Pictured: Best EFN Latent Space (Left); Best k=4 Moment EFN Latent Space (Right) for quark/gluon discrimination



Moment Pooling:

Gaining Performance and Interpretability Through Physics Inspired Product Structures

Rikab Gambhir

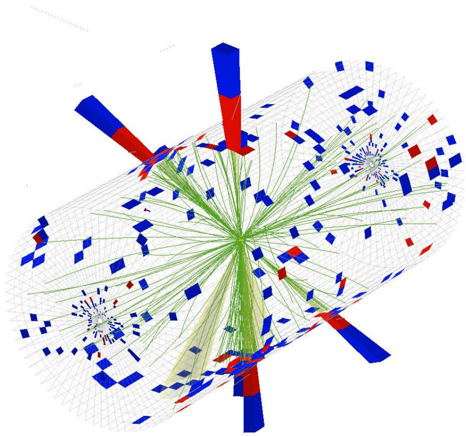
With Athis Osathapan and Jesse Thaler

Email me questions at rikab@mit.edu!

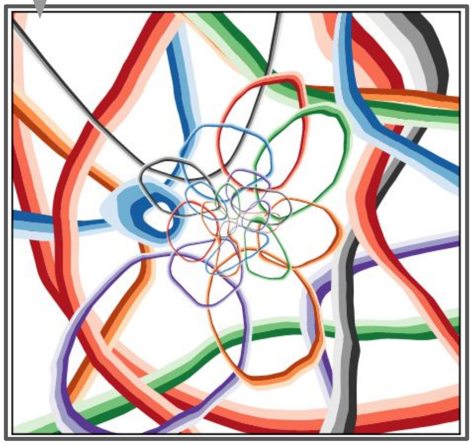
Based on [**RG**, Osathapan, Thaler, 23XX.XXXX]

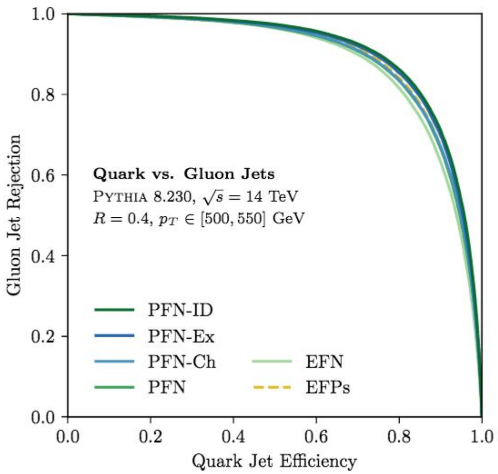
[Komiske, Metodiev, Thaler, 1810.05165]

Typical **Machine Learning** Setup



Contours of the L different latent representations, ϕ





Collider Data → **Latent Space** → **Observables**

~1000 Dimensional | ~10-100 Dimensional | ~1-10 Dimensional

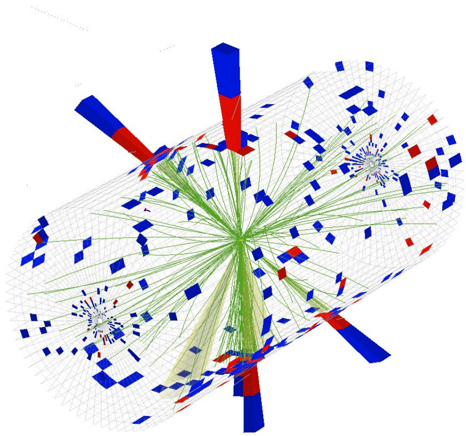
Pictured: An Energy Flow Network (**EFN**):

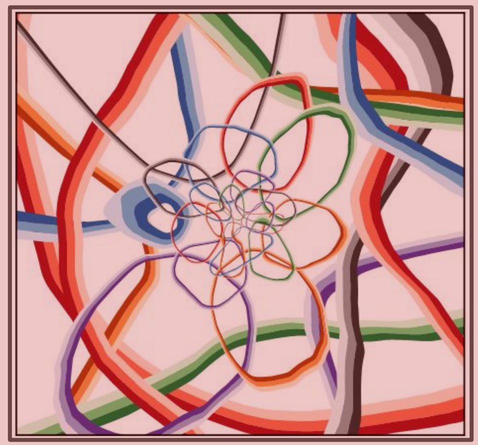
Particle weight: Energy Fraction

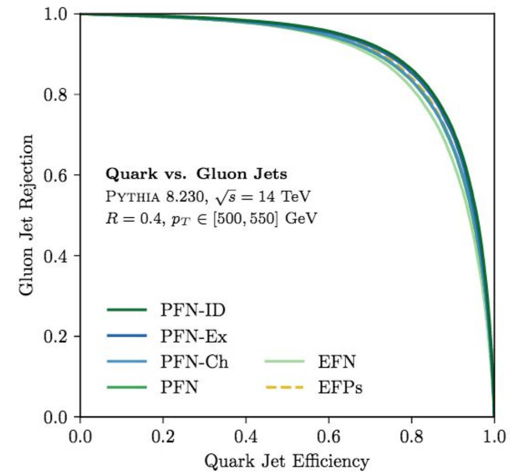
$$\mathcal{O}(\{p_1, \ldots, p_M\}) = F\left(\sum_{i=1}^{M} z_i \Phi(\hat{p}_i)\right)$$

L-dimensional latent representation, per particle

Typical Machine Learning Setup







Collider Data → **Latent Space** → **Observables**

~1000 Dimensional | ~10-100 Dimensional | ~1-10 Dimensional

Pictured: An Energy Flow Network (**EFN**):

$$\mathcal{O}(\{p_1, \ldots, p_M\}) = F\left(\sum_{i=1}^{M} z_i \Phi(\hat{p}_i)\right)$$

(How) can we understand and constrain this?

(How) can we be more efficient?

The Moment-EFN

$$\mathcal{O}(\mathcal{P}) = F\left(\langle\phi^a\rangle_\mathcal{P}\right)$$

EFNs* can be thought of as taking the (weighted) **mean** of a latent particle representation ϕ – Let's generalize to *any* **moment**!

Generalize!

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\langle\phi^a\rangle_\mathcal{P}, \langle\phi^{a_1}\phi^{a_2}\rangle_\mathcal{P}, \ldots, \langle\phi^{a_1}\ldots\phi^{a_k}\rangle_\mathcal{P}\right)$$

This is a natural way to encode **multiplication** of distributions in neural nets.

Hope: This "Moment-EFN" might give more efficient representations?!

*Most of what I say here today also applies to Particle Flow networks or any other Deep-Sets inspired architecture!

The **Moment-EFN** (Details)

Set of momenta

a = 1 … *L*, the *Latent Dimension* index

$$\mathcal{O}(\mathcal{P}) = F\left(\langle \phi^a \rangle_{\mathcal{P}}\right)$$

$$\langle \phi \rangle_{\mathcal{P}} \equiv \sum_i z_i \phi(\hat{p}_i)$$

The **Deep Sets Theorem** guarantees that any function on sets $\mathscr{P}$ can be written this way, for "sufficiently complex" *F* and ϕ

Generalize!

k = Highest order moment considered

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\underbrace{\langle \phi^a \rangle_{\mathcal{P}}}_{\text{1st Moment}}, \underbrace{\langle \phi^{a_1}\phi^{a_2} \rangle_{\mathcal{P}}}_{\text{2nd Moment}}, \dots, \underbrace{\langle \phi^{a_1}\dots\phi^{a_k} \rangle_{\mathcal{P}}}_{k\text{th Moment}}\right)$$

$$L_{\text{eff}} = \frac{k+1}{L}\binom{k+L}{k+1}$$

Hope: This "Moment-EFN" might give more efficient representations?!

The **Moment-EFN** (Details)

More precisely …

"More efficient representation" means ϕ, *F* could be…

1. Simpler elementary functions? *e.g. linear* ϕ, *F*
2. More simply parameterized? *i.e. Fewer total parameters*
3. Easier to embed? *i.e. Smaller L, fewer* ϕ *functions*

that any
way, for

k = Highest order moment considered

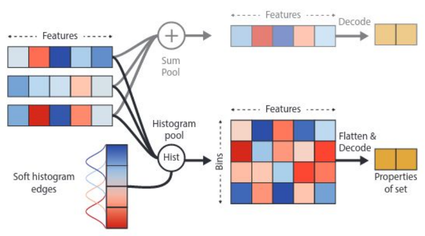
$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\langle\phi^{a}\rangle_{\mathcal{P}}, \langle\phi^{a_1}\phi^{a_2}\rangle_{\mathcal{P}}, \ldots, \langle\phi^{a_1}\ldots\phi^{a_k}\rangle_{\mathcal{P}}\right)$$

1st Moment — 2nd Moment — kth Moment

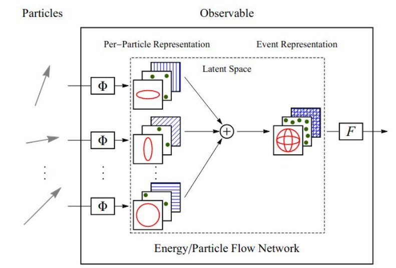
$$L_{\text{eff}} = \frac{k+1}{L}\binom{k+L}{k+1}$$

Hope: This "Moment-EFN" might give more efficient representations?!

NSF AI Institute for Artificial Intelligence & Fundamental Interactions



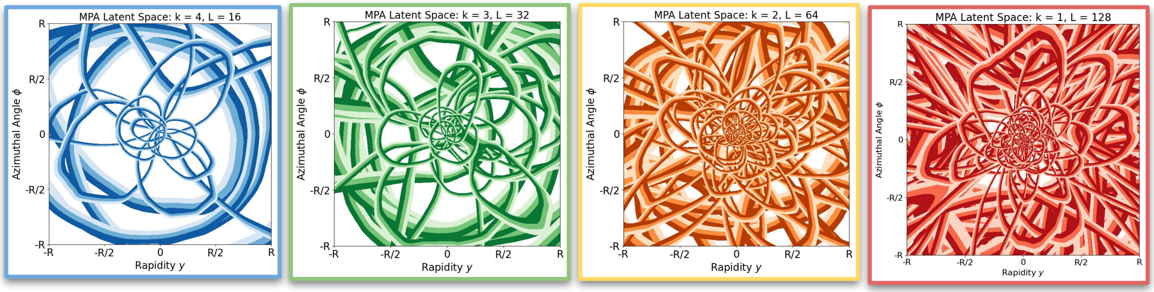
Extending Deep Sets to Distributions
[Histogram Pooling, M. Cranmer et. al.]



IRC-safe Neural Networks
[Energy Flow Networks, 1810.05165]

Moment-EFNs

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\langle\phi^a\rangle_{\mathcal{P}}, \langle\phi^{a_1}\phi^{a_2}\rangle_{\mathcal{P}}, \ldots, \langle\phi^{a_1}\ldots\phi^{a_k}\rangle_{\mathcal{P}}\right)$$



Same Information

Using summary statistics of energy distributions within events to improve information per latent dimension

e.g. Jet Angularities

In the moment language, even integer* β jet angularities ⇔ $k = \beta^{\text{th}}$ moments!

$$\lambda^{(\beta)}(\mathcal{P}) = \sum_i z_i \left(\eta_i^2 + \phi_i^2\right)^{\beta/2}$$
$$= \langle \eta^\beta \rangle_{\mathcal{P}} + \langle \phi^\beta \rangle_{\mathcal{P}} + \text{ Cross Moments}$$

For the normal (k = 1) EFN, this would require learning nonlinear functions!

Test: Train three networks to regress $\lambda^{(2)}$ from 100k QCD jet samples, with a latent dimension L:

- Linear Network: ϕ, F are 1 layer, linear functions, $L = 2$
- Small Network: ϕ, F are 2 layers, each with 4 nodes and *LeakyReLU*, $L = 2$
- "Large" Network: ϕ, F are 3 layers, each with 32 nodes and *LeakyReLU*, $L = 8$

Expect $k = 2$ to outperform $k = 1$ for smaller networks!

*Ask later about non-even or non-integer β angularities

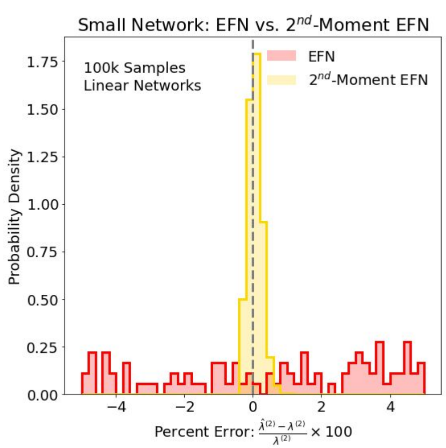
e.g. Jet Angularities

In the moment language, even integer* β jet angularities ⇔ $k = \beta^{\text{th}}$ moments!

$$\lambda^{(\beta)}(\mathcal{P}) = \sum_i z_i \left(\eta_i^2 + \phi_i^2\right)^{\beta/2}$$
$$= \langle \eta^\beta \rangle_{\mathcal{P}} + \langle \phi^\beta \rangle_{\mathcal{P}} + \text{ Cross Moments}$$

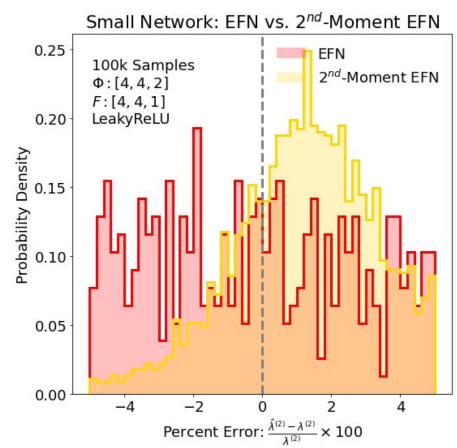
For the normal (k = 1) EFN, this would require learning nonlinear functions!

Linear Networks



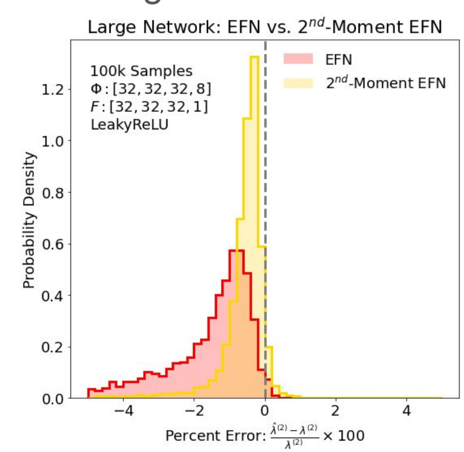
~10 Parameters

Small Networks



~100 Parameters

Large Networks



~6000 Parameters

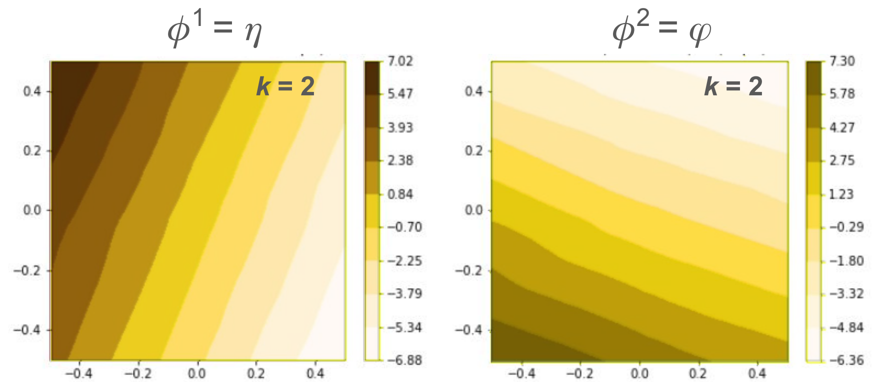
Training times are identical for k =*1* and *2*!

e.g. **Jet Angularit**

In the moment language, even i

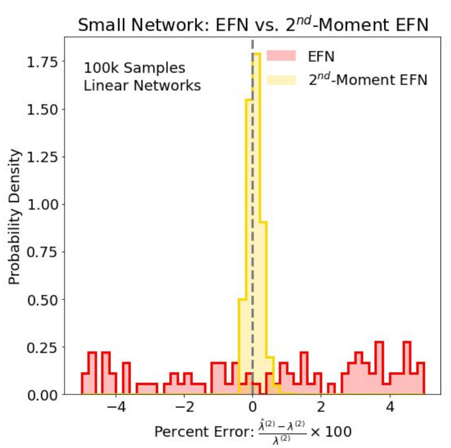
$$\lambda^{(\beta)}(\mathcal{P}) = \sum_i z_i \left(\eta_i^2 + \phi_i^2\right)^{\beta/2}$$
$$= \langle \eta^\beta \rangle_{\mathcal{P}} + \langle \phi^\beta \rangle_{\mathcal{P}} + \text{ Cro}$$

Learns the simplest latent representations!



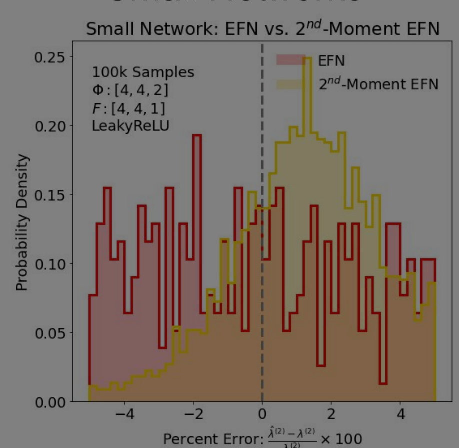
Encourage learning a diagonal basis in (η,φ) using L1 Regularization on F

Linear Networks



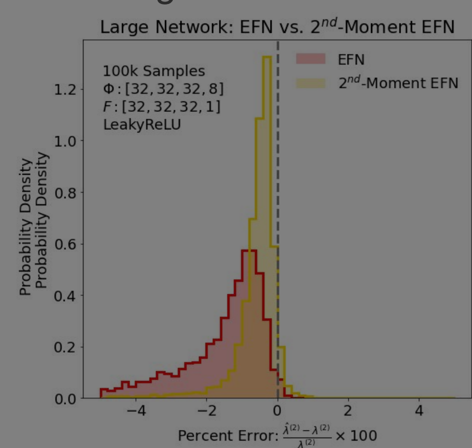
~10 Parameters

Small Networks



~100 Parameters

Large Networks



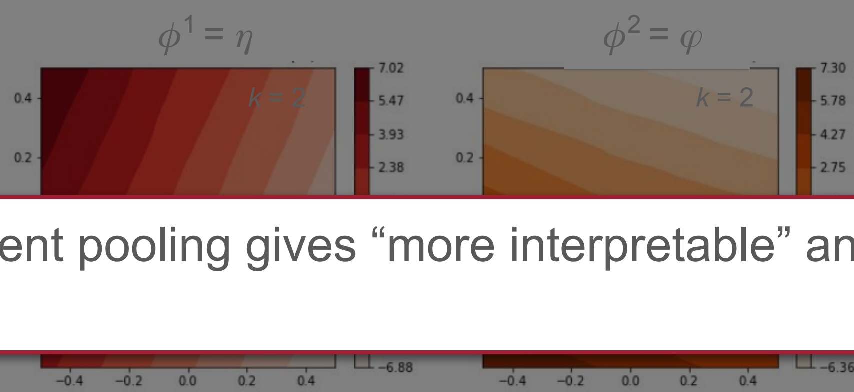
~6000 Parameters

Training times are identical for k =1 and 2!

e.g. Jet Angularit

In the moment language, even i

Learns the simplest latent representations!



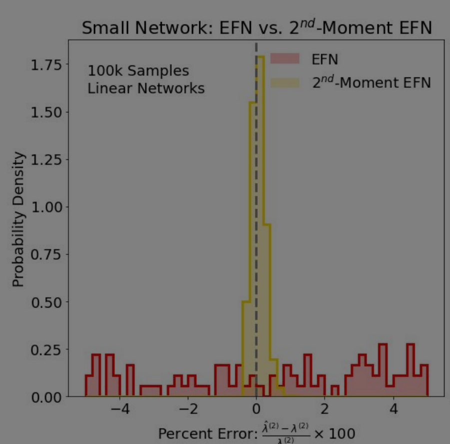
Encourage learning a diagonal basis in (η,φ) using L1 Regularization on F

… At least *one* case where moment pooling gives “more interpretable” and “more accurate” networks! ✓

λ

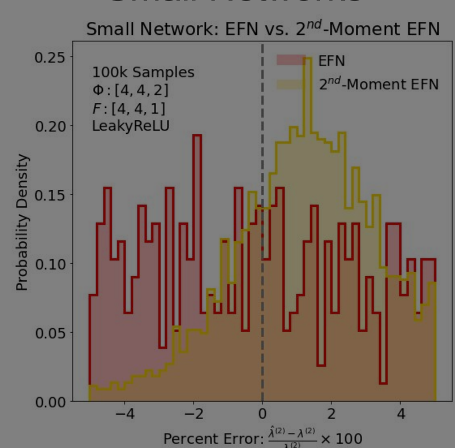
$= \langle \eta^2 \rangle_{\mathcal{P}} + \langle \varphi^2 \rangle_{\mathcal{P}} +$ Cro

Linear Networks



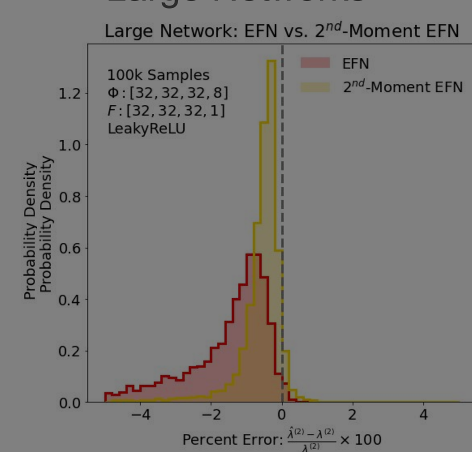
~10 Parameters

Small Networks



~100 Parameters

Large Networks



~6000 Parameters

Training times are identical for k =1 and 2!

A more **complex task …**

In principle, with a large enough k, we can approximate any* observable with a **linear** F – just like we did with angularities!

Is it possible to simplify complex observables, like a Q/G discriminant**, into linear F networks with just a few powers of k?

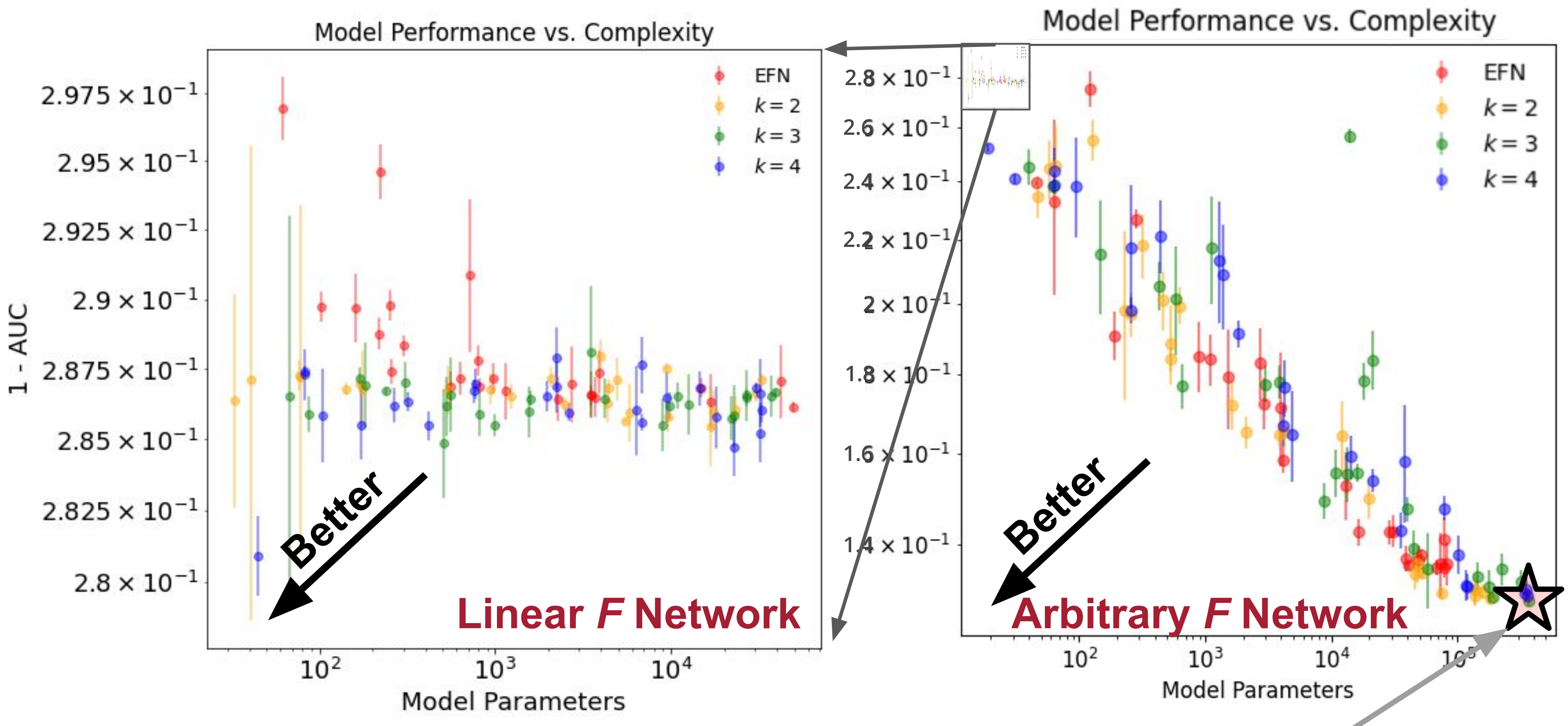
$$\begin{aligned}\mathcal{O}(\mathcal{P}) \approx F_a \langle \phi^a \rangle_{\mathcal{P}} \\ + F_{a_1 a_2} \langle \phi^{a_1} \phi^{a_2} \rangle_{\mathcal{P}} \\ + F_{a_1 a_2 a_3} \langle \phi^{a_1} \phi^{a_2} \phi^{a_3} \rangle_{\mathcal{P}} \\ + \ldots\end{aligned}$$

Train many Q/G of different sizes for different values of k, and see if $k > 1$ can be used to build linear functions, with less parameters, and with a smaller latent dimension!

*For well-behaved functions

**For classification tasks, we linearize the log likelihood and apply a sigmoid or softmax at the end

Quark/Gluon Discrimination



Saturates at AUC ~ 0.88, consistent with 1810.05165!

See backup slides for training details and dataset details.
If we have time – see backup for performance versus latent dimension! Same performance for *lower* latent dimensions!

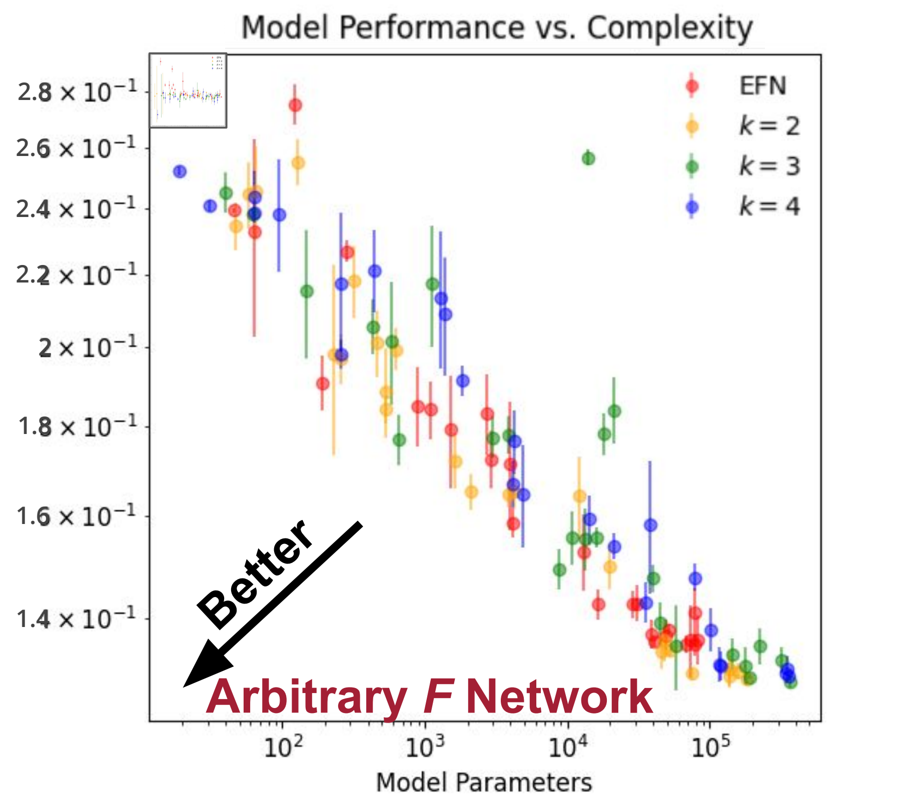
Quark/Gluon Discrimination

Quark/Gluon discriminators are inherently complex! We *can't* reduce them to linear functions* like with angularities – some problems are irreducibly hard.

Moments also don't reduce the number of necessary model parameters – "information is conserved" in difficult problems!

... But we can still do better!

*Not all simple functions are ruled out, e.g. Padé approximants on *F*



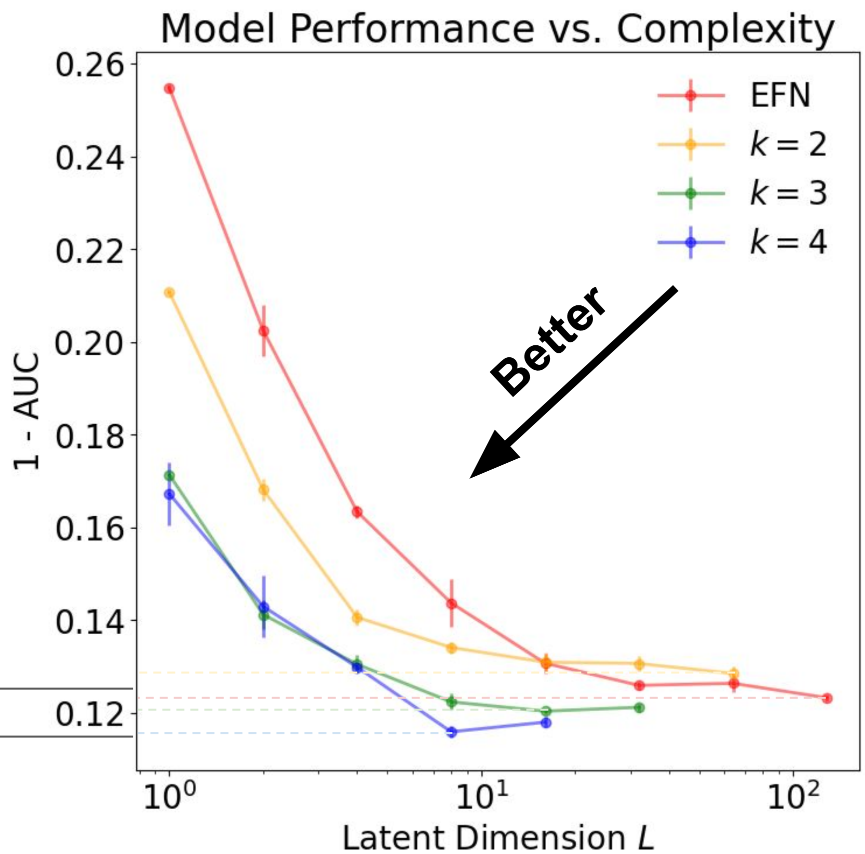
Saturates at AUC = 0.88, consistent with 1810.05165!

See backup slides for training details and dataset details.
If we have time – see backup for performance versus latent dimension! Same performance for *lower* latent dimensions!

Quark/Gluon **Latent Spaces**

Let's look at the **latent space embeddings**!

You get ***much*** better performance at a given *L*, especially at low *L*



As *k* goes up, you can get away with a ***much*** lower dimensional ϕ embedding!

Better peak performance!

This was visible in the previous plots too, but much easier to see here

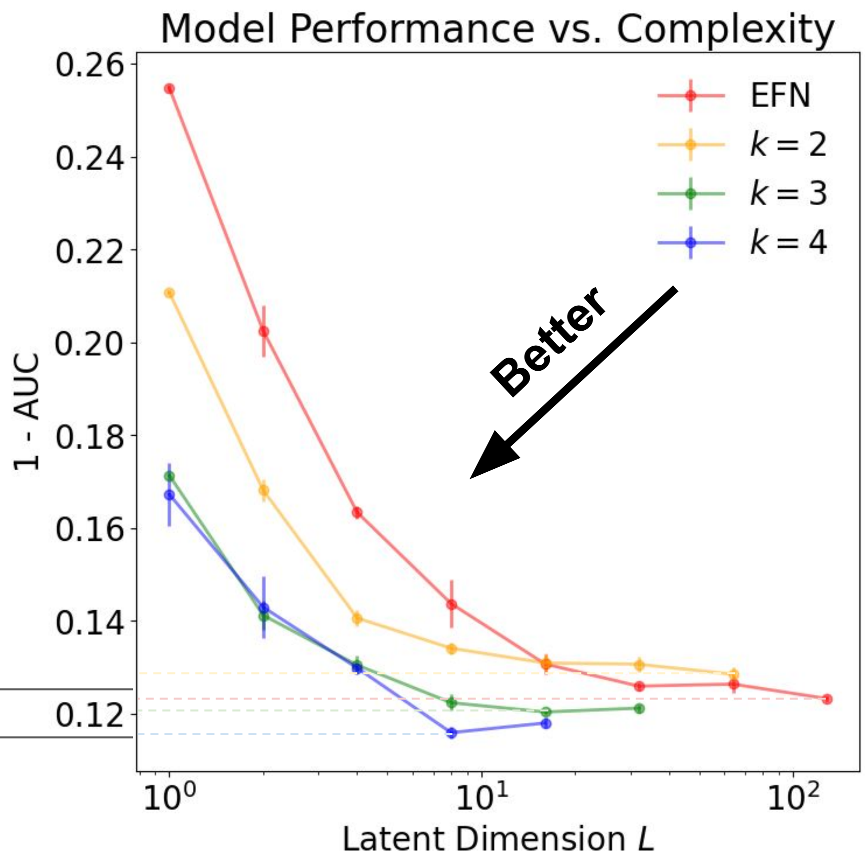
Quark/Gluon **Latent Spaces**

Let's look at the **latent space embeddings**!

You get ***much*** better performance at a given L, especially at low L

Better peak performance!

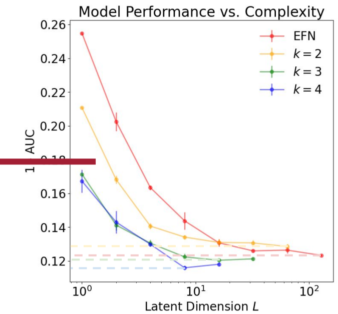
This was visible in the previous plots too, but much easier to see here



As k goes up, you can get away with a ***much*** lower dimensional ϕ embedding!

The same information is being more **compactly encoded**, completely losslessly!

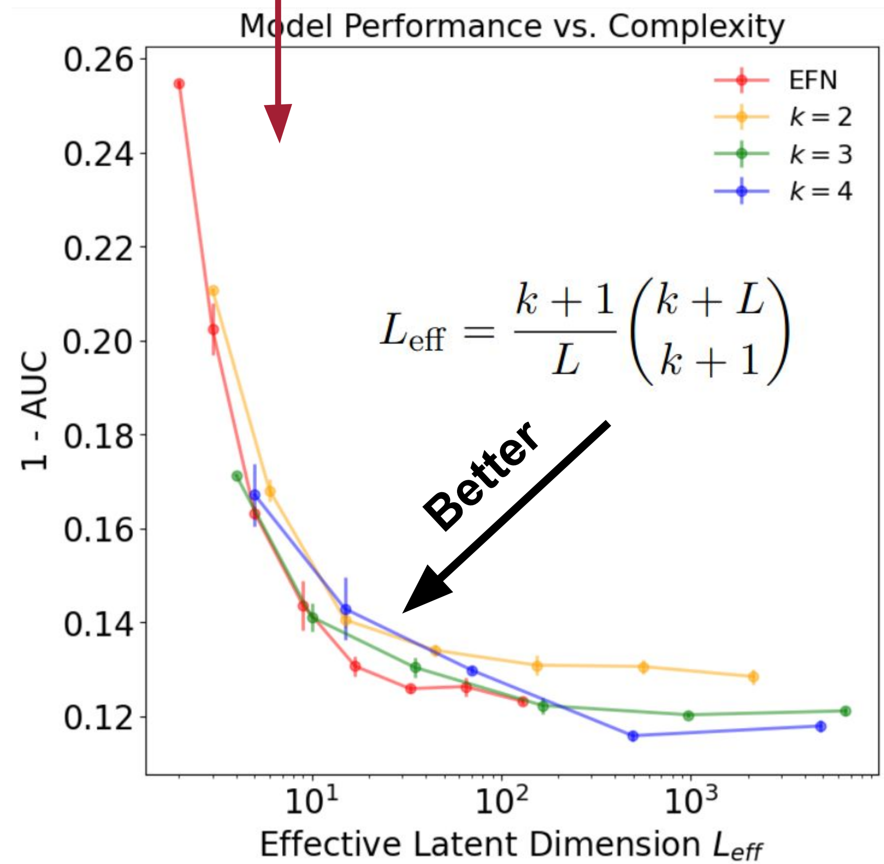
Aside, if we have time:
The Effective Latent Space



The moment structure allows for information to be more efficiently encoded and decoded.

This encoding and decoding can be incredibly complicated – moments help by allowing neural networks to do multiplication!

The complexity is reflected in the very large **effective latent dimension** – the number of combinations of all moments.



The info stored in a tiny L can be unraveled to L_{eff}, but info is conserved!

Quark/Gluon Latent S

Let's look at the **latent space embeddin**

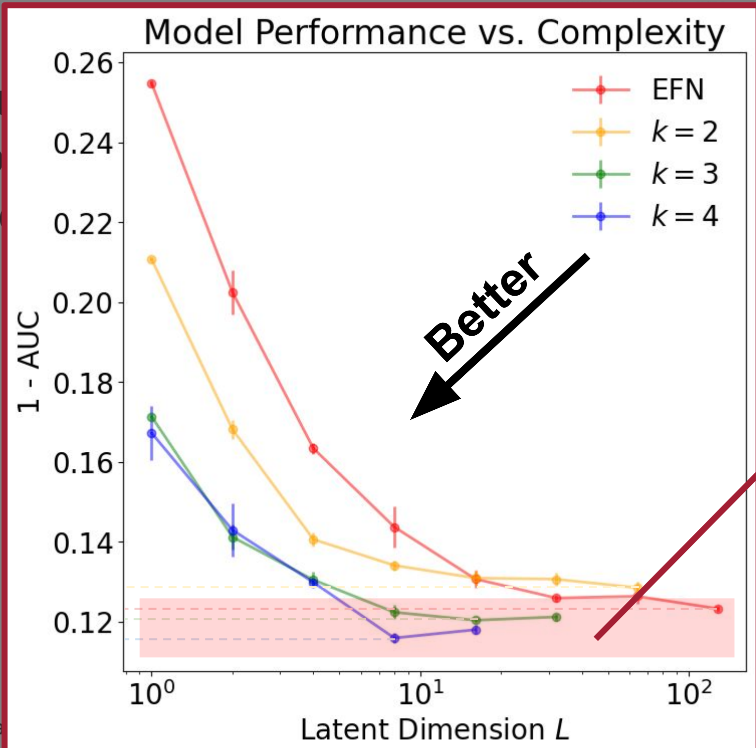
form

You

p

giv

This wa

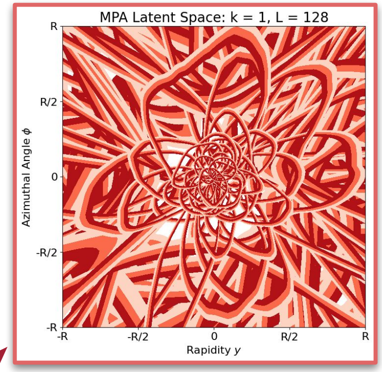


10^0

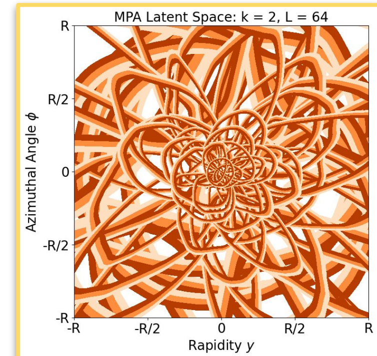
Latent

Latent Spaces

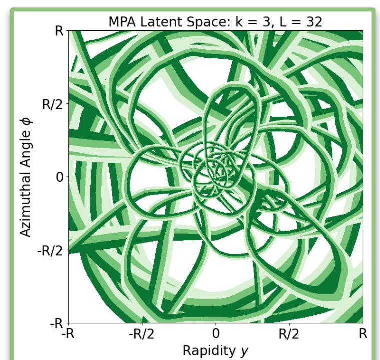
EFN (k = 1)
L = 128



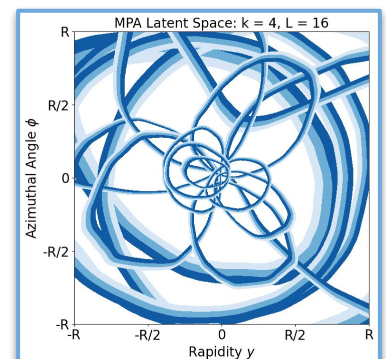
k = 2
L = 64



k = 3
L = 32



k = 4
L = 16



The 45% - 55% contours of each of the L different ϕ functions

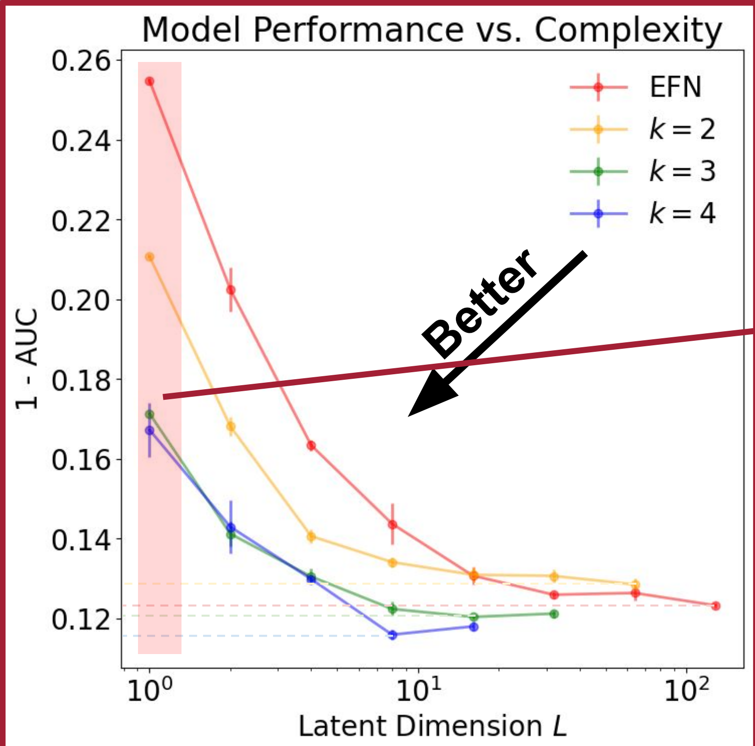
Quark/Gluon Latent S

Let's look at the **latent space embeddin**

Yo
p
giv

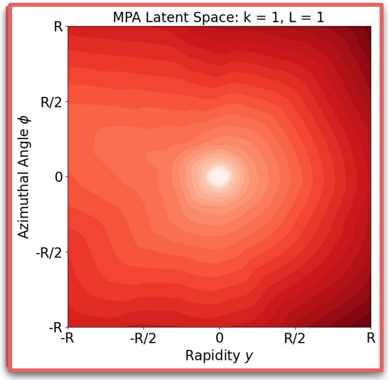
form

This wa

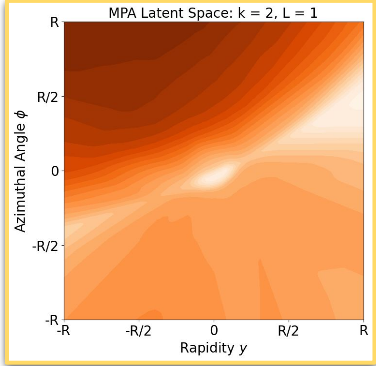


Latent Spaces

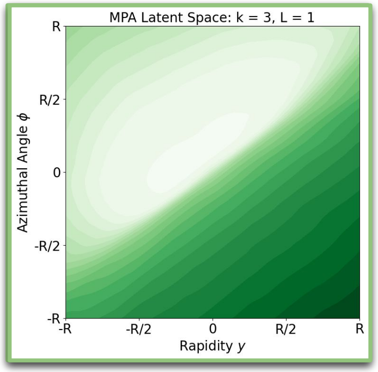
EFN (k = 1)
L = 1



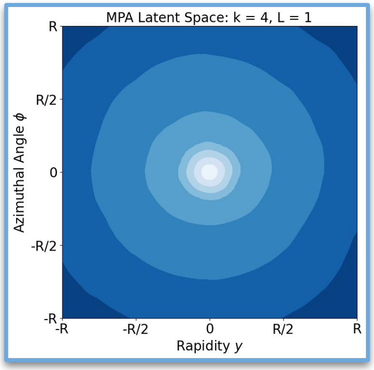
k = 2
L = 1



k = 3
L = 1



k = 4
L = 1



The contours of each of the L different ϕ functions

The Moment(s) of Truth*

Just a single (radially symmetric!) function $\phi \sim f(r) \sim \log(r)$

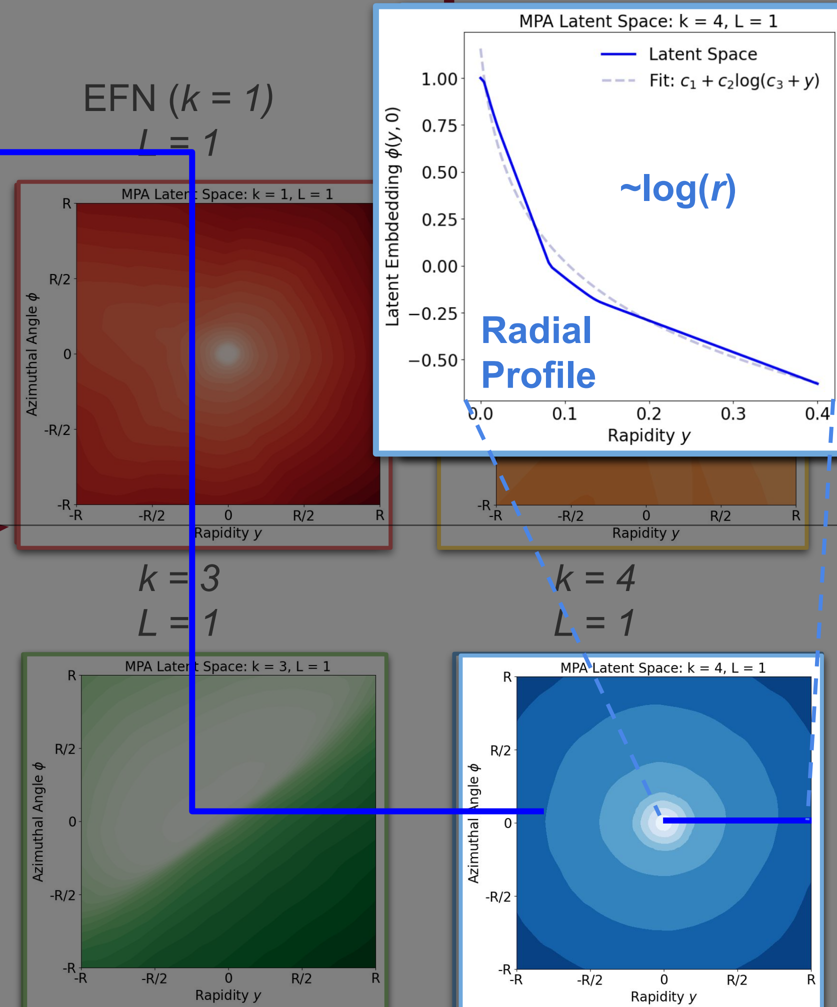
Simply compute the average values of $\phi^1, \phi^2, \phi^3, \phi^4$ on each event (akin to angularities), and feed these 4 numbers** through a simple neural net F.

This alone is enough to get an IRC-safe quark-gluon classifier with an AUC ~ 0.83!

*I apologize if I have repeated this same joke in other contexts during this talk. It would have been an even better joke if I had used top samples here.

**Interestingly, an EFN with $L=4$ has roughly the same performance, suggesting it learns something equivalent to this one function

Latent Spaces



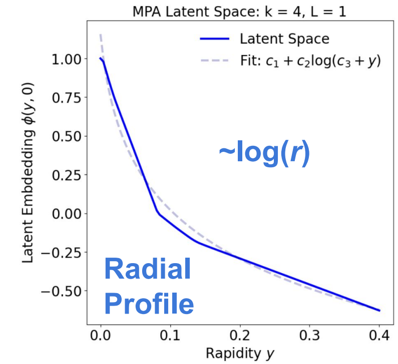
The contours of each of the L different ϕ functions

Email me questions at rikab@mit.edu!

Conclusion

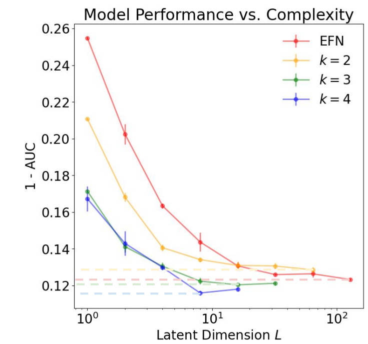
Hope: The “Moment-EFN” gives more efficient representations!

$$\mathcal{O}_k(\mathcal{P}) = F_k\left(\langle\phi^a\rangle_\mathcal{P}, \langle\phi^{a_1}\phi^{a_2}\rangle_\mathcal{P}, \ldots, \langle\phi^{a_1}\ldots\phi^{a_k}\rangle_\mathcal{P}\right)$$



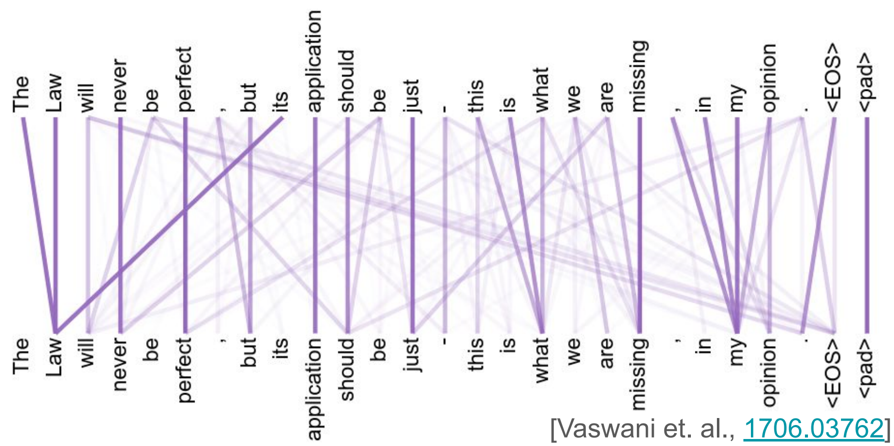
“More efficient representation” means ϕ, F could be…

1. Simpler elementary functions? ~ *Yes for simple observables, maybe for complex ones*
2. More simply parameterized? ✗ *No, independent of k*
3. Easier to embed? ✓ *Yes! **Much** smaller L’s for larger k*



Appendices

Aside, if we have time:
Attention is all you need



[Vaswani et. al., 1706.03762]

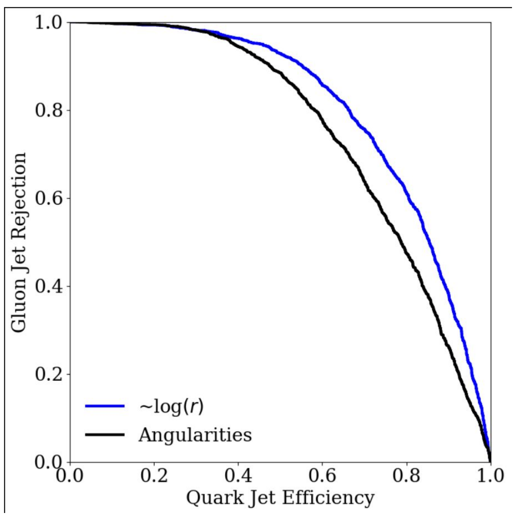
$$\sim \quad \langle \phi^{a_1} \phi^{a_2} \rangle_{\mathcal{P}}$$

Can view moment pooling as a multi-headed self-attention-like mechanism
Each latent variable weights each other latent variable

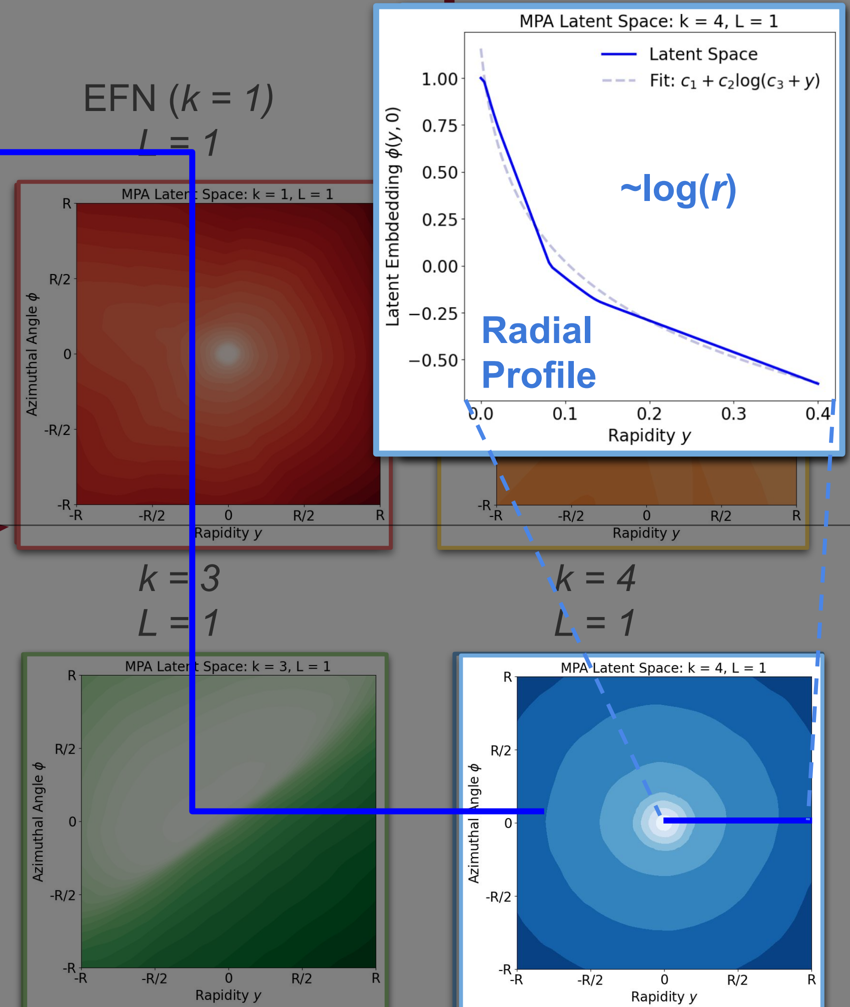
Analytic Observables

Define $\phi(r) = 1 + 0.5\log(r + 0.01)$

Feed the first 4 moments to a simple DNN – AUC of ~0.82! Angularities AUC ~0.75.



Latent Spaces



The contours of each of the L different ϕ functions

Angularities Study (Details)

Dataset:

- 14 TeV Z+jet[g, uds] events generated in Pythia 8.226
- Jets clustered using AK4 (Fastjet 3.3.0)
- Keep p_T between 500 GeV and 550 GeV, $|y| < 1.7$
- 100k Train, 2.5k Val, 2.5k Test
- Angularities normalized to unit mean and standard deviation
- Particle p_T normalized to one.

Training:

- Batch Size: 512
- Epochs: 100
- Optimizer: Adam with learning rate 0.001

Q/G Study (Details)

Same dataset as angularity study, but with 500k training samples

For each of k = 1 … 4:

1. Choose random integers F_{size} and ϕ_{size} from 1 … 128, and L from 1 … L_{max}, where L_{max}= 128 for $k = 1$, 64 for $k = 2$, 32 for $k = 3$, and 16 for $k = 4$.
   a. Choose such that the number of network parameters is uniform in log scale.
   b. For the linear F study, set F_{size}=1.
2. Initialize $N = 3$ Moment-EFNs (with different seeds), where the F and ϕ networks have three layers of the above size and latent dimension L.
   a. For the linear F study, instead choose the F network to be a single linear layer.
3. Train all N Moment-EFNs, using BCE loss, with the same hyperparameters are the angularities study. Record their AUCs.
4. Record the mean and standard deviation of the N AUCs and plot a single point. Repeat for 25 total points.