Faculty of Physics

WARSAW UNIVERSITY OF TECHNOLOGY

# Particle identification with machine learning from incomplete data in the ALICE experiment

**Maja Kabus**,
Monika Jakubowska, Kamil Deja,
Łukasz Graczykowski, Miłosz Kasak

*on behalf of the ALICE Collaboration*

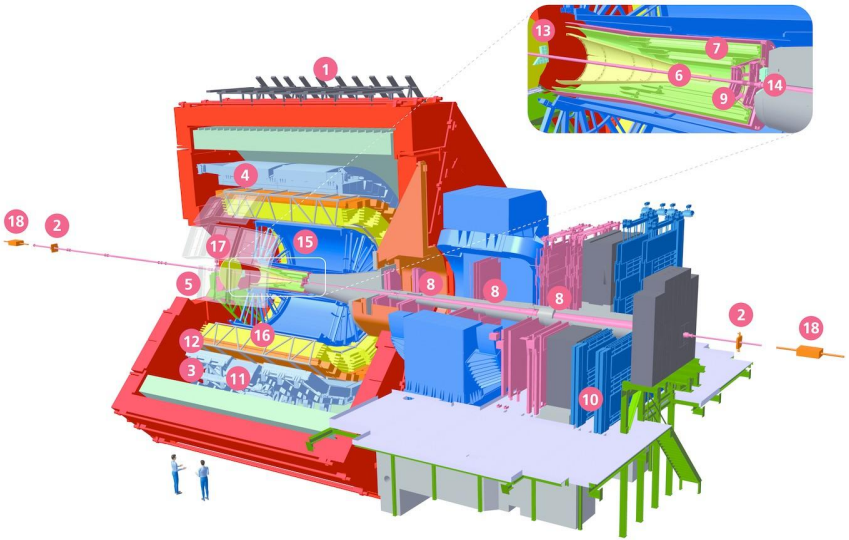
ALICE

CERN

AI4EIC, 30.11.2023

# The ALICE experiment

**ALICE** – one of the experiments at the **Large Hadron Collider (LHC)** at CERN




LHC Run 1+2 configuration

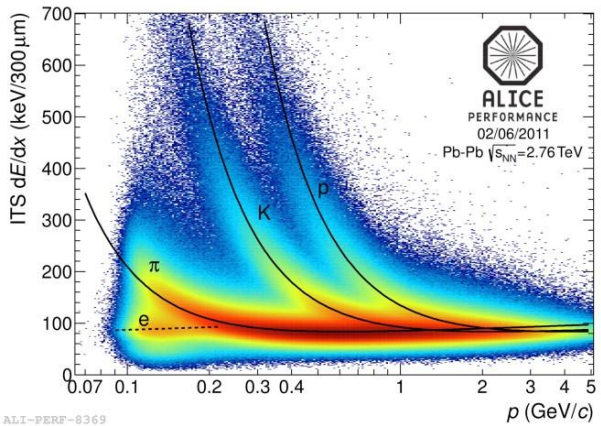
Heavy-ion collisions → production of **quark-gluon plasma (QGP)**

- beginnings of the Universe
- neutron stars

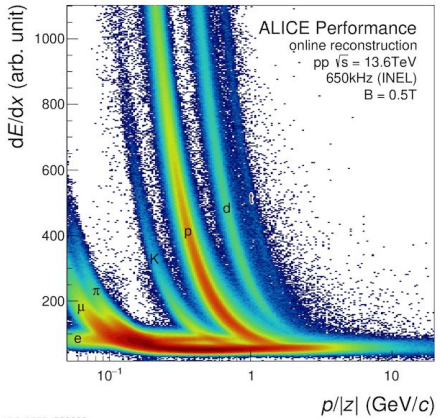
# Particle identification (PID)

**Aim:** provide high purity samples of particles of a given type

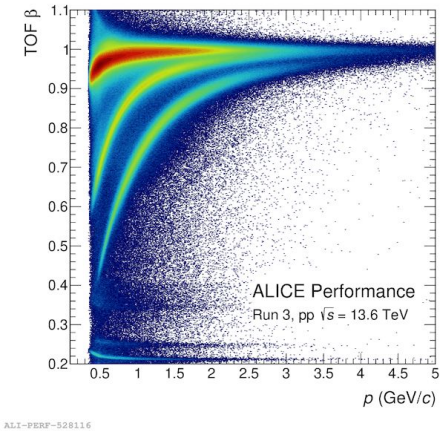
- **an essential step** for many physics analyses, especially **quark-gluon plasma** measurements
- **a distinguishing feature** of ALICE among the LHC experiments:
  - identification of particles of momenta from **100 MeV/*c* up to 20 GeV/*c***
  - **very good separation** of pion, kaons, protons, electrons
  - **all known techniques** employed: dE/dx energy loss, time-of-flight, Cherenkov radiation for hadrons and transition radiation for electrons




ITS




TPC

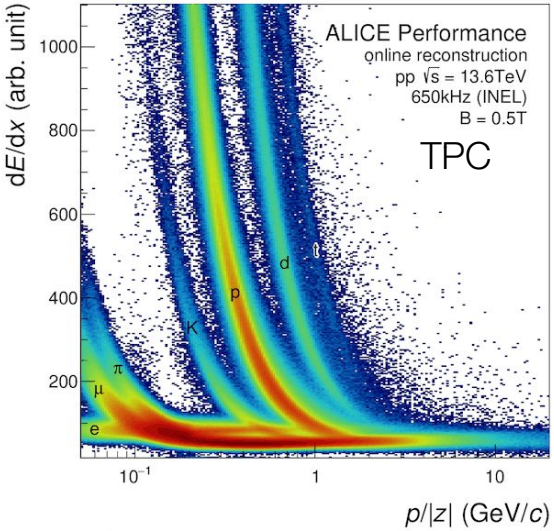


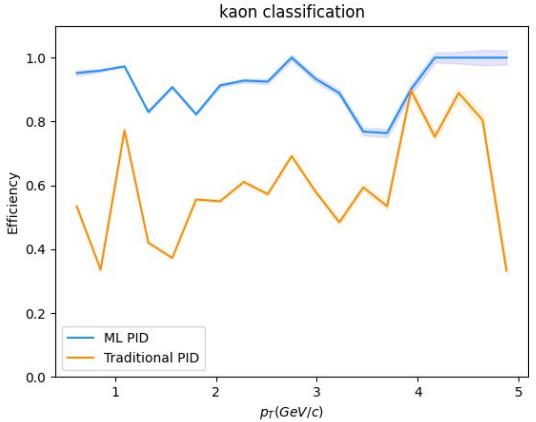

TOF

# Present state-of-art

1. **Traditional method:**
   - hand-crafted selections of selected quantities, e.g., nσ
   - problems:
     - overlapping signals
     - time-consuming optimization

2. **Bayesian method (arxiv:1602.01392):**
   - updating probability of an hypothesis with each new evidence
   - priors = best guess of true particle yields per events
   - posteriors ~ purity
   - increased purity, results consistent with the traditional method



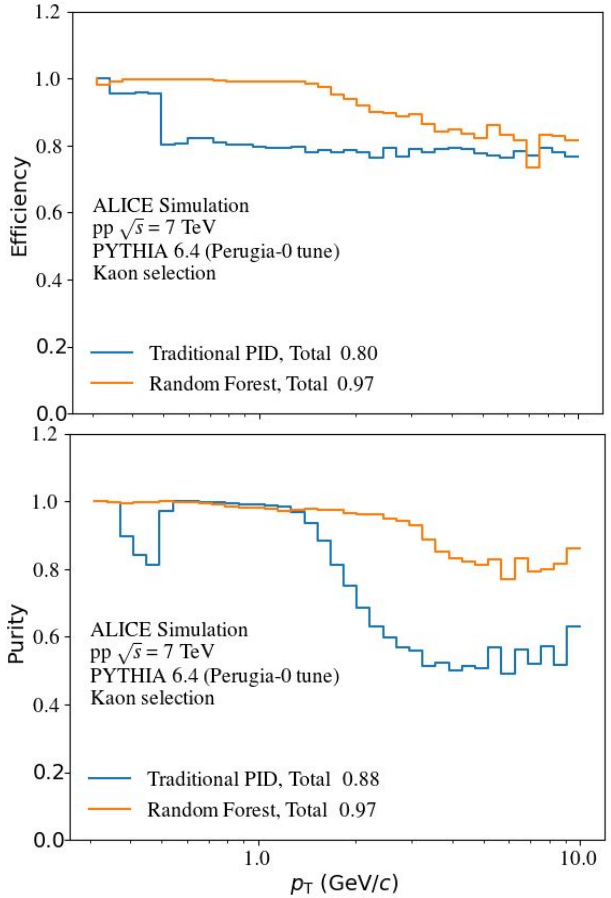




Both methods available in O$^2$Physics – ALICE Run 3 software:
https://aliceo2group.github.io/analysis-framework/

**Can we go any better?**

$n_\sigma$ method: high purity above 0.9 at the cost of low efficiency → can we balance?

# Machine learning for PID




- **classification** problem – a ML ''standard''
- can use more track parameters as input
- can learn **more complex relationships**
- many software libraries available

Note also **the limitations**:

- good quality of the training data
- hard to obtain systematic uncertainties
- hard to follow classifier's ''reasoning''

Machine learning can **greatly improve** purity and efficiency

- random forest: T. Trzciński, Ł. Graczykowski, M. Glinka, ALICE Collaboration. Using Random Forest classifier for particle identification in the ALICE experiment. Conference on Information Technology, Systems Research and Computational Physics, pp. 3-17. 2018
- domain adaptation: M. Kabus, M. Jakubowska, Ł. Graczykowski, K. Deja, ALICE Collaboration. Using machine learning for particle identification in ALICE. JINST, v. 17, p. C07016. 2022
- details in backup

# Dealing with incomplete data

**At present:** simple neural network, **19 features:** momenta, spatial coordinates, charge sign, DCA XY, DCA Z, alpha angle, track type, TPC shared clusters, detector signals

Data might be missing from one or more detectors due to, e.g., too small $p_T$
**Challenge:** Classify without making any assumptions about the missing values

**Feature Set Embedding** (article):

- instead of vectors, use (feature, value) pairs; no value → no pair
- map pairs into an embedding space of fixed dimension: similar features close to each other
- predict output class from embedded vectors
- **2 functions (networks) to learn:** (feature, value) pairs→embeddings, embeddings→class

**Bonus:** simultaneous learning of variants with and without given feature

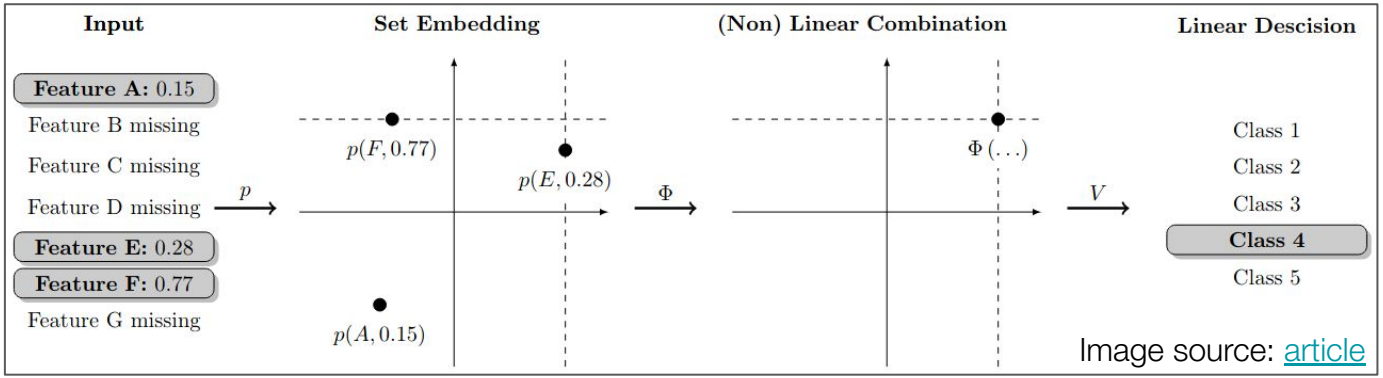
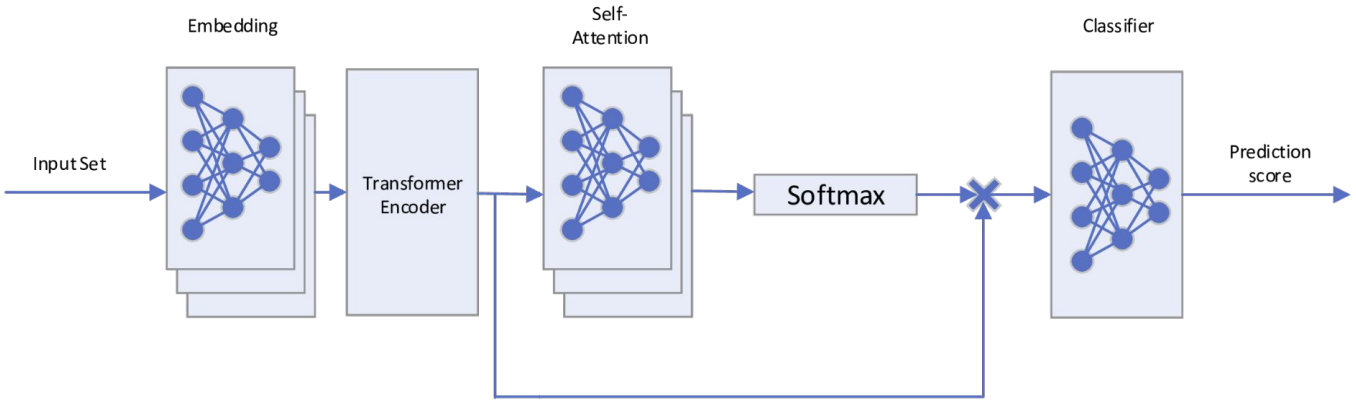



Image source: article

# One step further: the attention mechanism

Inspired by AMI-Net proposed for medical diagnosis from incomplete data




1. Feature Set Embedding to encode the inputs
   a. one-hot encoding of feature indices for easier processing
2. Transformer Encoder to detect patterns in the input
3. Self-attention to pool the encoder output set into a single vector
4. **Classifier:** a simple neural network for **a specific particle specie**
   a. "certainty" in range (0, 1) that a given particle belongs to the given specie

details in backup

# Test setup

5 methods for incomplete data:

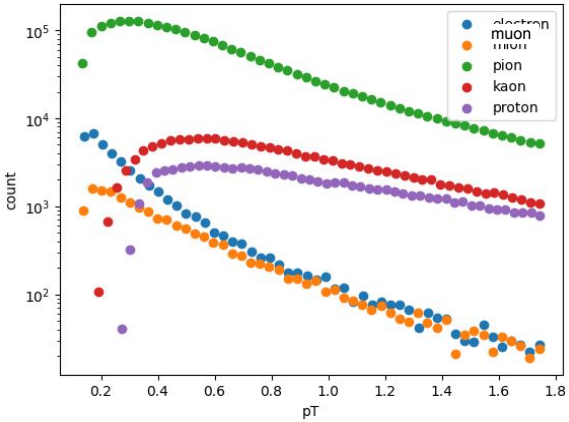
- imputation
  - mean
  - linear regression
- case deletion
- neural networks ensemble
- attention + FSE

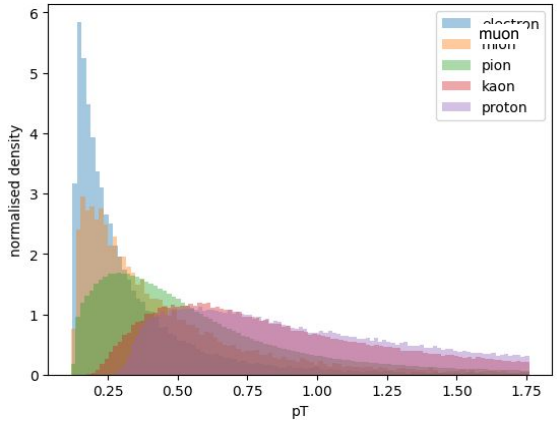
architecture details in backup

Standard nσ method: $|n_{\sigma,\,TPC}| < 3$ for $p_T < 0.5$ GeV/$c$, $\sqrt{n_{\sigma,\,TPC}^2 + n_{\sigma,\,TOF}^2} < 3$ for $p_T \geq 0.5$ GeV/$c$

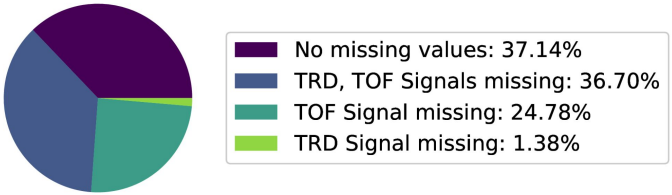
**Hyperparameter sweep** to choose best model for each method

**Dataset:** Run 2 general-purpose MC pp at $\sqrt{s}$ = 13 TeV simulated with Pythia8 and Geant4




Missing data distribution



# Results

$F_1$ = 2 x (purity x efficiency) / (purity + efficiency)

**best model**, **2nd best model**

ML outperforms the standard way

**FSE + attention** with **very good scores** of $\mathbf{F_1}$

**No flaws of other methods:**

- imputation:
  artificial bias in data
- case deletion:
  no ability to analyze samples with missing detector signals
- NN ensemble:
  potentially large complexity

| | $\pi$ | $p$ | $K$ | $\bar{\pi}$ | $\bar{p}$ | $\bar{K}$ |
|---|---|---|---|---|---|---|
| standard | 87.87 ± 0.87 | 74.61 ± 1.88 | 73.17 ± 1.57 | 87.66 ± 0.87 | 69.12 ± 1.93 | 69.44 ± 1.60 |
| NN ensemble | 98.45 ± 0.04 | 95.42 ± 0.12 | 86.74 ± 0.16 | 98.27 ± 0.42 | 94.60 ± 0.10 | 84.91 ± 0.48 |
| mean | 98.40 ± 0.01 | 95.54 ± 0.06 | 86.36 ± 0.34 | 98.34 ± 0.01 | 94.75 ± 0.20 | 84.67 ± 0.38 |
| attention + FSE | 98.50 ± 0.02 | 95.79 ± 0.07 | 87.44 ± 0.14 | 98.44 ± 0.02 | 94.89 ± 0.14 | 86.00 ± 0.13 |
| regression | 98.40 ± 0.04 | 95.49 ± 0.15 | 86.22 ± 0.46 | 98.36 ± 0.03 | 94.57 ± 0.13 | 85.01 ± 0.13 |

| | $\pi$, only complete data | $p$, only complete data | $K$, only complete data | $\bar{\pi}$, only complete data | $\bar{p}$, only complete data | $\bar{K}$, only complete data |
|---|---|---|---|---|---|---|
| case deletion | 99.37 ± 0.01 | 99.43 ± 0.16 | 96.95 ± 0.06 | 99.37 ± 0.01 | 99.13 ± 0.26 | 96.33 ± 0.11 |
| NN ensemble | 99.38 ± 0.01 | 99.46 ± 0.13 | 97.23 ± 0.10 | 99.34 ± 0.18 | 99.33 ± 0.10 | 96.87 ± 0.09 |
| mean | 99.27 ± 0.04 | 99.47 ± 0.08 | 96.08 ± 0.36 | 99.27 ± 0.04 | 99.20 ± 0.27 | 95.45 ± 0.33 |
| attention + FSE | 99.36 ± 0.01 | 99.48 ± 0.02 | 97.04 ± 0.17 | 99.37 ± 0.03 | 99.44 ± 0.08 | 96.91 ± 0.11 |
| regression | 99.25 ± 0.07 | 99.37 ± 0.07 | 95.62 ± 0.39 | 99.28 ± 0.02 | 99.10 ± 0.13 | 95.11 ± 0.58 |

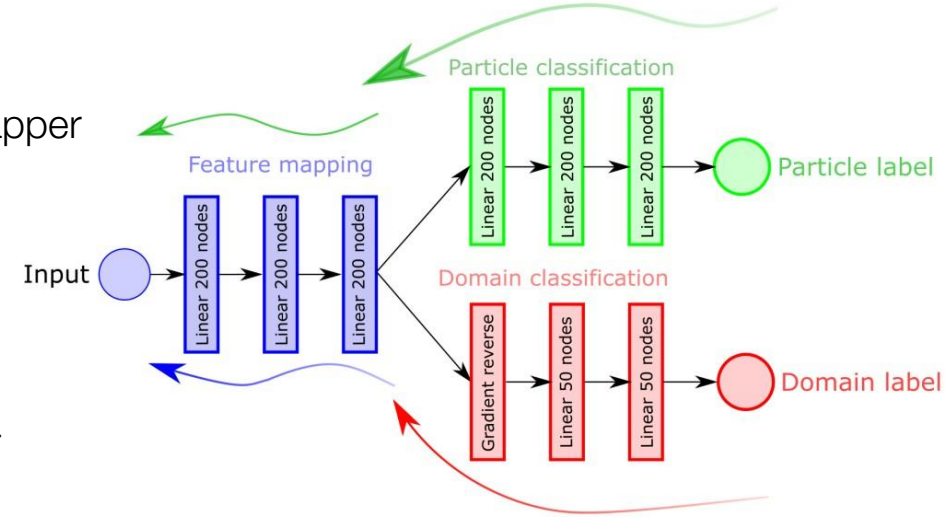
# Domain Adversarial Neural Networks (DANNs)

**feature mapping:** input → domain invariant features
**particle classifier:** recognize particles based on domain invariant latent space
**domain classifier:** recognize MC vs real samples

Training more complicated:

1. Train the domain classifier independently.
2. Freeze the domain classifier.
3. Train jointly particle classifier and feature mapper **adversarially** to the domain classifier.
4. Weights of the feature mapper:
   gradient from particle classifier
   + reversed gradient from domain classifier

**Application time similar** to a standard classifier



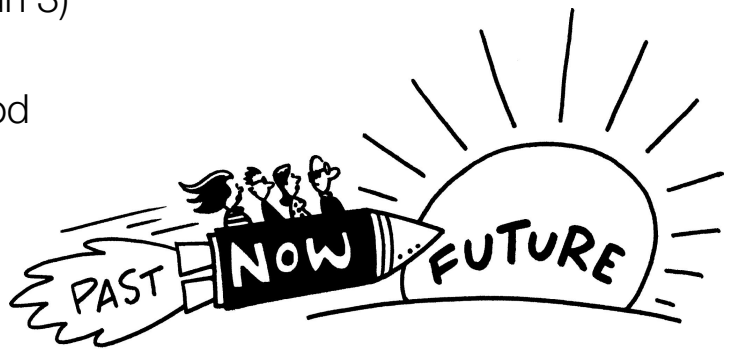
# Summary and outlook

**Summary:**

- **machine learning** is a promising way to identify particles with **higher purity and efficiency**
- **Feature Set Embedding with Multi-Head Attention** improve $F_1$ score for PID on **incomplete data**

**Plans:**

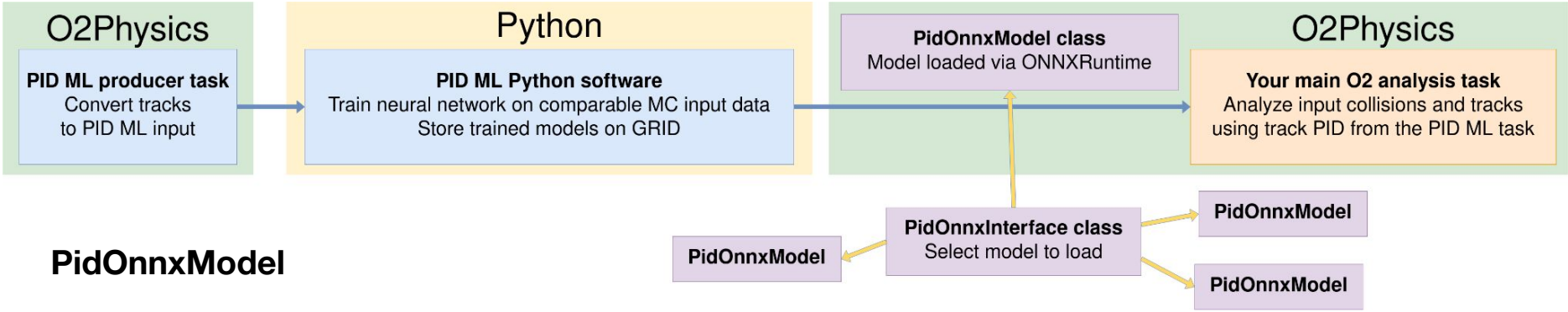
- test in an analysis task
- test on MC data from the next LHC data-taking period (Run 3)
- add domain adaptation and test on the new real data
- regular production of models for the new data-taking period




Thank you for your attention!

# Backup

# Integration with O$^2$Physics: user interface




**PidOnnxModel**

- 1 instance = 1 model = 1 particle specie recognized (yes / no)
- **convenient interface** clearly separated from the rest of analysis
- using all capabilities of **Python ML libraries** for training
- ONNX file format and **ONNXRuntime** software used for inference in O$^2$ C++ environment

**PidOnnxInterface**

- **automatically select most suitable model** for user needs or manual mode
- as **little additional knowledge** from the analyser as possible

# Random Forest (RF) on Run 2 data

Preliminary work in 2019 for LHC Run 2

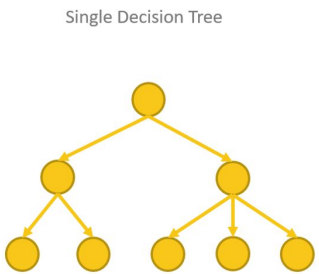
Tomasz Trzciński, Łukasz Graczykowski, Michał Glinka, ALICE Collaboration, et al. Using Random Forest classifier for particle identification in the ALICE experiment. In Conference on Information Technology, Systems Research and Computational Physics, pages 3–17. Springer, 2018

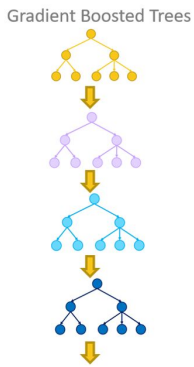
## Why Random Forest?

- a set of decision trees, each trained on a random subset of the training data
- easy to parallelize, e.g., on GRID
- resistant to overfitting

## Our approach

- tree generation: Gini index
- selection: majority of votes by trees
- adaptive boosting




# Run 2 results

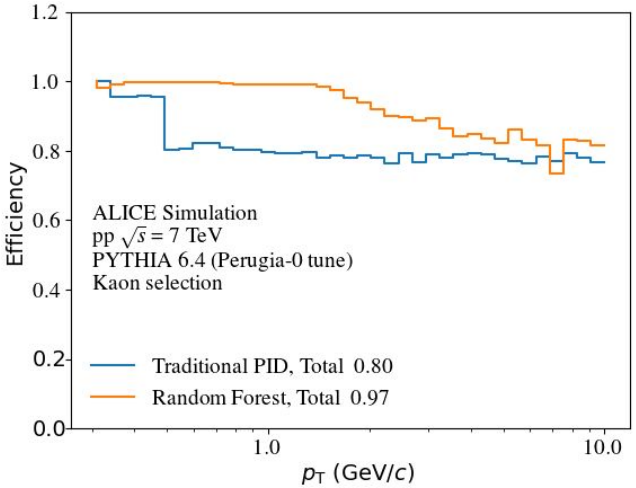
- pp at 7 TeV, Pythia 6 Perugia-0
- kaons vs other particles

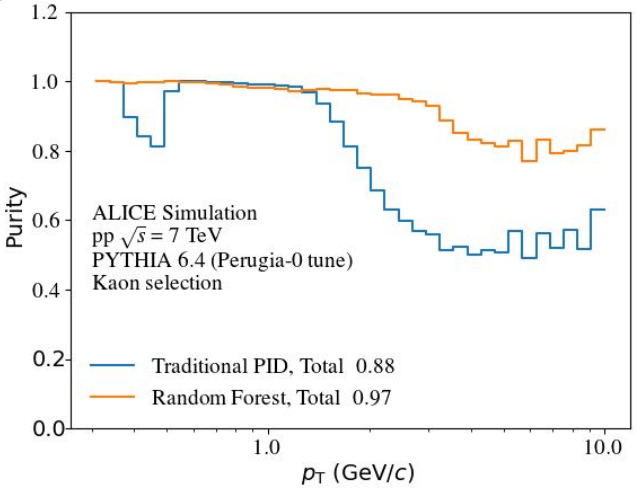
**Traditional PID:**

$$n^2_{\sigma,\mathrm{TPC}} \qquad p_\mathrm{T} \leq 0.5\mathrm{GeV/c}$$

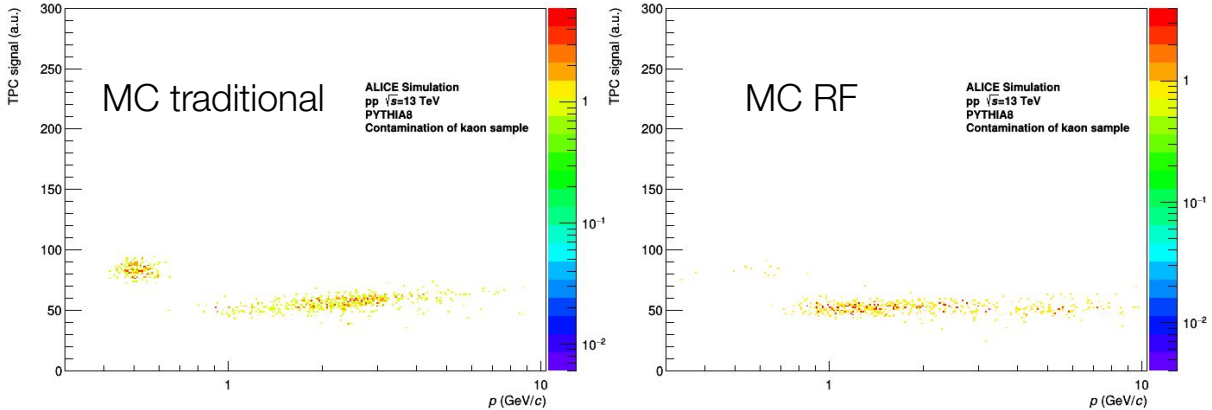
$$\sqrt{n^2_{\sigma,\mathrm{TPC}} + n^2_{\sigma,\mathrm{TOF}}} \qquad p_\mathrm{T} > 0.5\mathrm{GeV/c}$$

**much higher efficiency** and **purity** with Random Forest



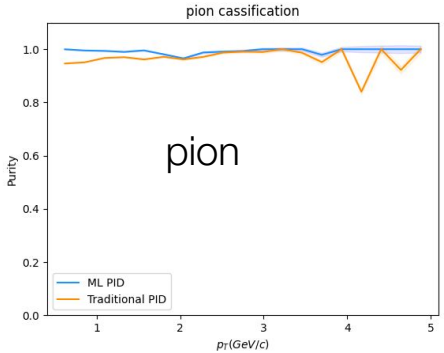

Contamination of kaon samples

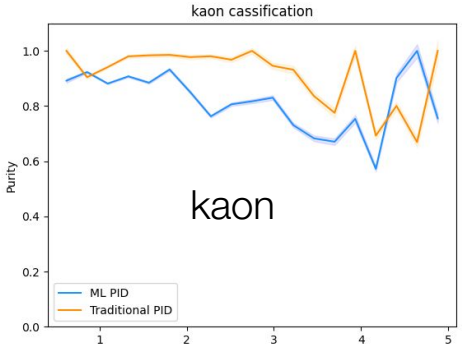


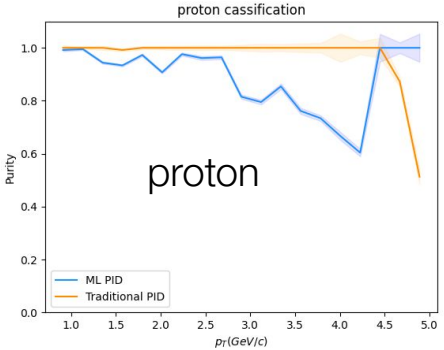
# Baseline – plain vanilla neural networks

- one neural network model per particle and per set of detectors
- results for using all detectors; **ML PID**, **traditional approach**

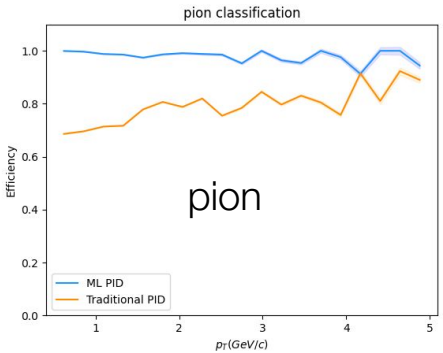
purity = precision / specificity

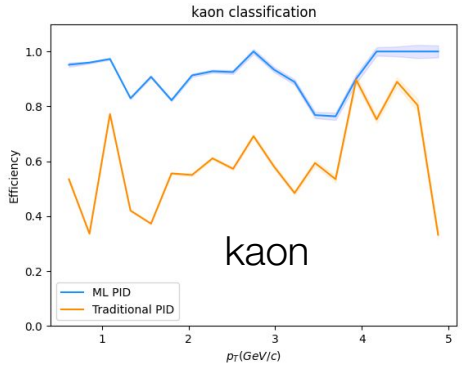


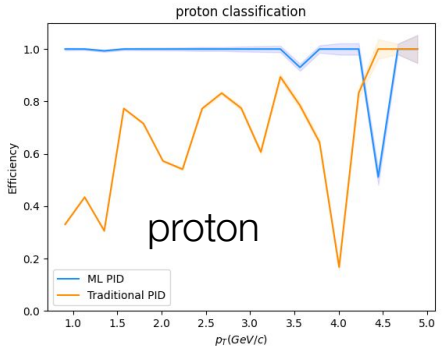




efficiency = recall / sensitivity



# Results – particles

$F_1$ = 2(purity x efficiency) / (purity + efficiency)

**best model**, **2nd best model**

ML outperforms the standard way

**FSE + attention** with **very good scores** of **$F_1$, purity and efficiency**

**No flaws of other methods:**

- imputation:
  artificial bias in data
- case deletion:
  no ability to analyze samples with missing detector signals
- NN ensemble:
  potentially large complexity

| | *π* | | | *p* | | | *K* | | |
|---|---|---|---|---|---|---|---|---|---|
| model | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ |
| standard | 99.99 ± 0.01 | 78.37 ± 0.01 | 87.87 ± 0.87 | 99.40 ± 0.01 | 59.72 ± 0.03 | 74.61 ± 1.88 | 92.87 ± 0.01 | 60.37 ± 0.05 | 73.17 ± 1.57 |
| NN ensemble | 97.47 ± 0.25 | 99.46 ± 0.21 | 98.45 ± 0.04 | 97.16 ± 0.46 | 93.74 ± 0.30 | 95.42 ± 0.12 | 91.18 ± 2.00 | 82.72 ± 1.42 | 86.74 ± 0.16 |
| mean | 97.31 ± 0.07 | 99.52 ± 0.07 | 98.40 ± 0.01 | 97.85 ± 0.41 | 93.34 ± 0.32 | 95.54 ± 0.06 | 90.83 ± 1.71 | 82.32 ± 0.96 | 86.36 ± 0.34 |
| attention + FSE | 97.49 ± 0.06 | 99.54 ± 0.05 | 98.50 ± 0.02 | 97.80 ± 0.44 | 93.86 ± 0.27 | 95.79 ± 0.07 | 91.55 ± 0.71 | 83.68 ± 0.82 | 87.44 ± 0.14 |
| regression | 97.33 ± 0.06 | 99.49 ± 0.07 | 98.40 ± 0.04 | 97.38 ± 0.40 | 93.67 ± 0.38 | 95.49 ± 0.15 | 91.17 ± 1.00 | 81.78 ± 0.21 | 86.22 ± 0.46 |

| | *π*, only complete data | | | *p*, only complete data | | | *K*, only complete data | | |
|---|---|---|---|---|---|---|---|---|---|
| model | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ |
| case deletion | 99.08 ± 0.07 | 99.67 ± 0.04 | 99.37 ± 0.01 | 99.23 ± 0.32 | 99.63 ± 0.05 | 99.43 ± 0.16 | 96.93 ± 0.37 | 96.98 ± 0.26 | 96.95 ± 0.06 |
| NN ensemble | 99.11 ± 0.04 | 99.64 ± 0.06 | 99.38 ± 0.01 | 99.16 ± 0.24 | 99.76 ± 0.07 | 99.46 ± 0.13 | 96.65 ± 0.38 | 97.82 ± 0.31 | 97.23 ± 0.10 |
| mean | 98.85 ± 0.09 | 99.69 ± 0.04 | 99.27 ± 0.04 | 99.22 ± 0.19 | 99.72 ± 0.04 | 99.47 ± 0.08 | 96.83 ± 0.17 | 95.33 ± 0.67 | 96.08 ± 0.36 |
| attention + FSE | 99.08 ± 0.02 | 99.64 ± 0.03 | 99.36 ± 0.01 | 99.28 ± 0.10 | 99.68 ± 0.09 | 99.48 ± 0.02 | 96.03 ± 0.98 | 98.06 ± 0.72 | 97.04 ± 0.17 |
| regression | 99.02 ± 0.02 | 99.49 ± 0.14 | 99.25 ± 0.07 | 99.10 ± 0.09 | 99.65 ± 0.09 | 99.37 ± 0.07 | 94.27 ± 0.98 | 97.01 ± 0.51 | 95.62 ± 0.39 |

# Results – antiparticles

$F_1$ = 2(purity x efficiency) / (purity + efficiency)

**best model**, **2nd best model**

ML outperforms the standard way

**FSE + attention** with **very good scores** of **$F_1$, purity and efficiency**

**No flaws of other methods:**

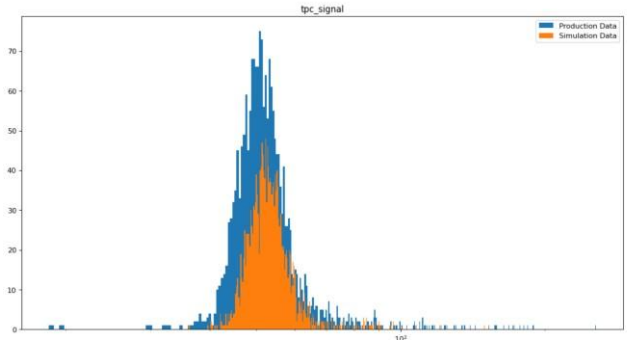
- imputation:
  artificial bias in data
- case deletion:
  no ability to analyze samples with missing detector signals
- NN ensemble:
  potentially large complexity

| | $\pi^-$ | | | $\bar{p}$ | | | $K^-$ | | |
|---|---|---|---|---|---|---|---|---|---|
| model | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ |
| standard | 99.99 ± 0.01 | 78.03 ± 0.01 | 87.66 ± 0.87 | 99.24 ± 0.01 | 53.02 ± 0.03 | 69.12 ± 1.93 | 92.22 ± 0.01 | 55.68 ± 0.04 | 69.44 ± 1.60 |
| NN ensemble | 97.01 ± 0.87 | 99.56 ± 0.11 | 98.27 ± 0.42 | 96.90 ± 0.24 | 92.41 ± 0.21 | 94.60 ± 0.10 | 89.16 ± 1.51 | 81.06 ± 1.74 | 84.91 ± 0.48 |
| mean | 97.23 ± 0.03 | 99.47 ± 0.03 | 98.34 ± 0.01 | 97.24 ± 0.39 | 92.39 ± 0.15 | 94.75 ± 0.20 | 89.75 ± 0.80 | 80.14 ± 1.18 | 84.67 ± 0.38 |
| attention + FSE | 97.38 ± 0.04 | 99.51 ± 0.02 | 98.44 ± 0.02 | 97.51 ± 0.55 | 92.40 ± 0.74 | 94.89 ± 0.14 | 90.86 ± 0.70 | 81.64 ± 0.63 | 86.00 ± 0.13 |
| regression | 97.22 ± 0.15 | 99.52 ± 0.12 | 98.36 ± 0.03 | 96.89 ± 0.50 | 92.36 ± 0.22 | 94.57 ± 0.13 | 91.63 ± 0.58 | 79.29 ± 0.59 | 85.01 ± 0.13 |

| | $\pi^-$, only complete data | | | $\bar{p}$, only complete data | | | $K^-$, only complete data | | |
|---|---|---|---|---|---|---|---|---|---|
| model | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ | purity | efficiency | $F_1$ |
| case deletion | 99.08 ± 0.04 | 99.67 ± 0.02 | 99.37 ± 0.01 | 98.75 ± 0.37 | 99.52 ± 0.17 | 99.13 ± 0.26 | 95.82 ± 0.69 | 96.84 ± 0.66 | 96.33 ± 0.11 |
| NN ensemble | 98.93 ± 0.51 | 99.76 ± 0.16 | 99.34 ± 0.18 | 99.12 ± 0.08 | 99.53 ± 0.16 | 99.33 ± 0.10 | 96.14 ± 0.32 | 97.60 ± 0.24 | 96.87 ± 0.09 |
| mean | 98.86 ± 0.11 | 99.69 ± 0.03 | 99.27 ± 0.04 | 98.79 ± 0.58 | 99.62 ± 0.18 | 99.20 ± 0.27 | 96.14 ± 0.45 | 94.77 ± 0.99 | 95.45 ± 0.33 |
| attention + FSE | 99.08 ± 0.04 | 99.67 ± 0.02 | 99.37 ± 0.03 | 99.25 ± 0.06 | 99.63 ± 0.20 | 99.44 ± 0.08 | 96.00 ± 0.11 | 97.85 ± 0.12 | 96.91 ± 0.11 |
| regression | 99.04 ± 0.03 | 99.51 ± 0.05 | 99.28 ± 0.02 | 98.57 ± 0.33 | 99.64 ± 0.25 | 99.10 ± 0.13 | 93.85 ± 1.11 | 96.41 ± 0.18 | 95.11 ± 0.58 |

# Domain adaptation

**Training set:** labeled data → MC samples
**Apply set:** unlabeled real data with **different distributions of attributes**
→ worse performance on real data




How can we transfer the knowledge from training to inference?

Standard PID example: **"tune on data"**

- get parametrization from data → real data
- generate a random detector signal → MC data
- equivalent distributions of real and MC samples – the differences are statistical fluctuations

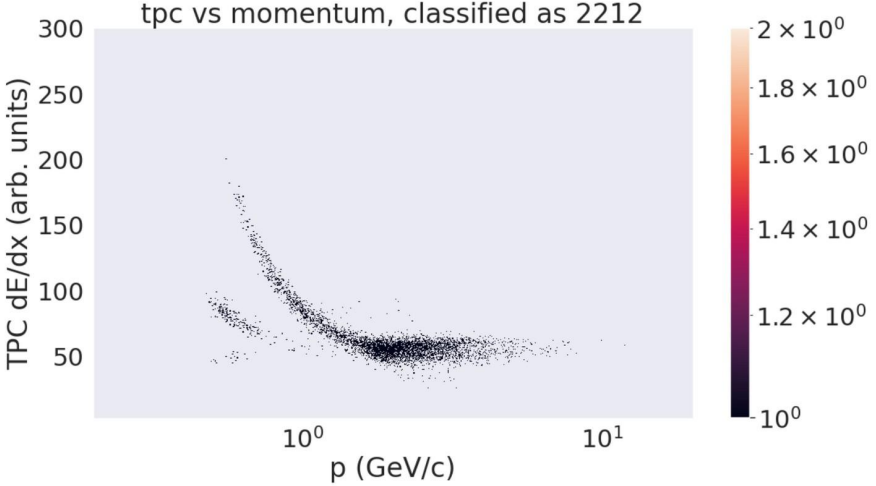
**Machine learning:**

- actually **learn** the difference between data domains
- translate both data to a single common hyperspace

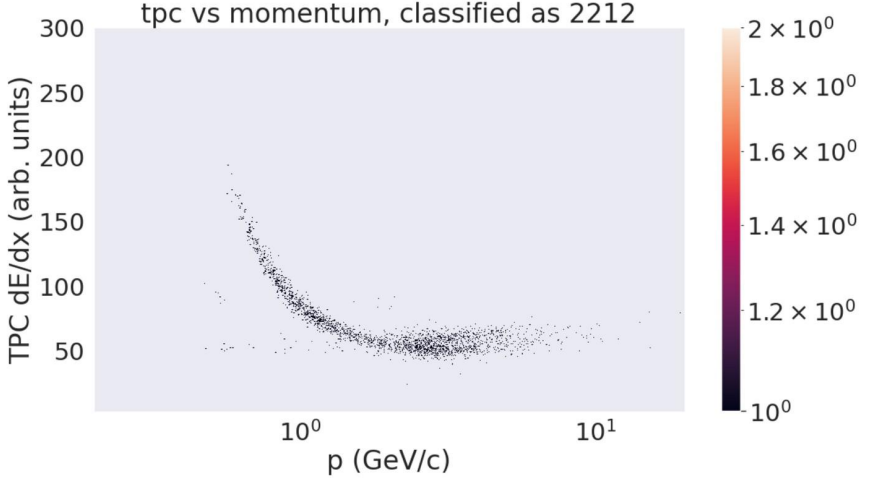
# First results of domain adaptation

- pp data at 13 TeV, LHC Run 2
- training: PYTHIA 8 with Monash tune
- classification improved – **reduction of contamination**
- more research ongoing

No domain adaptation




With domain adaptation



# Simple network implementation

- linear layers with Leaky ReLU, sigmoid at the end
- simple: dropout after each linear layer


Parameters:

- optimizer: Adam
- output layer: 1 node (yes / no for a given particle)
- loss function: binary cross entropy
- scheduler: exponential with rate 0.98
- learning rate: 0.0005
- batch size: 64
- epochs: 30

# Example: FSE with one-hot encoding

From the article in preparation

Table 1: Preprocessing of data samples into feature set values – example.

(a) 3 data samples with 5 attributes with different amount of missing values.

| id | momentum | TOF | TPC | TRD | ITS |
|---|---|---|---|---|---|
| 1 | 0.1 | | 3 | | 5 |
| 2 | 7 | 70 | 24 | 13 | 88 |
| 3 | | 78 | | | |

(b) First particle

| key | | | | | value |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0.1 |
| 0 | 0 | 1 | 0 | 0 | 3 |
| 0 | 0 | 0 | 0 | 1 | 5 |

(c) Second particle.

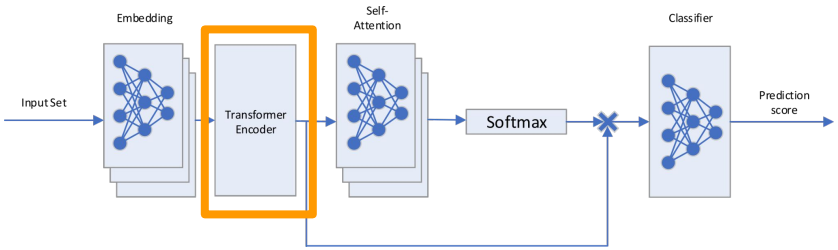
| key | | | | | value |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 7 |
| 0 | 1 | 0 | 0 | 0 | 70 |
| 0 | 0 | 1 | 0 | 0 | 24 |
| 0 | 0 | 0 | 1 | 0 | 13 |
| 0 | 0 | 0 | 0 | 1 | 88 |

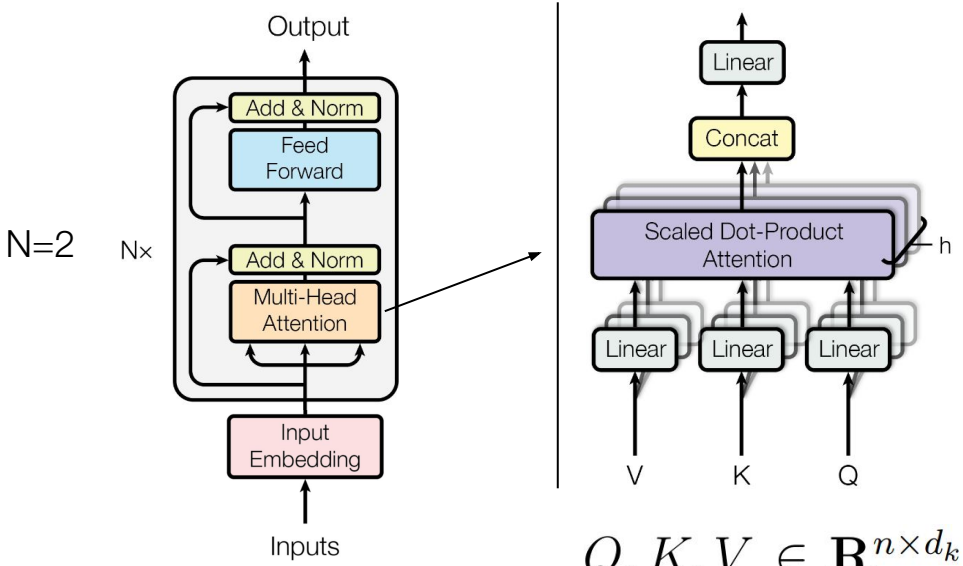
(d) Third particle.

| key | | | | | value |
|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 78 |

# The attention continued

2. [Transformer Encoder]



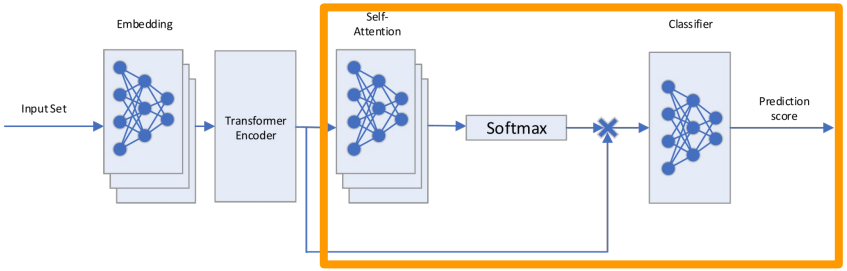

modified diagram from the article

- adjusted original Transformer Encoder
- attention without convolutions and recurrence
- finding self-correlations in an instance set of vectors
- example: a specific detector signal could be used if and only if the momentum is in a specific range

$$Q, K, V \in \mathbf{R}^{n \times d_k}$$

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

# Pooling and final classification




**Classifier:** a simple neural network
expects a single vector as an input

**Solution: self-attention** to pool the variable-size vector set from Transformer Encoder

$$\{v_1, v_2, ..., v_n\}, \; v_i \in \mathbf{R}^{d_{model}}$$

$$e_i = NN(v_i) \qquad \forall i \in [1, n]$$ self-attention values

$$\alpha'_j = softmax(e'_j) \qquad \forall j \in [1, d_{model}]$$ self-attention weights

$$o_j = \sum_{k=1}^{n} \alpha_{kj} v_{kj} \qquad \forall j \in [1, d_{model}]$$ pooled output vector

**Classifier score:** logistic function $f(x) = \frac{1}{1+e^{-x}}$ , range (0, 1)
"certainty" that a given particle belongs to the given specie

# Architecture of tested neural networks

**Imputations, case deletion, and NN ensemble**

- 3 hidden layers of sizes 64, 32, 16 with ReLU activation
- dropout 0.1 after each activation layer
- input size:
  - imputations and case deletion: 19 as all missing features are imputed
  - ensemble: 4 networks with input sizes 19, 17, 17, 15

**Attention + FSE**

- embedding layers: 20 – 128 – 32 neurons
- Transformer Encoder:
  - Multi-Head Attention: dimension 32, 2 heads
  - neural network layers: 32 – 128 – 32 neurons
  - 2 layers of Multi-Head Attention + neural network
- Self-Attention layers: 32 – 64 – 32 neurons
- classifier layers: 32 – 64 – 1 neurons
- dropout 0.1 at the output of embedding and each Transformer Encoder layer
- ReLU activation between neural network layers

# Sample ROC curves

**FSE+attention** achieves **best results.**

**Little variation** between particle species.