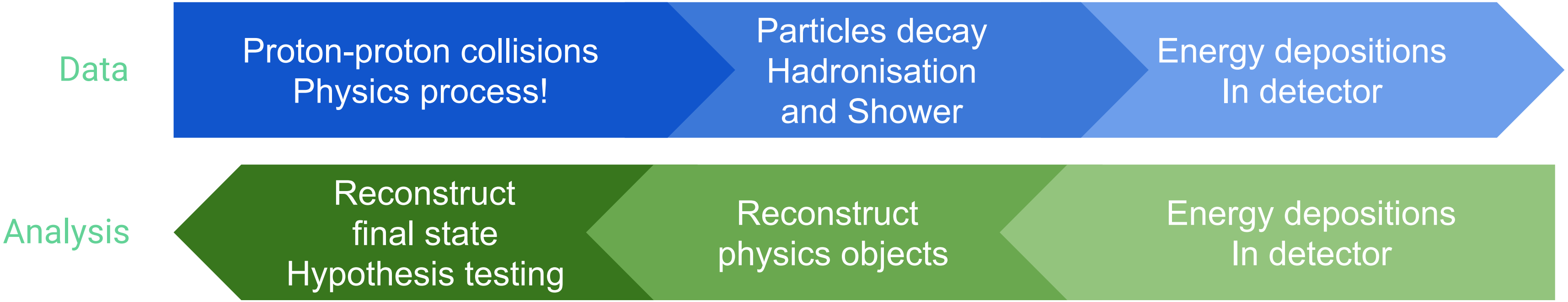
CoCo: Contrastive Framework for Combinatorics

Debajyoti Sengupta, Johnny Raine, Tobias Golling
ML4Jets, Hamburg 2023

UNIVERSITÉ DE GENÈVE

Reconstructing collisions

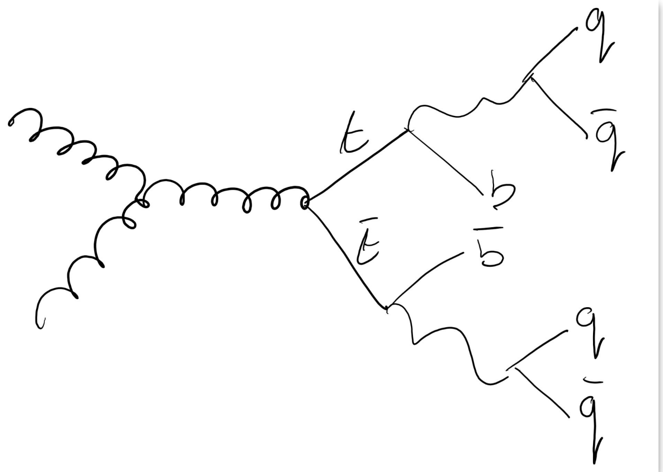
- In our detectors, all the interesting physics happens within ~3mm
  - The detector exists to capture the remnants
  - But we never see exactly what happened!

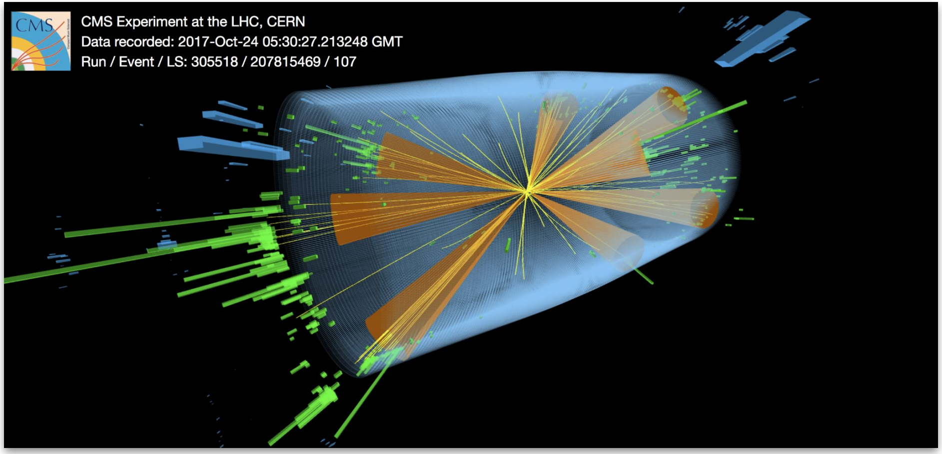


- One of the biggest challenges in our physics programme is

Reconstruction and inference

Problems with high multiplicity processes!





What we observe
Six jets, two b-tagged
All pointing back to interaction point

But which jet is which?
Need to match jets to correct quarks

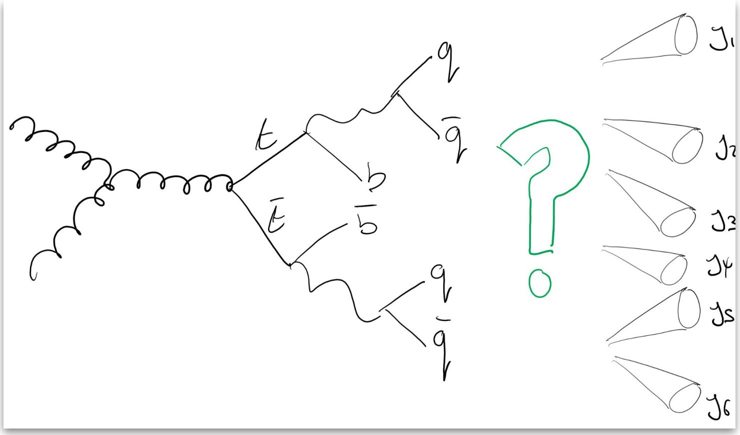
Simple approaches

For top quark reconstruction very common approach:

Solve combinatorics of **assigning jets to quarks**

The standard method is to test every possible combination and pick “most likely”

nJets	6	7	8	9
nChoose6 (removing symmetries)	92	644	2576	7728



Huge combinatorial explosion!

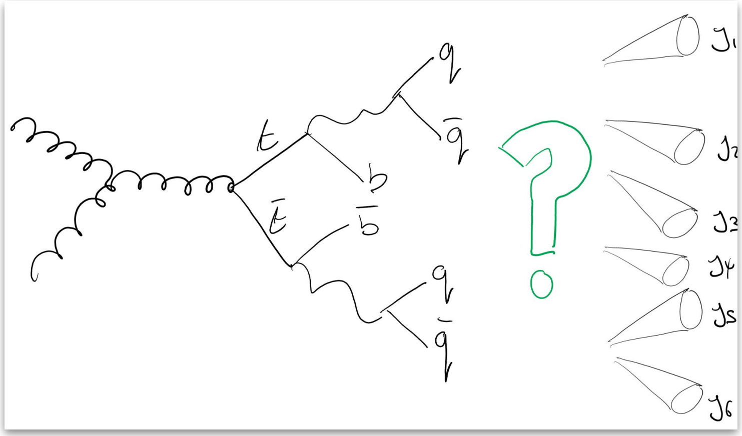
Simple approaches

For top quark reconstruction very common approach:

Solve combinatorics of **assigning jets to quarks**

The standard method is to test every possible combination and pick “most likely”

$$\chi^2 = \frac{(m_{b_1 q_1 q_2} - m_t)^2}{\sigma_t^2} + \frac{(m_{b_2 q_3 q_4} - m_t)^2}{\sigma_t^2} + \frac{(m_{q_1 q_2} - m_W)^2}{\sigma_W^2} + \frac{(m_{q_3 q_4} - m_W)^2}{\sigma_W^2},$$

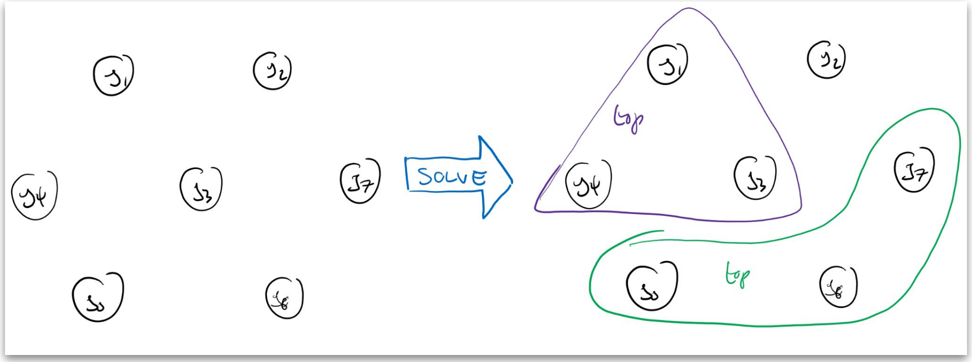


“Most likely” often chosen to minimise this

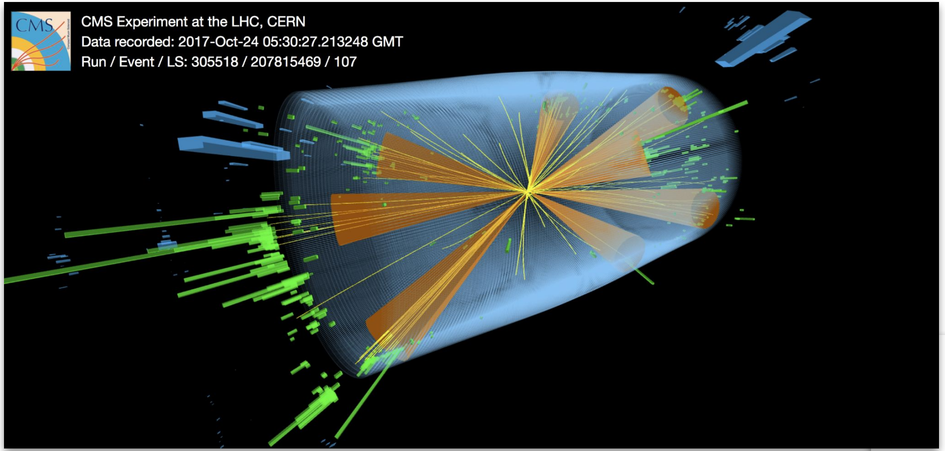
Modern ML techniques

Graphs and sets natural way to represent physics data

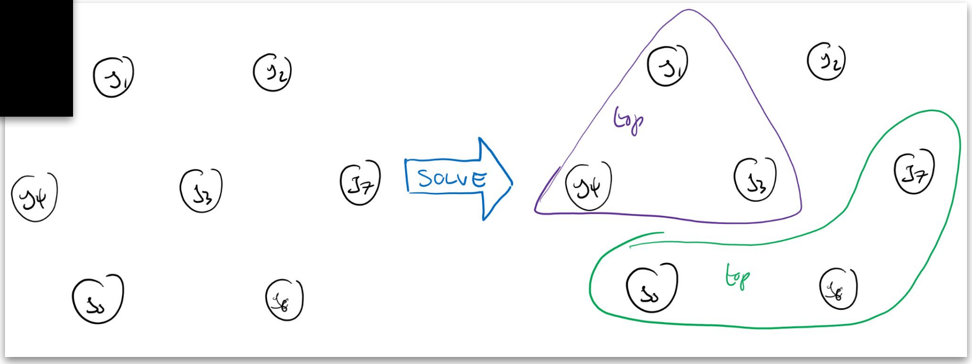
- Use graph networks to identify objects (jets/tracks) from same origin
  - Connect every object together N(N-1) edges, **identify “true” edges**
- Attention transformers on sets
  - Evaluate all triplets directly
- Notable literature:
  - SPANet
  - Topograph



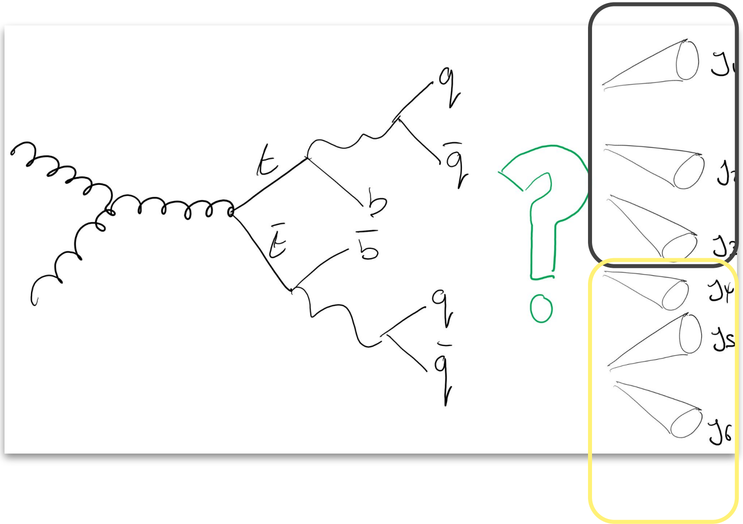
Unfortunately...



... detector environment doesn't end up with jets spatially grouped based on origin



Can we still do this?



YES!

Take inspiration from Feynman diagram

Can just cluster this space:

- Learn an embedding that groups based on decay tree

How can we do this?

Contrastive learning!

Learn an embedding/feature space

For some definition of ‘sameness’, pull positive pairs, push negative pairs apart.

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)},$$

The contrastive loss function

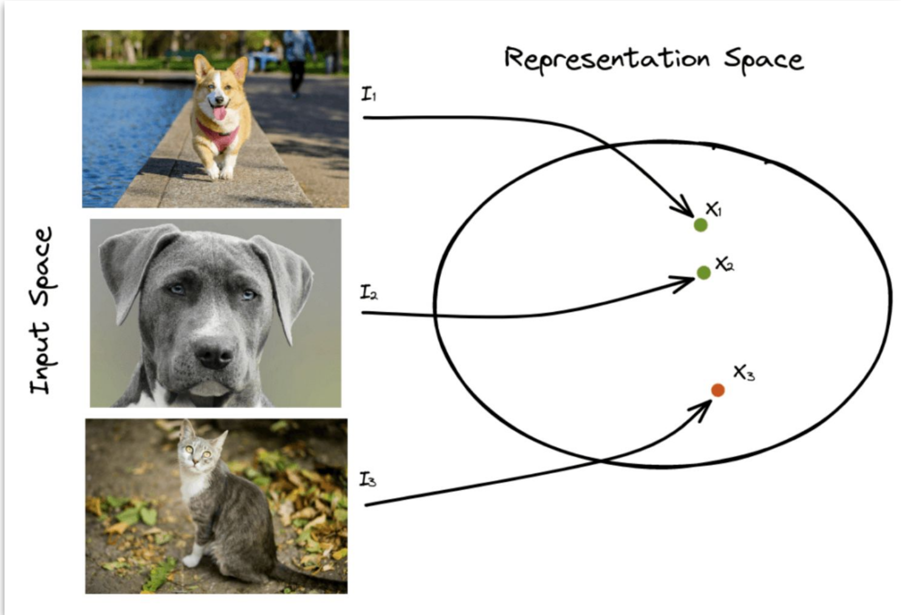


Image credits

How can we do this?

Contrastive learning!

Learn an embedding/feature space

For some definition of ‘sameness’, pull positive pairs, push negative pairs apart.

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)},$$

The contrastive loss function

Embed jets instead

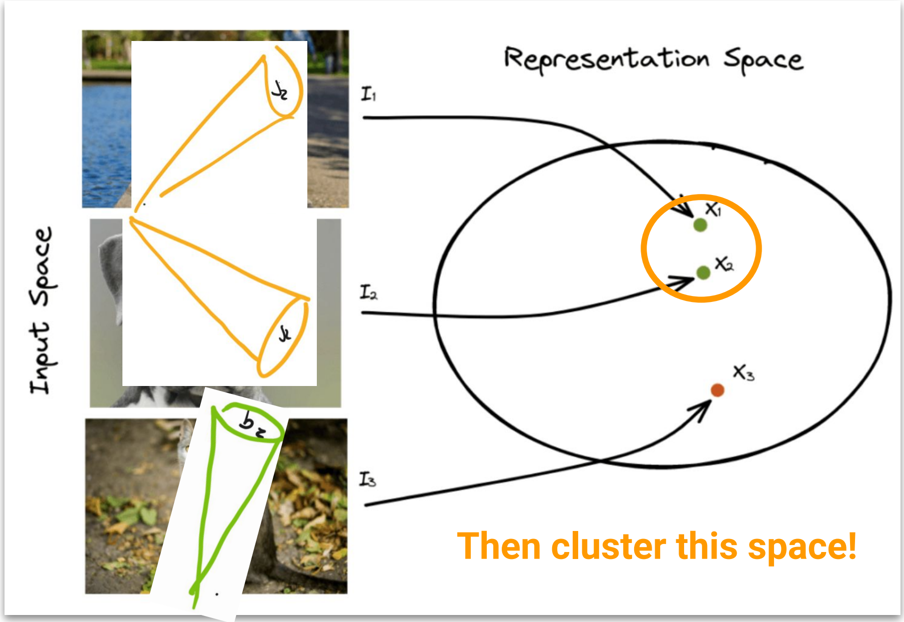


Image credits

Advantages

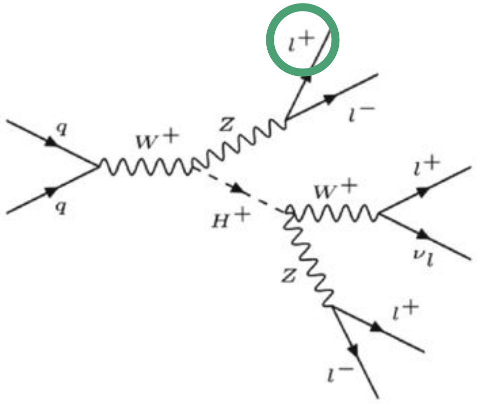
- Any Feynman diagram can be cast into this setup:
  - Method requires labels to identify common parents

Advantages

- Any Feynman diagram can be cast into this setup:
  - Method requires labels to identify common parents
- Missing objects are fine:
  - No kinematic reconstruction required → Method just learns to group things together
  - Can deal with asymmetric decay structures.

Advantages

- Any Feynman diagram can be cast into this setup:
  - Method requires labels to identify common parents
- Missing objects are fine:
  - No kinematic reconstruction required ➔ Method just learns to group things together



(c)

Say - this object is somehow outside the acceptance, and missing.

No positive partner ➔ But is still regularised by all the other negative partners.

An example case

t t hadronic channel

Casting the problem

For t t ➔ W (qq) b

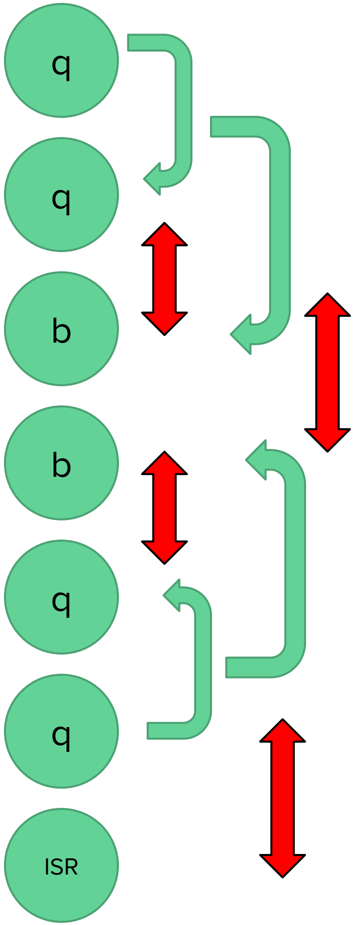
Level 0:

1. Pull q's and b from same top together. Push the other q's and b from the other top apart.

Level 1:

1. Pull q's together, push b apart within this cluster.
2. Push everything else apart.

- Train an encoder with: Loss = Level 0 + w * Level 1
- After training an encoder ➔ **Perform a bottom up clustering.**

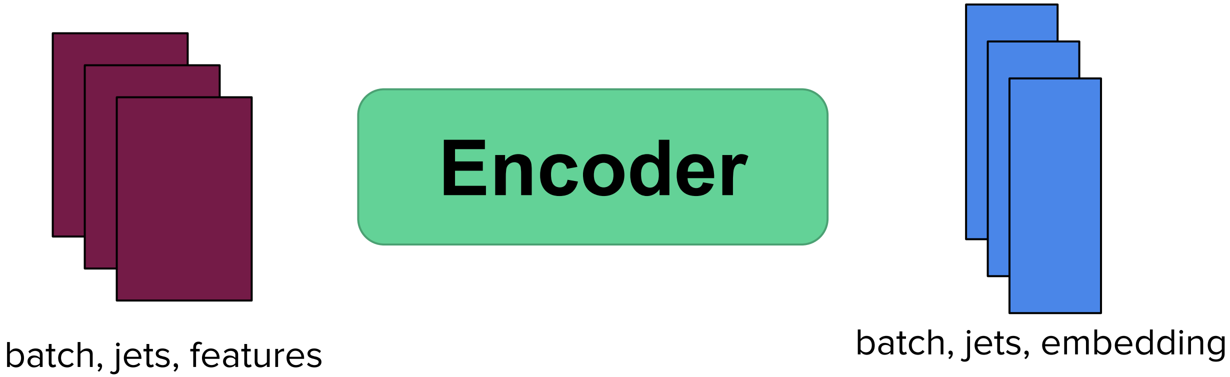


Workflow

Features:

1. features: jet kinematics, b-tag
2. Condition on number of jets in events.

Encoder used: Transformer



The clustering

Perform a bottom up clustering

- Merge two objects into a cluster, closer than a preset threshold
- If Cluster reaches a preset cardinality ➔ remove from process.
- If Cluster exceeds cardinality ➔ undo last merge and remove from process.
- Stop: When no two points are closer than preset threshold.

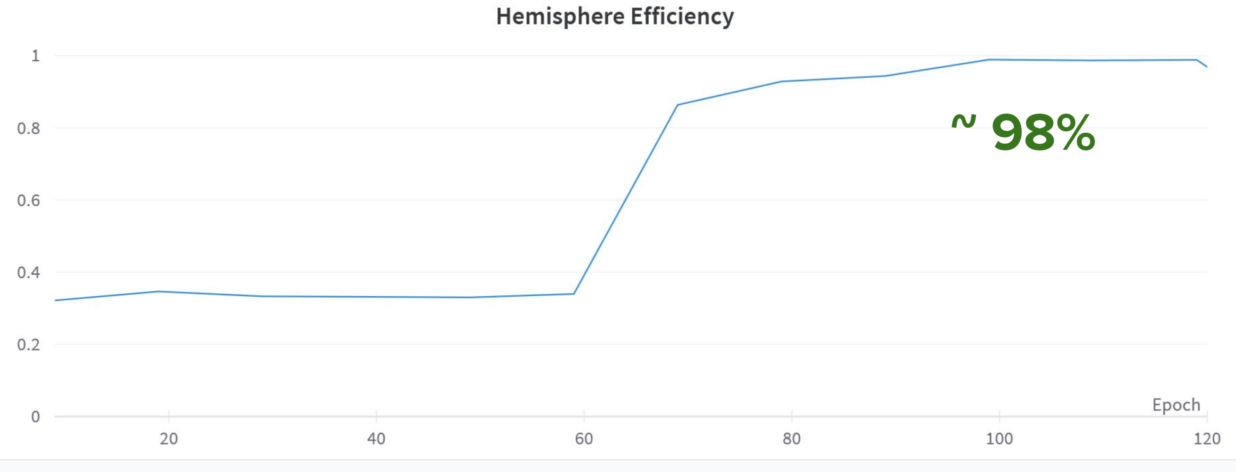
```
Define N: max cluster size
Define t: max distance threshold
Define e_i: embedding of object i
Define ℓ_i: cluster label
Define N_ℓ: cluster number
d_ij = pairwise_distance(e_i, e_j)
C = identity(len(i))
N_ℓ ← 0
d_ij = mask_lower_triangle(d_ij)
d_ij = mask_diagonal(d_ij)
while any d_ij < t do:
    find (i,j) for min(d_ij)
    if hard N requirement is True then
        C′ ← C
    end if
    C[i,j] ← 1                                        ▷ Set element i,j to 1
    C[i,:] ← C[i,:] ∨ C[j,:]                          ▷ Set nonzero elements in j to 1 in i
    C[j,:] ← C[j,:] ∨ C[i,:]                          ▷ Set nonzero elements in i to 1 in j
    C| = transpose(C)                                 ▷ Transpose to make symmetric on diagonal
    if sum(C[i,:]) ≥ N then
        if hard N requirement and sum(C[i,:]) > N then
            C ← C′
        end if
        k ← 0
        for v in C[i,:] do
            if v is 1 then
                ℓ_k ← N_ℓ
                mask(d_ij[k,:])
            end if
            k ← k + 1
        end for
        N_ℓ ← N_ℓ + 1
        if hard N requirement and sum C[i,:] > N then
            k ← 0
            for v in C[j,:] do
                if v is 1 then
                    ℓ_k ← N_ℓ
                    mask(d_ij[k,:])
                end if
                k ← k + 1
            end for
            N_ℓ ← N_ℓ + 1
        end if
    end if
end while
```

The ideal clustering limit

Given a set of ‘informed’ encodings:

- If you could seed a centre ➜ group the nearest neighbours

- In practice, we do a bottom up clustering.
- Performance to be gained here.



Top reconstruction efficiencies:

- Performs better than Chi Squared, approaching SotA.

	CoCo	**Topograph**	**Chi Squared**
6 jets = 2b	85.3%	**88.3%**	79.1%
6 jets >= 2b	84.3%	**87.8%**	79.2%
7 jets = 2b	62.5%	**70.6%**	57.9%
7 jets >= 2b	61.1%	**69.7%**	57.1%

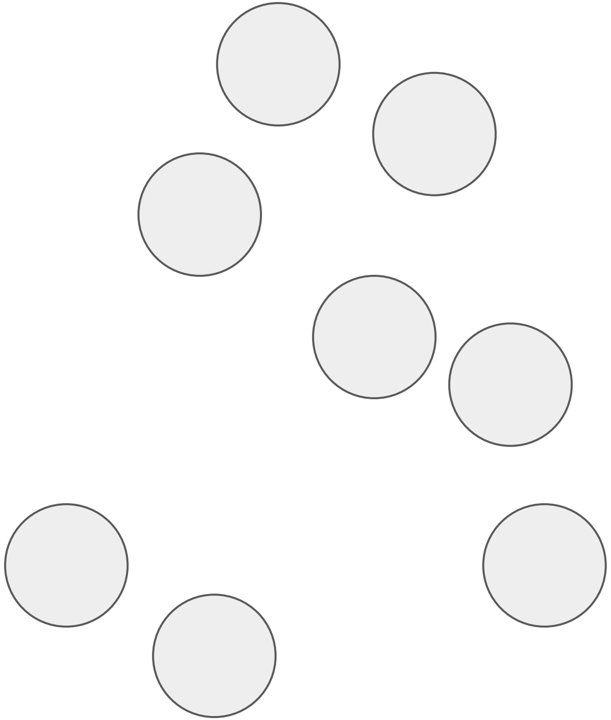
Top reco. eff. as a function of number of jets.

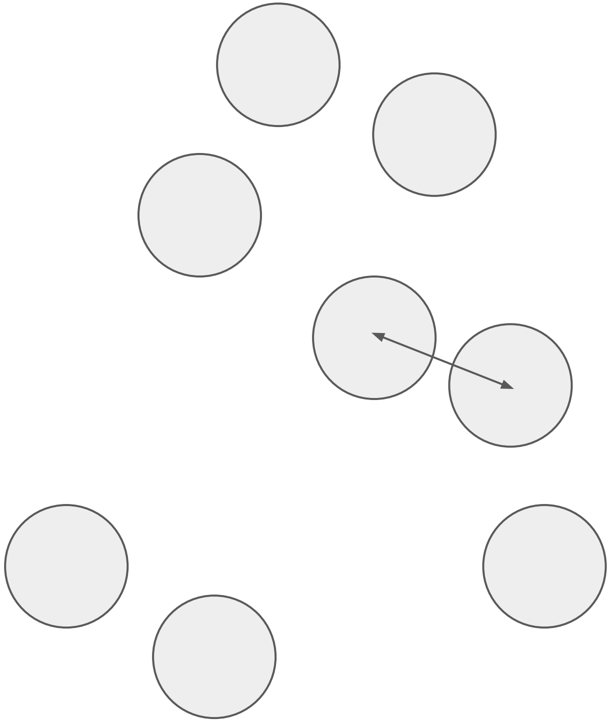
Outlook

- Contrastive losses a natural framework for combinatoric solving.
  - Flexible, generisable, performant

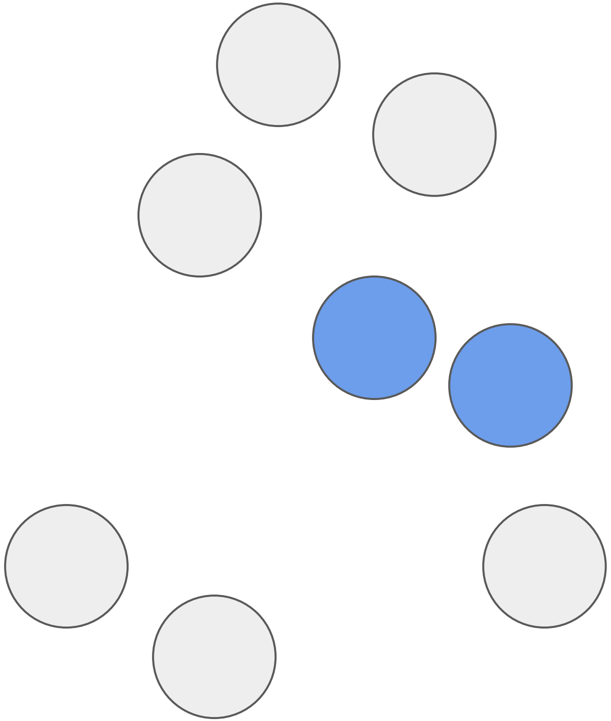
Backup

- Clustering demo





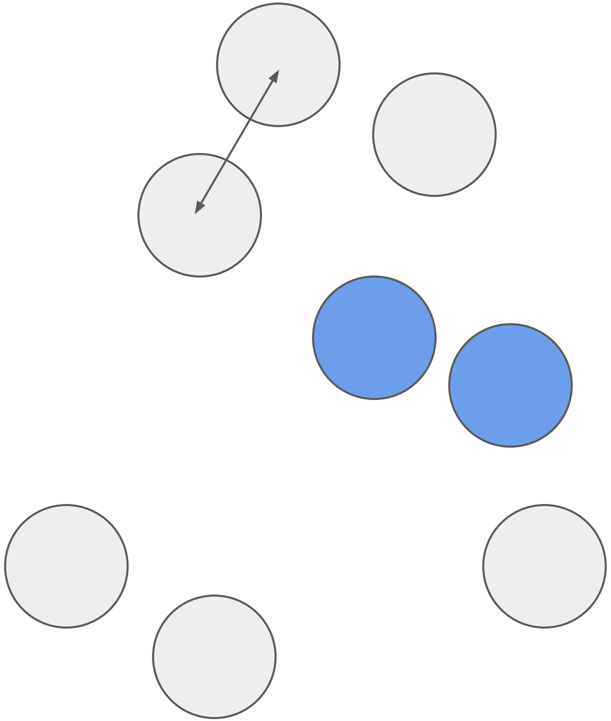
Find nearest two embeddings



Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

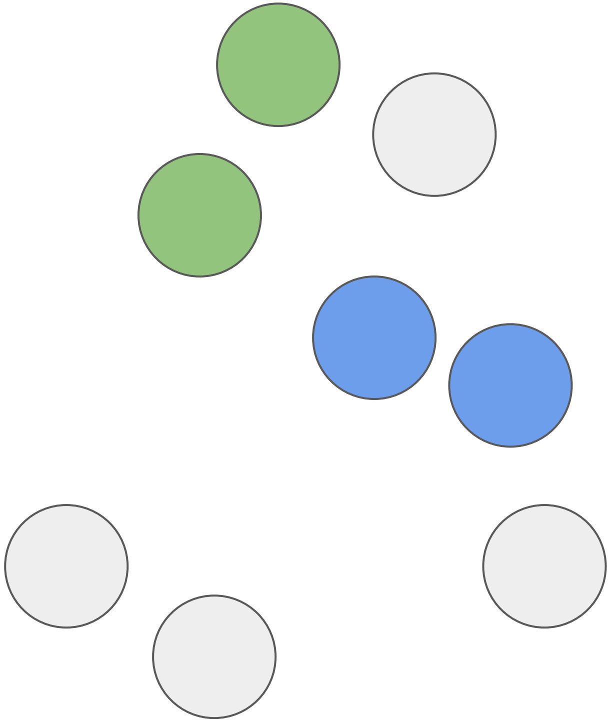


Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

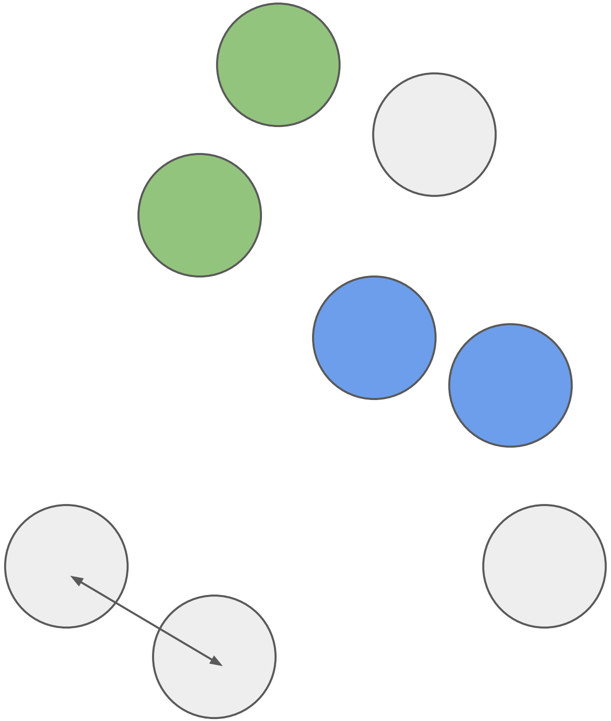


Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

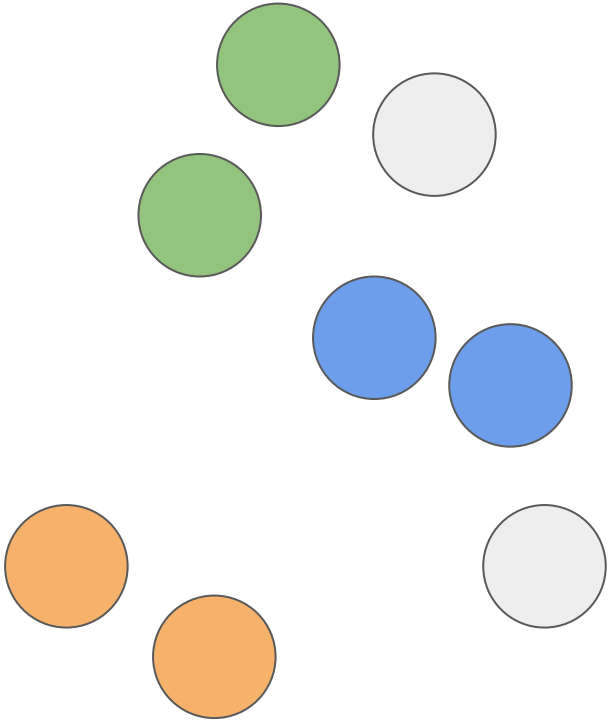


Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

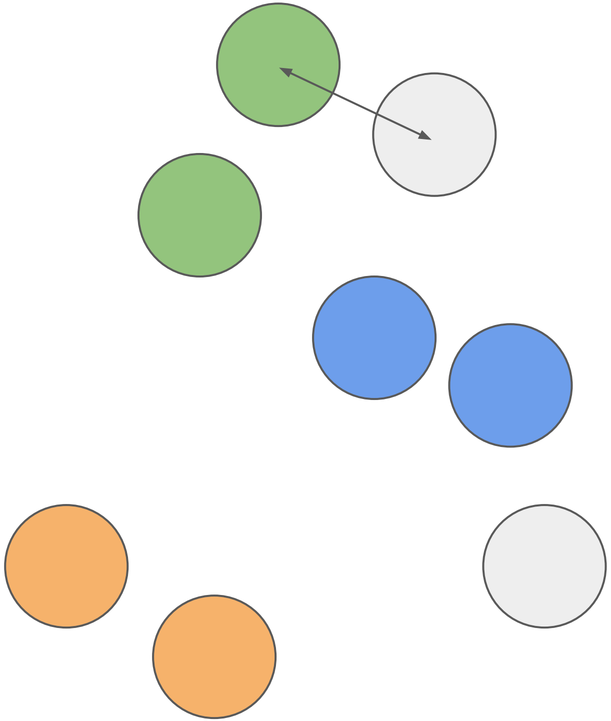


Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

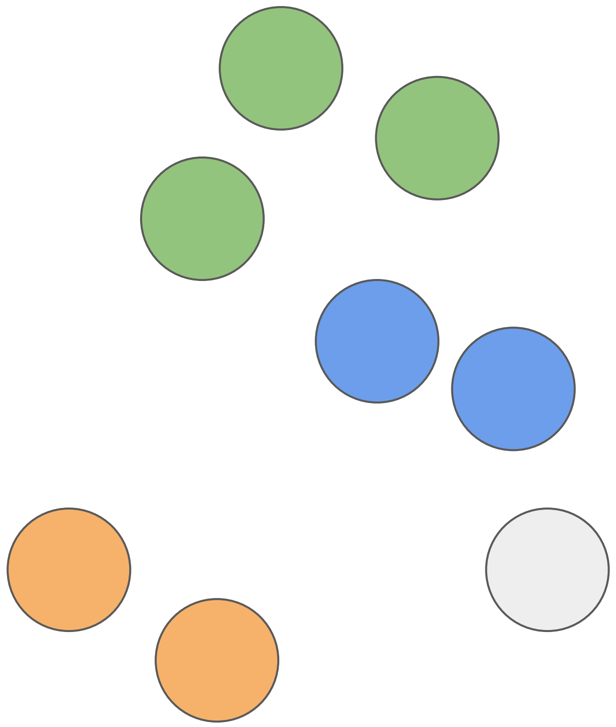
Repeat until hit max cluster size



Find nearest two embeddings

Merge into same cluster
- If already in a cluster, merge clusters

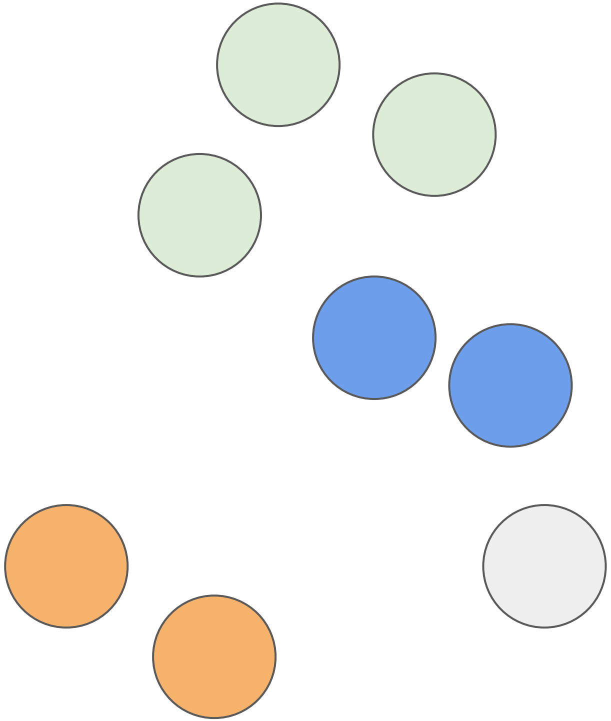
Repeat until hit max cluster size



Find nearest two embeddings

Merge into same cluster
- If already in a cluster, merge clusters

Repeat until hit max cluster size



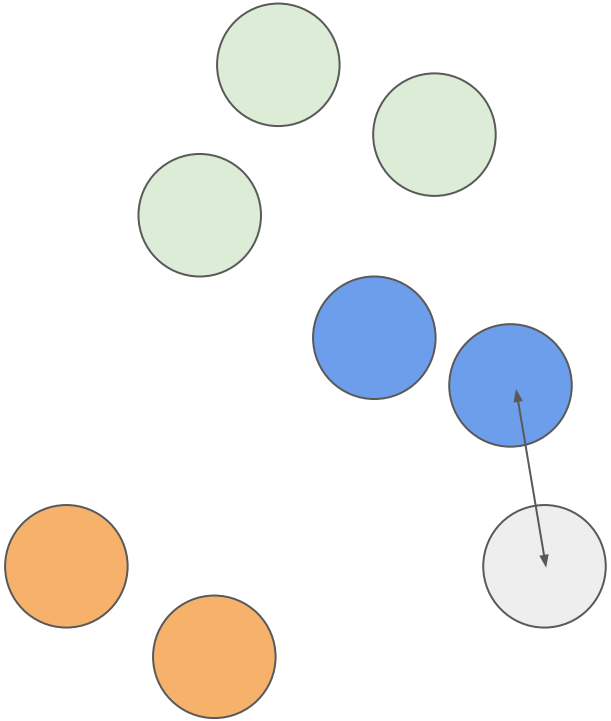
Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

- Lock cluster embeddings



Find nearest two embeddings

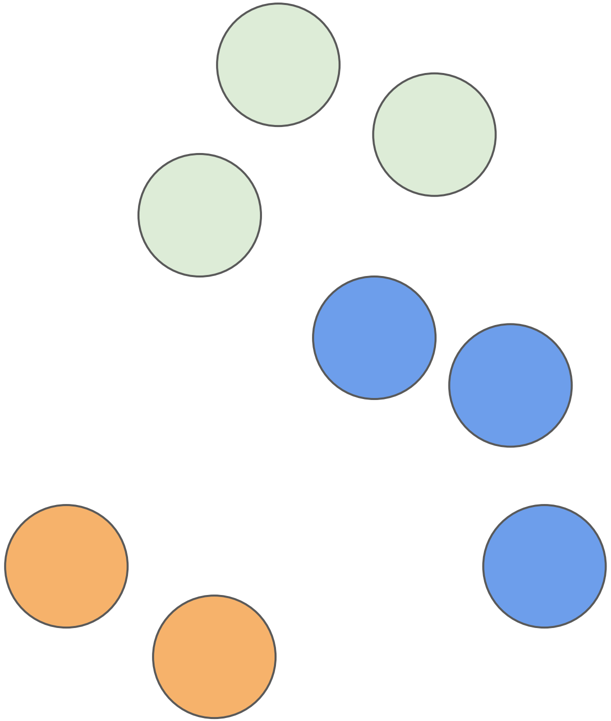
Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

- Lock cluster embeddings

Repeat until all in cluster (or no embeddings within a threshold distance)



Find nearest two embeddings

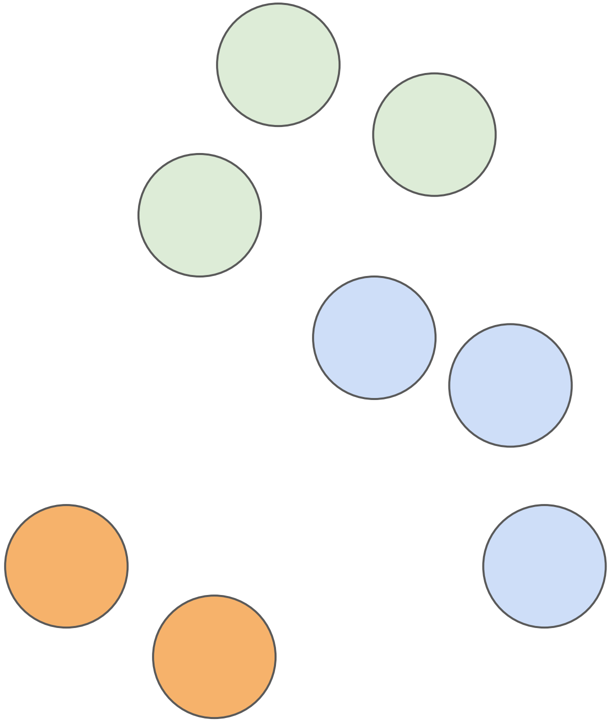
Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

- Lock cluster embeddings

Repeat until all in cluster (or no embeddings within a threshold distance)



Find nearest two embeddings

Merge into same cluster

- If already in a cluster, merge clusters

Repeat until hit max cluster size

- Lock cluster embeddings

Repeat until all in cluster (or no embeddings within a threshold distance)