_1, j_2, \ldots) = p(j_2, j_1, \ldots)$.

We encode this as a symmetry group on the indices of $\boldsymbol{\Theta}$

Suppose $\boldsymbol{G_P} \subseteq \boldsymbol{S_K}$ is a permutation group acting on the indices $\{j_1, j_2, \ldots, j_K\}$.

1. Create a symmetric weights tensor $\boldsymbol{S}$ by summing over the symmetric indices of $\boldsymbol{\theta}$ according to $\boldsymbol{G_P}$.
2. Perform generalized dot-product on the list of input vectors $\boldsymbol{X}$ with a **symmetric** weights tensor $\boldsymbol{S}$.
3. Create a **symmetric** joint distribution $\boldsymbol{P}$ by normalizing $\boldsymbol{O}$ just like before.

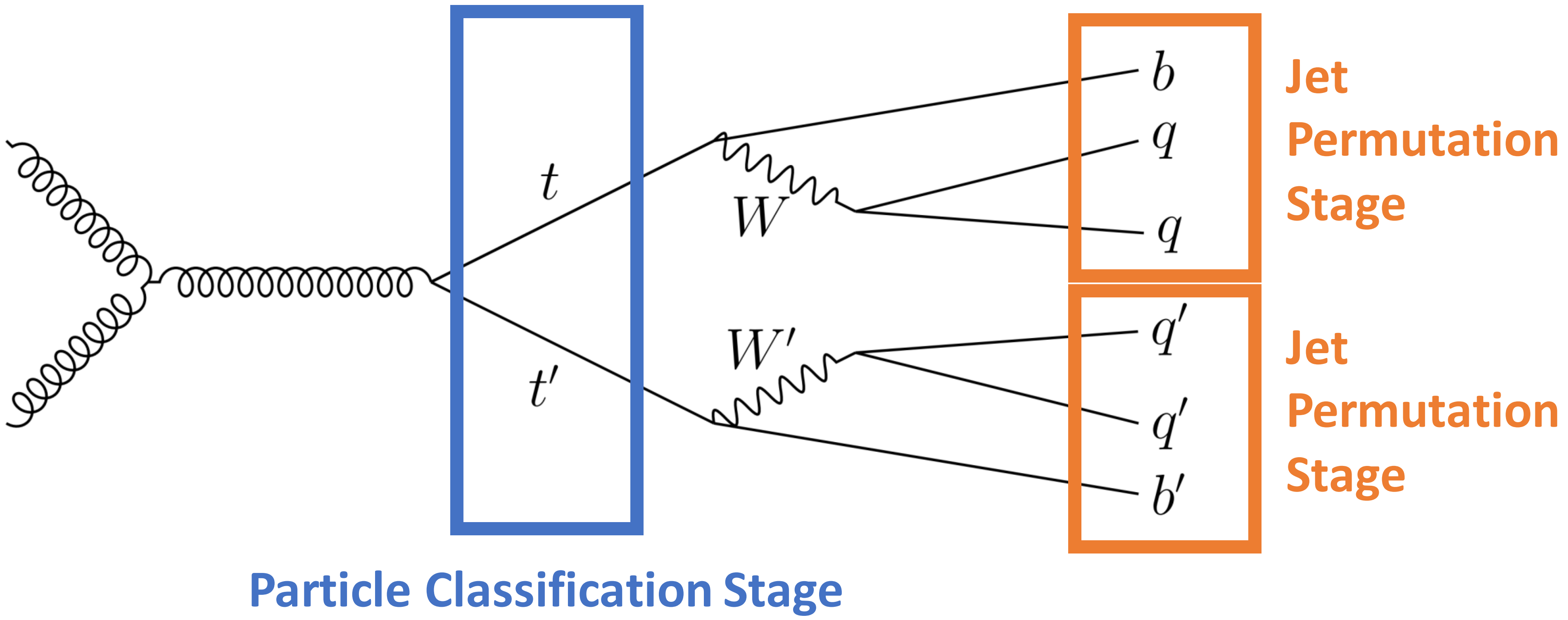
$$S^{i_1 i_2 \ldots i_K} = \sum_{\sigma \in G_P} \Theta^{i_{\sigma(1)} i_{\sigma(2)} \cdots i_{\sigma(K)}}$$

$$O^{j_1 j_2 \cdots j_K} = X_{i_1}^{j_1} X_{i_2}^{j_2} \ldots X_{i_K}^{j_K} S^{i_1 i_2 \ldots i_K}$$

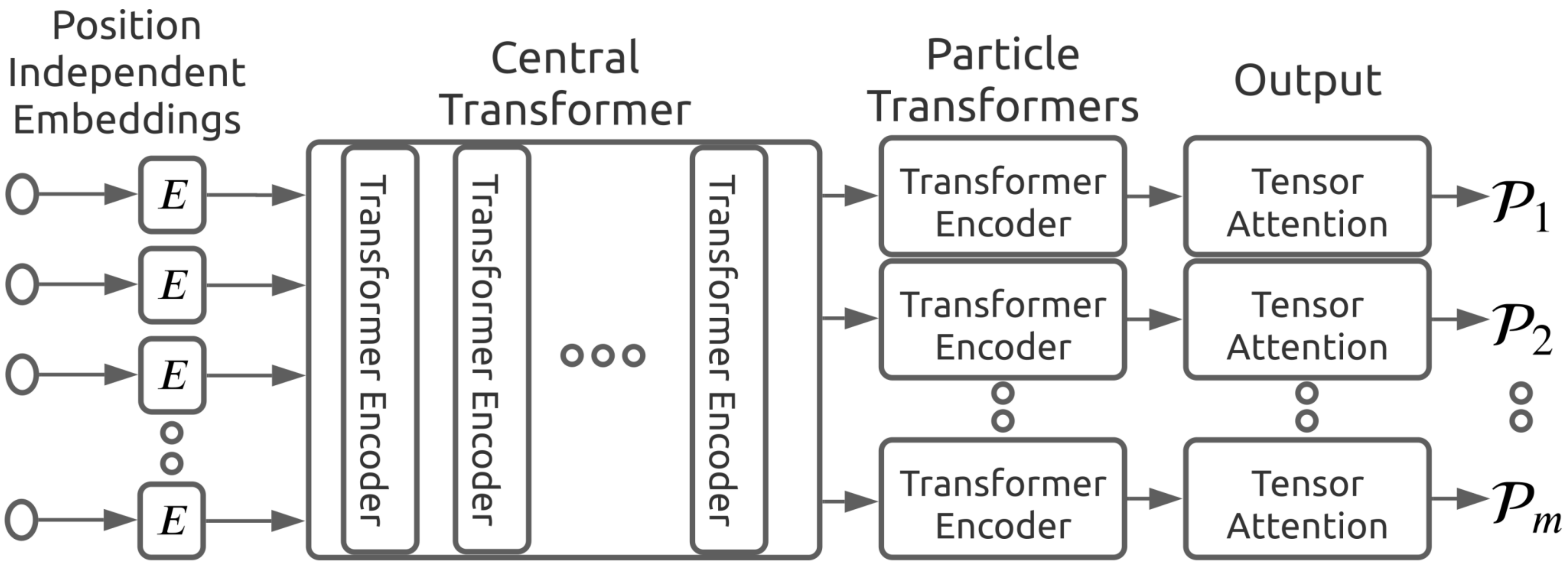
$$\mathcal{P}^{j_1 j_2 \cdots j_K} = \frac{\exp\left(O^{j_1 j_2 \cdots j_K}\right)}{\sum_{j_1, j_2, \ldots, j_K} \exp\left(O^{j_1 j_2 \cdots j_K}\right)}$$

SPANet A Combined Approach

Tensor Attention produces relatively small (2-4) dimensional joint distributions over jets.



SPANet Architecture



SPANet Training the Permutation Ranker

- Train symmetric joint-distributions using a simple categorical cross-entropy.
- **One special difference**
  - The target $\mathcal{T}$ is not a delta distribution!
  - $\mathcal{T}$ will be non-zero for every valid symmetric assignment.

$$\mathcal{L}_P(\mathcal{P}, \mathcal{T}) = \sum_{j_1, j_2, \cdots, j_N} -\mathcal{T}^{j_1 j_2 \cdots j_N} \log \mathcal{P}^{j_1 j_2 \cdots j_N}$$

SPANet Symmetric Training

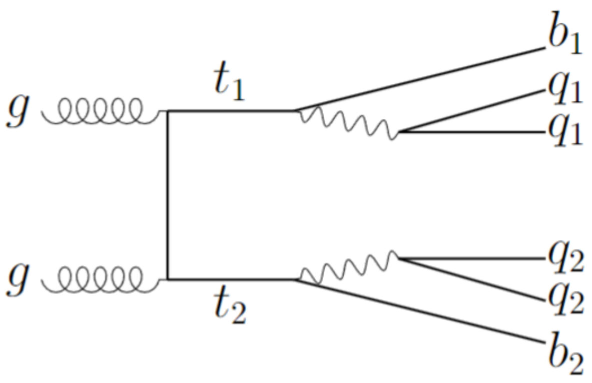
- Handle event symmetries using a symmetric loss function.
- Define an **event-level symmetry group** $G_E \subseteq \boldsymbol{S}_m$ acting on particles $\{P_1, P_2, \ldots, P_m\}$
- Loss function simply takes the **minimum** achievable loss over valid permutations.
- Also tried other methods such as *sum* of *soft-min*. Simple *min* works the best.

$$\mathcal{L} = \min_{\sigma \in G_E} \sum_{i=1}^{m} \mathcal{L}_P \left(\mathcal{P}_{\sigma(i)}, \mathcal{T}_{\sigma(i)} \right)$$

SPANet Results

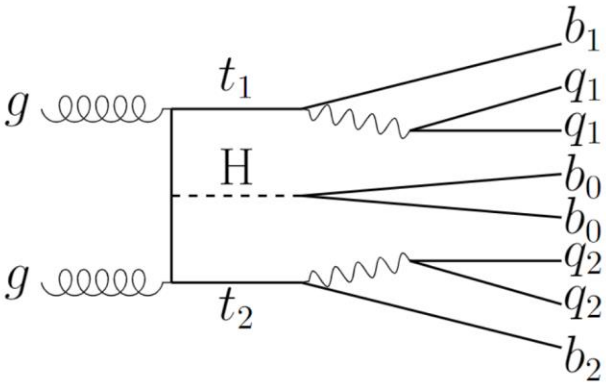
- Compare to a common baseline used in CERN analyses – A permutation-based χ^2 approach.
- Greatly improve accuracy over baseline methods. On average around ~25% improvement.
- Drastically increase runtime performance. Baseline method cannot tractably evaluate $tttt$!

$t\bar{t}$



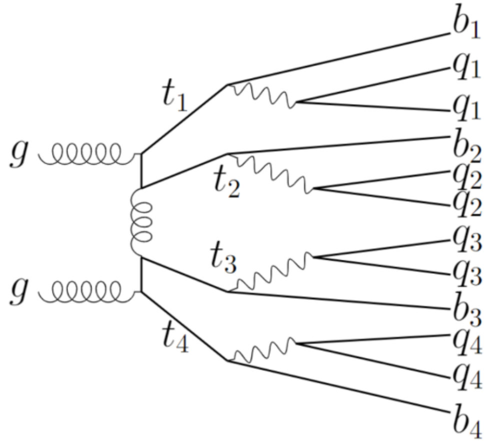
	N_{jets}	Event Fraction	SPA-NET Efficiency Event	SPA-NET Efficiency Top Quark	χ^2 Efficiency Event	χ^2 Efficiency Top Quark
All Events	== 6	0.245	0.643	0.696	0.461	0.523
	== 7	0.282	0.601	0.667	0.408	0.476
	≥ 8	0.320	0.528	0.613	0.313	0.395
	Inclusive	**0.848**	**0.586**	**0.653**	**0.387**	**0.457**
Complete Events	== 6	0.074	0.803	0.837	0.664	0.696
	== 7	0.105	0.667	0.754	0.457	0.556
	≥ 8	0.145	0.521	0.662	0.281	0.429
	Inclusive	**0.325**	**0.633**	**0.732**	**0.426**	**0.532**

ttH



N_{jets}	Event Fraction	SPA-NET Efficiency Event	SPA-NET Efficiency Higgs	SPA-NET Efficiency Top	χ^2 Efficiency Event	χ^2 Efficiency Higgs	χ^2 Efficiency Top
== 8	0.261	0.370	0.497	0.540	0.056	0.193	0.092
== 9	0.313	0.343	0.492	0.514	0.053	0.160	0.102
≥ 10	0.313	0.294	0.472	0.473	0.031	0.150	0.056
Inclusive	**0.972**	**0.330**	**0.485**	**0.502**	**0.045**	**0.164**	**0.081**
== 8	0.042	0.532	0.657	0.663	0.040	0.220	0.135
== 9	0.070	0.422	0.601	0.596	0.019	0.152	0.079
≥ 10	0.115	0.306	0.545	0.523	0.004	0.126	0.073
Inclusive	**0.228**	**0.383**	**0.583**	**0.572**	**0.016**	**0.153**	**0.087**

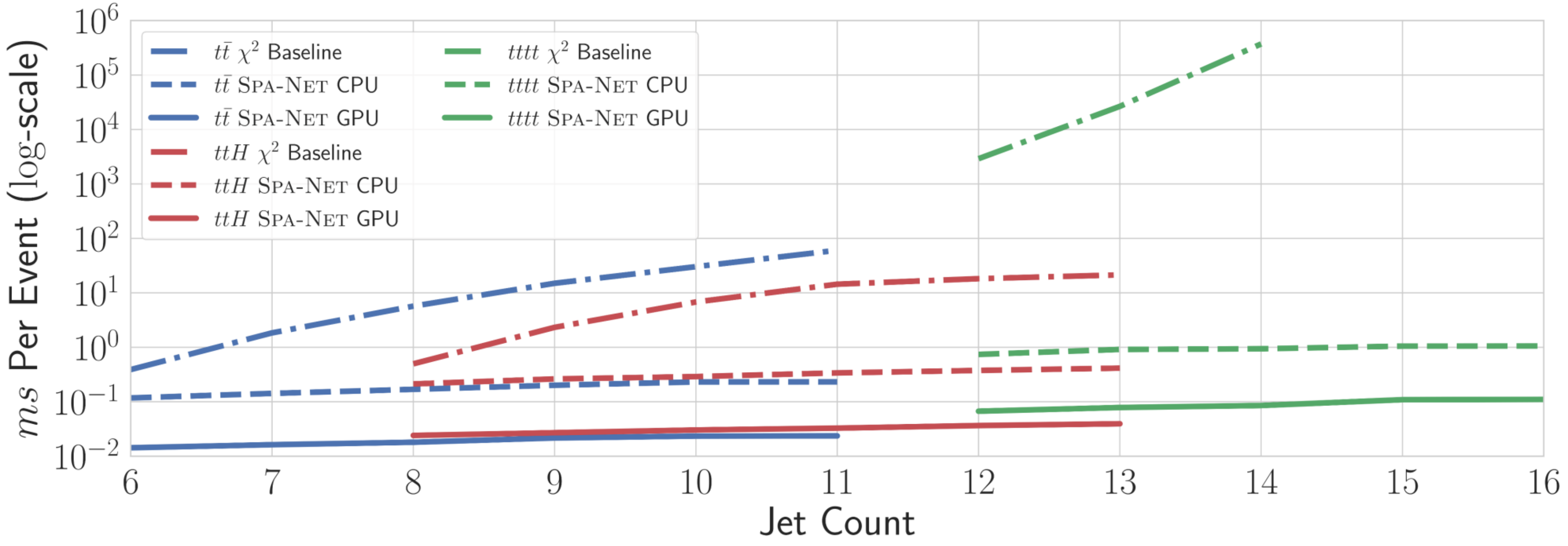
$tttt$



N_{jets}	Event Fraction	SPA-NET Efficiency Event	SPA-NET Efficiency Top Quark
== 12	0.219	0.276	0.484
== 13	0.304	0.247	0.474
≥ 14	0.450	0.198	0.450
Inclusive	**0.974**	**0.231**	**0.464**
== 12	0.005	0.350	0.617
== 13	0.016	0.249	0.567
≥ 14	0.044	0.149	0.504
Inclusive	**0.066**	**0.191**	**0.529**

SPANet Results

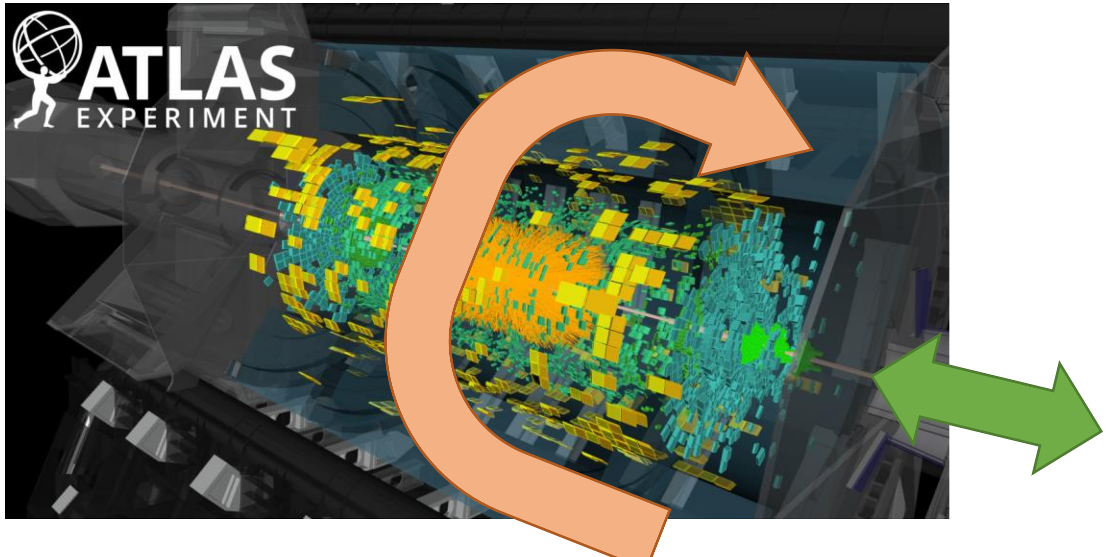
- Compare to a common baseline used in CERN analyses – A permutation-based χ^2 approach.
- Greatly improve accuracy over baseline methods. On average around ~25% improvement.
- Drastically increase runtime performance. Baseline method cannot tractably evaluate $tttt$!



Upcoming Work Continuous Input Symmetries

There are really three types symmetries in reconstruction.

- Mathematical permutation symmetry of sets
  - Handled with transformer attention.
- Discrete Physics symmetries such as CPT
  - Handled charge and parity symmetries with tensor attention and symmetric loss.
- Continuous Physics symmetries – **Lorentz symmetries** of space.
  - We can **rotate** or **flip** the entire detector and get the exact same event.
  - **Still present!**



Upcoming Work Continuous Input Symmetries

A **Reference Frame** is a conditional mapping $\phi(u; v)$ over the input vectors. In HEP, our transform is the **Lorentz boost,** which is preserved under a global Lorentz transform.

$$\phi(u; v) = \frac{1}{1 - v \cdot u}\left(\frac{u}{\gamma_v} - v + \frac{\gamma_v}{\gamma_v + 1}(u \cdot v)v\right)$$

Keys and values become **matrix** collections of vectors. Queries remain as lists of vectors.

$$\begin{aligned} Q &\in \mathbb{R}^{N \times D} && \textsc{Queries} \\ K &\in \mathbb{R}^{N \times N \times D} && \textsc{Keys} \\ V &\in \mathbb{R}^{N \times N \times D} && \textsc{Values} \end{aligned}$$

$K_{i,j} = \phi(k_j; k_i)$ represents the $j'th$ key **from** $\boldsymbol{i'}$**th reference frame**.
$V_{i,j} = \phi(v_j; v_i)$ represents the $j'th$ value **from** $\boldsymbol{i'}$**th reference frame**.
$Q_i = \phi(q_i; q_i)$ represents the $i'th$ query **from** $\boldsymbol{i'}$**th reference frame**.

Upcoming Work Continuous Input Symmetries

- Modified attention operation is nearly identically to regular attention.
- Just need to add some extra indices
- Output vector, O, is invariant to any global changes w.r.t the perspective operation.

$$S_{i,j} = \frac{Q_i \cdot K_{i,j}}{\sqrt{D}} \qquad \in \mathbb{R}$$

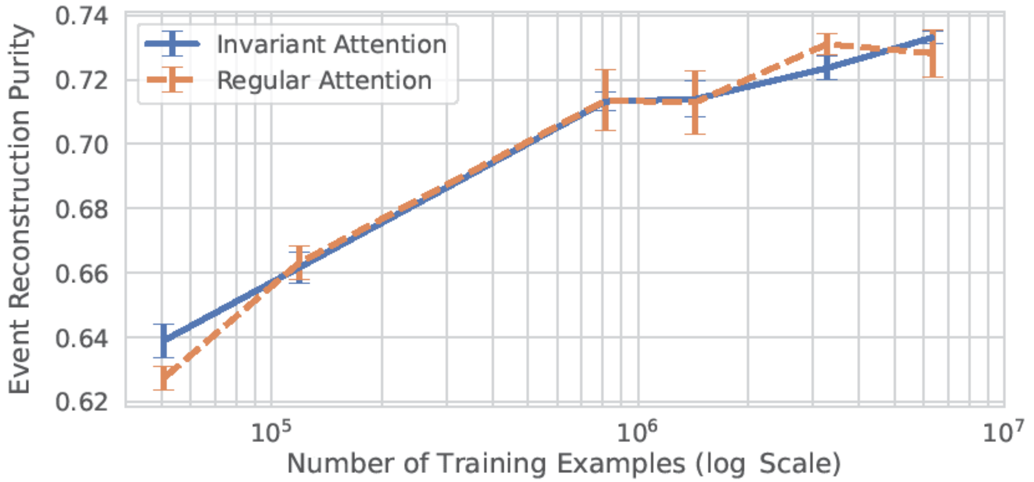
$$A_{i,:} = \textsc{Softmax}(S_{i,:}) \qquad \in \mathbb{R}^N$$

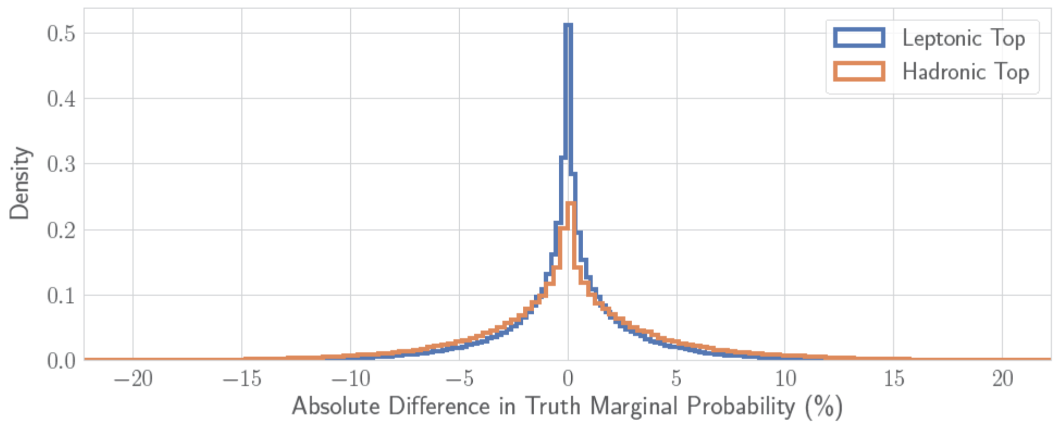
$$O_i = \sum_j (A_{i,j} \odot V_{i,j}) \qquad \in \mathbb{R}^D$$

A trivial reference frame function, $\phi(u; v) = u$, reduces this operation regular scaled dot-product attention.

Upcoming Work Continuous Input Symmetries

Almost no improvement in performance!





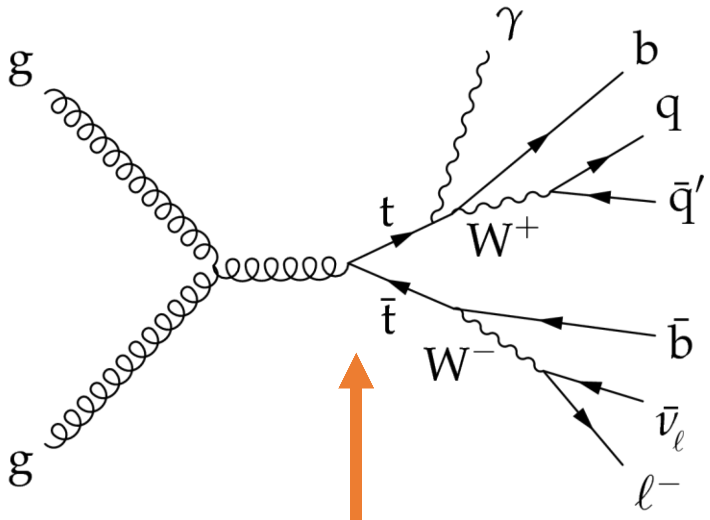
What's going on?

- The transformation function is just a *non-linear conditional function* of the original input.
- This is precisely a transformer!
- If Lorentz invariance is useful, then the network **should just learn it.**

Let's examine how much our assignment probability changes as you feed the **same event** rotated in several ways.

Even without Lorentz invariant attention, SPANet is **approximately invariant**!

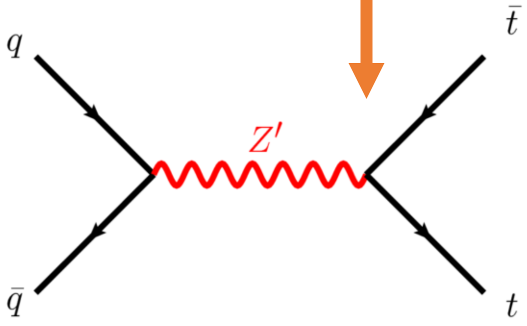
Upcoming Work Search for New Particles



Background
Standard model semi-leptonic $t\bar{t}$ event.

Events also have an additional *neutrino* term which needs to be reconstructed separately.

Need to separate events into a *signal* and a *background* class depending on if the hypothetical particle was present in the event.

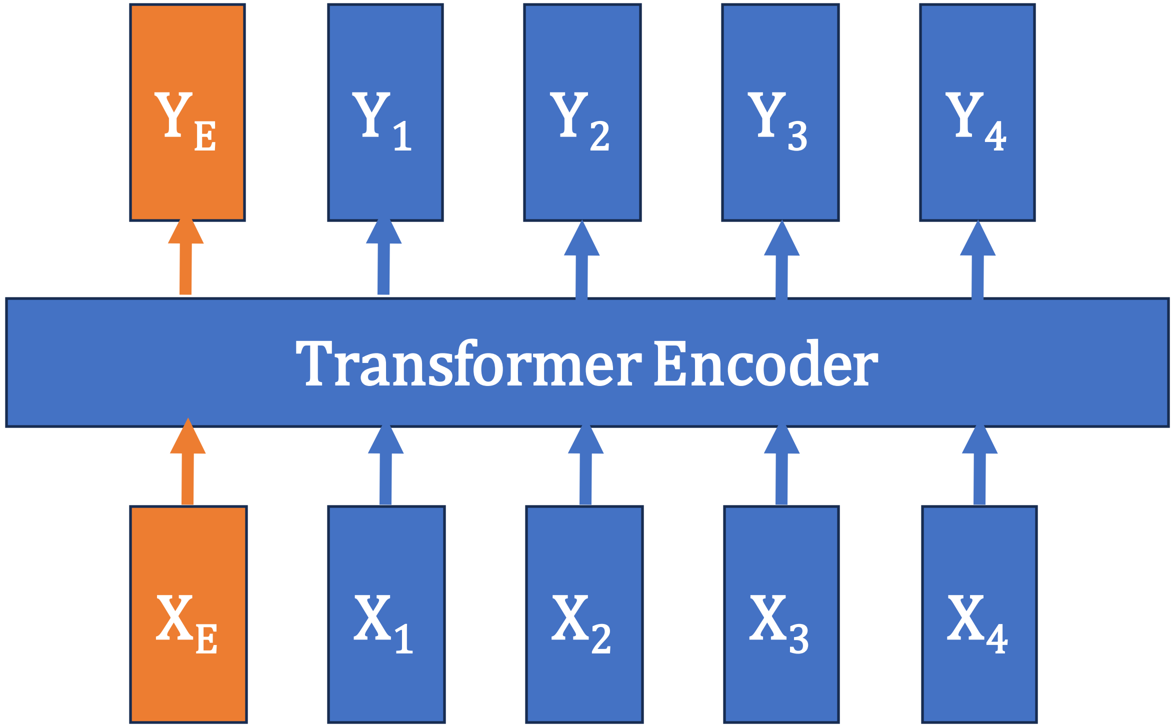


Signal
Hypothetical modified event with a non-standard-model Z' Boson before top decay.

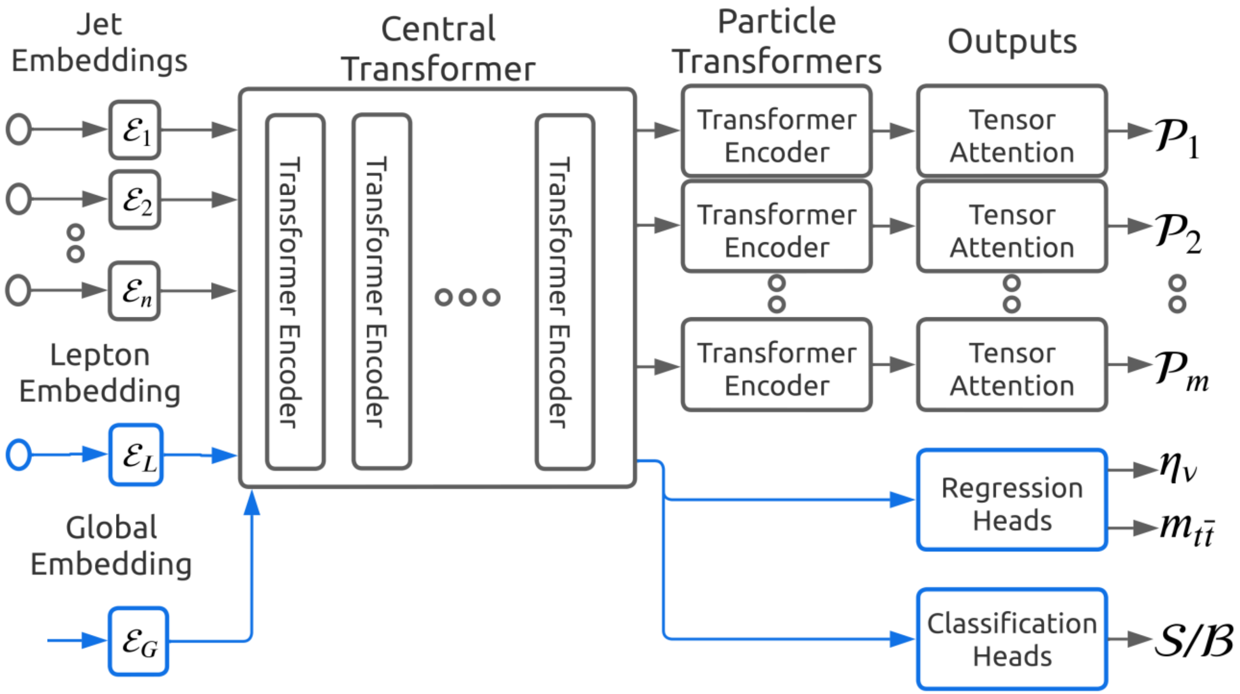
Upcoming Work Search for New Particles

We can add a special **Event-Level** output to attention with a neat trick.

Add a special *fixed* vector to our input stream, X_E, which will represent the *event* information. This special vector will have the same value for all events. The transformer's contextual learning will fill Y_E with relevant event information.



Upcoming Work Search for New Particles



Combined Training
Train SPANet on simulated data with the hypothetical particle present and not present. This will ensure reasonable performance regardless of what the real data holds.

Regress extra neutrino terms

Classify event into signal or background

Trainable Global Vector
Will store information about the complete event.

Global Outputs
Use the encoded global vector to predict additional event-level terms.

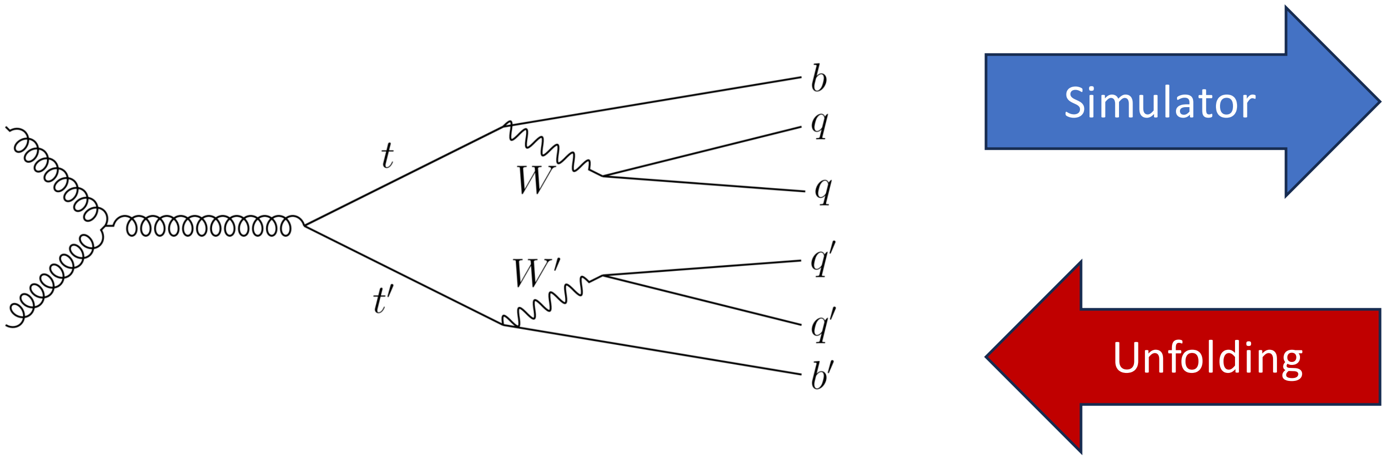
Upcoming Work Unfolding

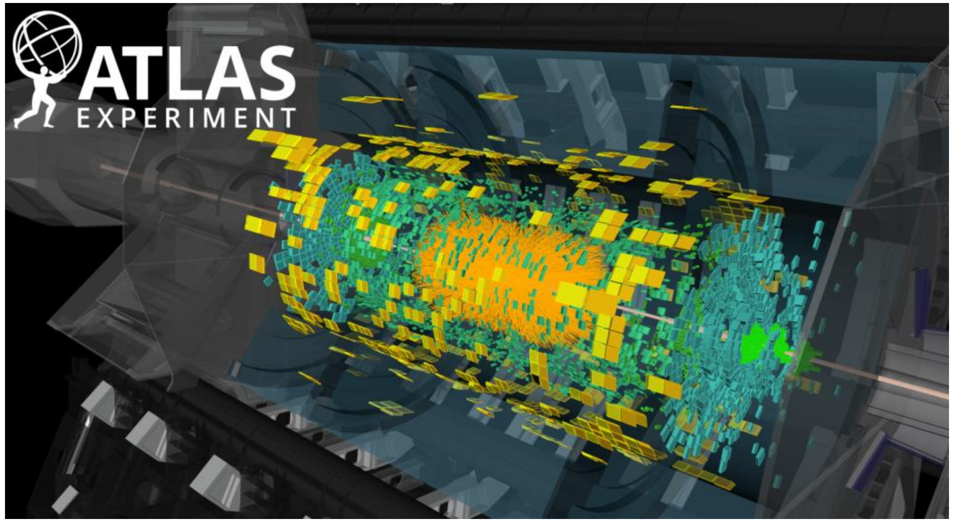
So far, we have been **assigning** jets to source partons. What if we could do more?

Unfolding Fully recovering the momenta of the source partons from observations.

Simulators (Madgraph, Delphes, etc) define the forward problem:
Parton -> Detector

We want to solve the inverse problem:
Detector -> Parton

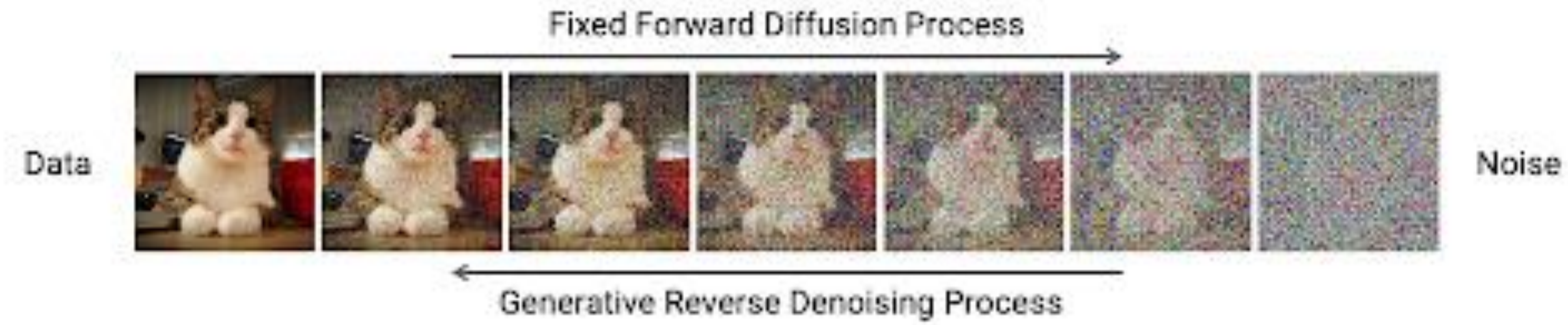




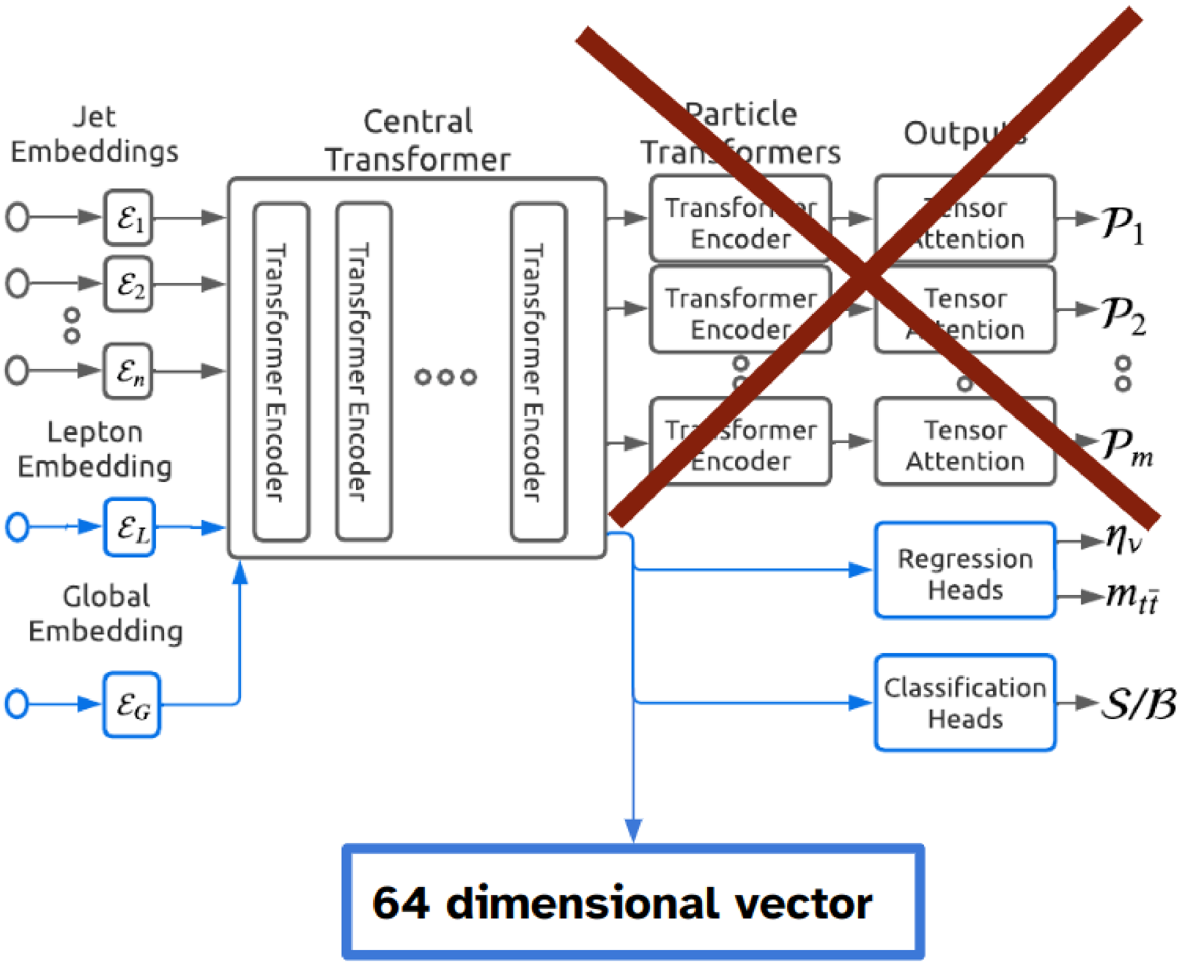
Upcoming Work Unfolding

We borrow a popular idea from machine learning: **Diffusion.**
Generate complex distributions *conditioned* on another observation.

Based on a principle of *denoising:* generate new samples by reversing gradual noise.



Upcoming Work Unfolding



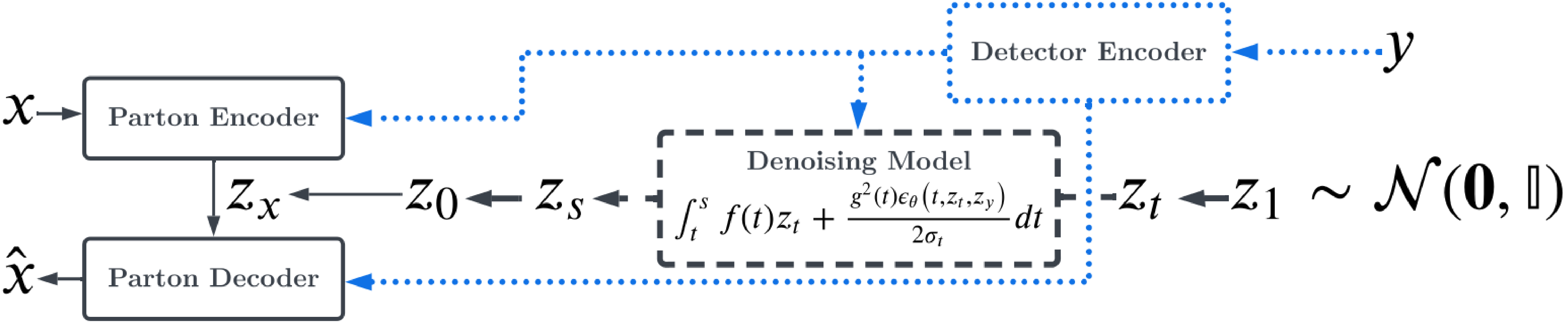
Detector Encoder

Use the special event-level output from SPANet to get an abstract conditional latent vector.

Convert a complex variable-length observation into a fixed-length vector to use for diffusion.

Upcoming Work Unfolding

The whole unfolding framework will generate Parton configurations conditioned on this event vector.



Upcoming Work **Unfolding**