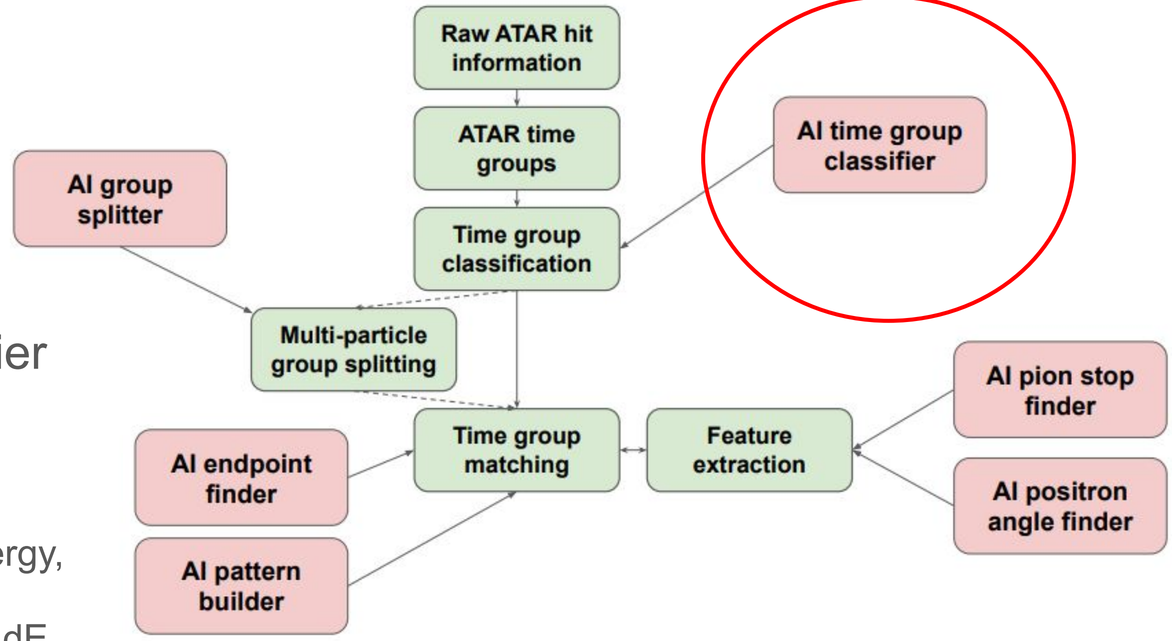
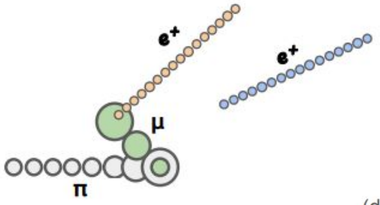
Training and Validation Details For Classification Models for PIONEER Reconstruction

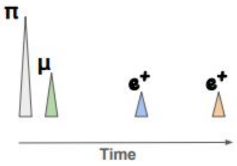
Jack Carlton
University of Kentucky

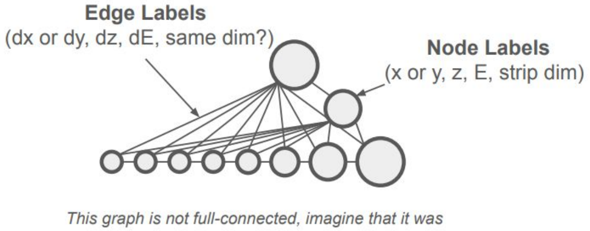
Time Group Classifier

- I'll mostly be talking about Omar's "time group classifier model"
- Input:
  - Time grouped hit graph
    - Nodes: [x (or y), z, energy, view, group_energy]
    - Edges: [dx (or dy), dz, dE, same_view]
  - Groups split by time (could potentially contain multiple particle types)
- Output
  - Class labels: [muon, pion, mip]
- Much better explained in [Omar's presentation]()



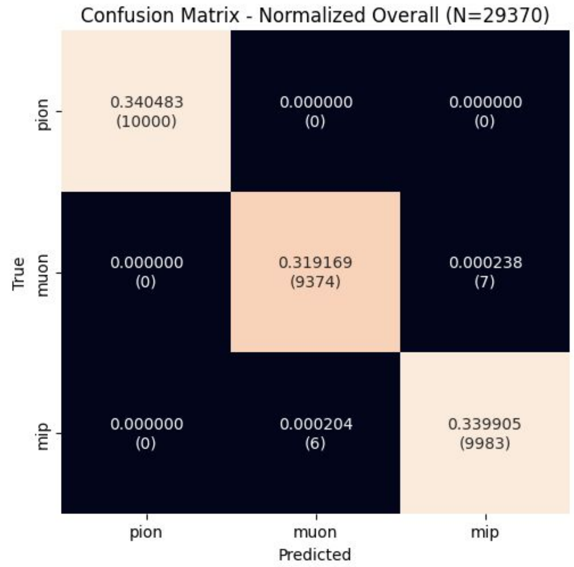


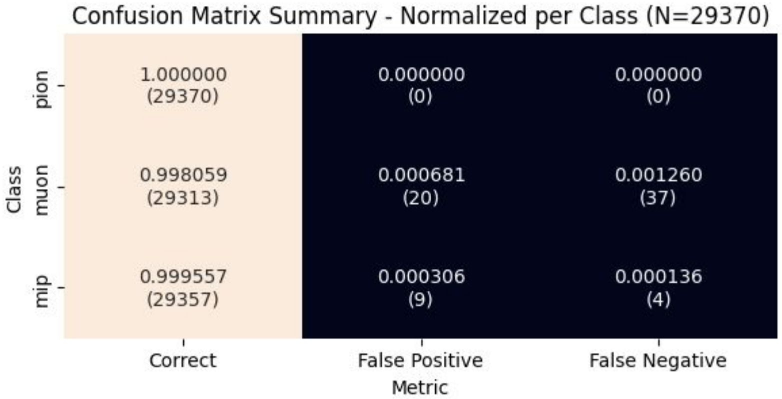




Performance of Model

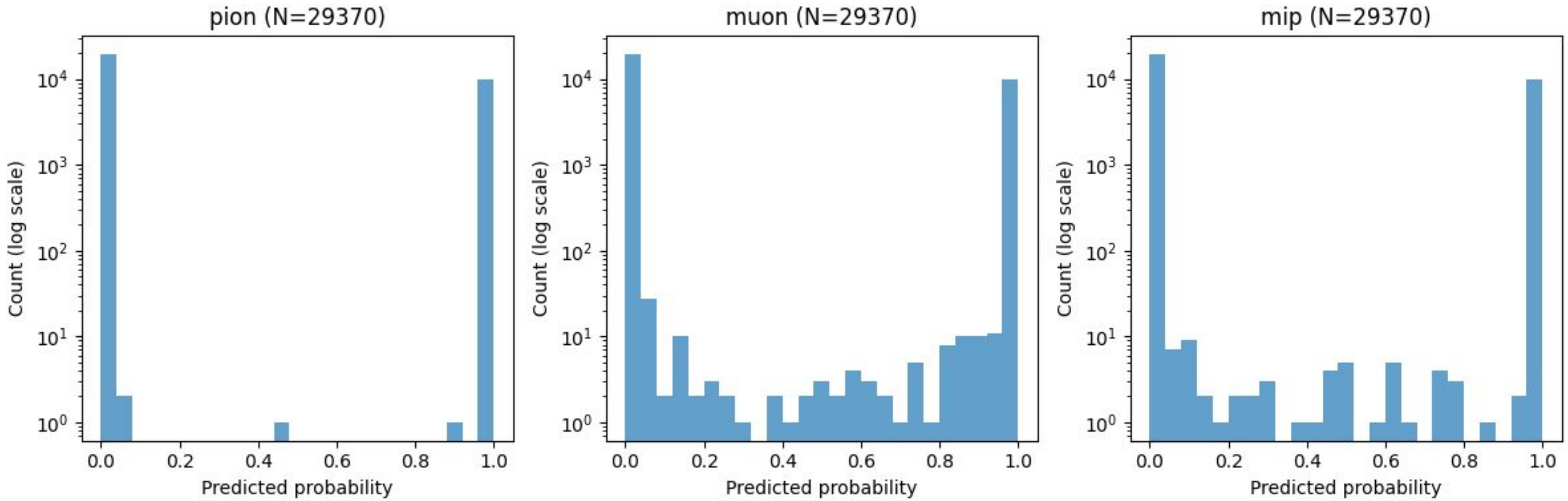
- Closely matches performance of Omar's work
- Differences from Omar's work
  - Used my machine to train
  - Did a hyperparameter search using Optuna
  - Incorporated into ZenML framework for creating pipelines in python
- Unsure how this compares to the traditional reco values(?)
- Unsure exact parameters in Omar's data set





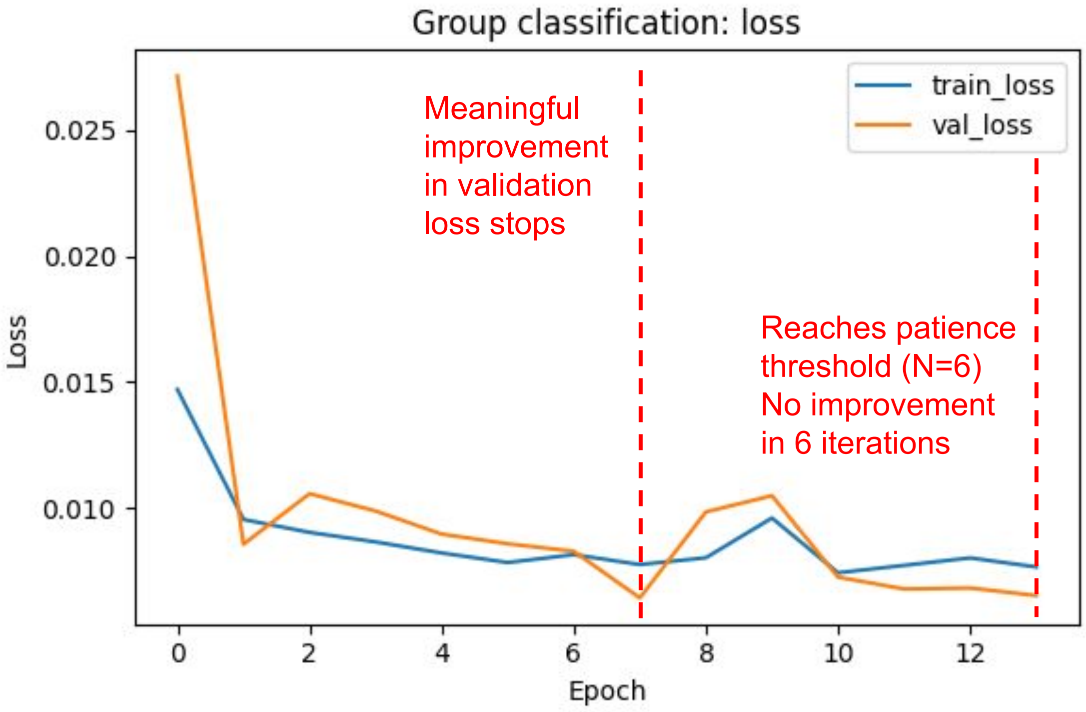
Performance of Model

- Histograms below shows the models “sureness” of each class
- Want to see large peaks at 0 and 1
  - 0 → this is definitely not this class
  - 1 → this is definitely this class
- The muon groups are the biggest struggle



Preventing Overtraining (Early Stopping)

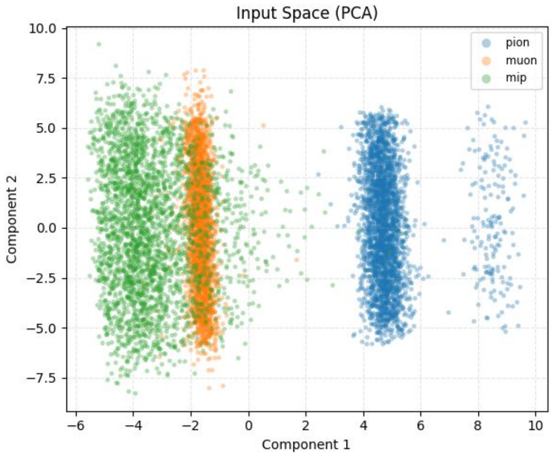
- Can set early stopping when training loss curves
  - If no meaningful (> δ) improvement within N epochs, stop training
  - N := “patience” usually set to ~5% of expected training epochs
- Potential improvements:
  - Loss curve smoothing
  - “Noise” estimations, only continue training if smoothed improvement > noise of previous iterations
- See documentation



*NOTE: This particular model is overtrained because N set to 6 (too high)

Preventing Overtraining (PCA)

- Need a way to visualize many dimensions
- Principal Component Analysis (PCA) good candidate for this
- Compare clusters in embedding space vs. input space
  - Input space shows data driven groupings
  - Embedding space shows learned groupings
  - # of groupings should match!

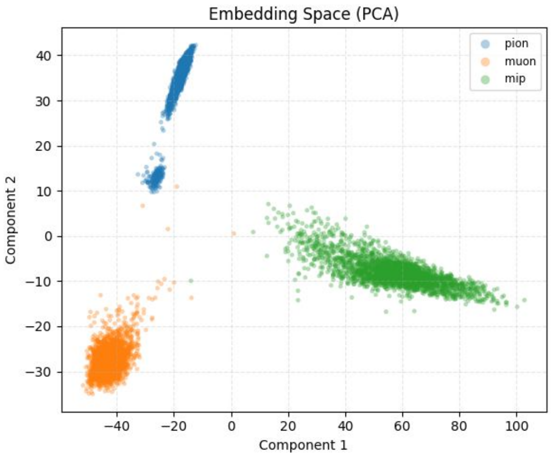


Average vector of all nodes in graph:
[coord,
 z_pos,
energy,
view,
group_energy]

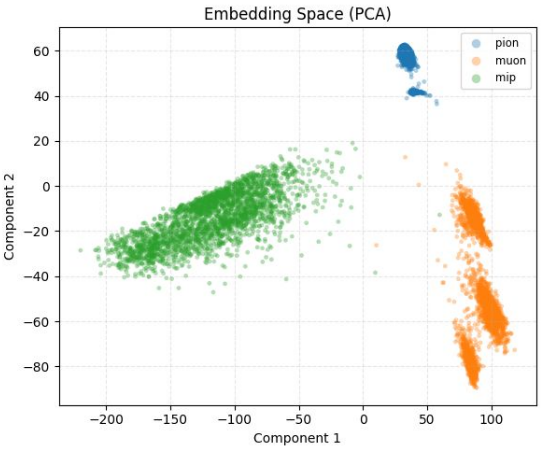
5D → 2D

Well-trained

Over-trained



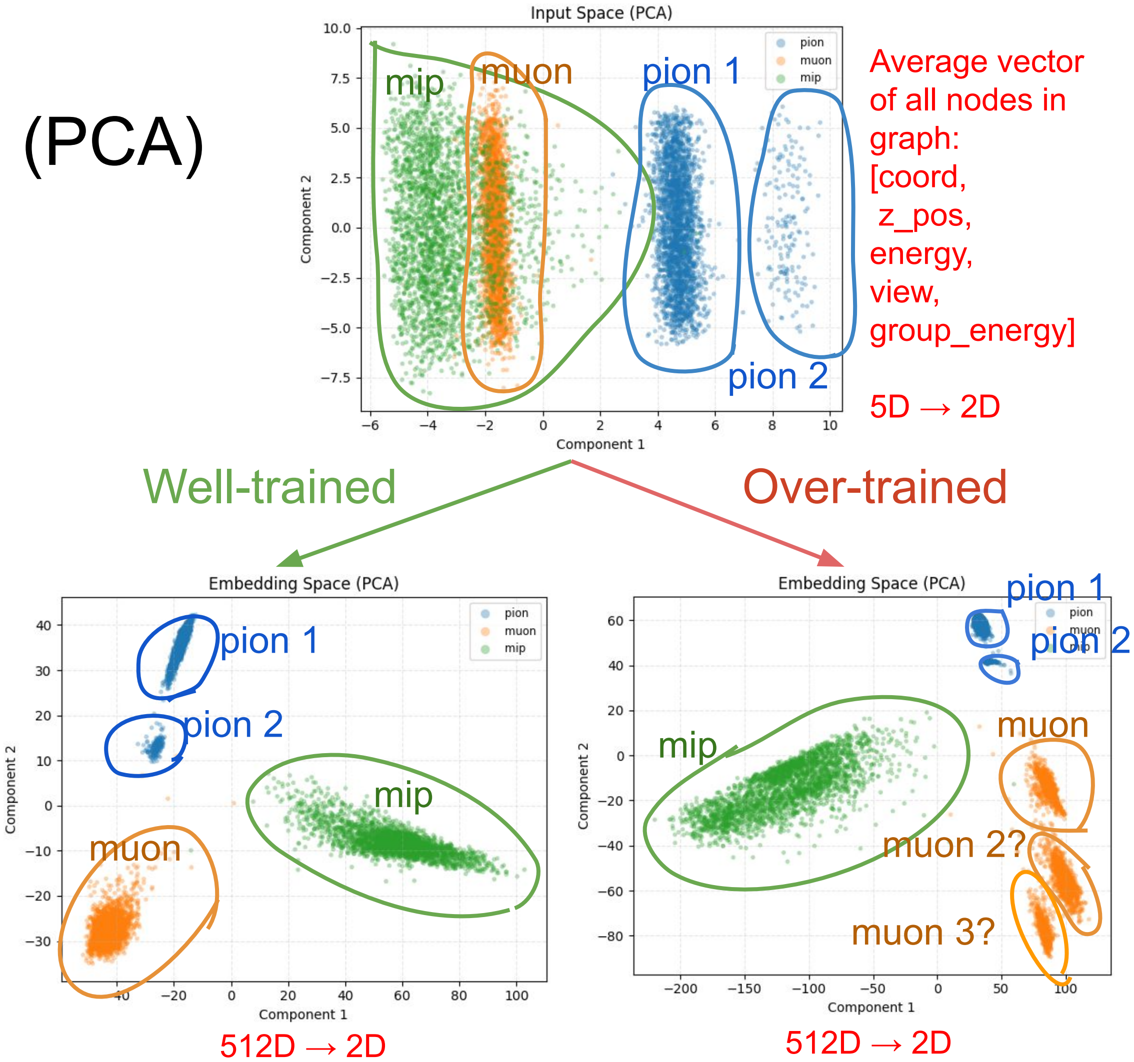
512D → 2D



512D → 2D

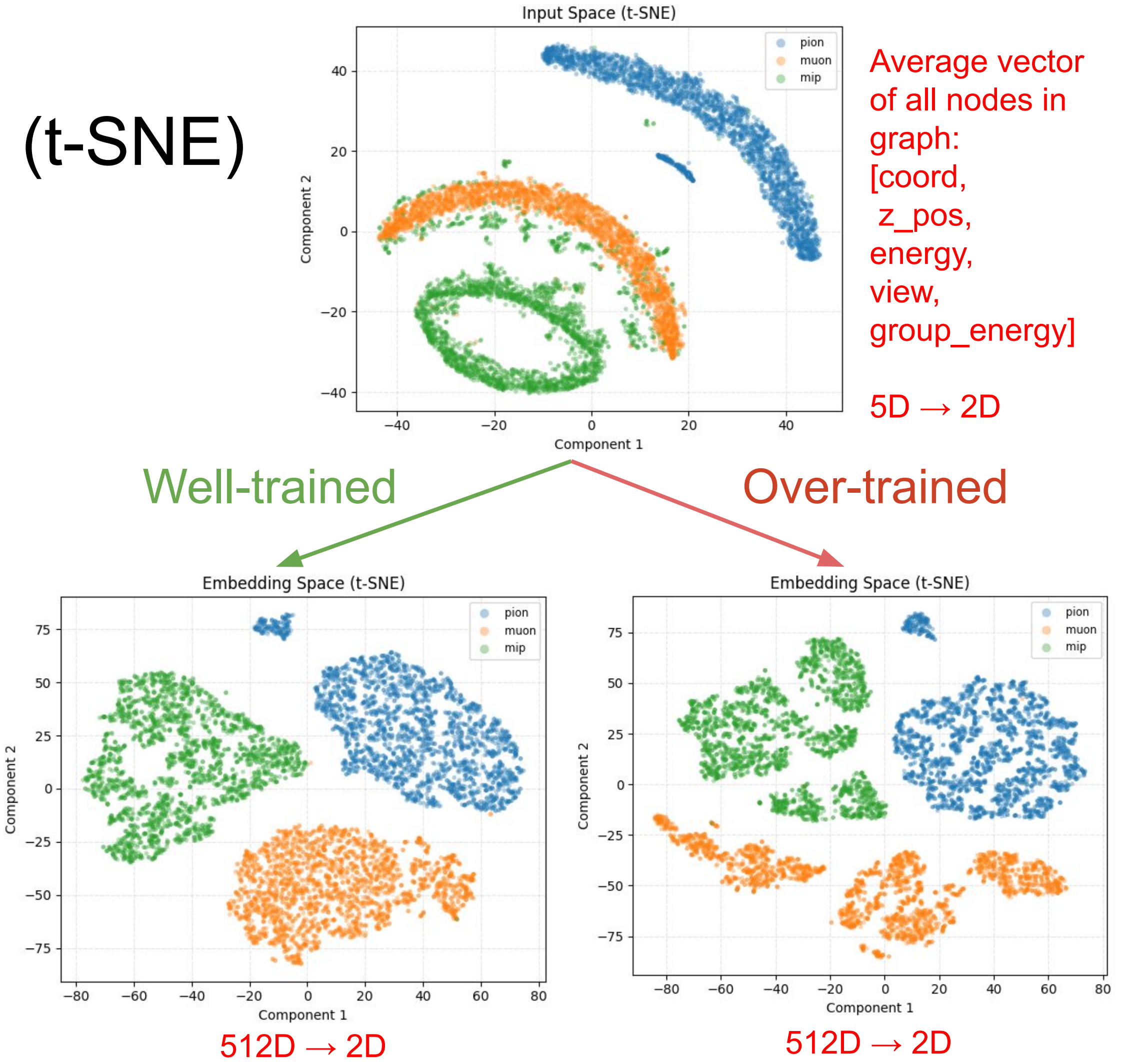
Preventing Overtraining (PCA)

- Need a way to visualize many dimensions
- Principal Component Analysis (PCA) good candidate for this
  - Choose to project down to d = 2 dimensions this way
- Compare clusters in embedding space vs. input space
  - Input space shows data driven groupings
  - Embedding space shows learned groupings
  - # of groupings should match!



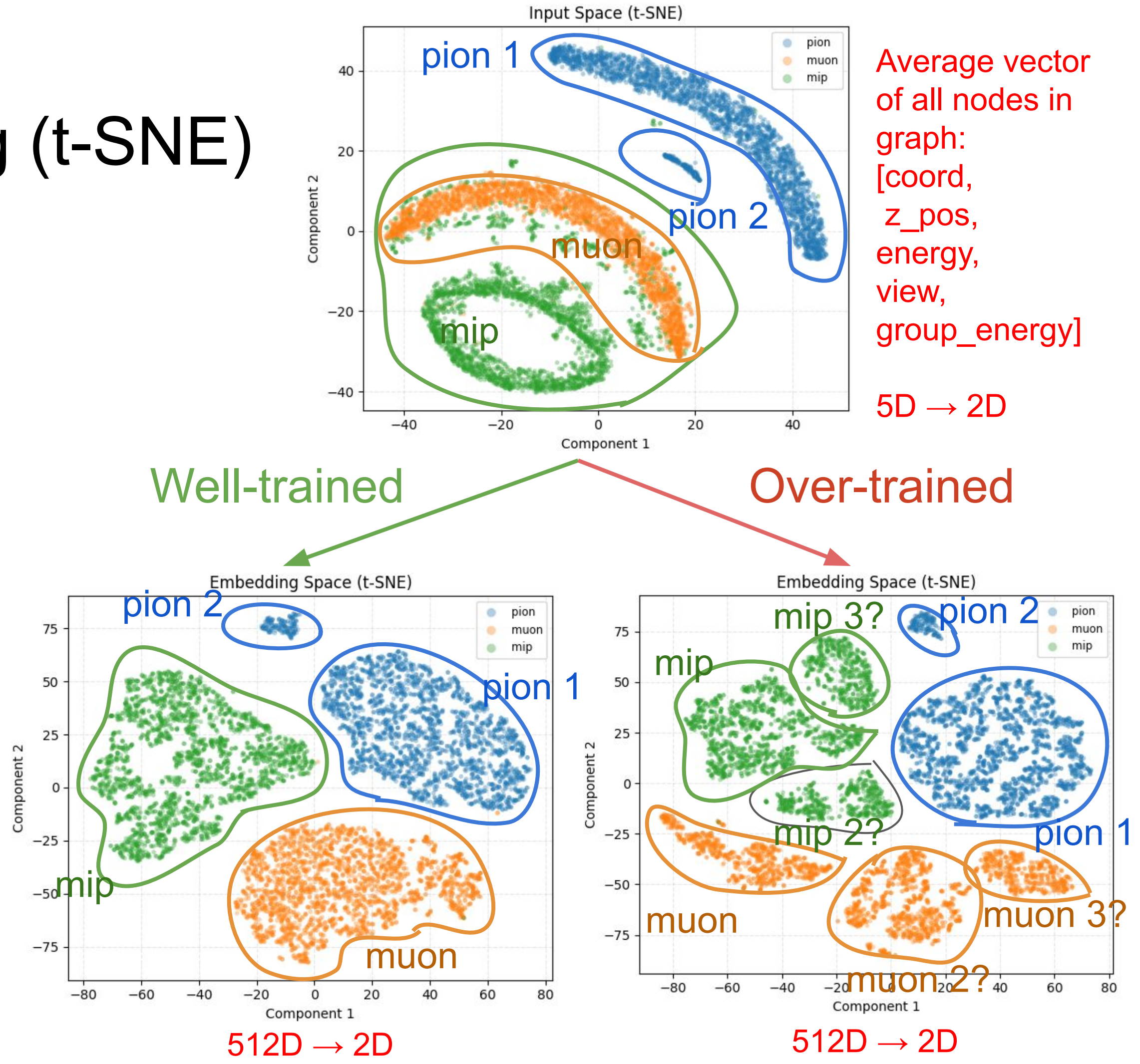
Preventing Overtraining (t-SNE)

- Problem with PCA
  - Global linear may not preserve local neighborhoods
    - May not show groups!
- t-distributed stochastic neighbor embedding (t-SNE) is designed to preserve local clusters
  - Maps N dims → d dims
    - We choose d = 2 for visualization ease
- Same ideas as PCA
  - compare input space and embedding space



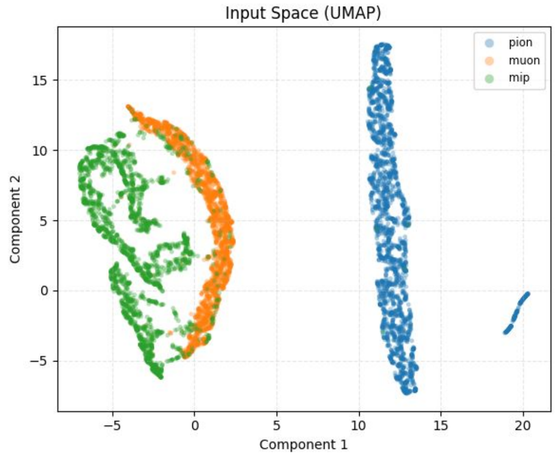
Preventing Overtraining (t-SNE)

- Problem with PCA
  - Global linear may not preserve local neighborhoods
    - May not show groups!
- [t-distributed stochastic neighbor embedding](#) (t-SNE) is designed to preserve local clusters
  - Maps N dims → d dims
    - We choose d = 2 for visualization ease
- Same ideas as PCA
  - compare input space and embedding space



Preventing Overtraining (UMAP)

- Problem with t-SNE
  - Depends on a “Perplexity”
    - Parameter, ~how many neighbors a point can have
  - May artificially split or group clusters
- Uniform Manifold Approximation and Projection (UMAP) is designed to preserve the underlying manifold structure
  - Tries to preserve both local neighborhoods *and* their global relationships
  - Maps N dims → d dims
    - We choose d = 2 for visualization ease
- Computational expensive
  - Use as a “tie breaker” if PCA and t-SNE disagree

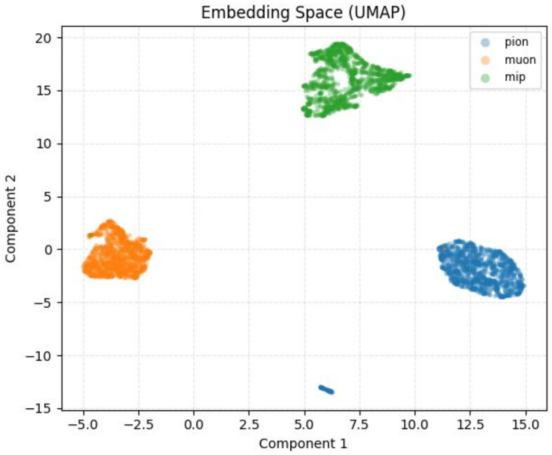


Average vector of all nodes in graph: [coord, z_pos, energy, view, group_energy]

5D → 2D

Well-trained

Over-trained



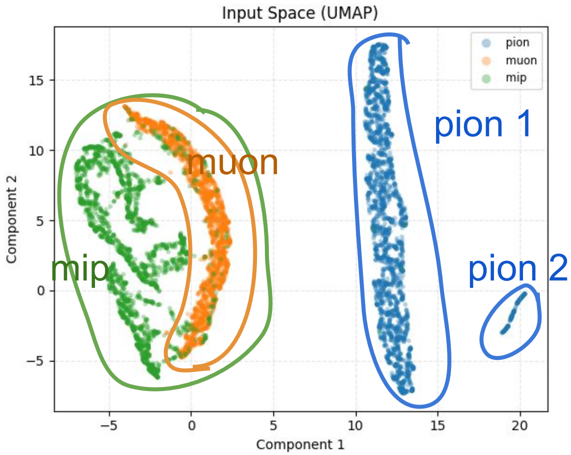
512D → 2D

Sorry!
I don’t have an example for this!

512D → 2D

Preventing Overtraining (UMAP)

- Problem with t-SNE
  - Depends on a “Perplexity”
    - Parameter, ~how many neighbors a point can have
  - May artificially split or group clusters
- Uniform Manifold Approximation and Projection (UMAP) is designed to preserve the underlying manifold structure
  - Tries to preserve both local neighborhoods *and* their global relationships
  - Maps N dims → d dims
    - We choose d = 2 for visualization ease
- Computational expensive
  - Use as a “tie breaker” if PCA and t-SNE disagree

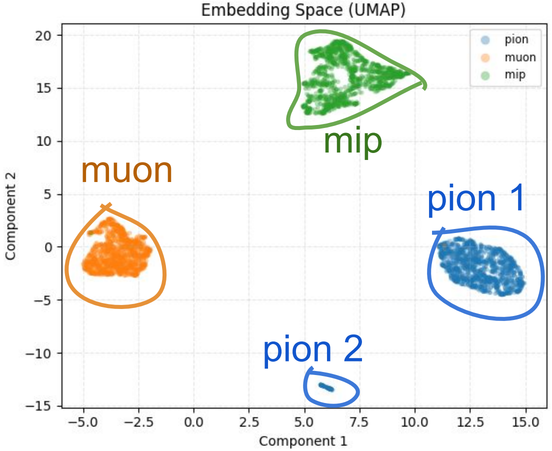


Average vector of all nodes in graph: [coord, z_pos, energy, view, group_energy]

5D → 2D

Well-trained

Over-trained



512D → 2D

Sorry!
I don’t have an example for this!

512D → 2D

Auxiliary Slides