NuGraph2

A Graph Neural Network for 3D Reconstruction in Liquid Argon Time Projection Chambers

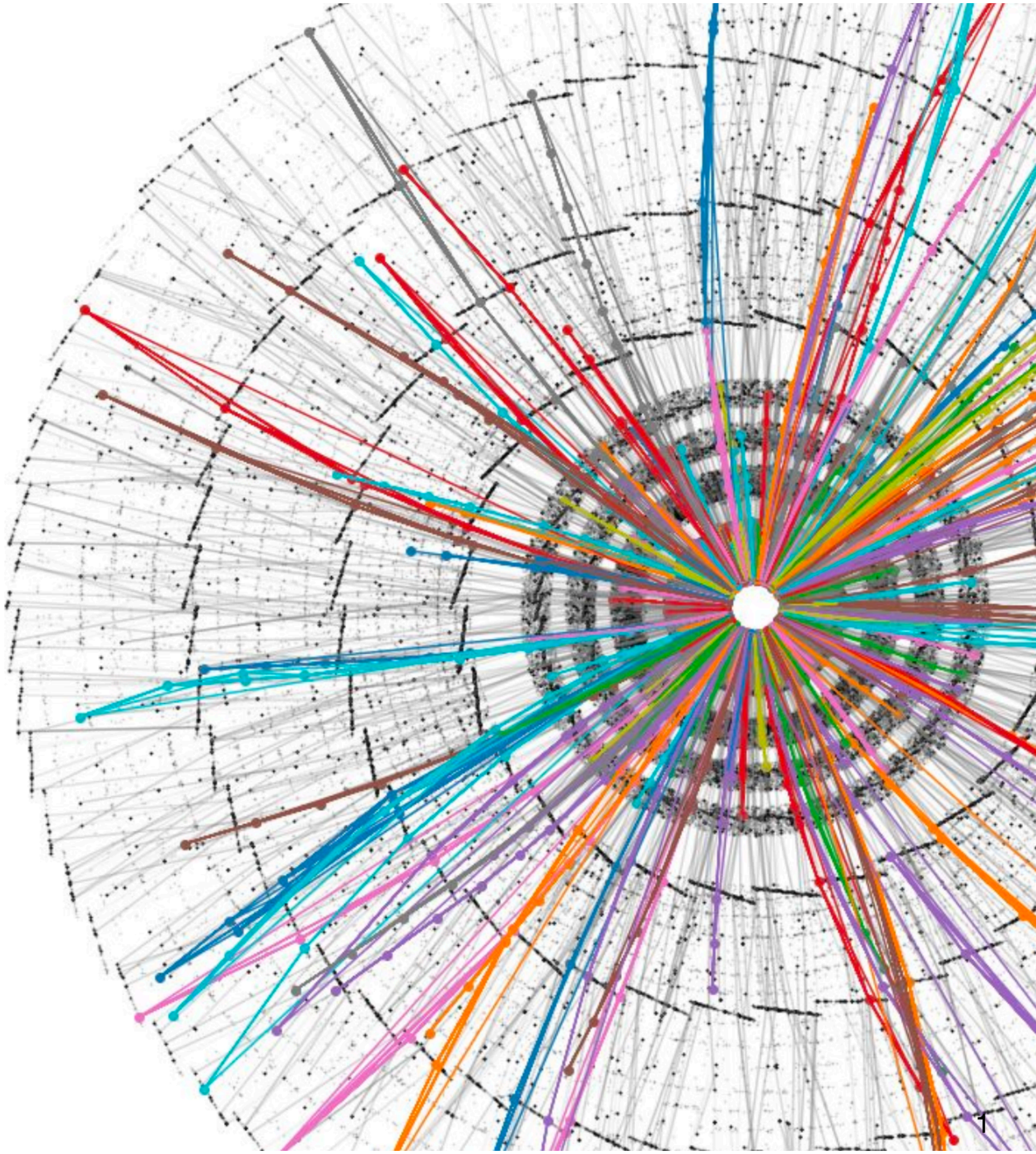
V Hewes

22nd August 2023

Neutrino Physics and Machine Learning Workshop 2023

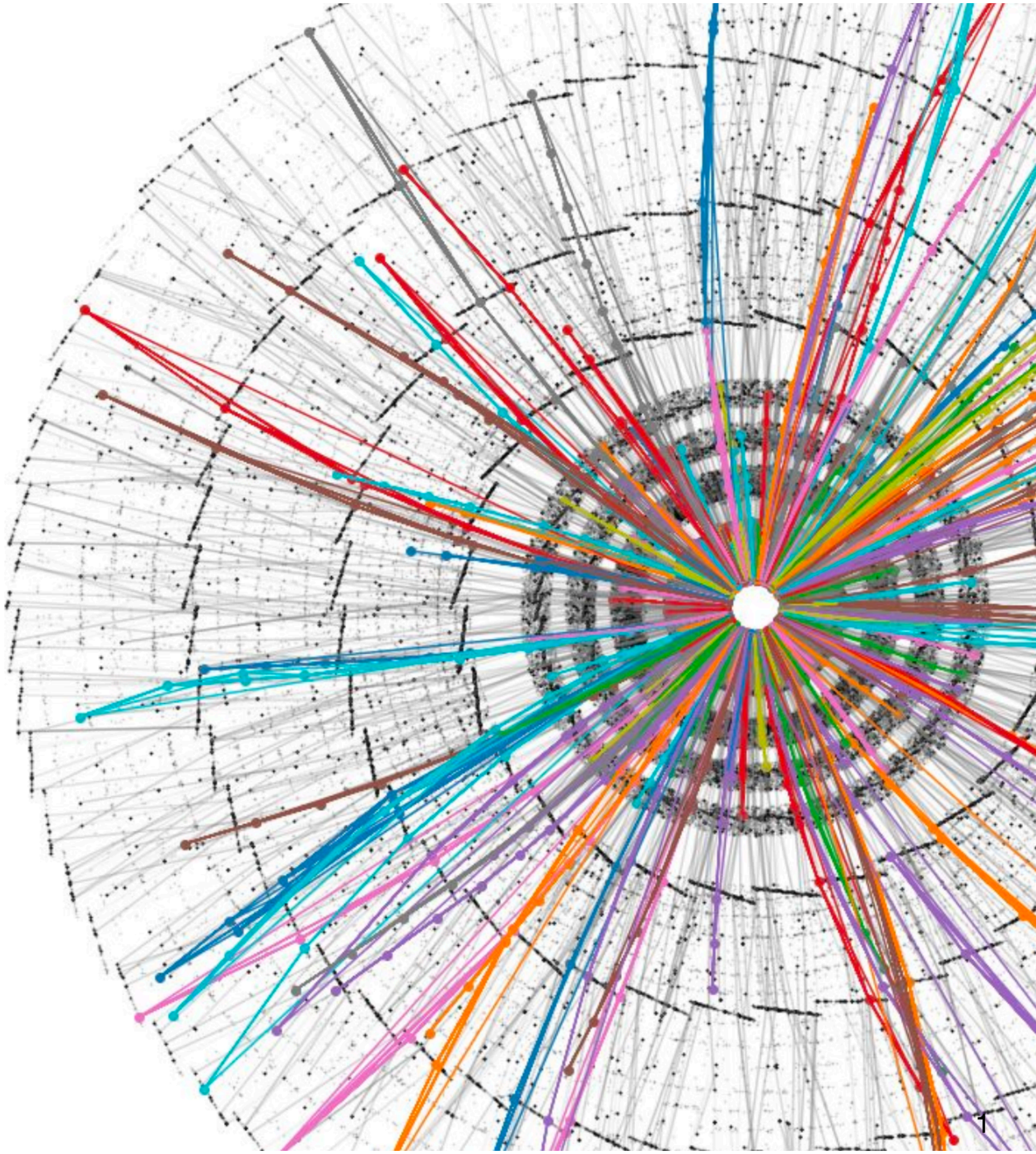
Exa.TrkX

- Exa.TrkX is a collaboration developing next-generation Graph Neural Network (GNN) reconstruction for HEP:



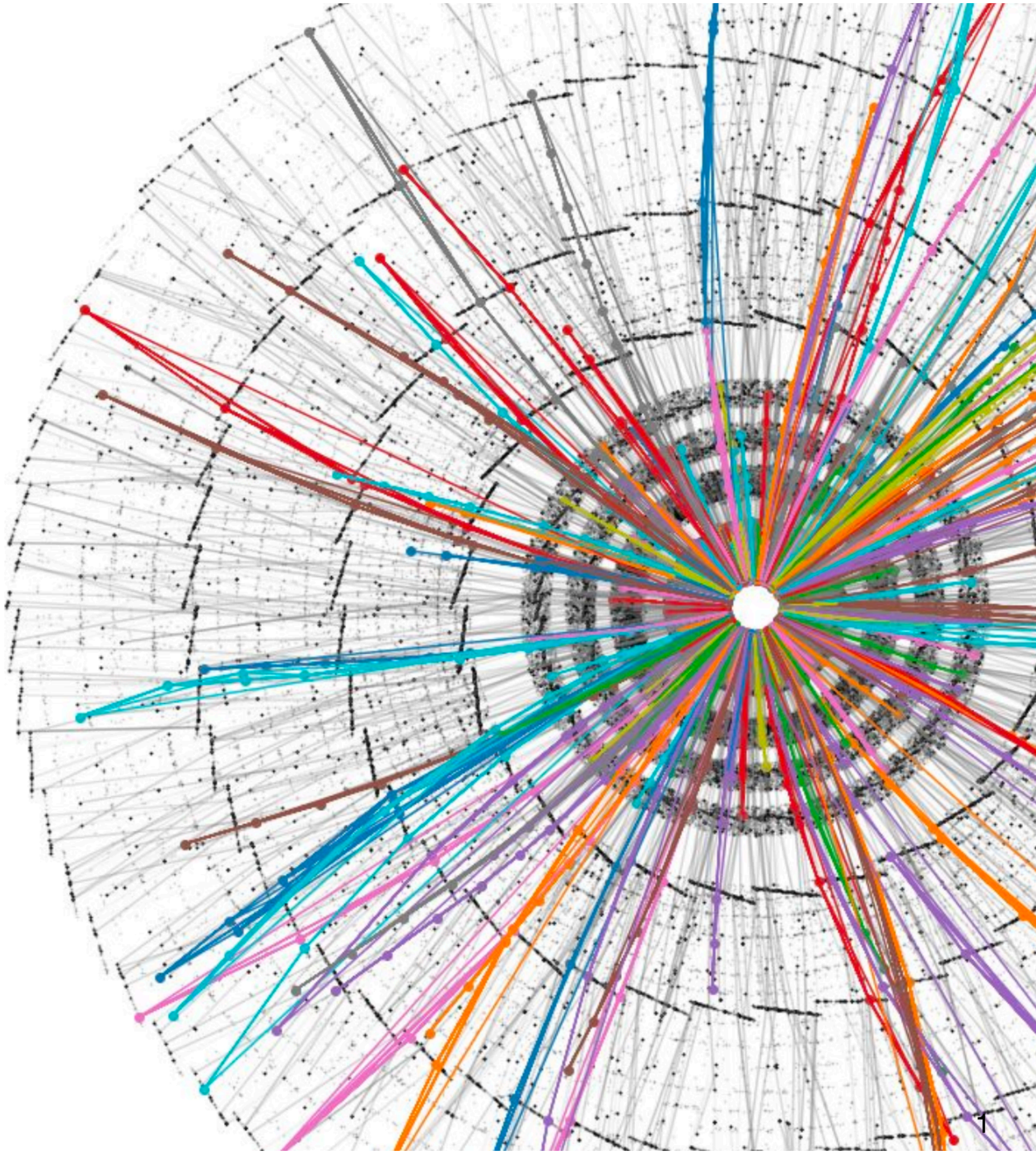
Exa.TrkX

- Exa.TrkX is a collaboration developing next-generation Graph Neural Network (GNN) reconstruction for HEP:
  - **Energy Frontier**
    - Expand on HEP.TrkX's prototype GNN for HL-LHC.
    - Incorporate into ATLAS's simulation and validation chain.



Exa.TrkX

- Exa.TrkX is a collaboration developing next-generation Graph Neural Network (GNN) reconstruction for HEP:
  - **Energy Frontier**
    - Expand on HEP.TrkX's prototype GNN for HL-LHC.
    - Incorporate into ATLAS's simulation and validation chain.
  - **Intensity Frontier**
    - Explore viability of HEP.TrkX network for neutrino physics.
    - Develop GNN-based reconstruction for Liquid Argon TPCs.

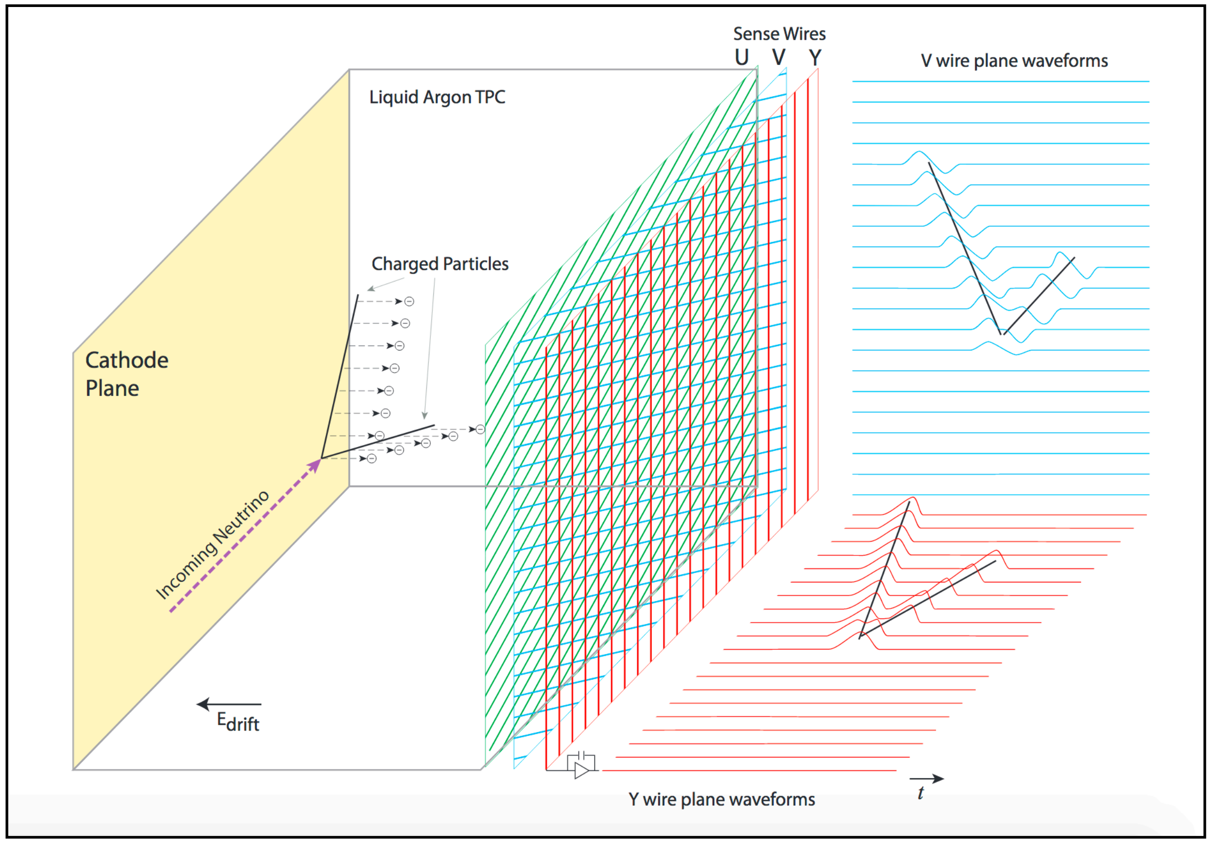


University of CINCINNATI

Liquid Argon TPCs

- Liquid Argon Time Projection Chambers (LArTPCs) currently a heavily utilised detector technology in neutrino physics.
  - At FNAL: MicroBooNE, Icarus, SBND.
  - Future: DUNE (70kT LArTPC deep underground, plus near detector).

- Charged particles ionize liquid argon as they travel.
- Ionisation electrons drift due to HV electrode field, and are collected by anode wires.
- Wire spacing ~3mm – **high-resolution detector**.



GNN reconstruction for Neutrino Physics

- The work presented here today utilises **simulated neutrinos** from **MicroBooNE's Open Data Release** (link).
  - Simulated neutrino interactions with cosmic data overlays.

GNN reconstruction for Neutrino Physics

- The work presented here today utilises **simulated neutrinos** from **MicroBooNE's Open Data Release** (link).
    - Simulated neutrino interactions with cosmic data overlays.
- This network architecture was originally developed in the context of the **DUNE Far Detector** geometry.
    - Motivation: reconstructing complex and high-multiplicity atmospheric and ν_τ interactions.

GNN reconstruction for Neutrino Physics

- The work presented here today utilises **simulated neutrinos** from **MicroBooNE's Open Data Release** (link).
  - Simulated neutrino interactions with cosmic data overlays.
- This network architecture was originally developed in the context of the **DUNE Far Detector** geometry.
  - Motivation: reconstructing complex and high-multiplicity atmospheric and ν_τ interactions.
- This network architecture is developed to have **broad applicability**, without being tied to any particular detector geometry.
  - Also deployed on non-LArTPC detector technology!

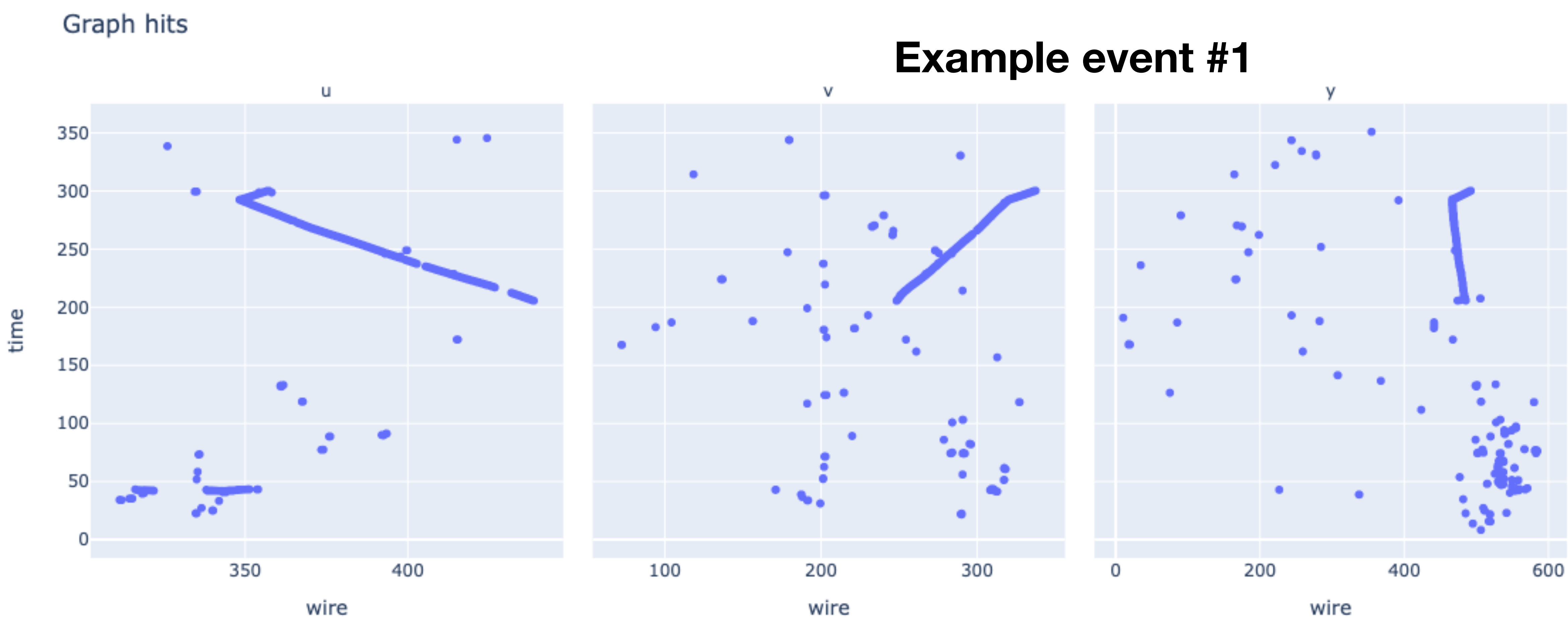
Event graphs for neutrino physics

- We describe a physics interaction as a **heterogeneous graph**, with each plane's detector hits acting as the nodes of an independent subgraph.

Event graphs for neutrino physics

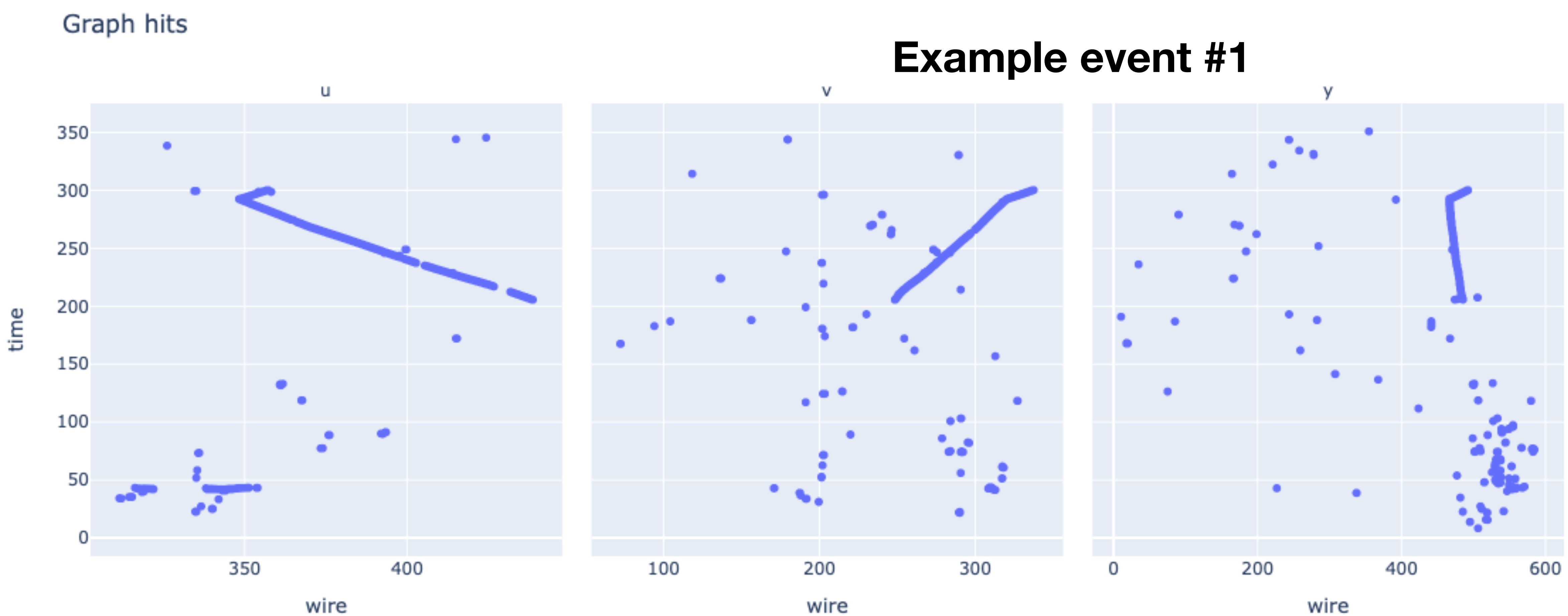
- We describe a physics interaction as a **heterogeneous graph**, with each plane's detector hits acting as the nodes of an independent subgraph.

Example event #1



Event graphs for neutrino physics

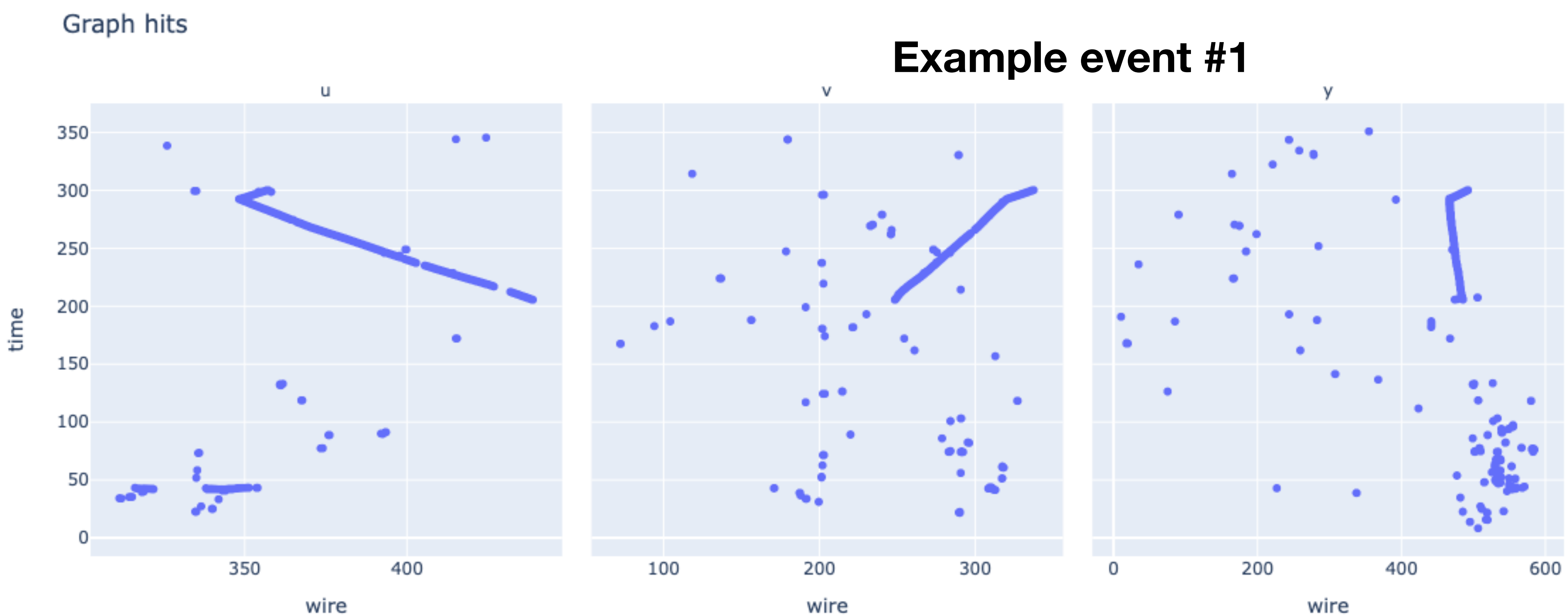
- We describe a physics interaction as a **heterogeneous graph**, with each plane's detector hits acting as the nodes of an independent subgraph.
  - Each hit node is described by four input features: **wire index**, **hit time**, **integral** and **RMS width**.



Event graphs for neutrino physics

- We describe a physics interaction as a **heterogeneous graph**, with each plane's detector hits acting as the nodes of an independent subgraph.
  - Each hit node is described by four input features: **wire index**, **hit time**, **integral** and **RMS width**.
  - Edges are formed for each planar subgraph using the **Delaunay triangulation** algorithm.

Example event #1



NuGraph2 architecture

- NuGraph2's core convolution engine is a **self-attention message passing network** utilising a **categorical embedding**.

NuGraph2 architecture

- NuGraph2's core convolution engine is a **self-attention message passing network** utilising a **categorical embedding**.
  - Each particle category is provided with a separate set of embedded features, which are convolved independently.

NuGraph2 architecture

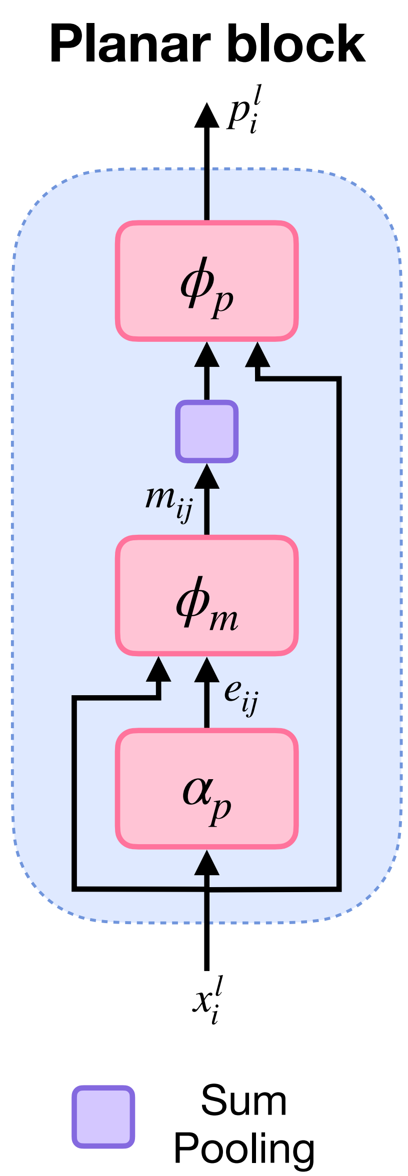
- NuGraph2's core convolution engine is a **self-attention message passing network** utilising a **categorical embedding**.
  - Each particle category is provided with a separate set of embedded features, which are convolved independently.
  - Context information is exchanged between different particle types via a **categorical cross-attention** mechanism.

NuGraph2 architecture

- NuGraph2's core convolution engine is a **self-attention message passing network** utilising a **categorical embedding**.
    - Each particle category is provided with a separate set of embedded features, which are convolved independently.
    - Context information is exchanged between different particle types via a **categorical cross-attention** mechanism.
- Each message-passing iteration consists of two phases, the **planar** step and the **nexus** step:

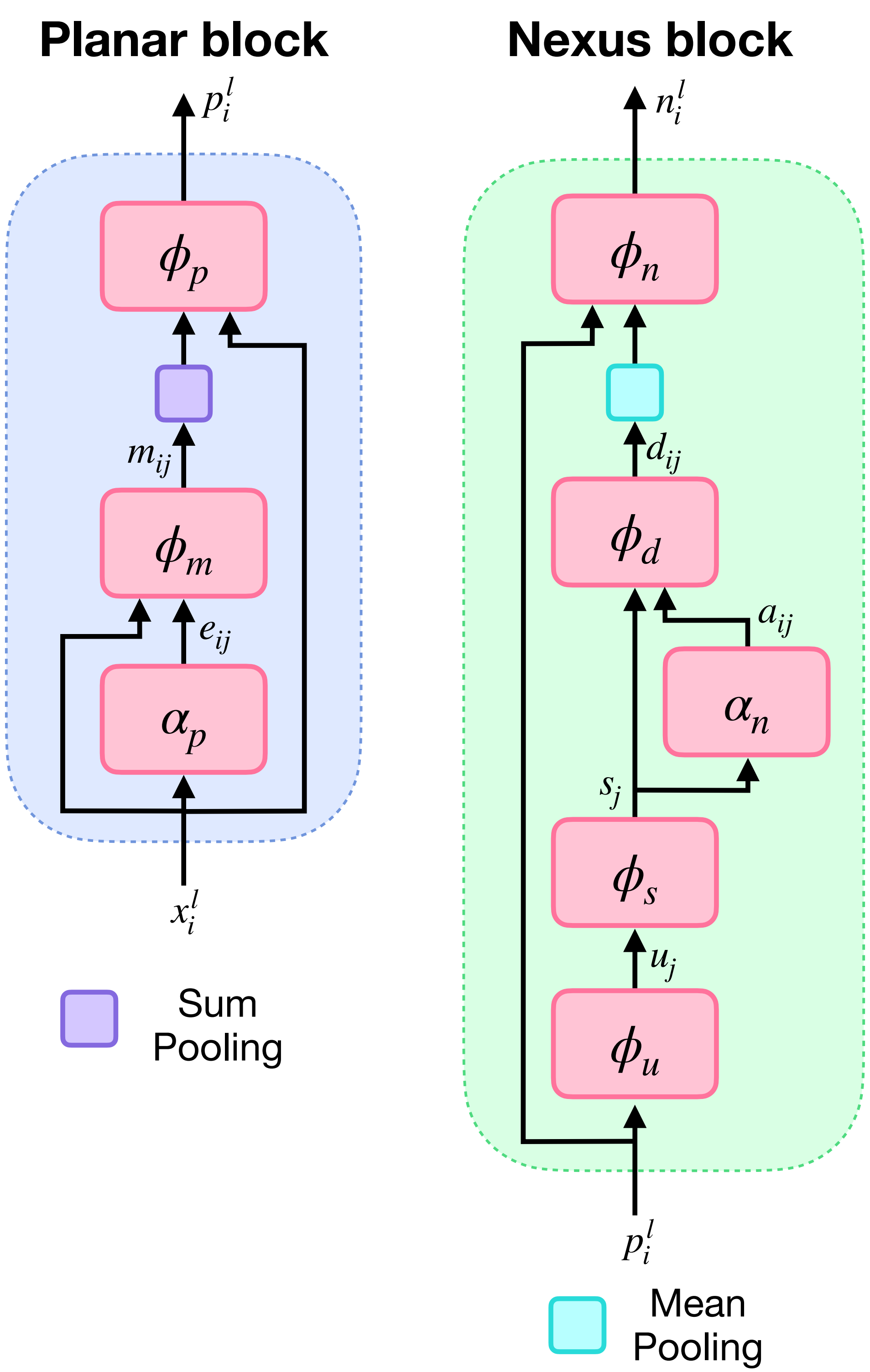
NuGraph2 architecture

- NuGraph2's core convolution engine is a **self-attention message passing network** utilising a **categorical embedding**.
  - Each particle category is provided with a separate set of embedded features, which are convolved independently.
  - Context information is exchanged between different particle types via a **categorical cross-attention** mechanism.
- Each message-passing iteration consists of two phases, the **planar** step and the **nexus** step:
  - Pass messages internally in each plane.



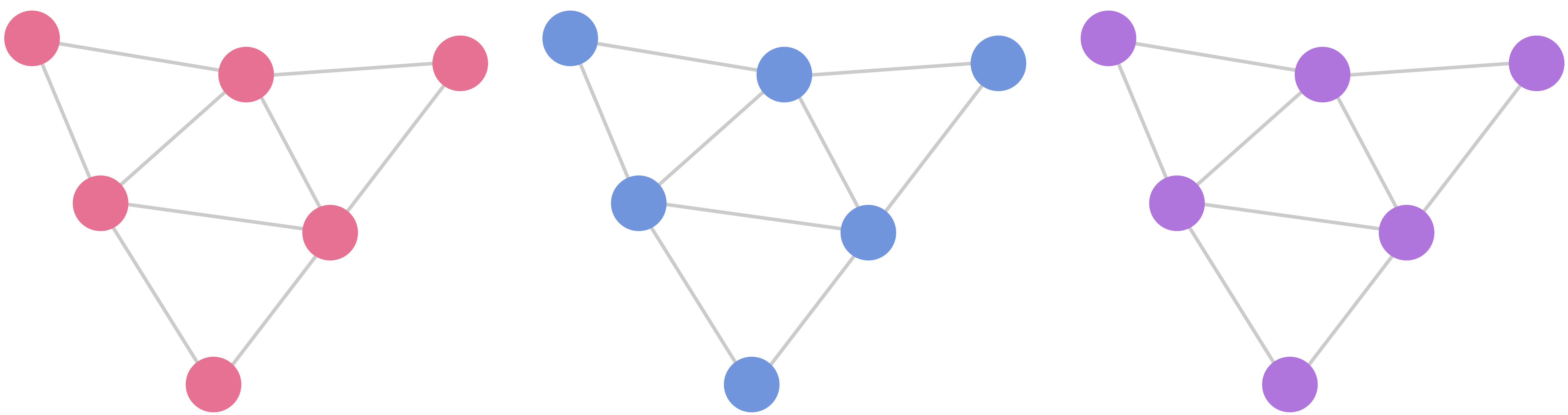
NuGraph2 architecture

- NuGraph2's core convolution engine is a **self-attention message passing network** utilising a **categorical embedding**.
  - Each particle category is provided with a separate set of embedded features, which are convolved independently.
  - Context information is exchanged between different particle types via a **categorical cross-attention** mechanism.
- Each message-passing iteration consists of two phases, the **planar** step and the **nexus** step:
  - Pass messages internally in each plane.
  - Pass messages up to 3D nexus nodes to share context information.



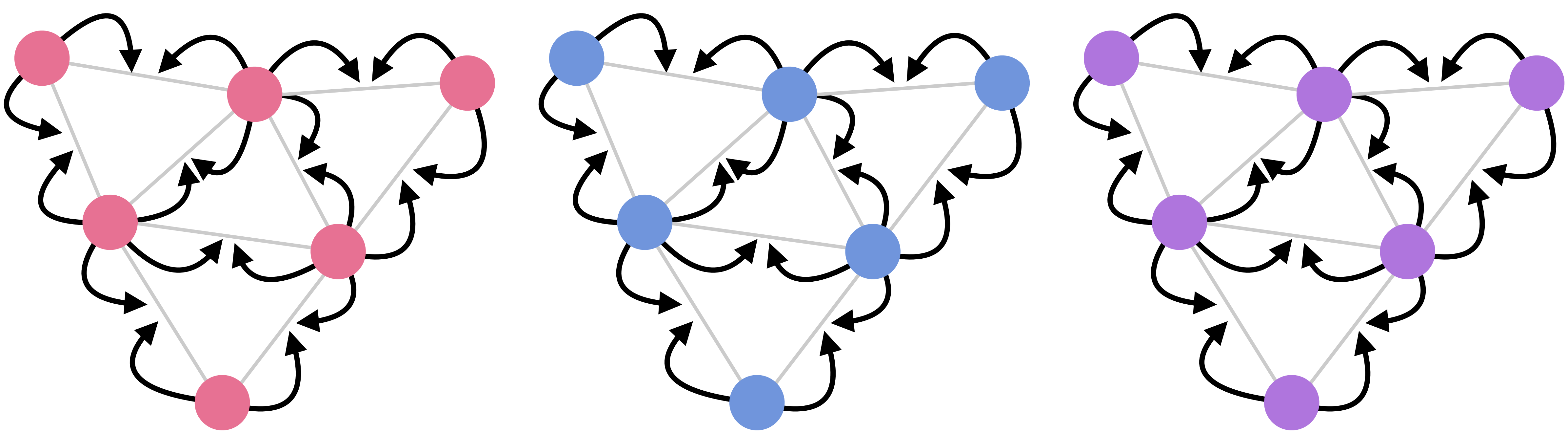
Nexus mechanism

- Perform message-passing independently in each detector view.



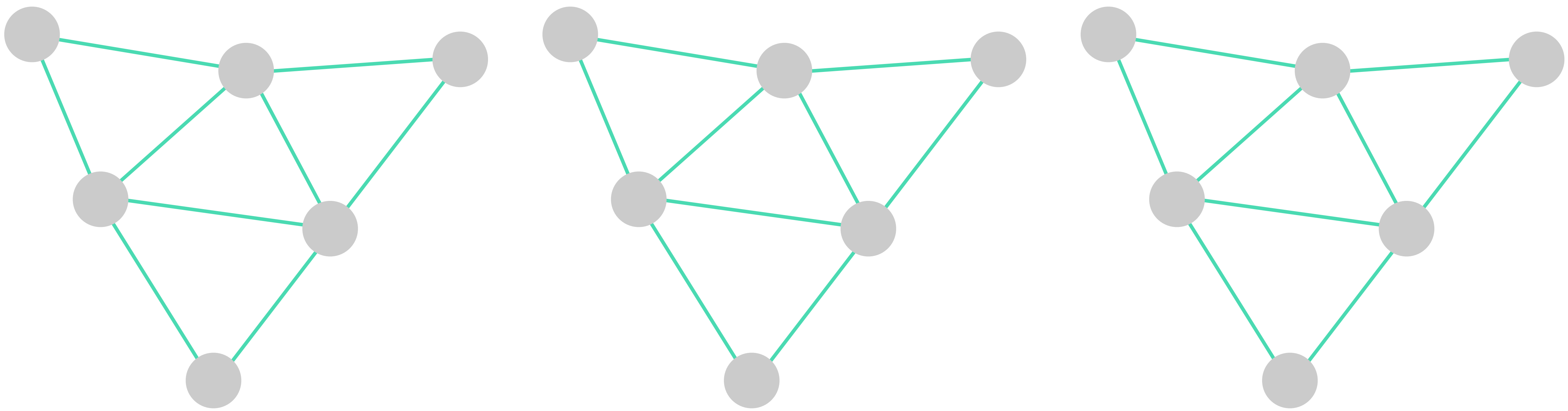
Nexus mechanism

- Perform message-passing independently in each detector view.



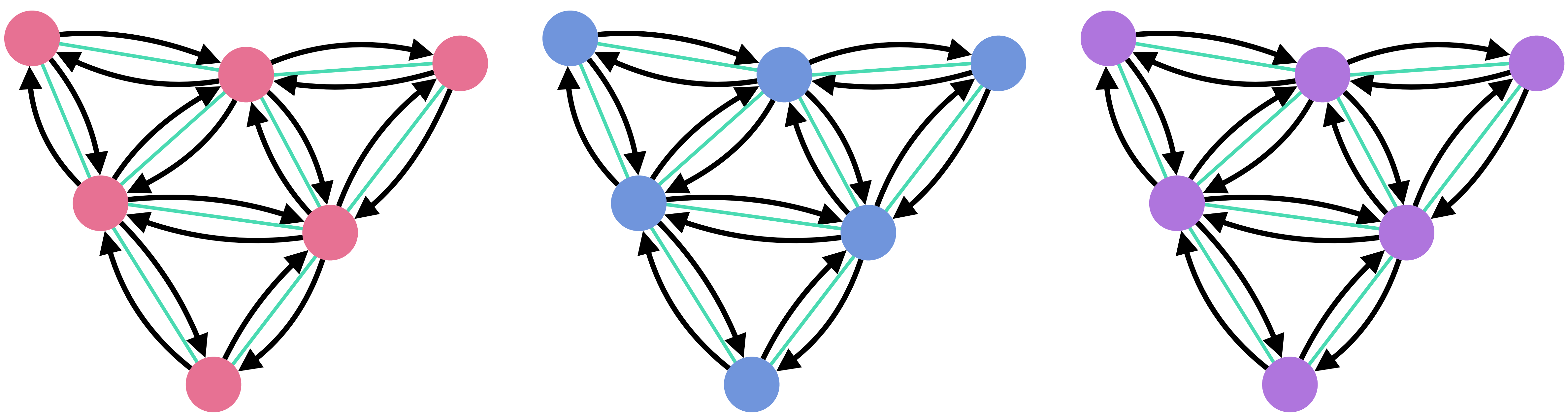
Nexus mechanism

- Perform message-passing independently in each detector view.



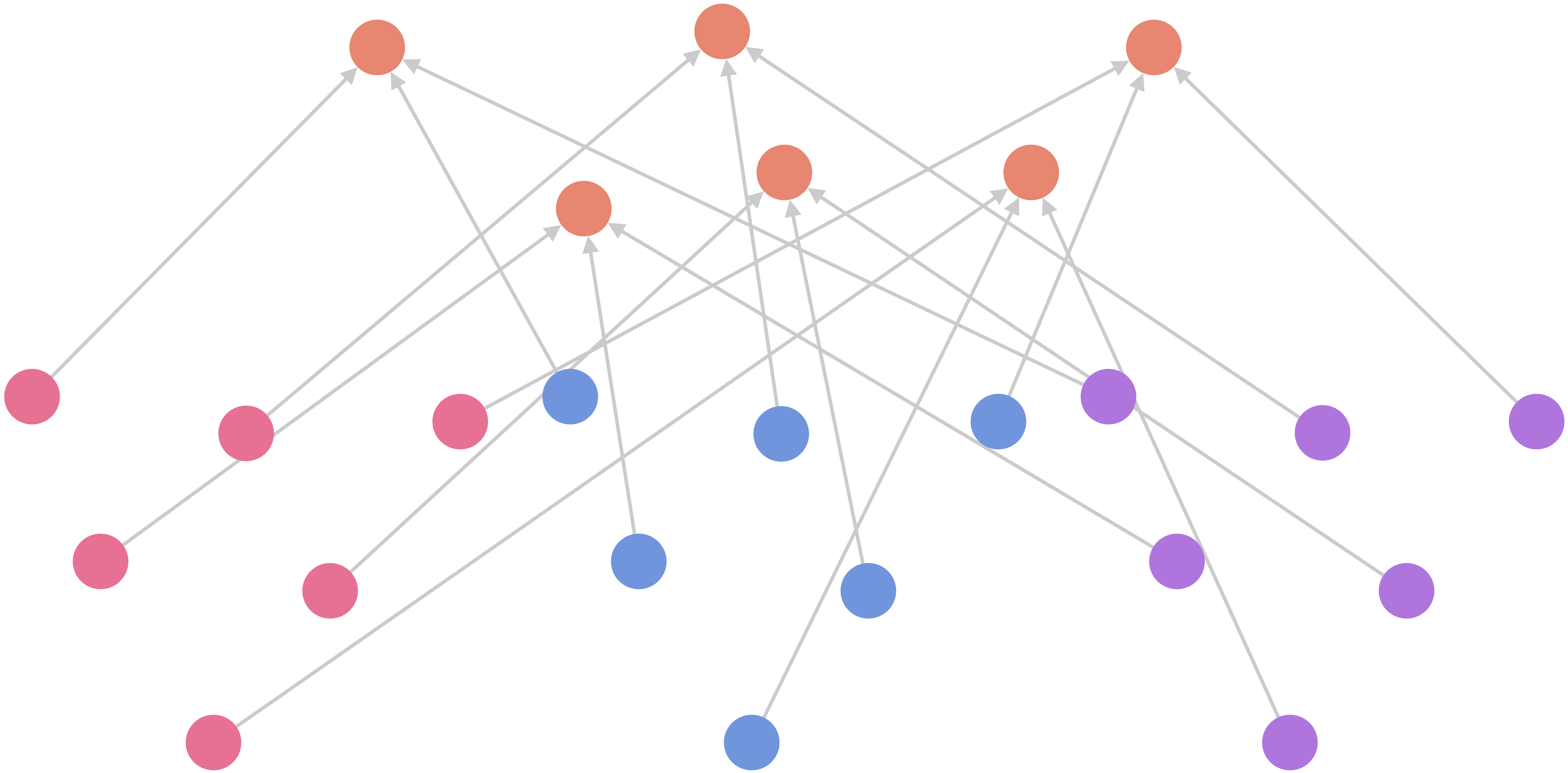
Nexus mechanism

- Perform message-passing independently in each detector view.



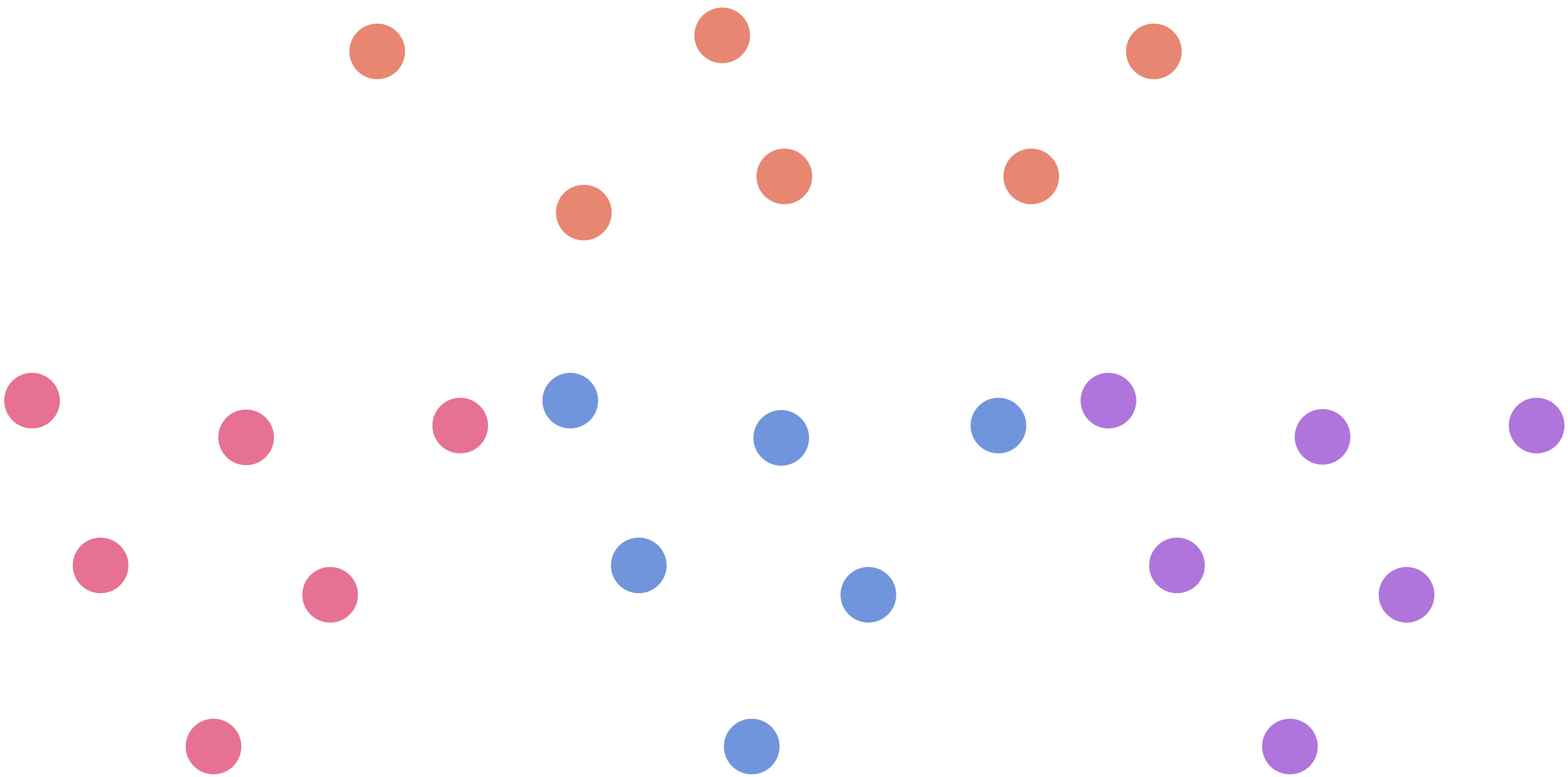
Nexus mechanism

- Propagate 2D node features to **nexus nodes** generated from simple spacepoint reconstruction.



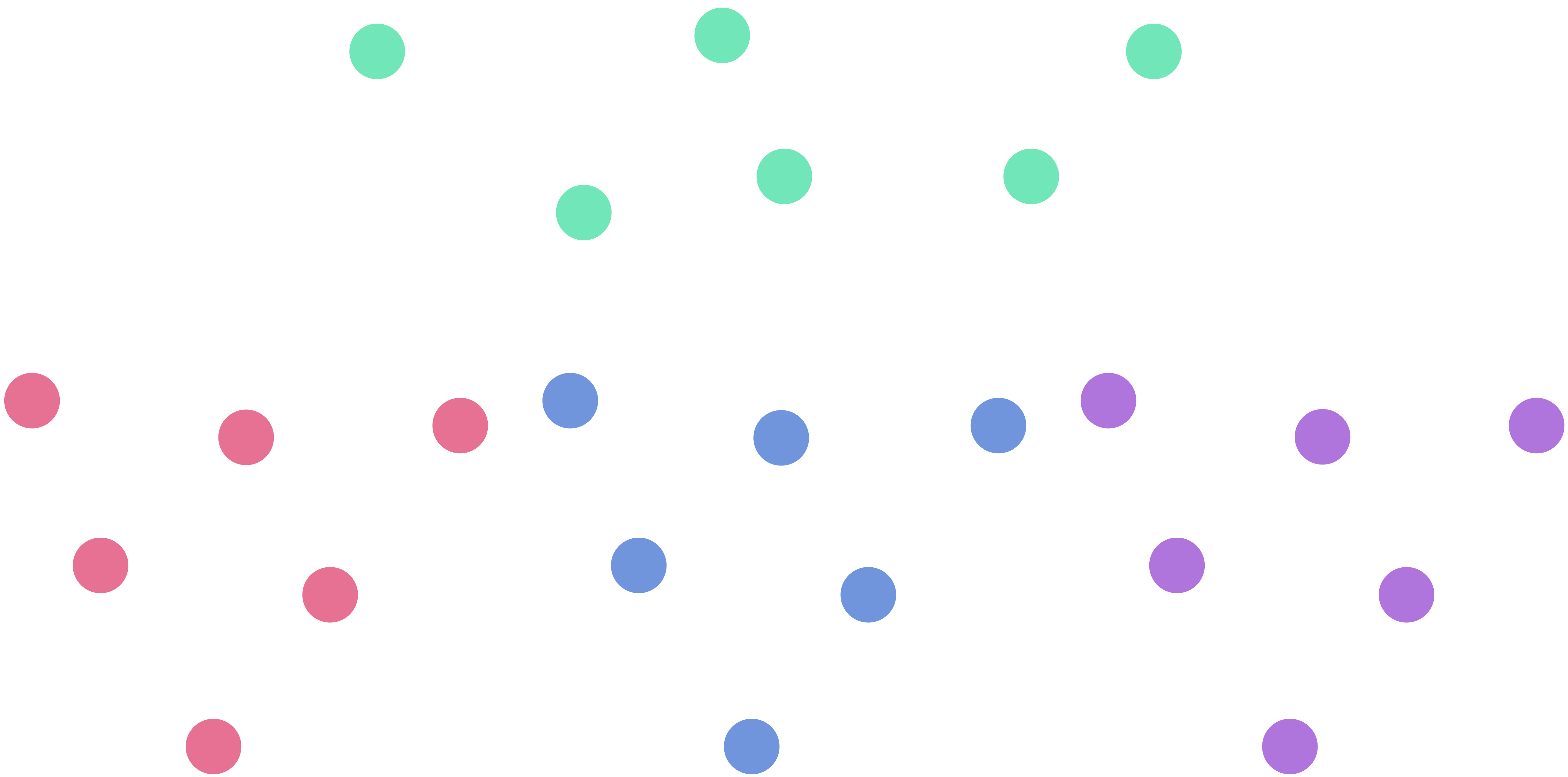
Nexus mechanism

- Convolve **nexus node features** to mix information between detector planes.



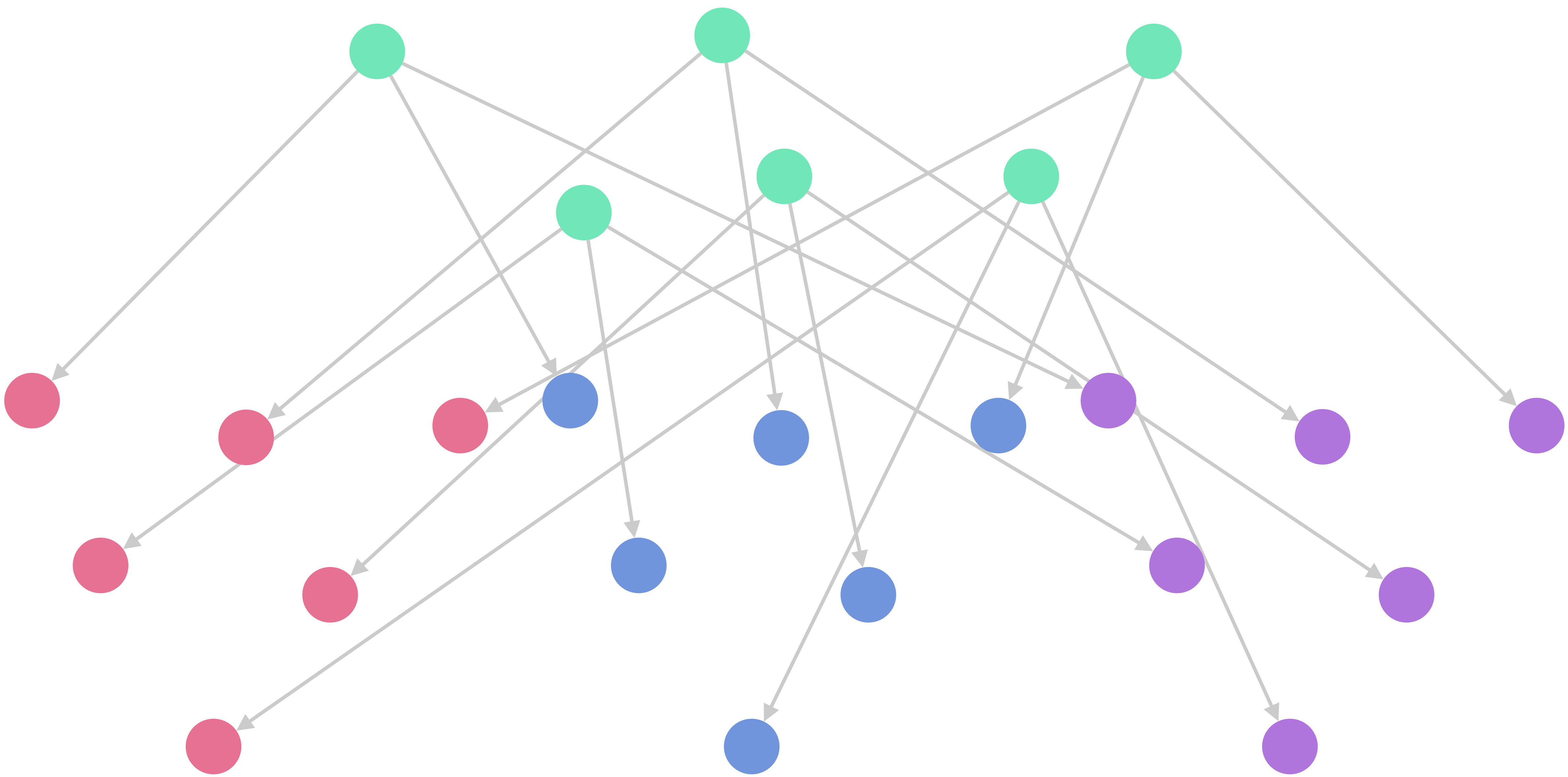
Nexus mechanism

- Convolve **nexus node features** to mix information between detector planes.



Nexus mechanism

- Propagate 3D **nexus nodes features** back down to 2D planar nodes.



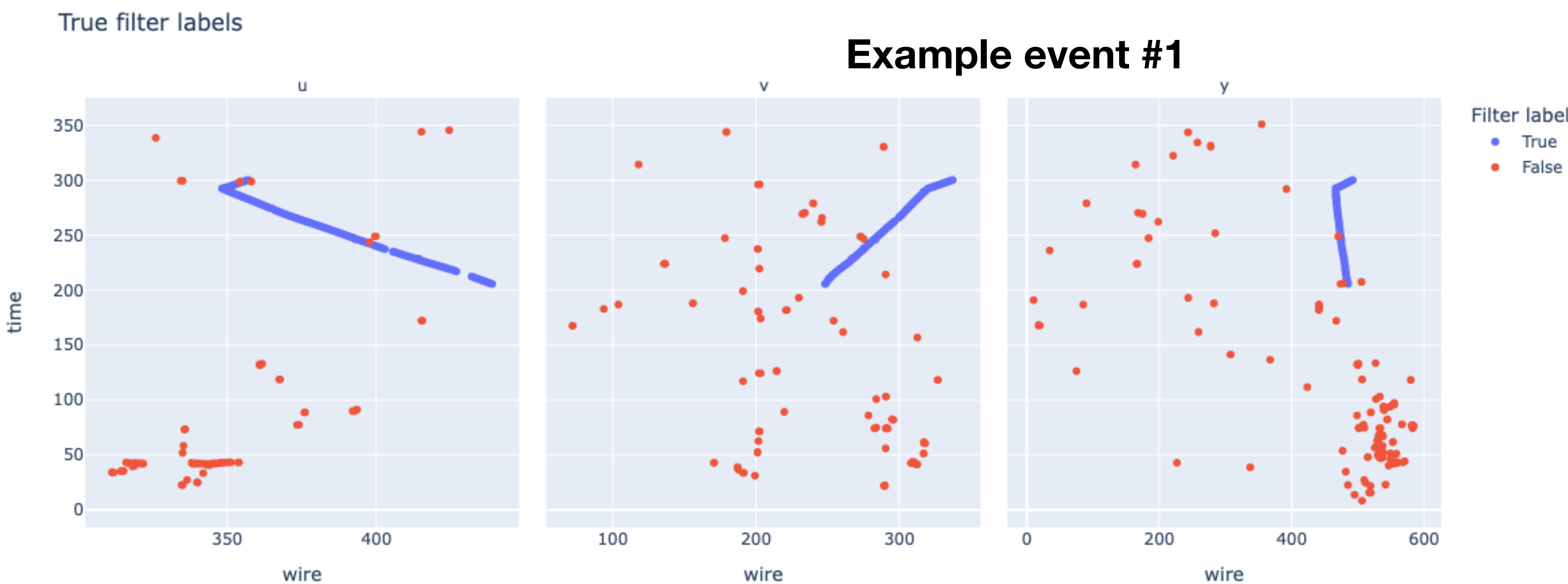
Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).

Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).

Example event #1

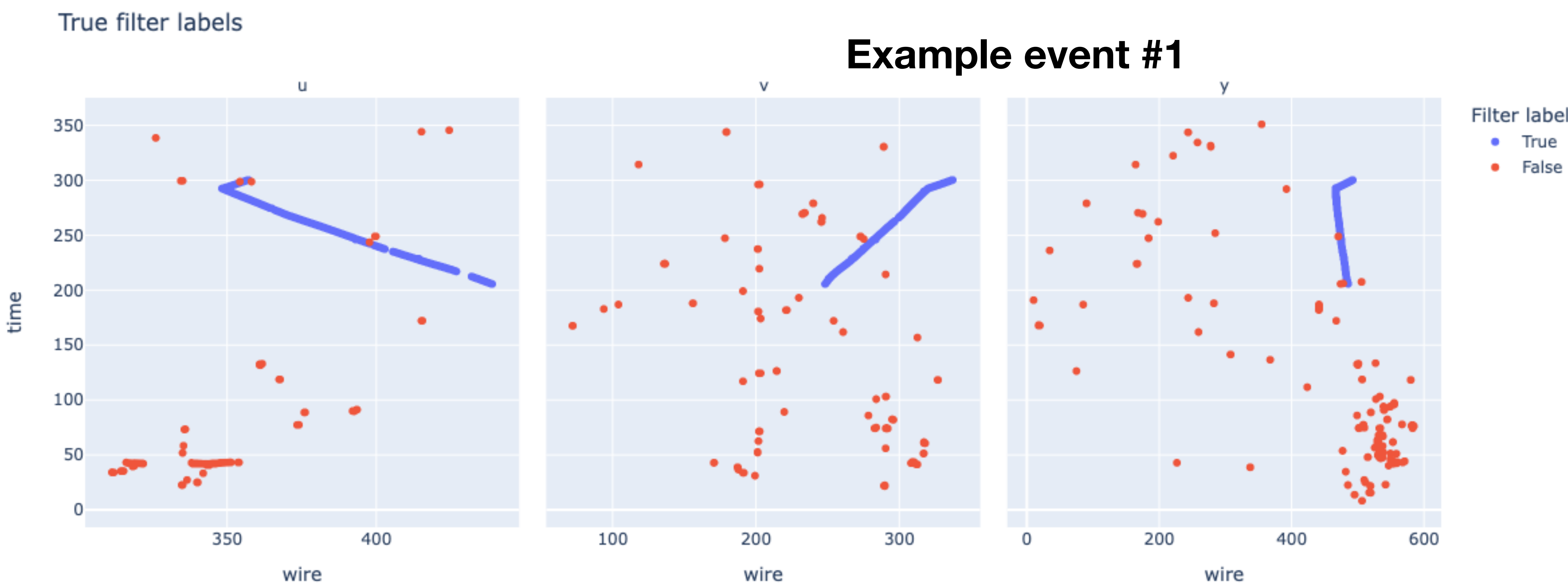


Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).

- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

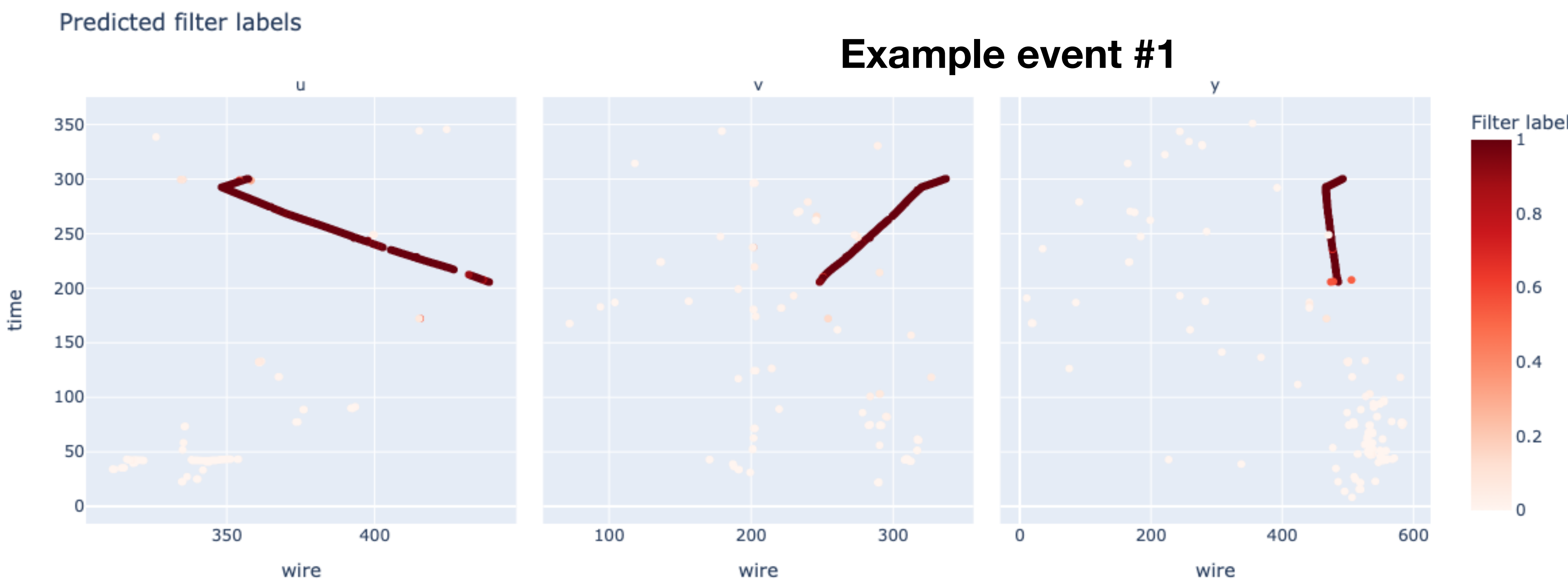
Example event #1



Background filtering

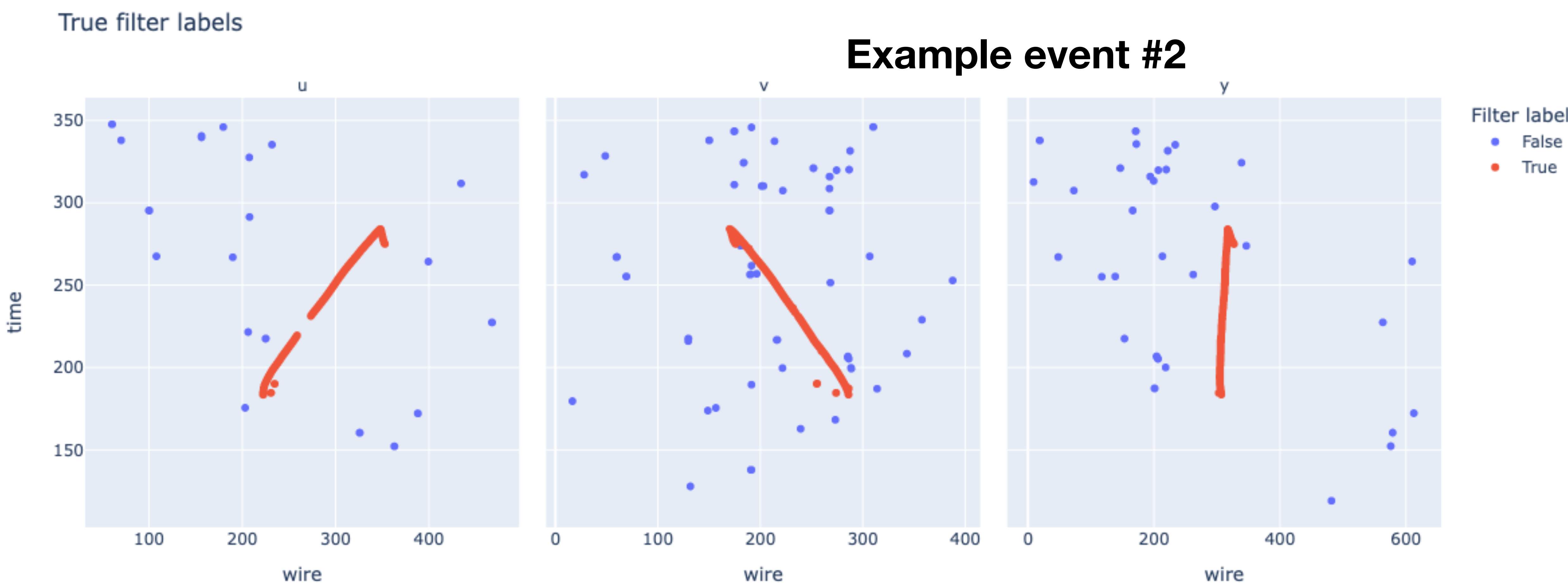
- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).
- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

Example event #1



Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).
- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

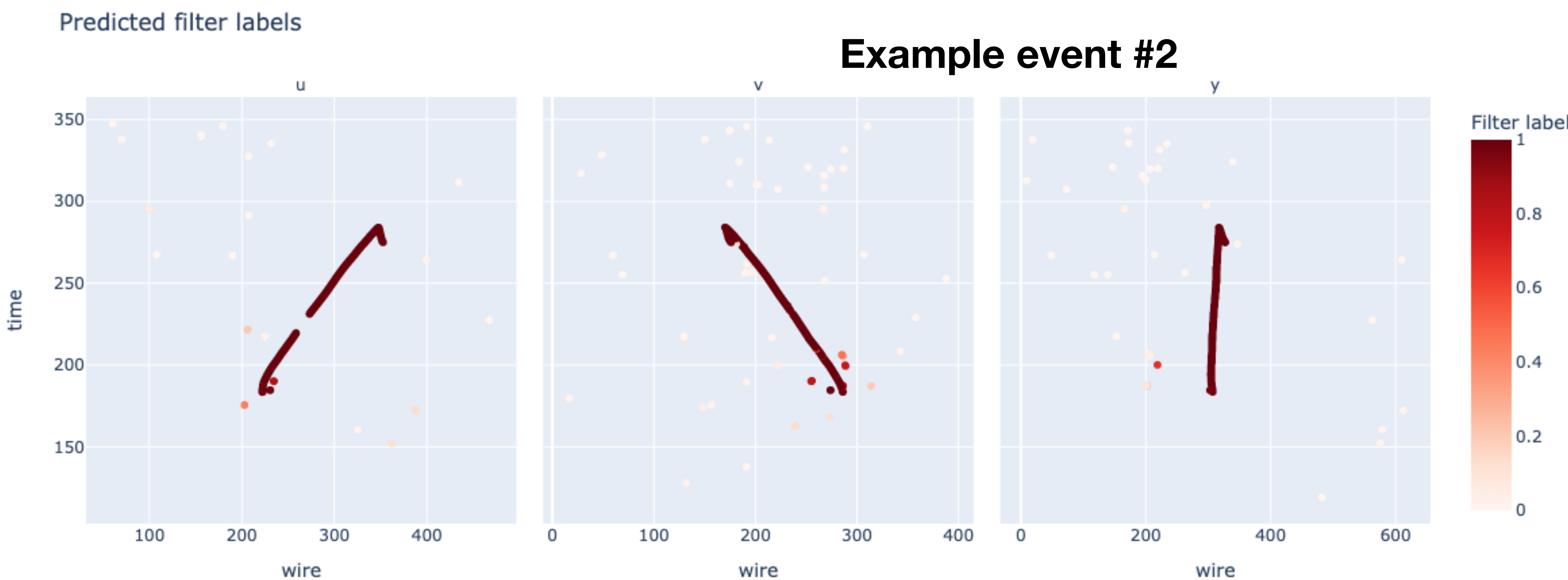


Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).

- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

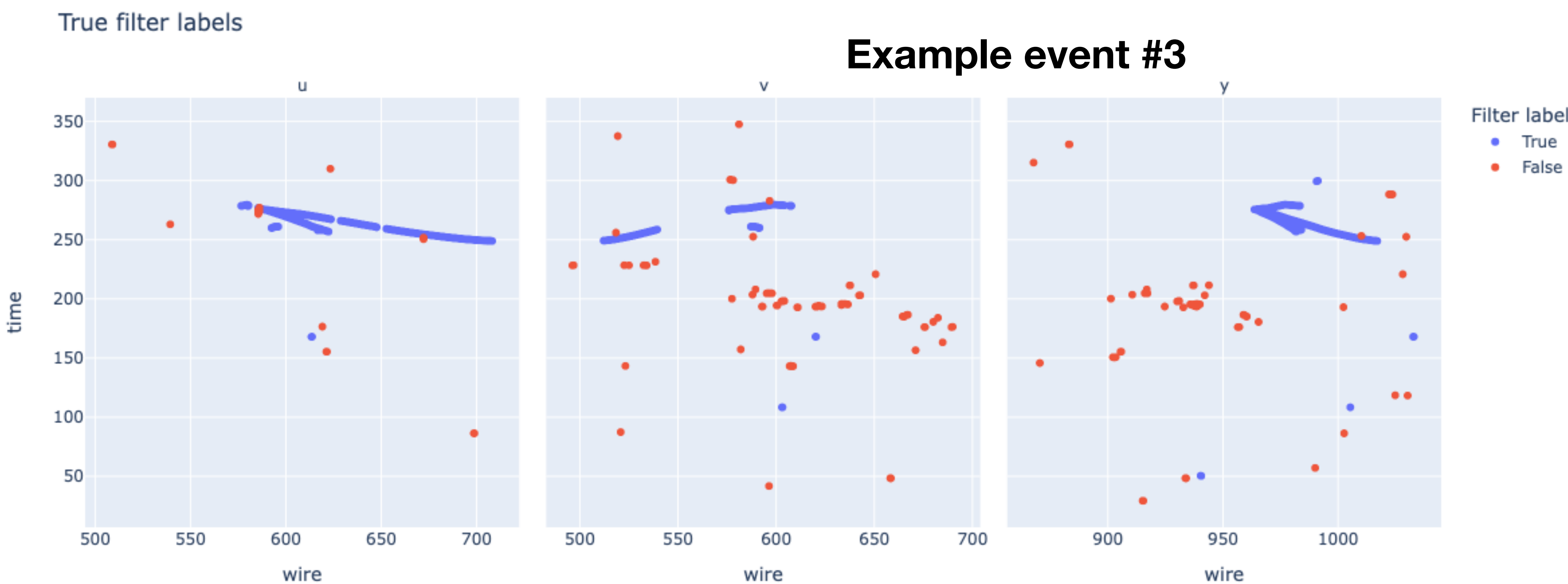
Example event #2



Background filtering

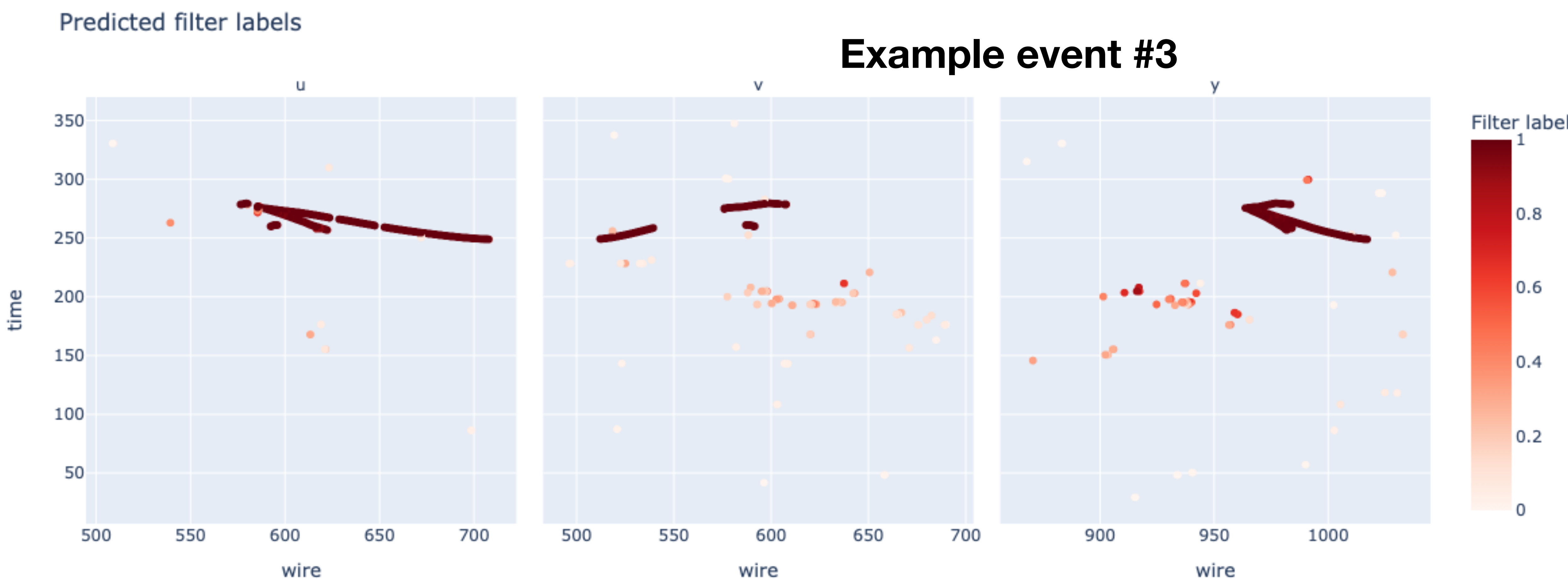
- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).
- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

Example event #3



Background filtering

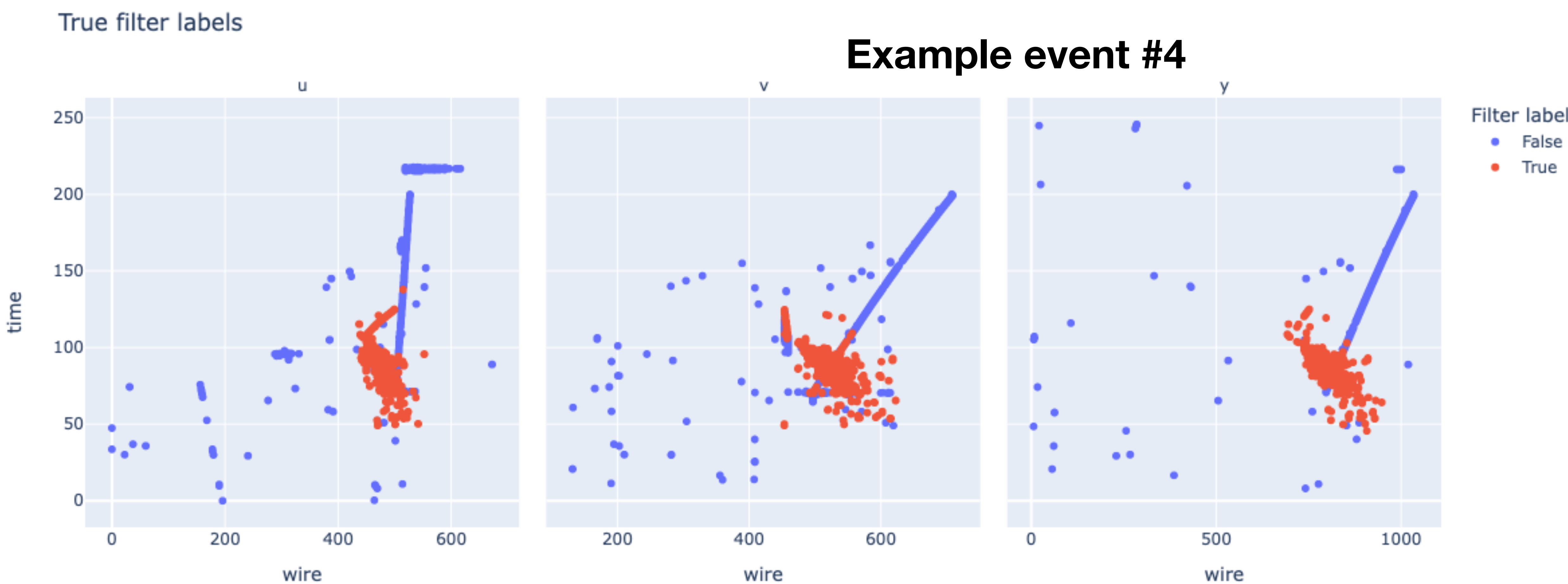
- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).
- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.



Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).
- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

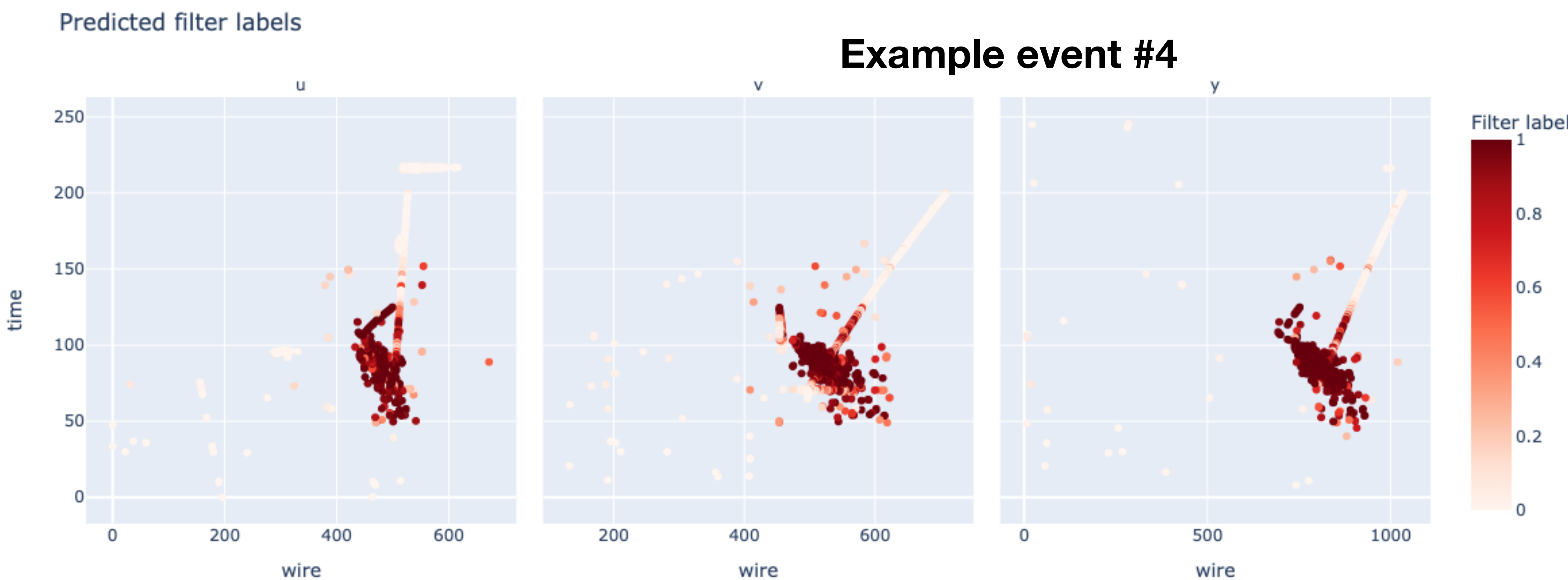
Example event #4



Background filtering

- For on-surface detectors, detector readout often contains a mixture of signal and background information (ie. cosmic interactions).
- Train a decoder to learn **a binary score for each hit**, determining whether it should be considered part of the physics interaction.

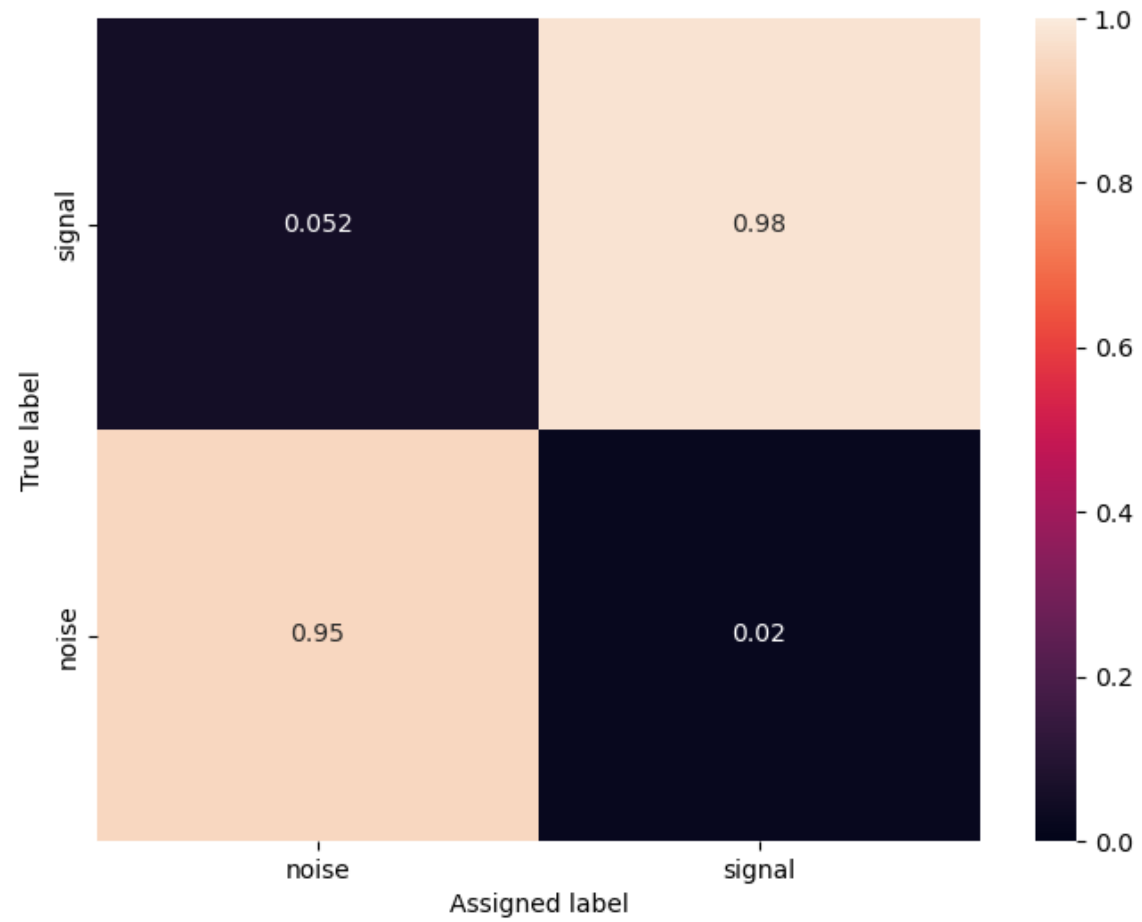
Example event #4



Background filtering results

- Performance metrics: **recall 0.978**, **precision 0.977**.

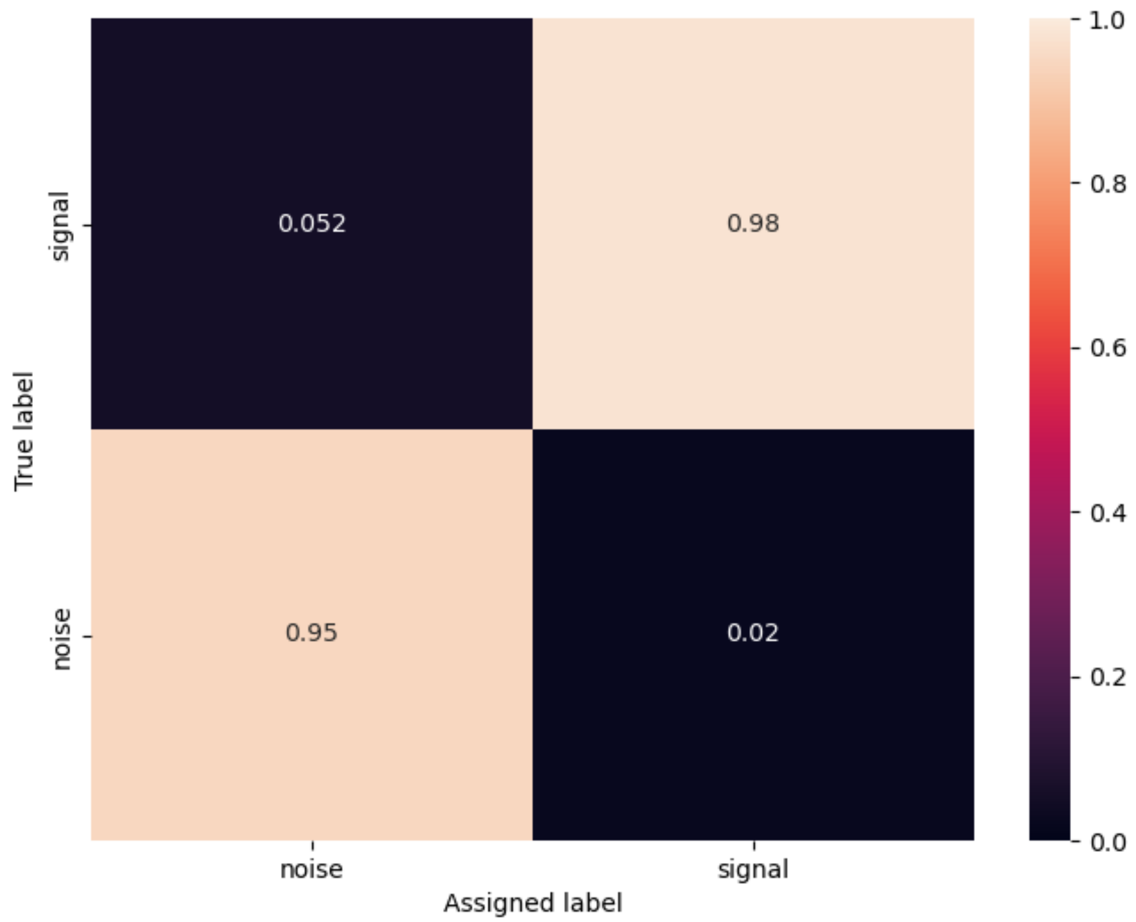
Recall matrix



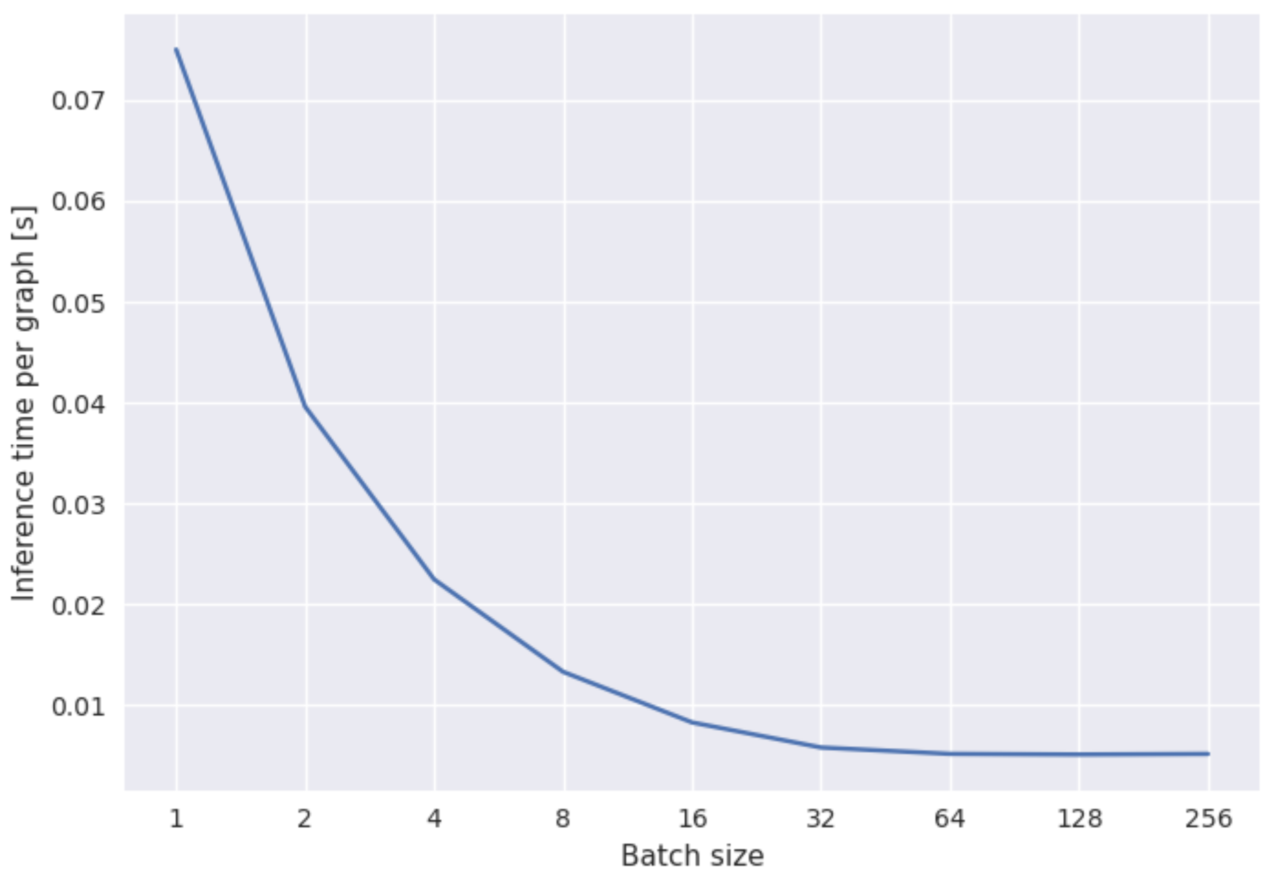
Background filtering results

- Performance metrics: **recall 0.978**, **precision 0.977**.
- Inference time: **0.12 s/evt on CPU**, **0.005s/evt batched on GPU**

Recall matrix



GPU inference time vs batch size



Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.
- Use five semantic categories:

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.

- Use five semantic categories:
  - **MIP: Minimum ionising particles (muons, pions etc)**

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.

- Use five semantic categories:
  - **MIP: Minimum ionising particles (muons, pions etc)**
  - **HIP: Highly ionising particles (protons, nucleons etc)**

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.

- Use five semantic categories:
  - **MIP: Minimum ionising particles (muons, pions etc)**
  - **HIP: Highly ionising particles (protons, nucleons etc)**
  - **EM showers**

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.
- Use five semantic categories:
  - **MIP: Minimum ionising particles (muons, pions etc)**
  - **HIP: Highly ionising particles (protons, nucleons etc)**
  - **EM showers**
  - **Michel electrons**

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.
- Use five semantic categories:
  - **MIP: Minimum ionising particles (muons, pions etc)**
  - **HIP: Highly ionising particles (protons, nucleons etc)**
  - **EM showers**
  - **Michel electrons**
  - **Diffuse activity (Compton scatters, neutrons etc)**

Hit classification

- The primary goal of NuGraph2 is to **classify each detector hit according to particle type**.

- Use five semantic categories:
  - **MIP: Minimum ionising particles (muons, pions etc)**
  - **HIP: Highly ionising particles (protons, nucleons etc)**
  - **EM showers**
  - **Michel electrons**
  - **Diffuse activity (Compton scatters, neutrons etc)**

- Going forward, will expand to more granular labelling schemes for possible $\mu/\pi/\kappa$ and e/γ separation.

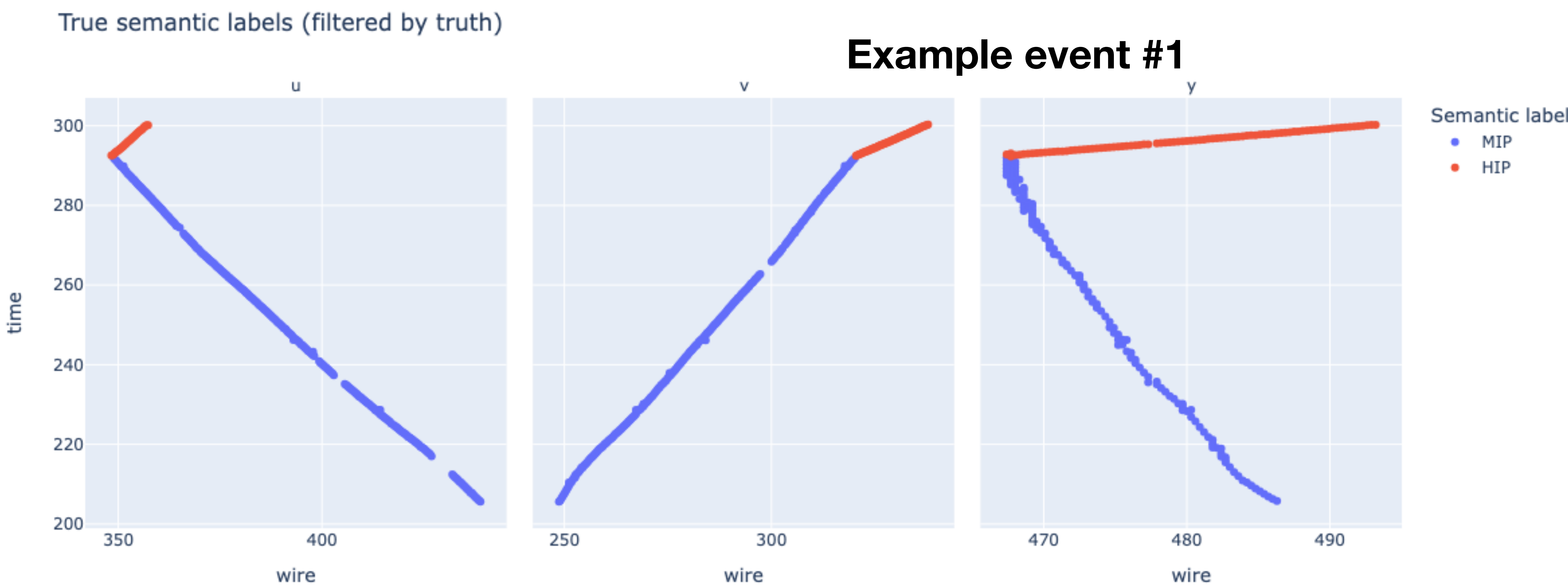
Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.

Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.

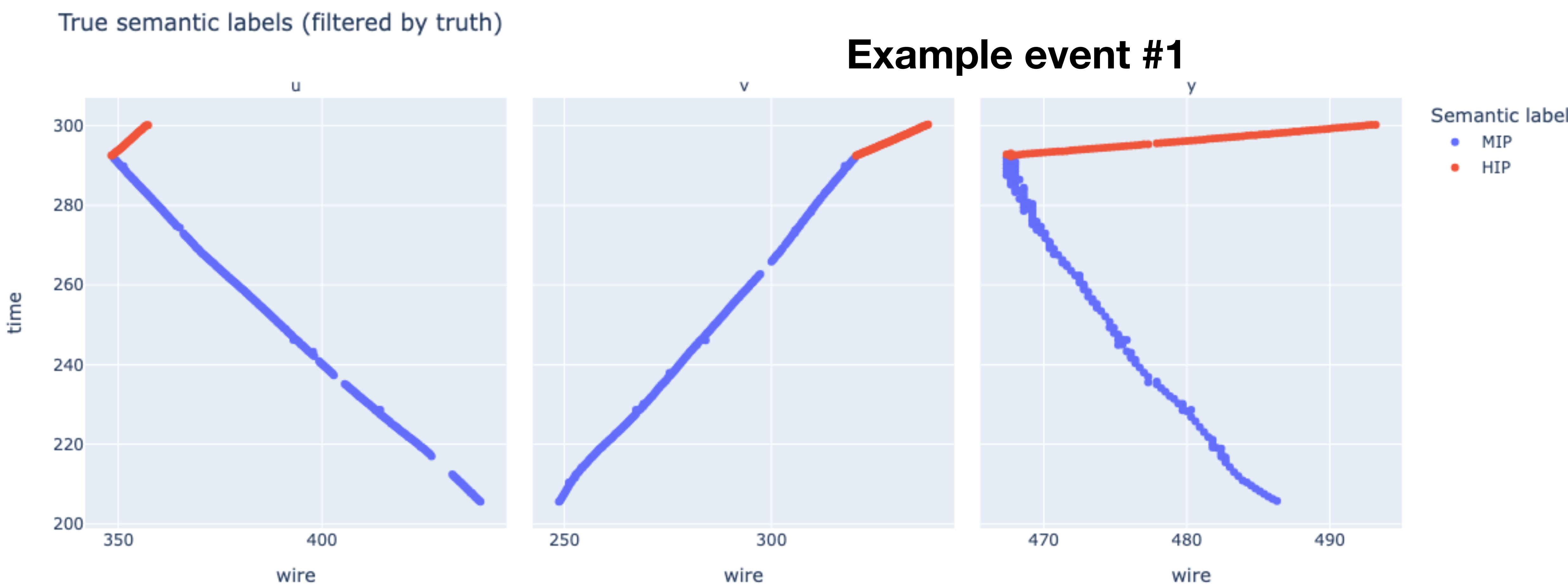
Example event #1



Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.
- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

Example event #1

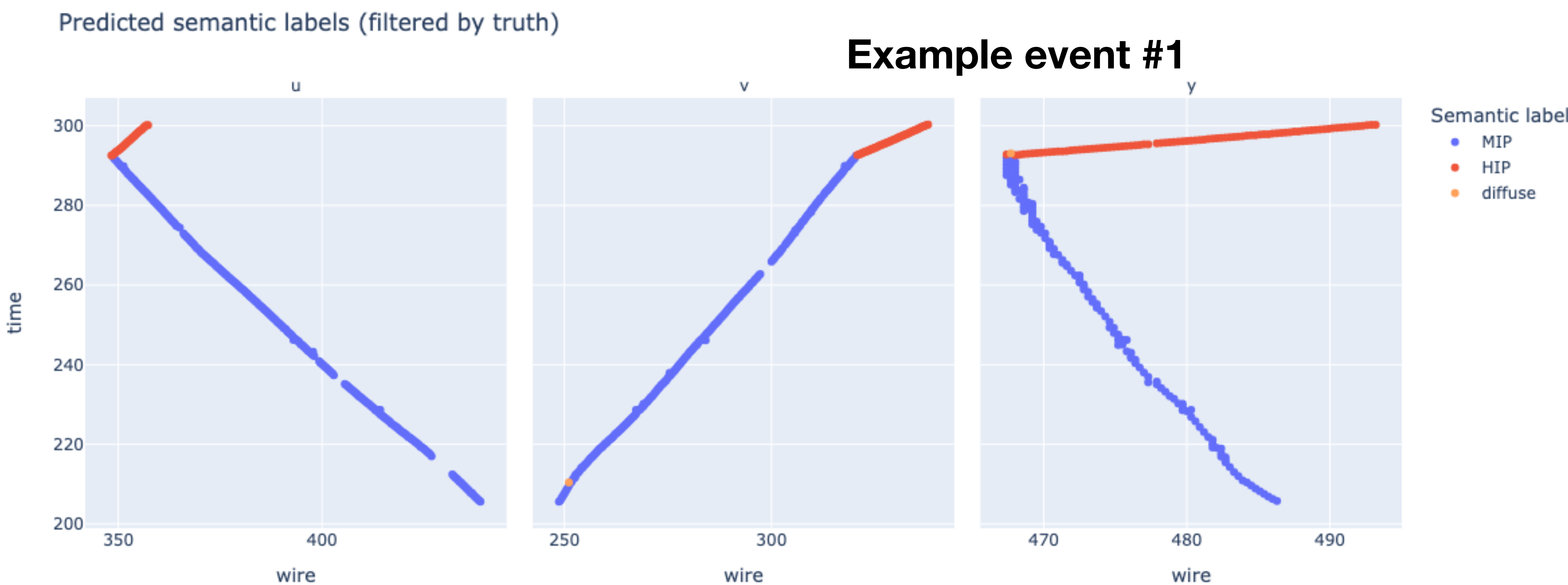


Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.

- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

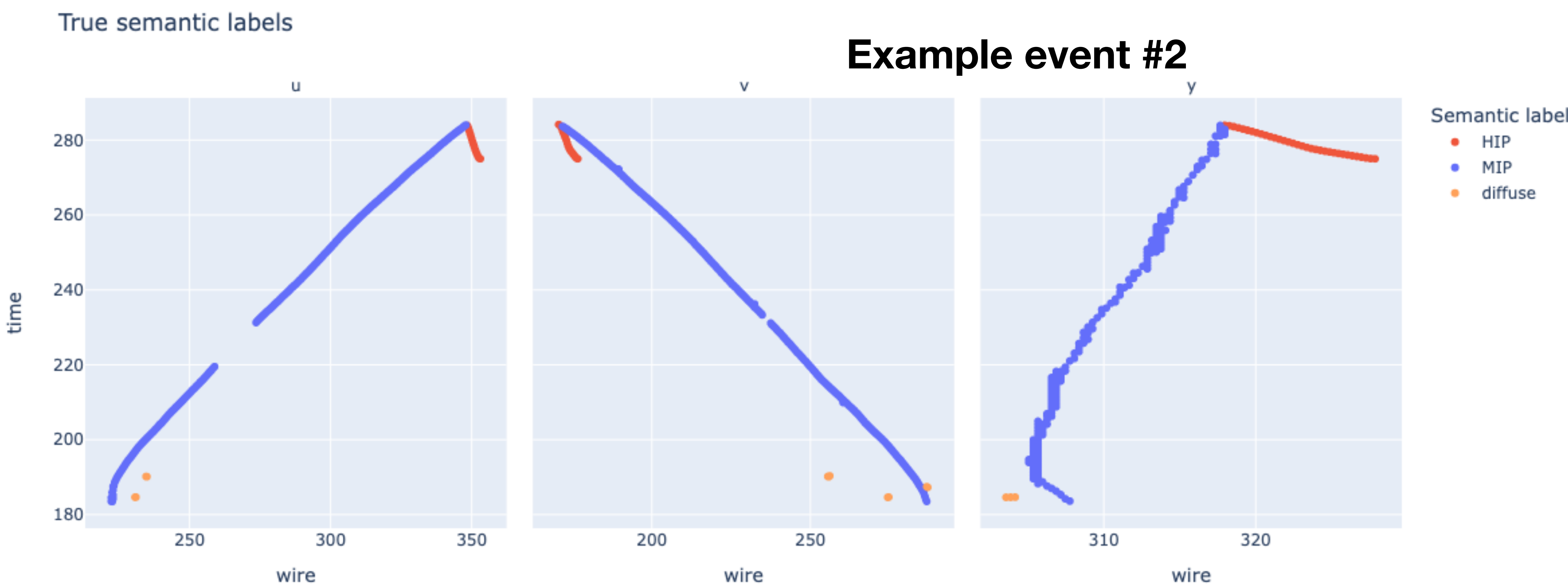
Example event #1



Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.
- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

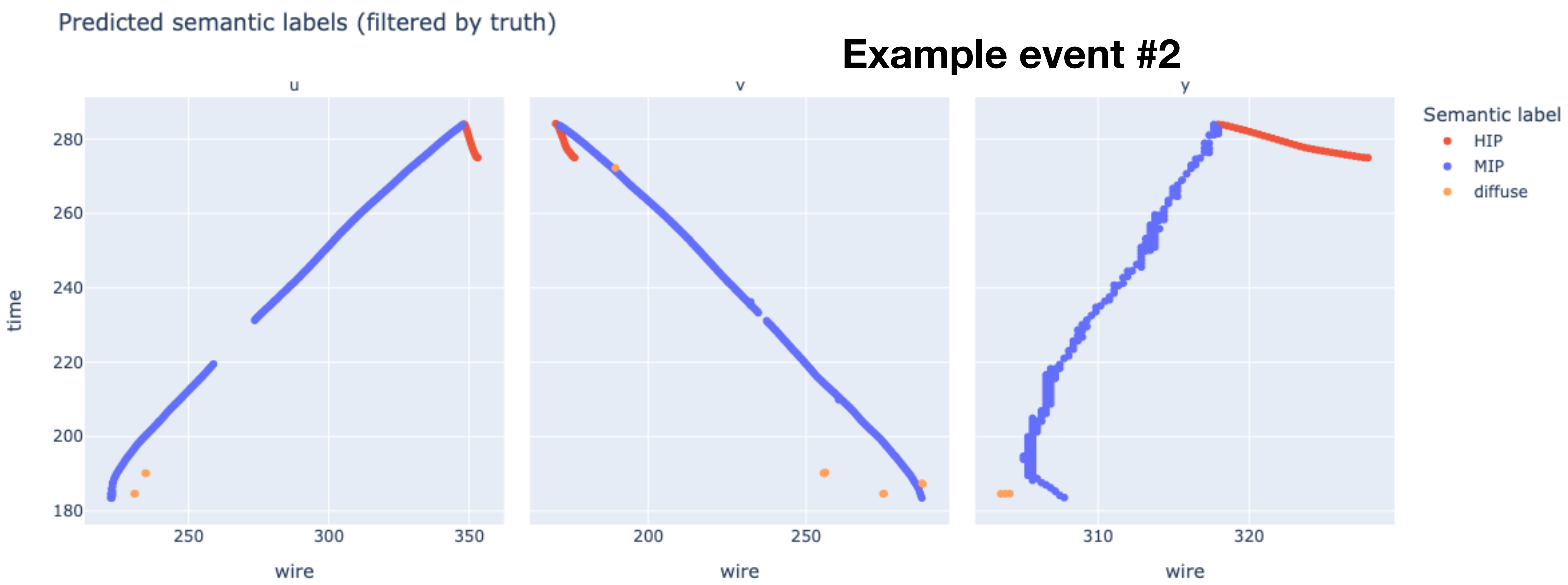
Example event #2



Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.
- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

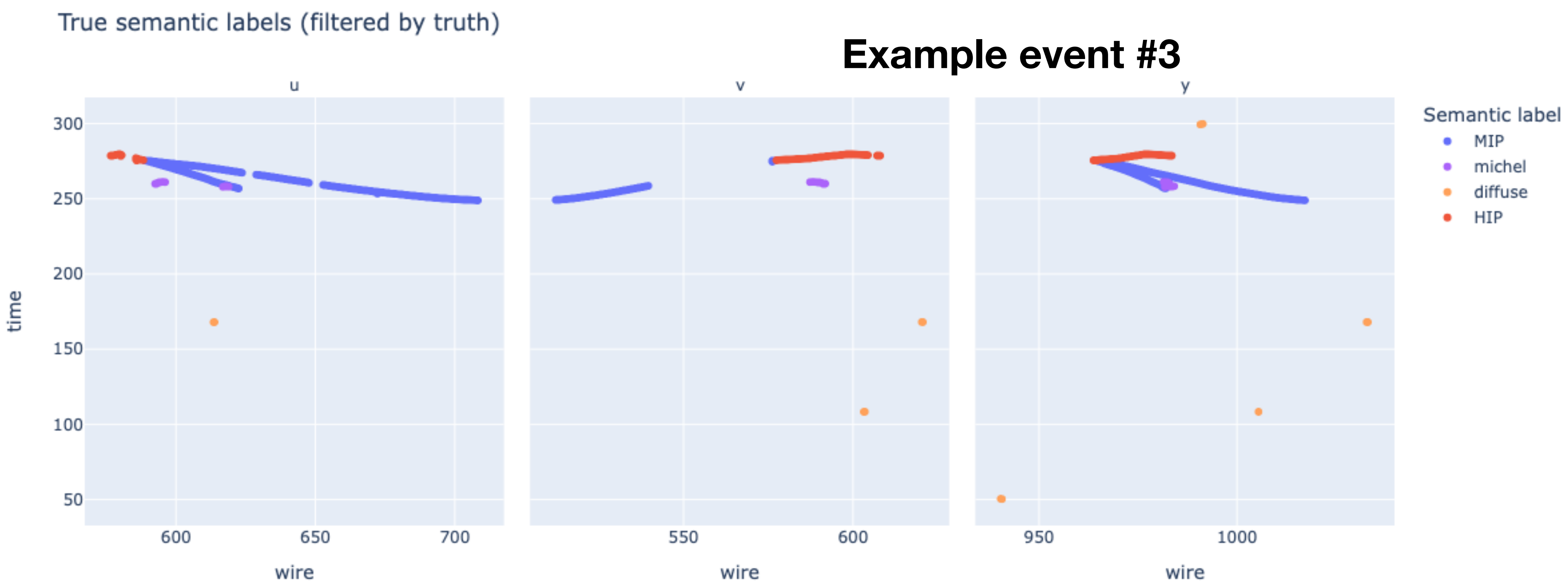
Example event #2



Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.
- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

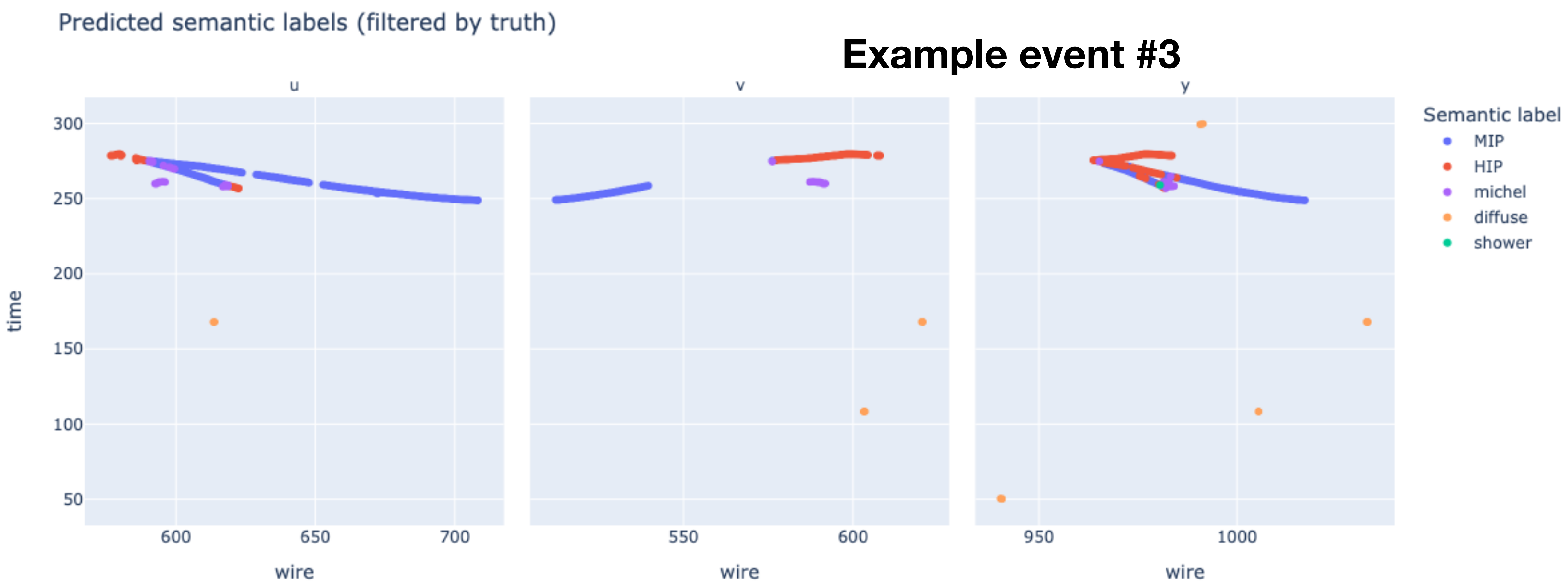
Example event #3



Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.
- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

Example event #3

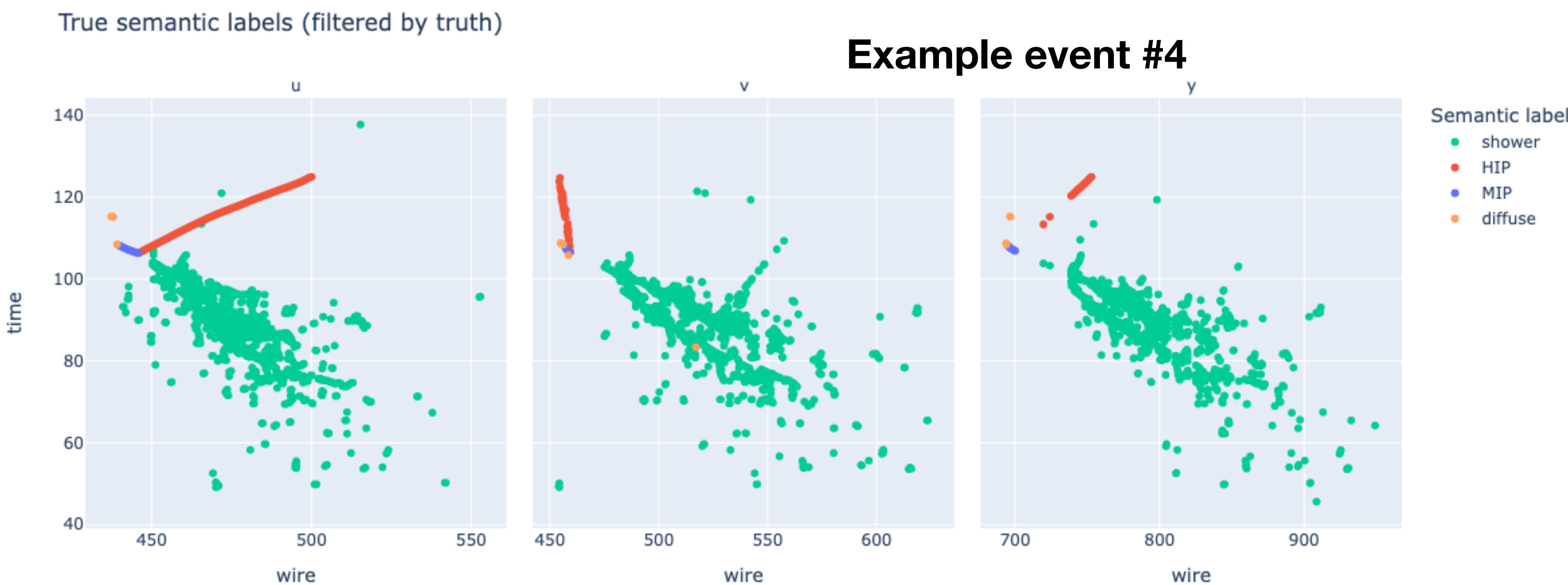


Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.

- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

Example event #4

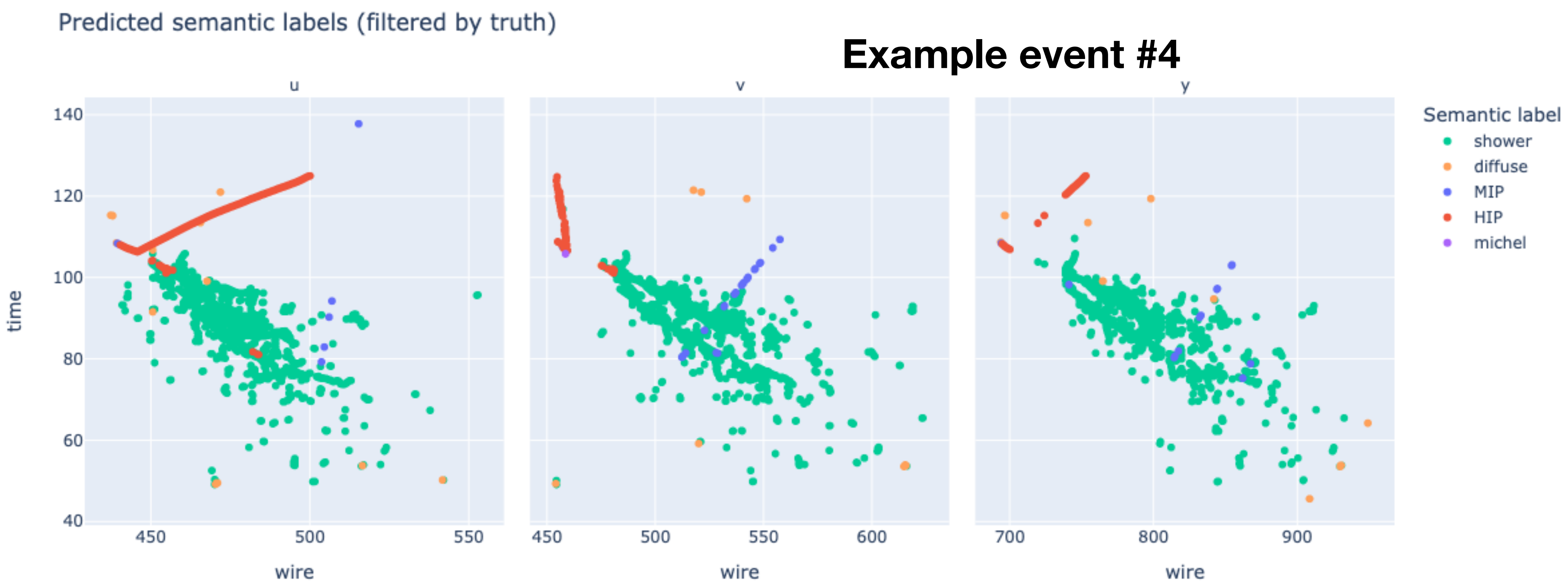


University of CINCINNATI

Hit classification

- For **signal hits** as defined for the background filtering task, we can use MC truth information to label the type of particle that produced each detector hit.
- Train a decoder to learn **a set of probabilities for each hit**, determining the particle type.

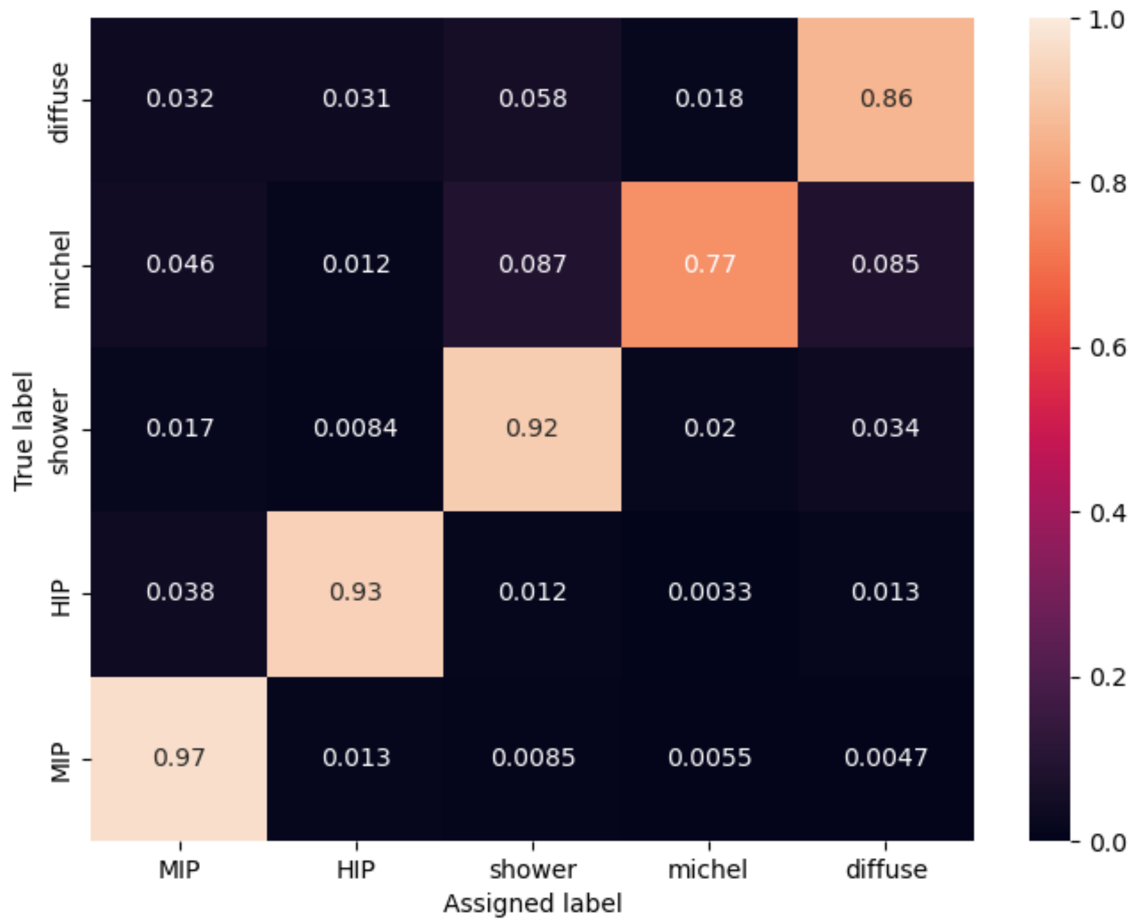
Example event #4



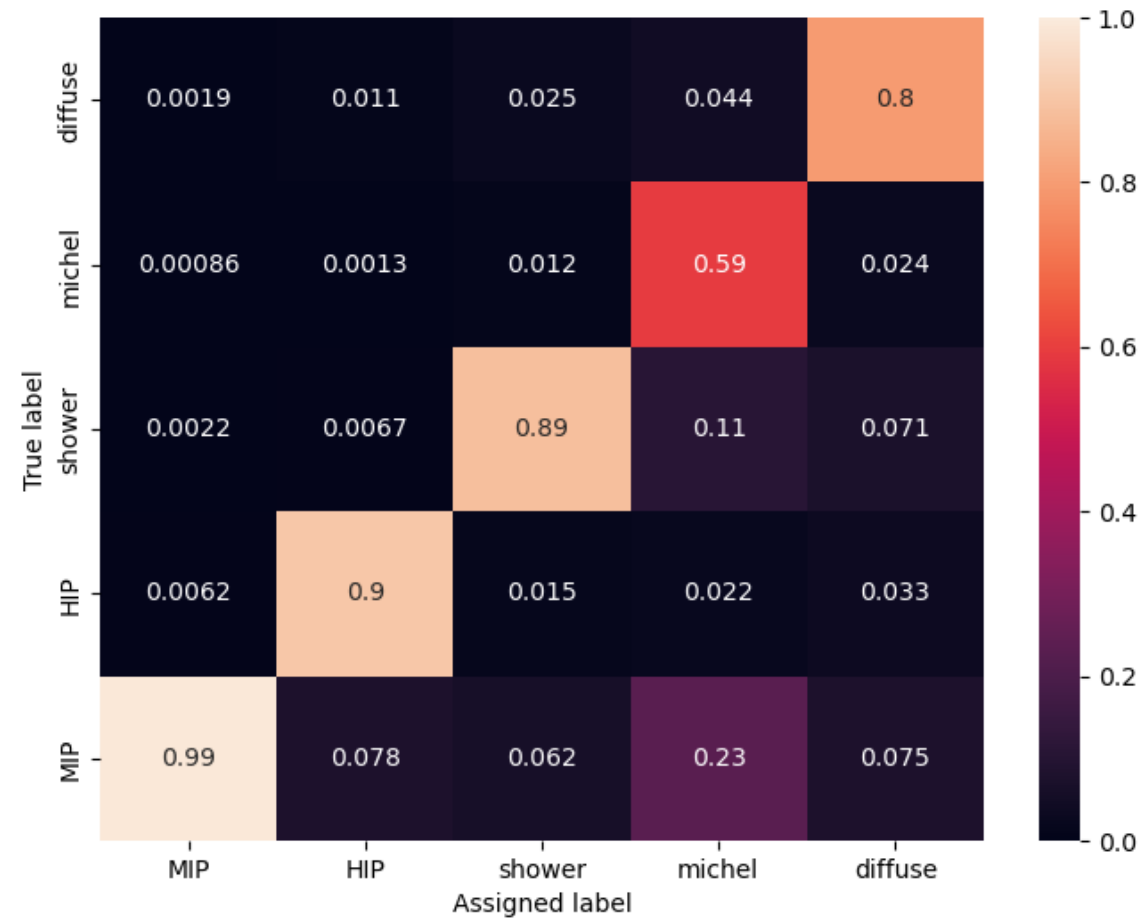
Hit classification results

- Performance metrics: **recall 0.948**, **precision 0.948**.
- Recently improved performance by enhancing ν_μ component of dataset, and using **recall loss** to counteract class imbalance.

Recall matrix



Precision matrix



Looking forward

- **Interaction vertexing** (Jonathan Huang, UChicago):

Looking forward

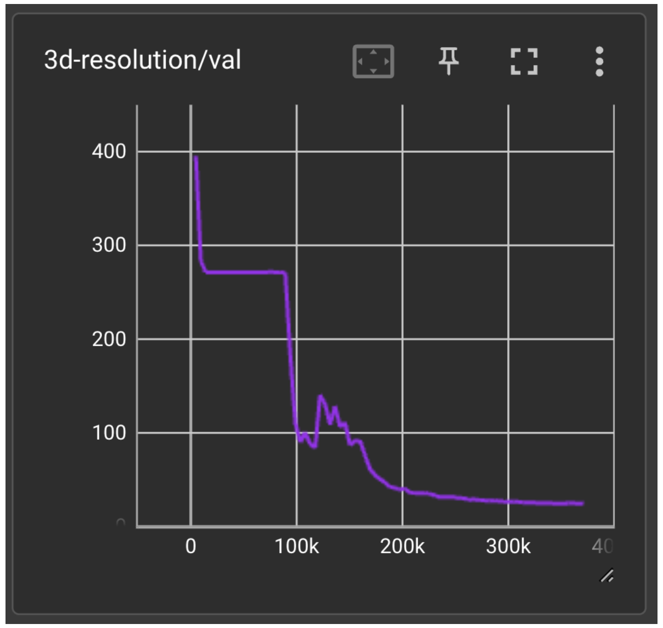
- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.

Looking forward

- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.
  - Learn a combination of physics (vertex position) and geometry (2D to 3D coordinate transformation).

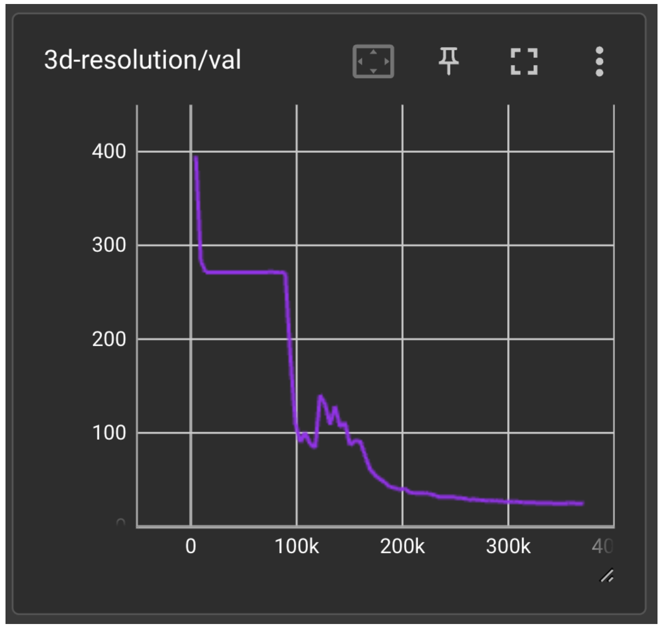
Looking forward

- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.
  - Learn a combination of physics (vertex position) and geometry (2D to 3D coordinate transformation).
  - Prototype decoder achieves ~25cm 3D resolution.



Looking forward

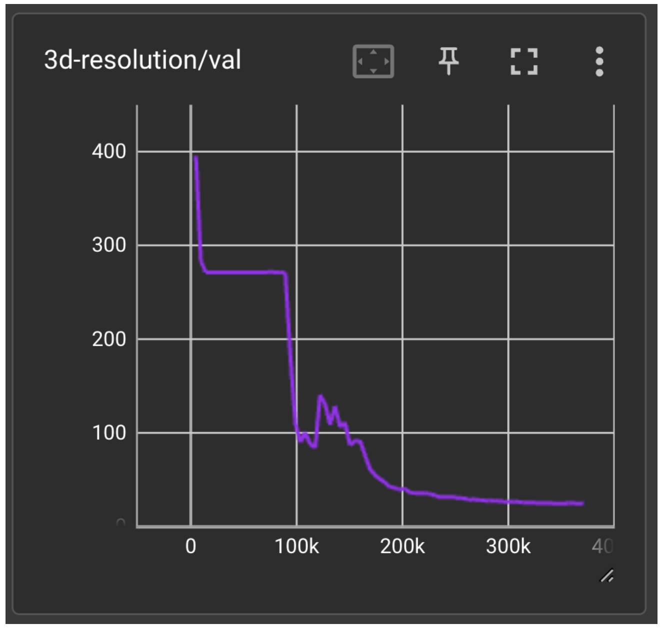
- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.
  - Learn a combination of physics (vertex position) and geometry (2D to 3D coordinate transformation).
  - Prototype decoder achieves ~25cm 3D resolution.

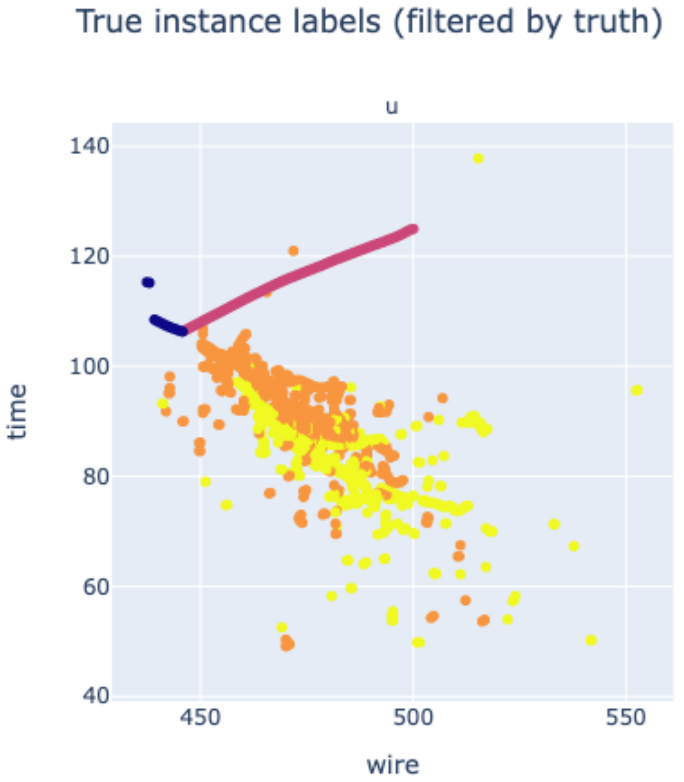


- **Clustering** (Marcelo Iovon, UCincinnati):

Looking forward

- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.
  - Learn a combination of physics (vertex position) and geometry (2D to 3D coordinate transformation).
  - Prototype decoder achieves ~25cm 3D resolution.

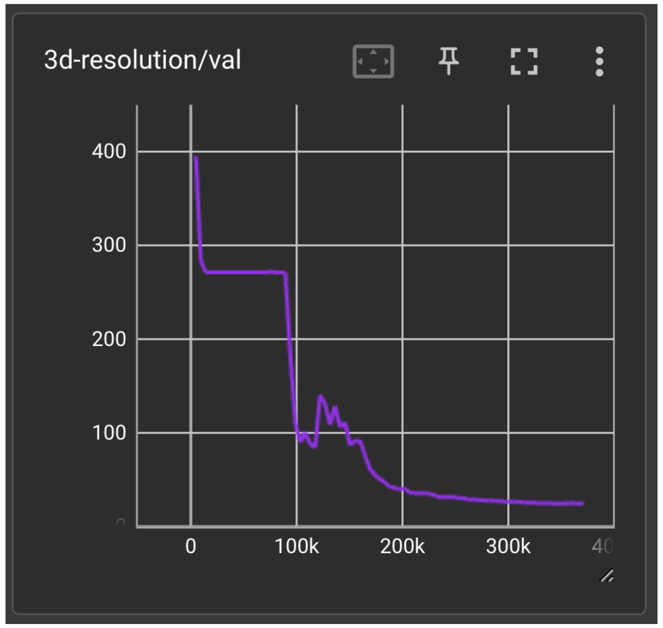


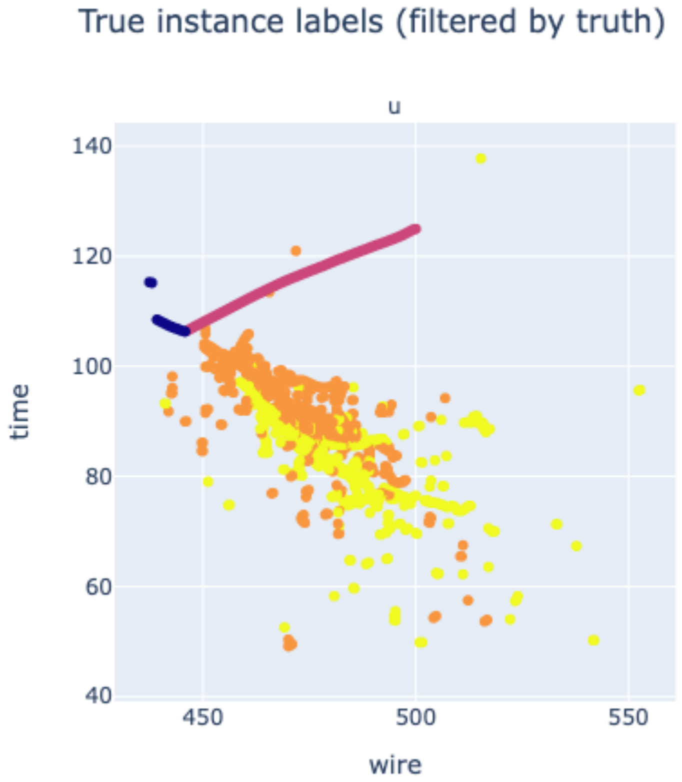


- **Clustering** (Marcelo Iovon, UCincinnati):
  - Object condensation decoder for grouping hits into particles.

Looking forward

- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.
  - Learn a combination of physics (vertex position) and geometry (2D to 3D coordinate transformation).
  - Prototype decoder achieves ~25cm 3D resolution.

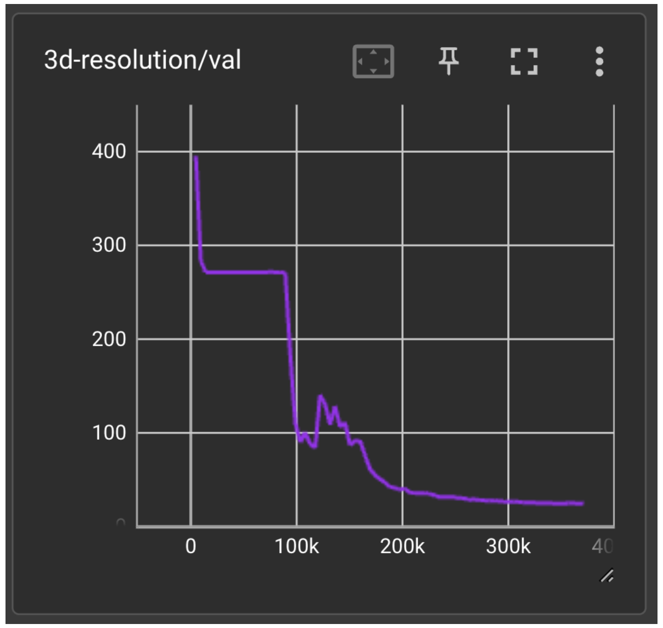


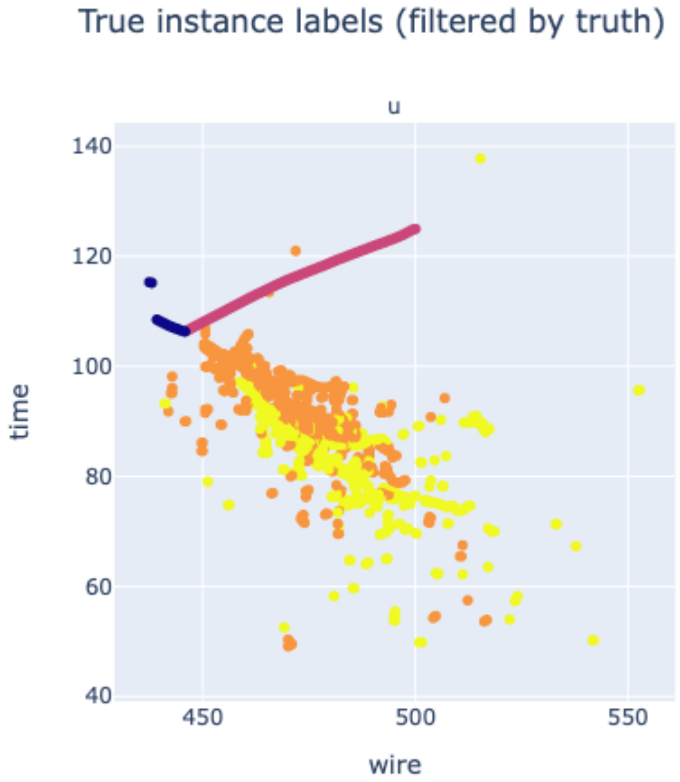


- **Clustering** (Marcelo Iovon, UCincinnati):
  - Object condensation decoder for grouping hits into particles.
  - Share instance labels between planes, to group 2D hits into natively 3D clusters.

Looking forward

- **Interaction vertexing** (Jonathan Huang, UChicago):
  - LSTM decoder for predicting 3D neutrino vertex position.
  - Learn a combination of physics (vertex position) and geometry (2D to 3D coordinate transformation).
  - Prototype decoder achieves ~25cm 3D resolution.

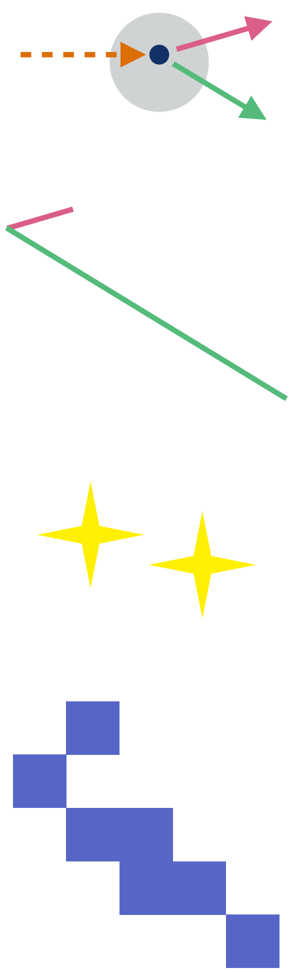


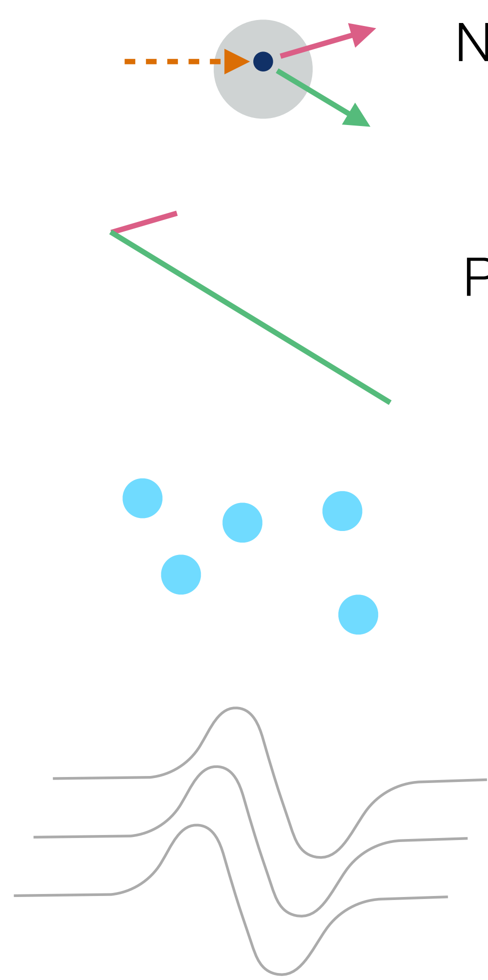


- **Clustering** (Marcelo Iovon, UCincinnati):
  - Object condensation decoder for grouping hits into particles.
  - Share instance labels between planes, to group 2D hits into natively 3D clusters.
  - Efficient clustering is our highest priority, since it enables a wealth of hierarchical graph approaches.

Common abstraction for neutrino experiments

- Although the details of many neutrino physics experiments vary, the majority of them share a common paradigm at a high level.





NOvA	MicroBooNE	Shared structure
Neutrino generator (GENIE)	Neutrino generator (GENIE)	**Event information**
Particle simulation (Geant4)	Particle simulation (Geant4)	**True particles**
True light depositions	True ionization electrons	**True energy deposits**
Photoelectrons on APDs	Pulses on TPC wires	**Detector hits**

NuML & PyNuML

- The **NuML** package is a toolkit for writing **physics event records** to an **HDF5 file format**.
  - Hold low-level information such as **simulated particles**, **hits**, **true energy depositions** etc.
  - Generic data structure can be **shared across experiments**.
  - Common interface with **PandAna** analysis toolkit (see CHEP 2021 talk).
  - Available as LArSoft package on GitHub.

NuML & PyNuML

- The **NuML** package is a toolkit for writing **physics event records** to an **HDF5 file format**.
  - Hold low-level information such as **simulated particles**, **hits**, **true energy depositions** etc.
  - Generic data structure can be **shared across experiments**.
  - Common interface with **PandAna** analysis toolkit (see CHEP 2021 talk).
  - Available as LArSoft package on GitHub.
- The **PyNuML** package is designed to provide a **generic**, **accessible**, **efficient** and **flexible** solution for many of the necessary tasks in leveraging ML for particle physics.
  - Define **particle ground truth labels** for Geant4-simulated particles.
  - **Arrange detector hits into ML objects**, ie. graphs, CNN pixel maps, etc.
  - Efficiently **preprocess ML inputs in parallel in HPC environments** using MPI.
  - Available as Python package on GitHub, or install with `pip install pynuml`!

Summary

- **NuGraph2** is a multi-purpose GNN architecture for reconstructing neutrino interactions in MicroBooNE, DUNE and elsewhere.
  - **Efficiently reject background detector hits**.
  - **Classify detector hits according to particle type**.
  - Future: **vertexing**, **clustering**, **hierarchical graphs**!
- **NeutrinoML** toolkit for standardising the process of producing ML inputs from HEP data for general use.
  - Utilised for MicroBooNE's public data release.
  - Open-source, easy-to-install code packages.