# A Deep Learning Approach to Proton Background Rejection for Positron Analysis with the AMS Electromagnetic Calorimeter

Raheem Hashmani

Advisor: Assist Prof. Dr. Emre Akbaş
Co-advisor: Prof. Dr. M. Bilge Demirköz

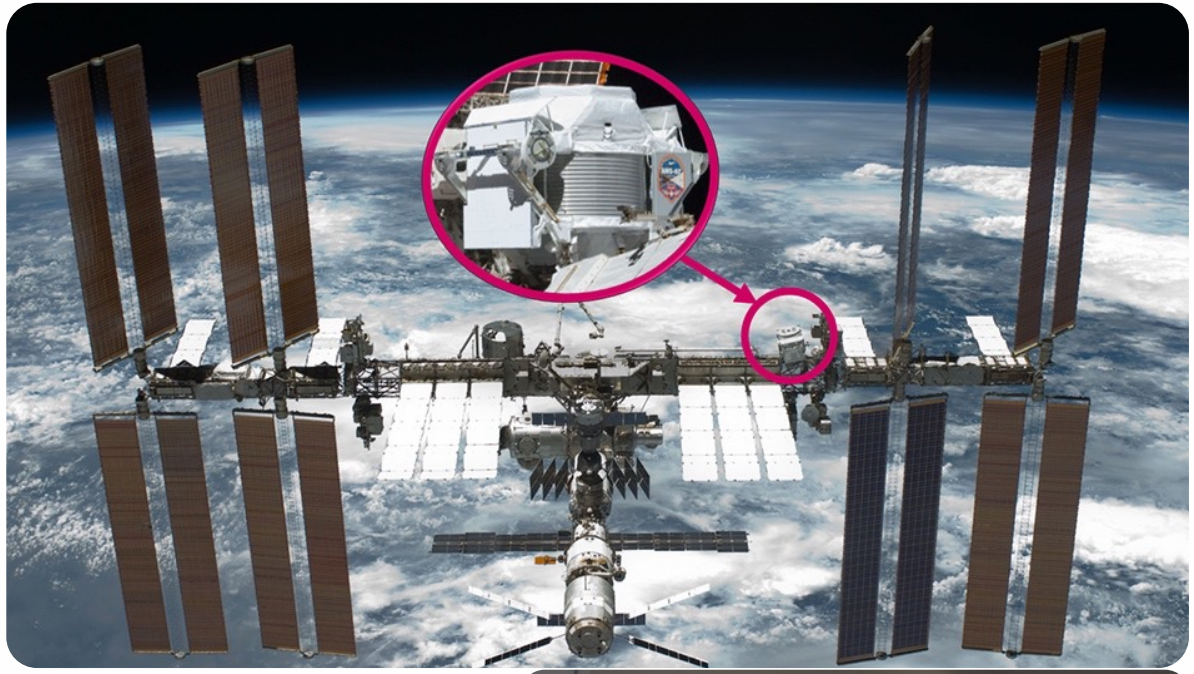


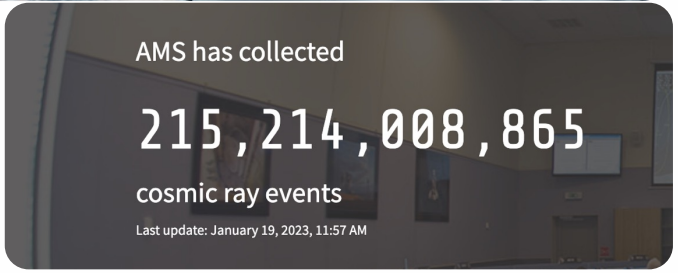

# Outline

- Introduction, Motivation, and Goal
- ECAL Dataset & Methodology
- Machine Learning Models Used
- Experiments & Results
- Conclusions and Future Direction

# Introduction - The Alpha Magnetic Spectrometer Experiment

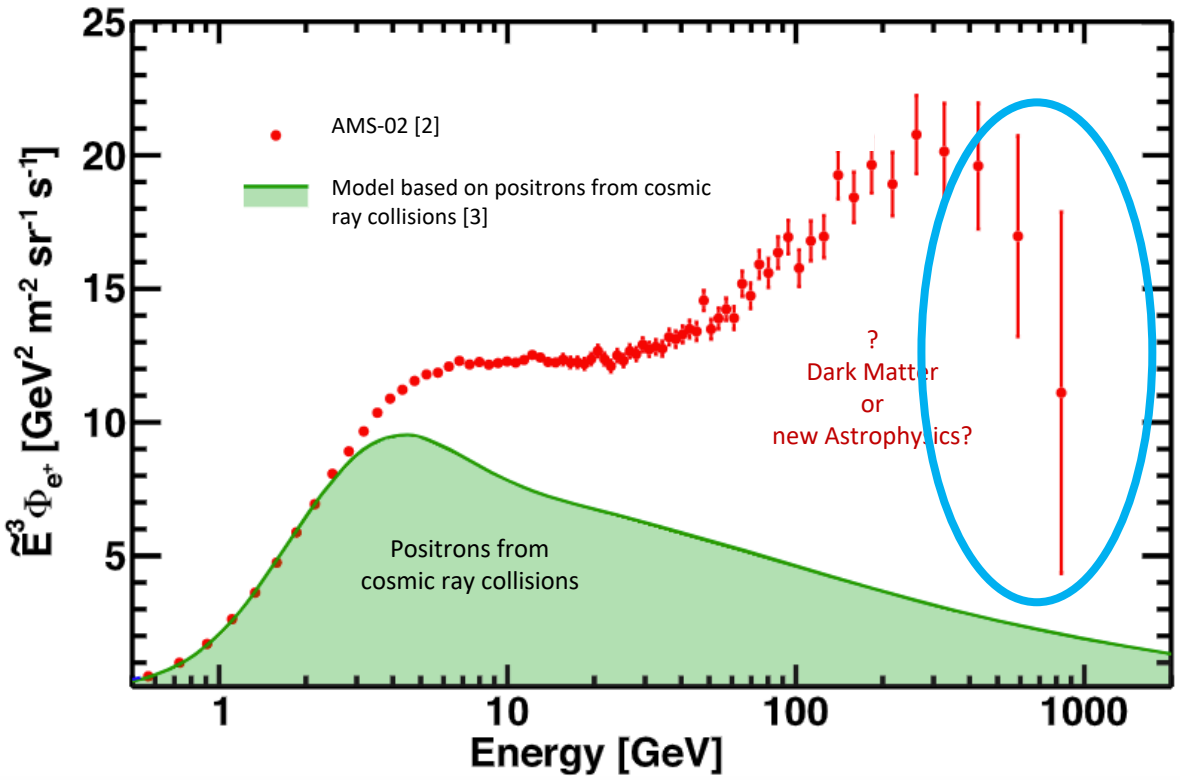
- General-purpose high-energy particle physics detector onboard the ISS [1].
- Installed on 19 May 2011
- Collected over 215 billion cosmic ray events
- Main objectives:
  - Searching for antimatter
  - Investigating dark matter
  - Analyzing cosmic rays






[1] S. Ting, Nuclear Physics B - Proceedings Supplements, Oct. 2013, doi: 10.1016/j.nuclphysbps.2013.09.028

# Motivation – Identifying Positrons at High Energies




These results can not be explained by traditional cosmic ray models.

To continue testing theoretical models, a purer measurement at higher energies is needed.

Difficulty in Separating Positrons from Protons at TeV Energies:

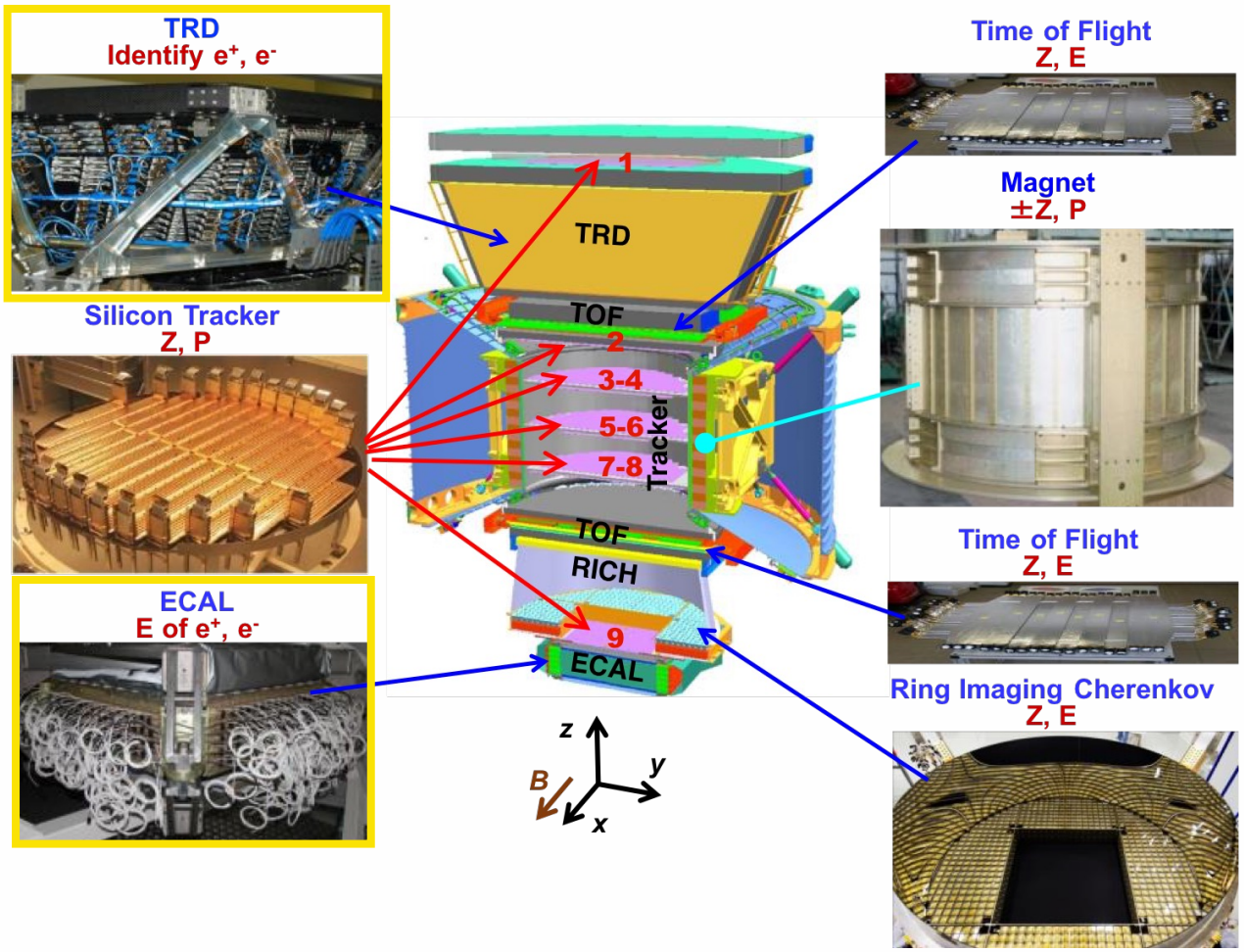
- Class Imbalance:
  - Around 1 TeV, ~4700 protons per positron.
- Label Inaccuracy / Detector Limitations:
  - TRD in inaccurate at separating positrons/electrons from protons at TeV energies.
  - ECAL depth not enough: only captures 75% of an electromagnetic shower's energy at 1 TeV [4].


[2] AMS Collaboration *et al.*, Phys. Rep., vol. 894, pp. 1–116, Feb. 2021, doi: 10.1016/J.PHYSREP.2020.09.003.
[3] R. Trotta *et al.*, Astrophys. J., vol. 729, no. 2, p. 106, Mar. 2011, doi: 10.1088/0004-637X/729/2/106.

# Our Goal – Deep Learning to Remedy These Problems

- Evaluate the potential of 4 deep learning architectures to separate electrons/positrons from protons:
  - Multilayer Perceptron (MLP)
  - Convolutional Neural Network (CNN)
  - Residual Neural Network (ResNet)
  - Convolutional Vision Transformer(CvT)
- Train a model to have a reduced dependency on energy.
  - To train a model on low energy and have it perform well on high energy data.
- Offer a deep learning model as a viable alternative for proton rejection in future physics analyses at AMS.

# Introduction – AMS Subdetectors



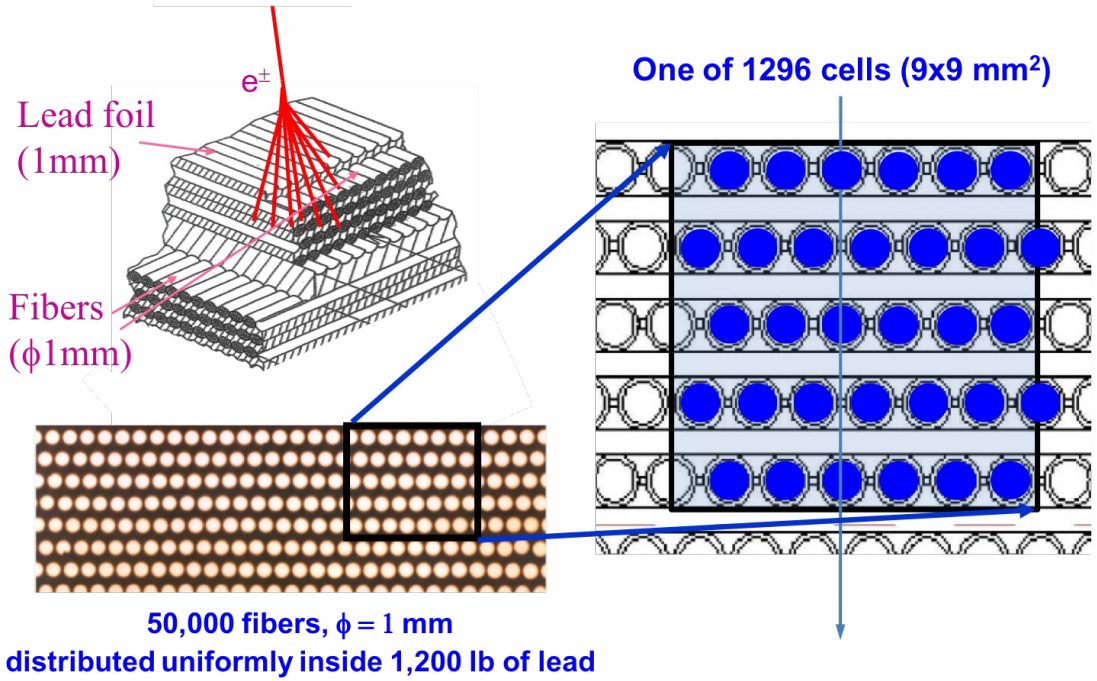

**Data Signature of Various Particles in Each Detector**

| | $e^-$ | P | Fe | $e^+$ | $\bar{P}$ | $\overline{He}$ |
|---|---|---|---|---|---|---|
| TRD | | | | | | |
| TOF | | | | | | |
| Tracker + Magnet | | | | | | |
| RICH | | | | | | |
| ECAL | | | | | | |
| Physics example | Cosmic Ray Physics Strangelets | | | Dark matter | | Antimatter |

Electrons and Positron practically the same for our purposes

# The AMS Electromagnetic Calorimeter

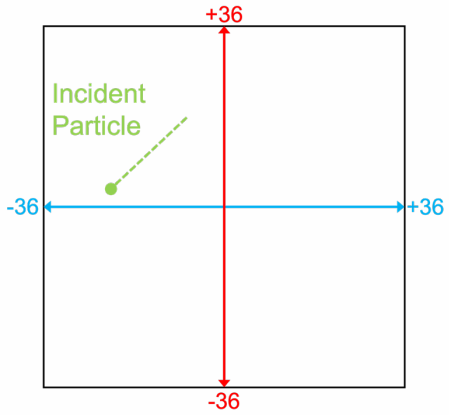
- Particles shower when they enter the ECAL.
- Lead-scintillating fibres, running along one direction [5], generate light.
- Photomultiplier tubes at the end collect the generated light.
- Measures Energy Deposition
- 3D imaging capability
  - 648 x 648 x 166 $mm^3$
  - Depth of 17 radiation lengths
  - 18 cells for depth, 72 cells for the x/y axis.
- 18 layers, 10 for X axis, 8 for y axis



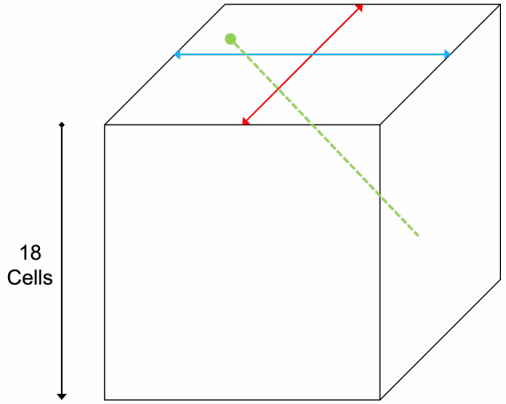

[5] C. Adloff *et al.*, Nucl Instrum Methods Phys Res A, vol. 714, pp. 147–154, Jun. 2013, doi: 10.1016/J.NIMA.2013.02.020

# ECAL – How We Handle ECAL Data

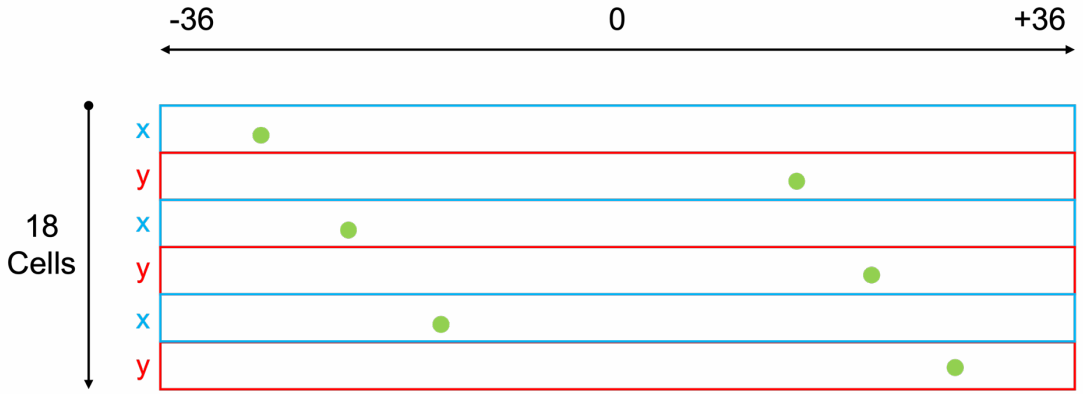
## Top-Down View



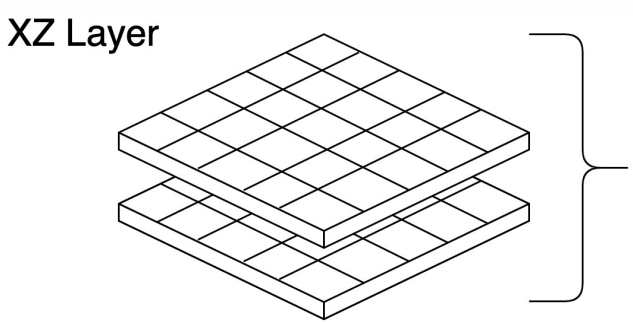

## 3D View




## 2D View



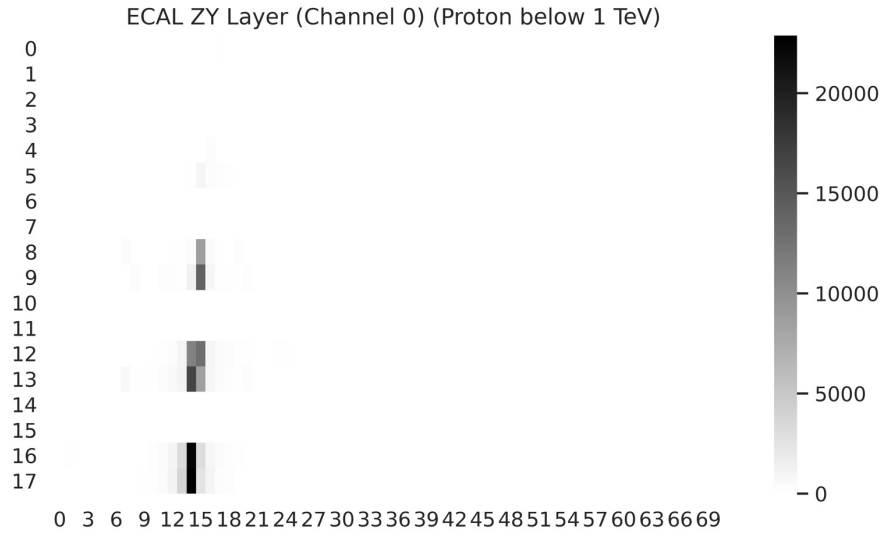

Dimensions: 1 x 18 x 72

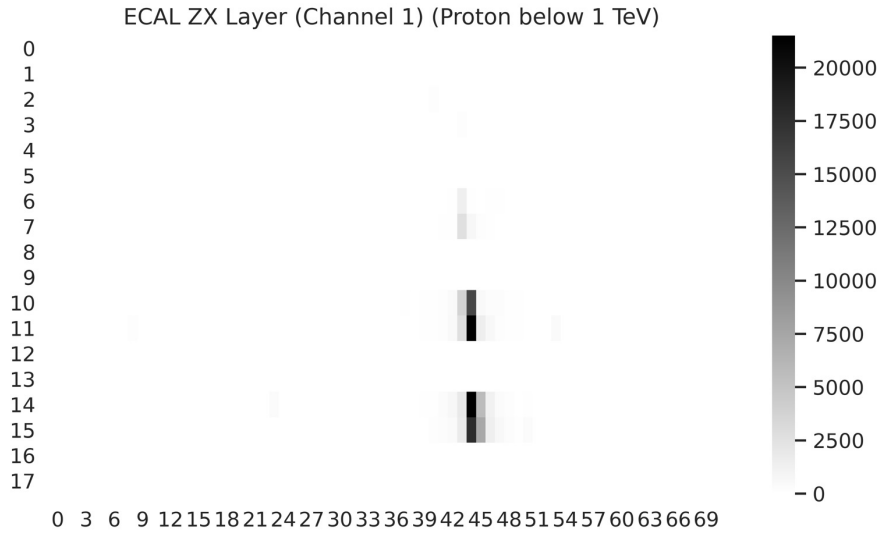


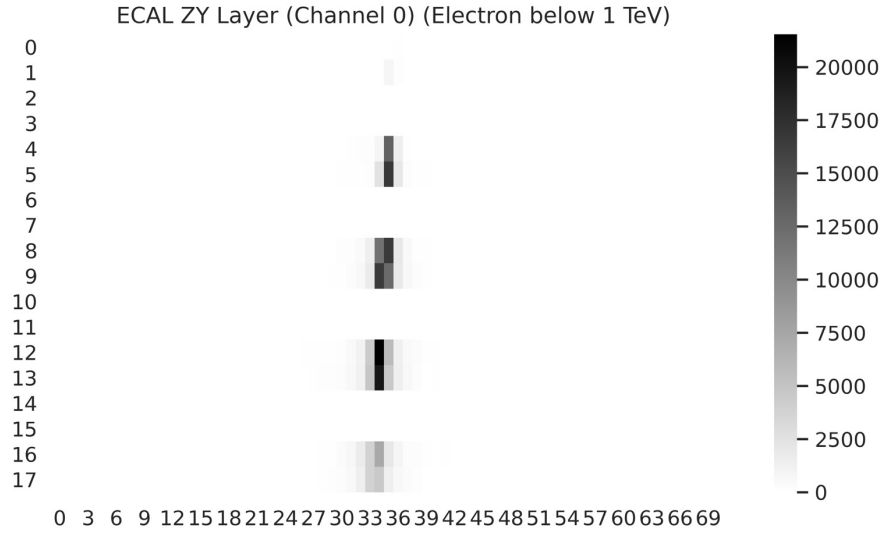

Dimensions:
2 x 18 x 72

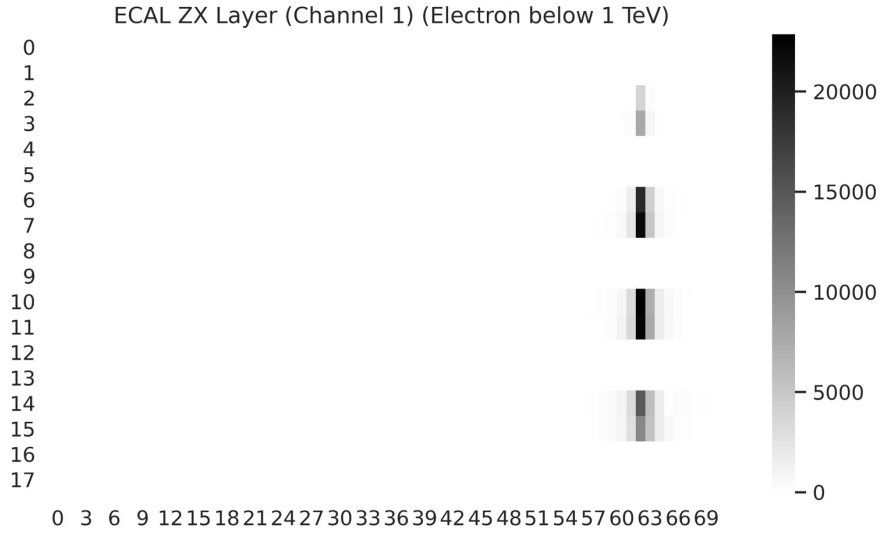
Reformat

# ECAL – Example Shower Images

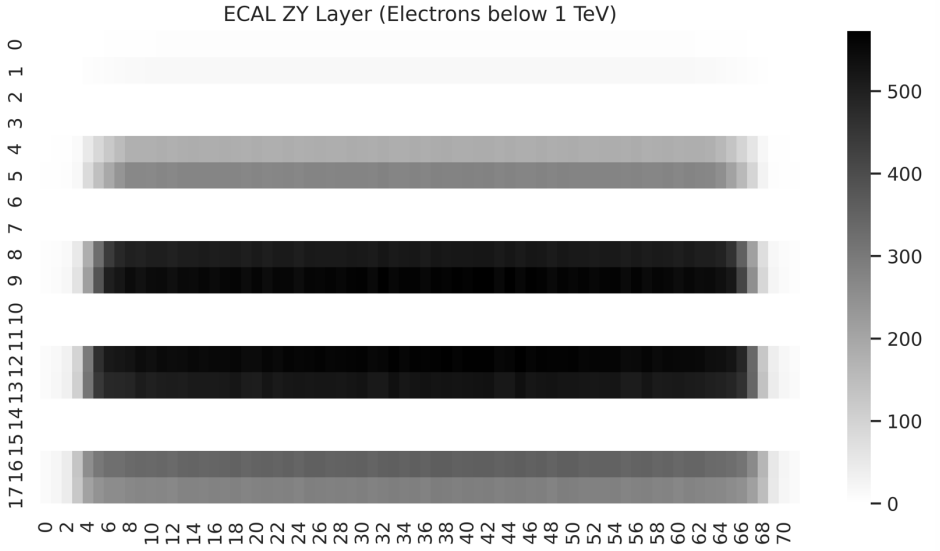


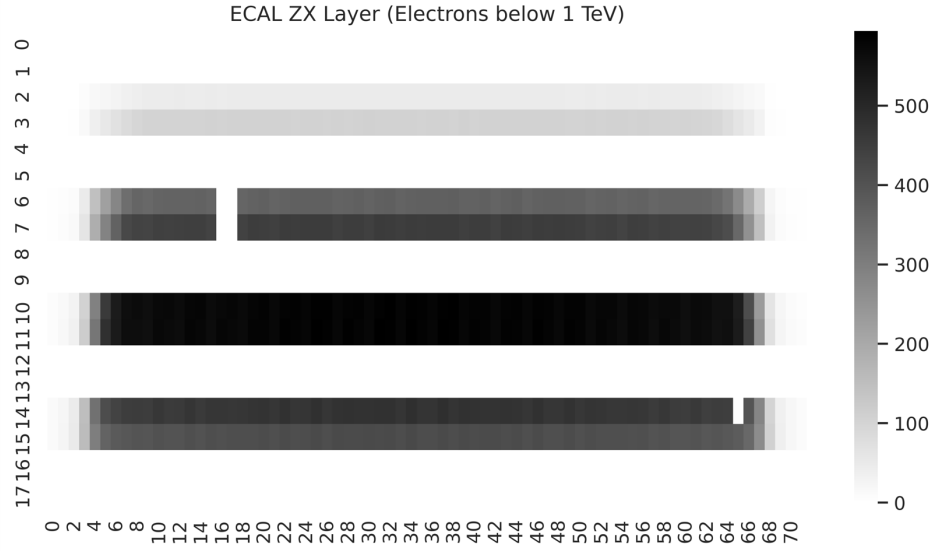



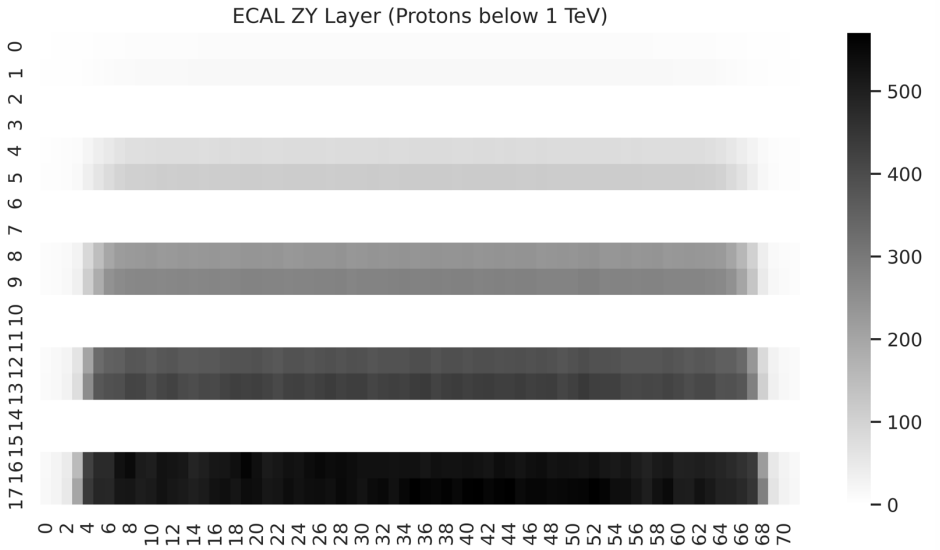



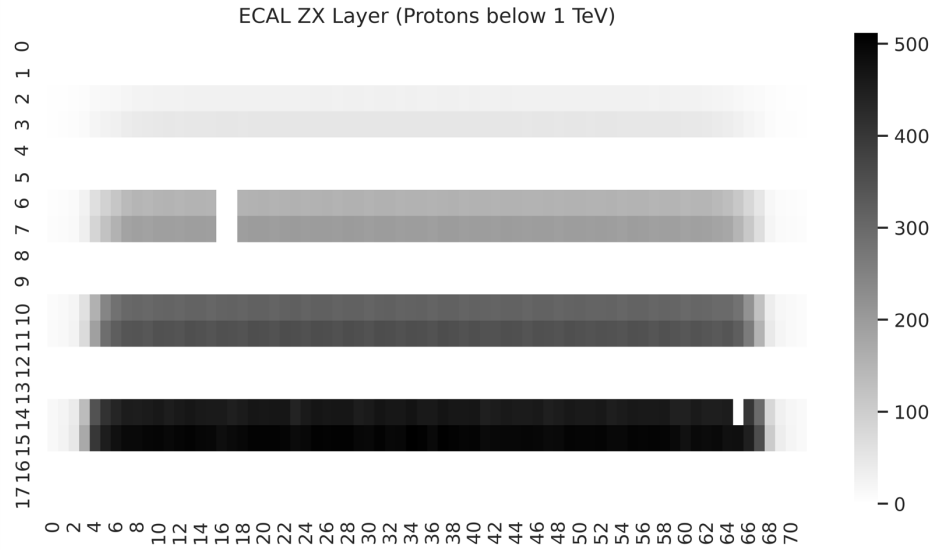



# ECAL – Average of Showers

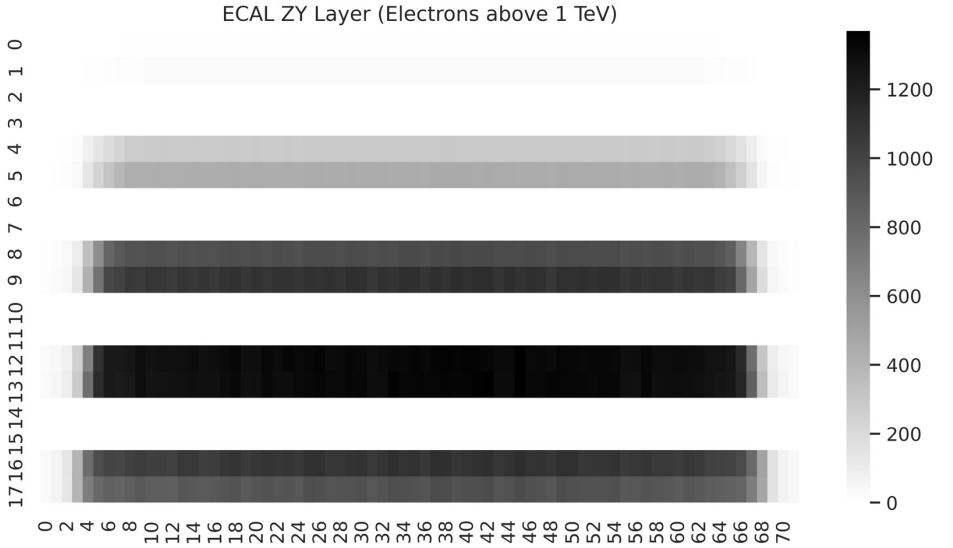


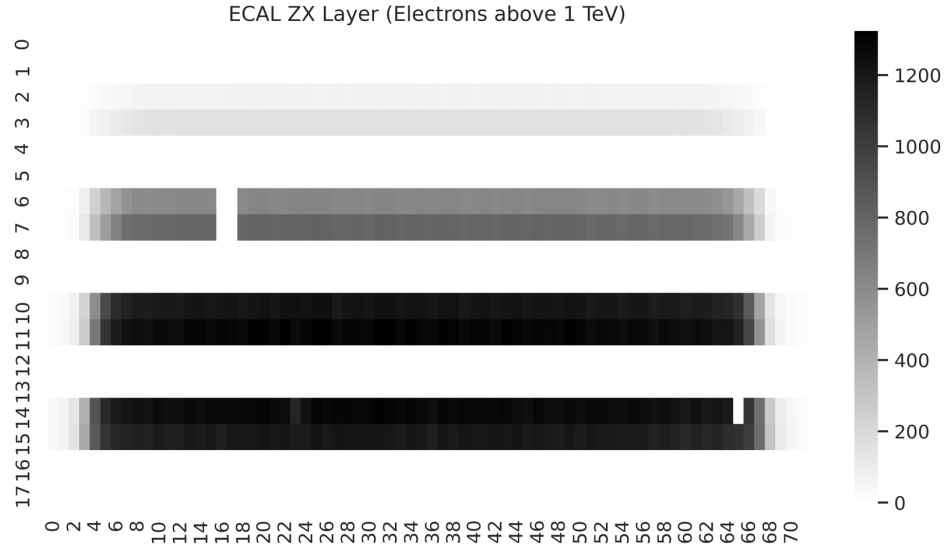



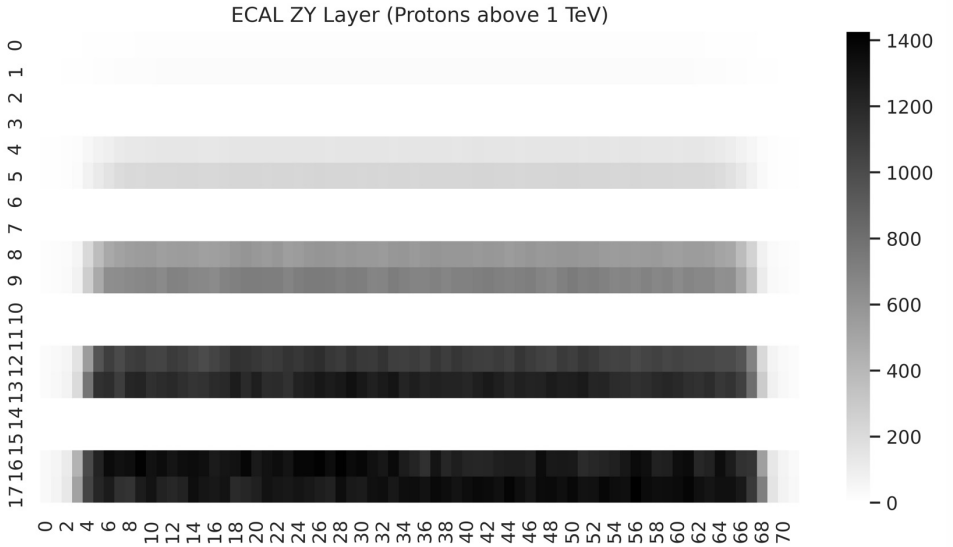



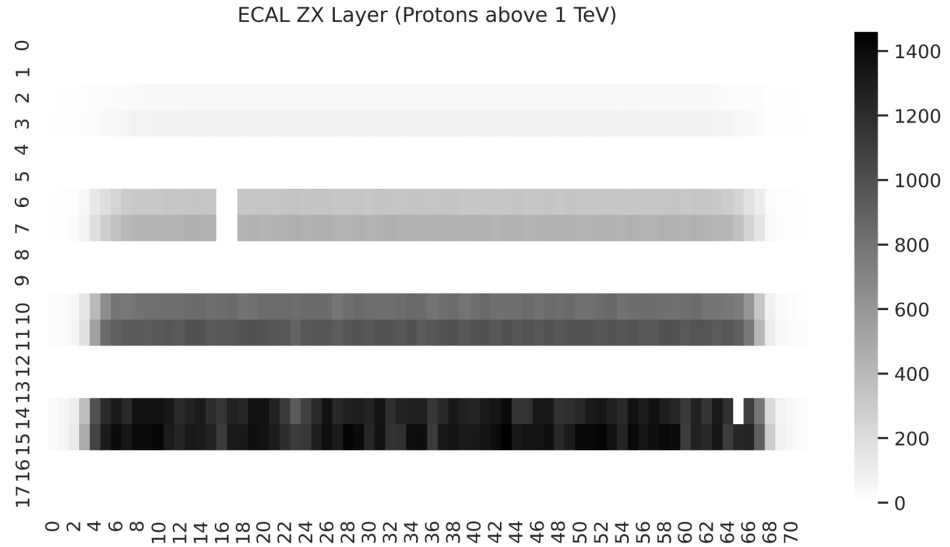



# ECAL – Average of Showers




Evidence of a domain shift: between above & below 1 TeV particles.

# ECAL – Datasets

We extract 4 datasets from AMS ROOT files to be used in Python using a script with defined cuts:

1. MC with a generated energy between 200 – 600 GeV.
2. MC with a reconstructed energy between 200 – 2000 GeV.
3. MC with a reconstructed energy between 200 – 2000 GeV + Additional variables from Tracker.
4. ISS data with a reconstructed energy between 50 – 70 GeV.
   - Used TRD as an independent method to get electrons and protons.
   - Energy range selected to get a pure (reliable labeling) and large dataset (large amounts of electrons and protons at this energy range in space).

Number of events (i.e. images) in each dataset.

| Source | Dataset | Below 1 TeV (in Millions) | | Above 1 TeV (in Millions) | |
|---|---|---|---|---|---|
| | | Electrons | Protons | Electrons | Protons |
| MC | 200-600 GeV, Generated | 4.60 | 0.16 | 0 | 0 |
| MC | 200-2000 GeV, Reconstructed | 7.03 | 3.90 | 2.69 | 1.19 |
| MC | 200-2000 GeV, Rec. + Tracker Variables | 7.51 | 3.98 | 2.89 | 1.21 |
| ISS | 50-70 GeV, Reconstructed | 0.03 | 1.19 | 0 | 0 |

# Data Influences Training Methodology

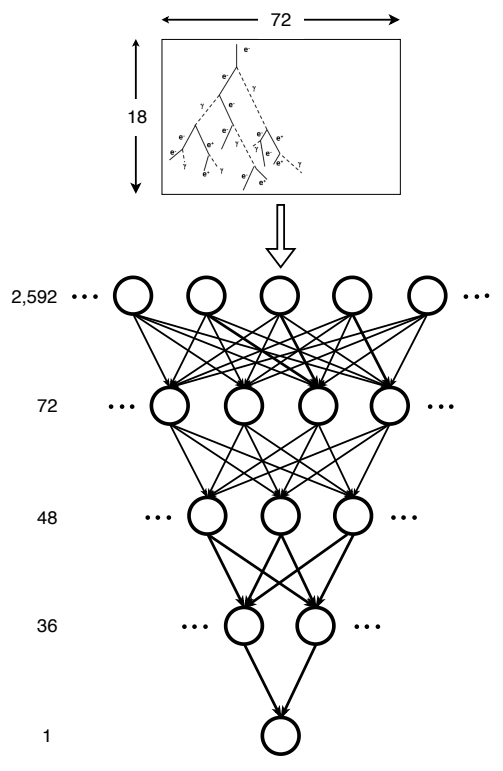
1. For dataset containing MC particles with gen. energy between 200 – 600 GeV:
   - Dataset split 60/20/20 into train/validation/test sets.
2. For dataset containing ISS data particles with rec. energy between 50 – 70 GeV:
   - Dataset split 60/20/20 into train/validation/test sets.
3. For dataset containing MC particles with rec. energy between 200 – 2000 GeV:
   - Dataset split into < 1 TeV and > 1 TeV sets.
   - < 1 TeV split 50/50 into train/test sets.
   - > 1 TeV split into 50/50 validation/test sets.
   - Thus, models that regularized, generalized, and focused on shower shape would perform better on the unseen higher energy events in the validation set. After training, models tested on both test sets.

# Simple Machine Learning Models with Scikit-learn [9]

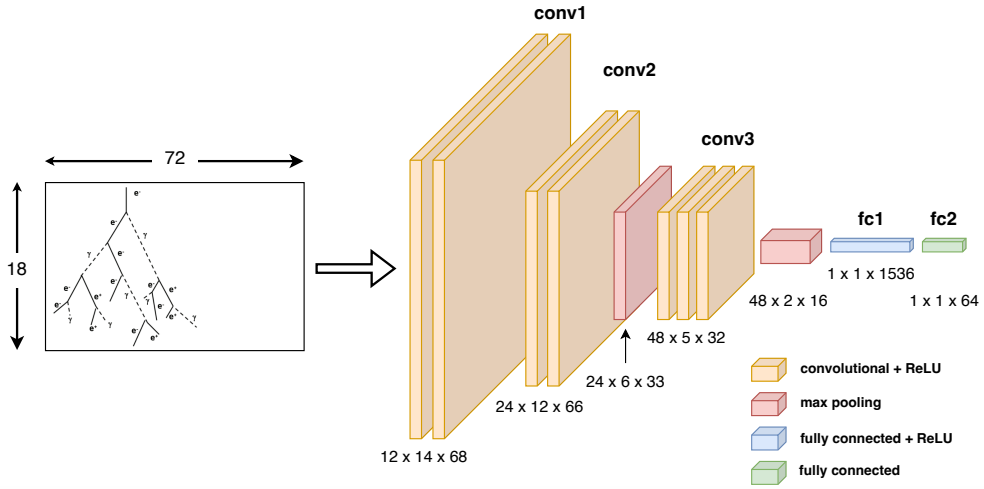
- To provide an initial benchmark for our deep learning models.
- Logistic Regression (LogReg)
- Support Vector Machine (SVM)
- Histogram-based gradient boosting decision tree (HistBDT)

[9] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, no. 85, pp. 2825–2830, 2011.

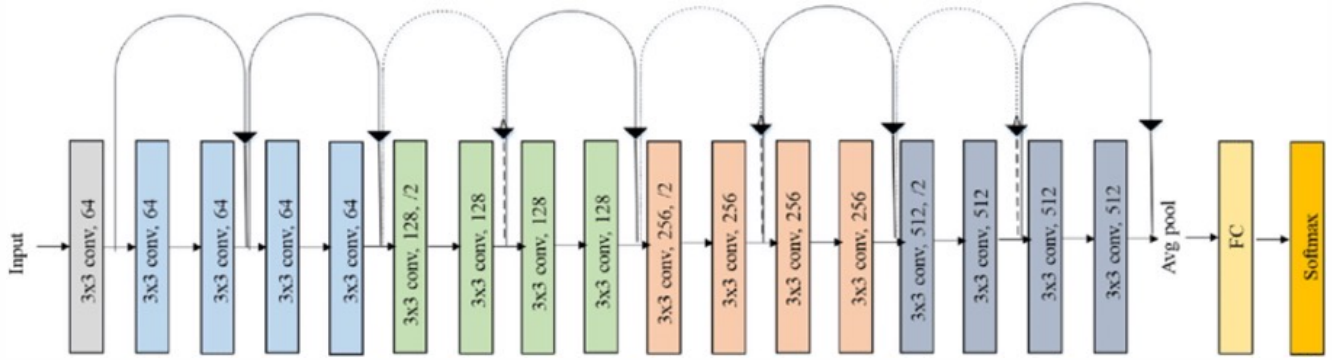
# Deep Learning Models with PyTorch [10]




SimpleMLP



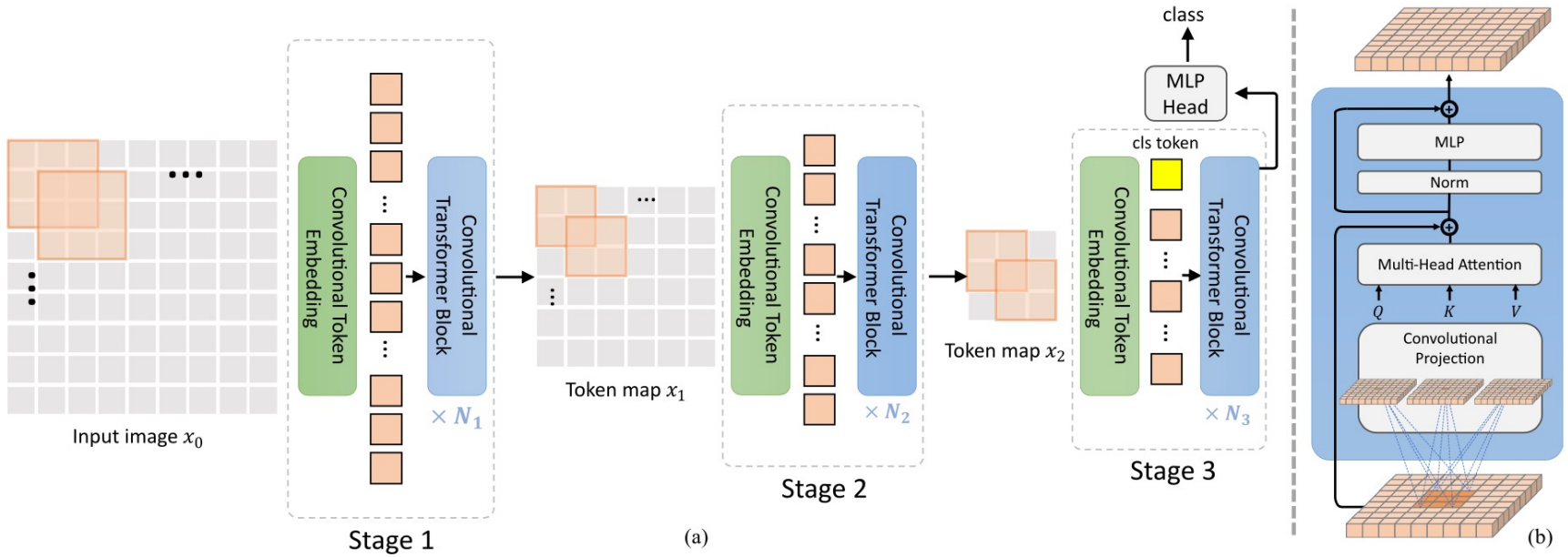

SimpleCNN




ResNet18 (also ResNet10) [11]


[10] A. Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in Adv Neural Inf Process Syst 32, 2019, pp. 8024–8035.
[11] K. He, X. Zhang, S. Ren, and J. Sun, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016

# ML Models – Convolutional Vision Transformer



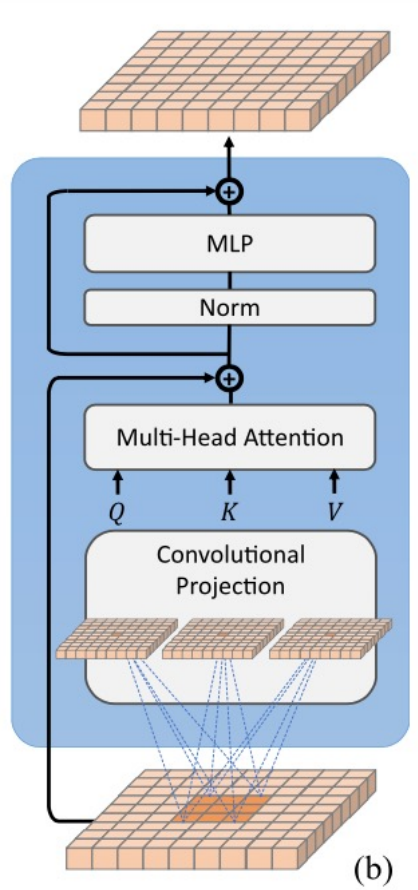

The pipeline of the CvT architecture [14]. (a) Overall architecture, showing the hierarchical multi-stage structure facilitated by the Convolutional Token Embedding layer. (b) Details of the Convolutional Transformer Block, which contains the convolution projection as the first layer.

[14] H. Wu et al., “CvT: Introducing Convolutions to Vision Transformers,” Proc. IEEE Int. Conf. Comput. Vis., pp. 22–31, 2021, doi: 10.1109/ICCV48922.2021.00009

# Attention Mechanism (AM)



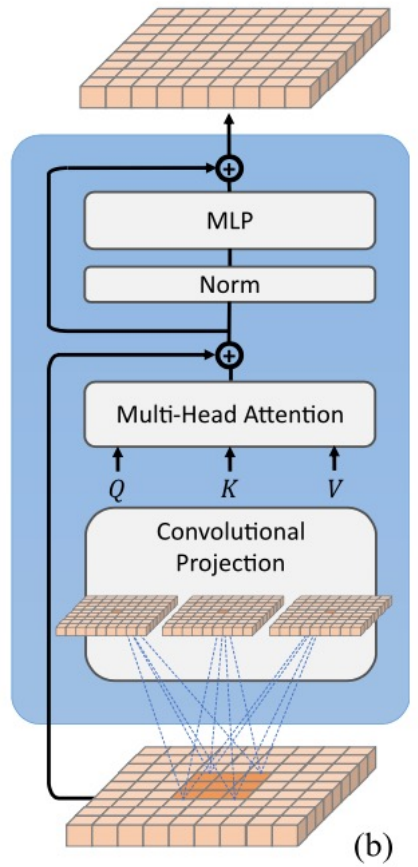
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



# Attention Mechanism (AM)

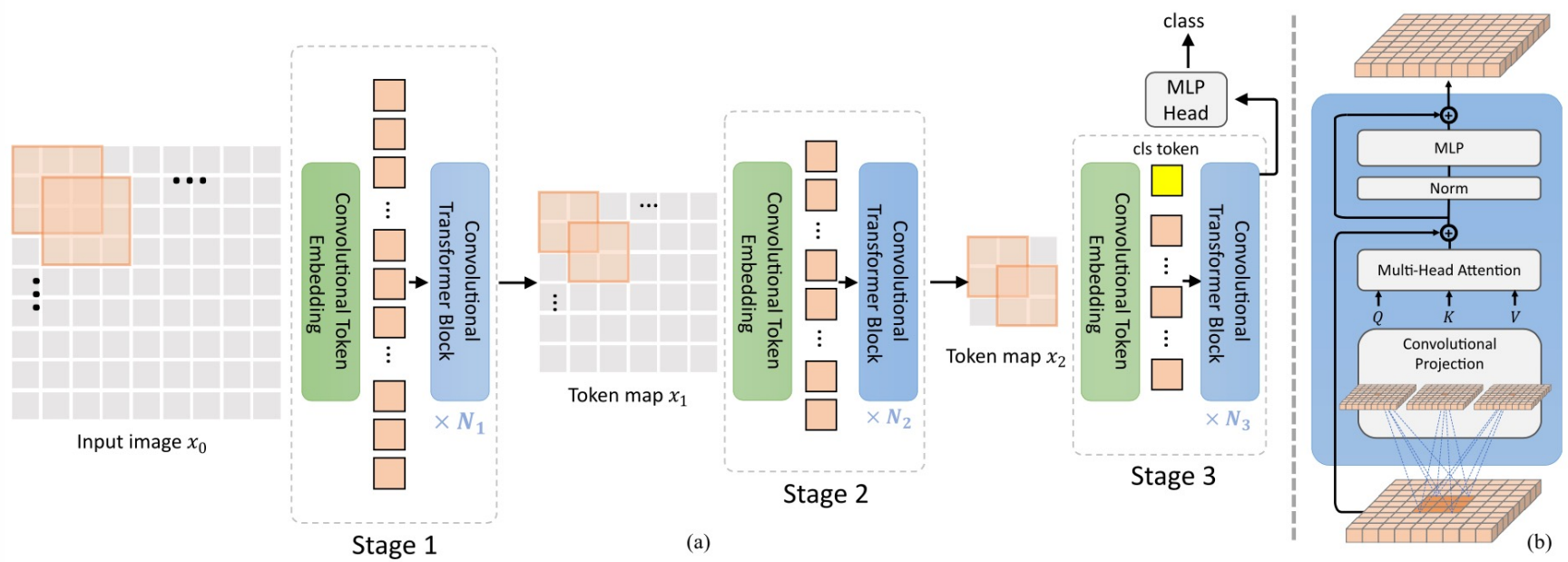


- V (Value): Represents the output values, i.e., the AM's outputted feature maps, that are to be weighted by the attention scores.
- Q (Query): Representation of the features that the AM is focusing on.
- K (Key): Representation of each element of the feature maps.
- Dot product of Q and K gives us the attention scores, which measure similarity between the query and each element of the feature maps.
- Divided by $\sqrt{d_k}$ to normalize the output values.
- AM can learn to focus on regions of the feature map that are most discriminative for a class.




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

# ML Models – Convolutional Vision Transformer




The pipeline of the CvT architecture [14]. (a) Overall architecture, showing the hierarchical multi-stage structure facilitated by the Convolutional Token Embedding layer. (b) Details of the Convolutional Transformer Block, which contains the convolution projection as the first layer.

[14] H. Wu et al., "CvT: Introducing Convolutions to Vision Transformers," Proc. IEEE Int. Conf. Comput. Vis., pp. 22–31, 2021, doi: 10.1109/ICCV48922.2021.00009

# ML Models – Convolutional Vision Transformer

- Combines the benefits of CNNs (translation equivariance) with Transformers (better generalization, better focus on key areas, global context).
- Important for ECAL showers ➔ shower “images” show different deposition shapes depending on the energy and type of particle, angle of incidence, and point of entry.
- Our implementation slightly modified ➔ uses 4 Transformer blocks (Stage 3 having 2 blocks), and 1, 3, and 6 attention heads for each stage, respectively.
- We use 2 variants:
  - CvT & Phys+CvT
  - Phys+CvT (small kernels)

| | | Stage 1 | Stage 2 | Stage 3 |
|---|---|---|---|---|
| CvT & Phys+CvT | Kernels | 2 x 6 | 3 x 4 | 3 x 3 |
| | Stride lengths | 1 x 2 | 1 x 2 | 1 x 1 |
| Phys+CvT (small kernels) | Kernels | 3 x 1 | 4 x 3 | 3 x 3 |
| | Stride lengths | 1 x 1 | 2 x 1 | 1 x 1 |

# ML Models – Some Values

| Hyperparameter | Value |
|---|---|
| Batch Size | 128 |
| No. of Workers | 4 |
| Loss Function | Binary Cross Entropy with Logits, Weighted |
| Activation Function after Last Layer | Sigmoid |
| Optimizer | Adam |
| Learning Rate for Optimizer | 1.0e-4 |
| Early Stopping Patience | 20 |
| Early Stopping Tolerance | 1.0e-5 |

| Model (1 Channel) | Trainable Parameters |
|---|---|
| SimpleMLP | 192,001 |
| SimpleCNN | 106,317 |
| ResNet10 | 4,900,033 |
| ResNet18 | 11,170,753 |
| CvT | 4,895,873 |
| Phys+CvT | 4,896,641 |
| Phys+CvT (Small Kernels) | 4,895,489 |

# ML Models – Performance Metrics

1. Loss and Accuracy Plots
2. Proton Rejection

# ML Models – Proton Rejection



- Proton rejection is then:
  - Proton Rejection = $\frac{\text{Total number of protons}}{\text{Number of proton misidentified}}$

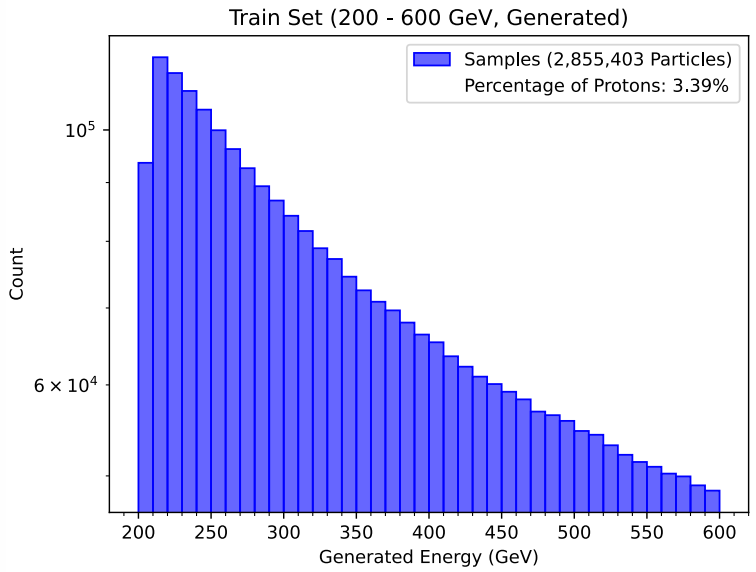
- We plot proton rejection vs. various electron efficiencies.

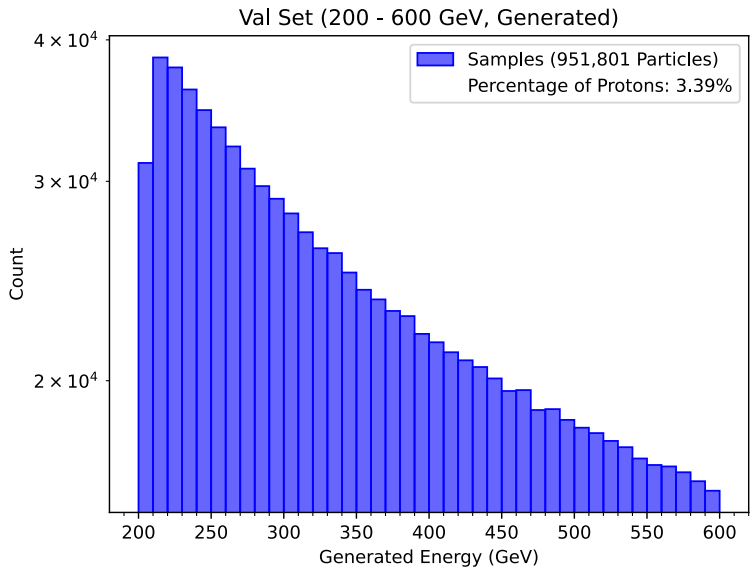
- We plot proton rej. vs. energy for a given electron efficiency to get an in-depth understanding of which energy bins our models perform poorly at.

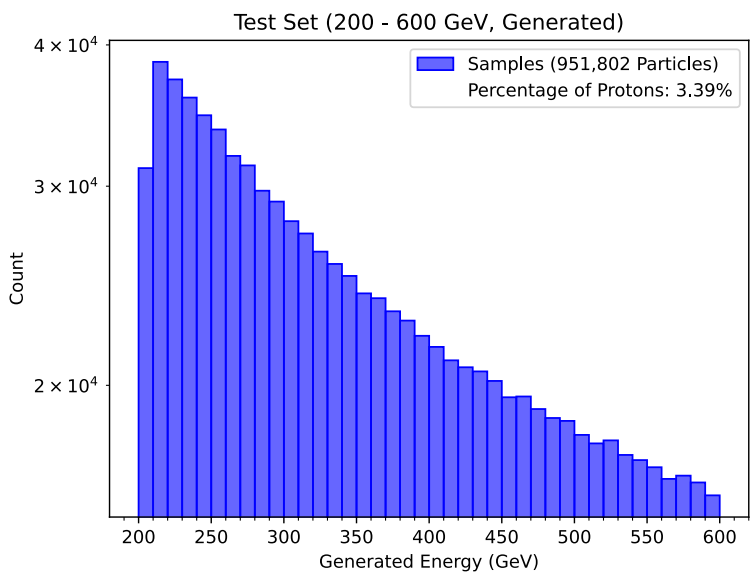
# Experiments & Results

# Exp 1 – Simple ML vs. Deep Learning Models

- Performed on Dataset 1 (MC particles with generated energy between 200 – 600 GeV).



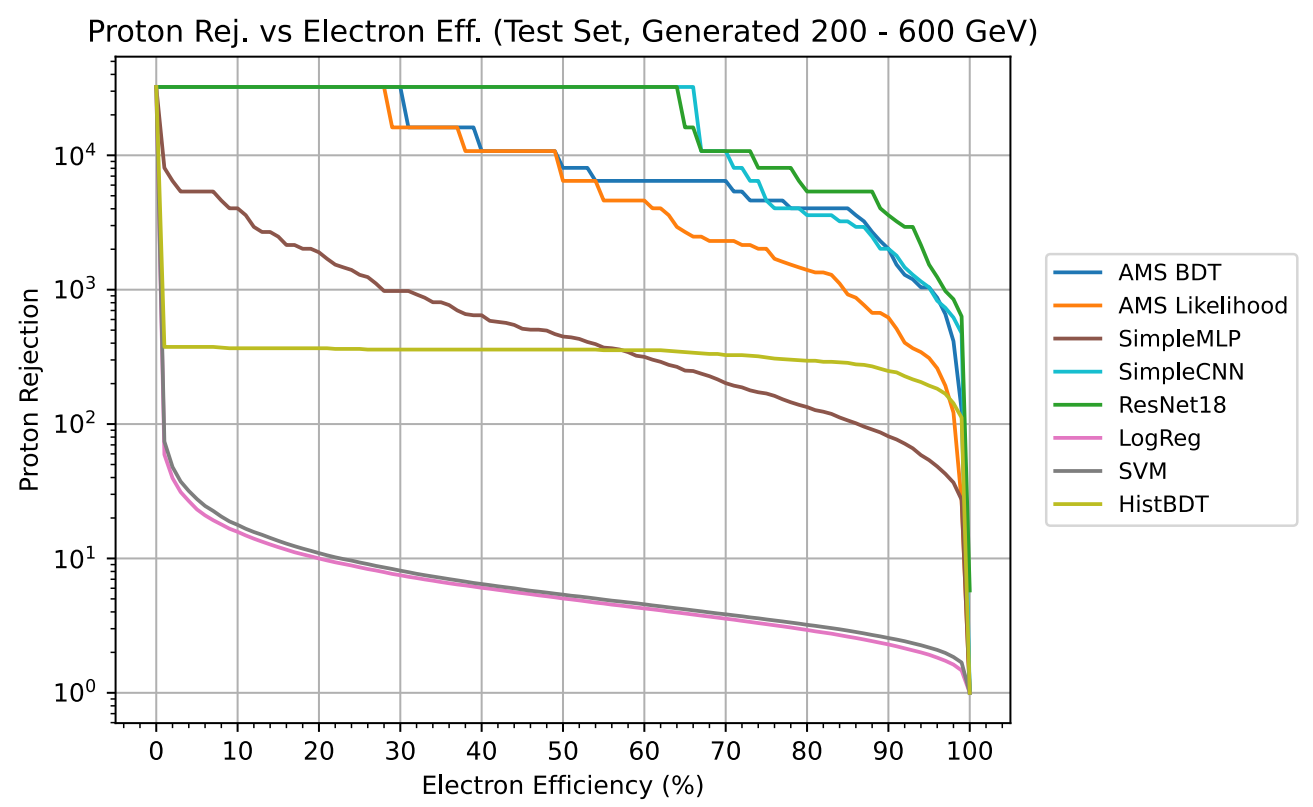
# Exp 1 – Simple ML vs. Deep Learning Models

- Performed on Dataset 1 (MC particles with generated energy between 200 – 600 GeV).
- Evaluated average accuracy ($^{Correctly\ Classified}/_{Total\ Test\ Set}$) ➔ Not very helpful

| Models | AMS BDT | AMS LHD | LogReg | SVM | HistBDT | SimpleMLP | SimpleCNN | ResNet18 |
|---|---|---|---|---|---|---|---|---|
| Accuracy | 0.973 | 0.967 | 0.843 | 0.911 | 0.998 | 0.988 | 0.999 | 0.998 |

# Exp 1 – Simple ML vs. Deep Learning Models

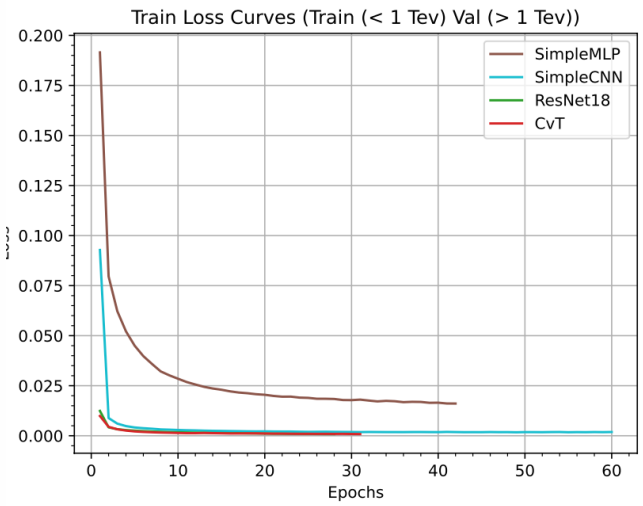
- Performed on Dataset 1 (MC particles with generated energy between 200 – 600 GeV).
- Evaluated average accuracy ($^{Correctly\ Classified}/_{Total\ Test\ Set}$) ➔ Not very helpful
- Plotted Proton Rej. vs.Electron Efficiency ➔ ResNet18 beats all models, Simple ML perform poorly
- **Conclusion**: Clear performance benefit of using DL models over simple ML models.

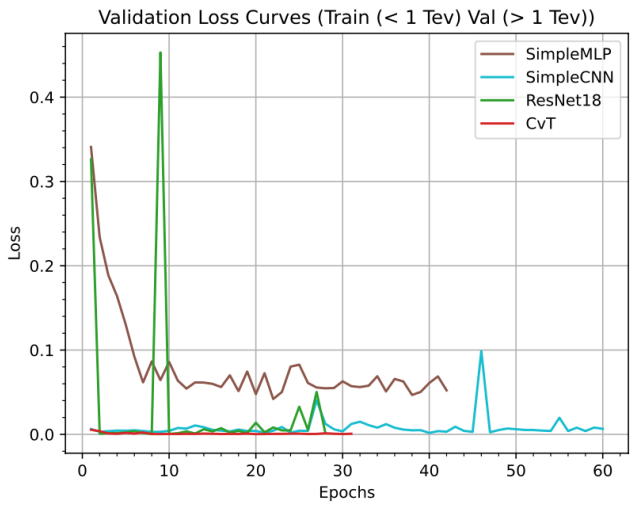


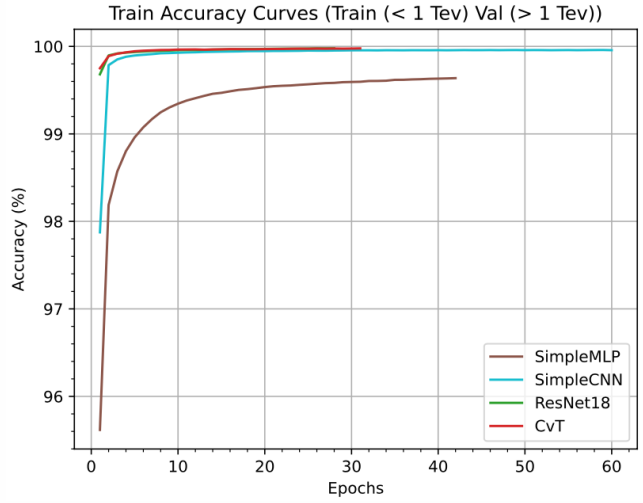
One Caveat: AMS Models were trained on data, not MC. A fairer comparison at the end.

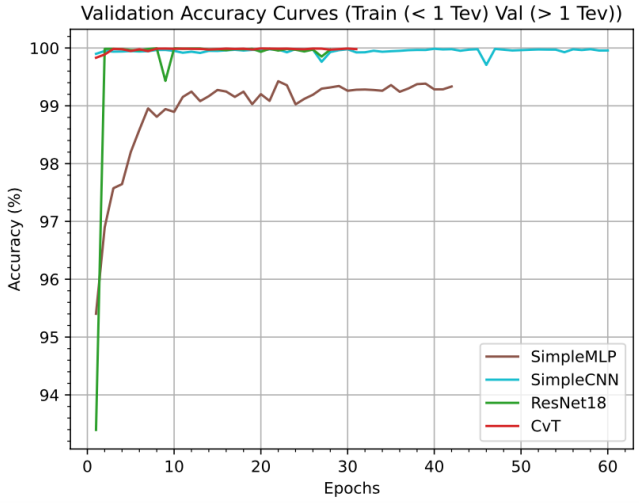
# Exp 2 – Comparison on 1st Reconstructed Energy Dataset

- Plotted loss and accuracy curves.



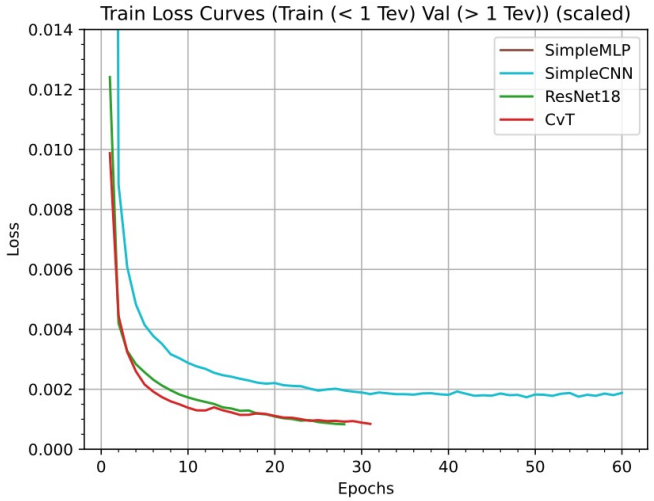



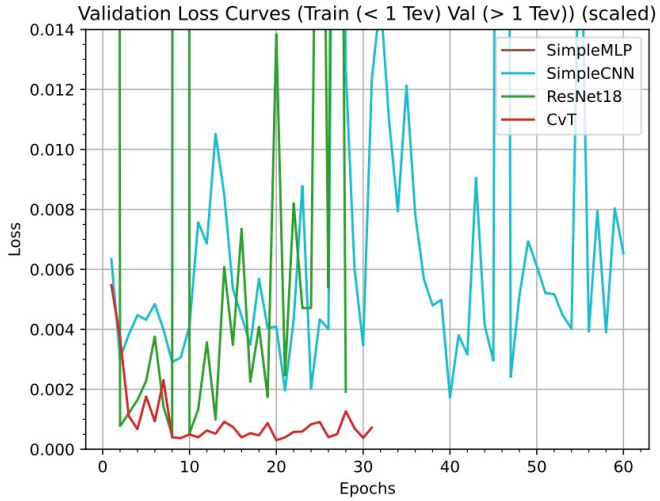



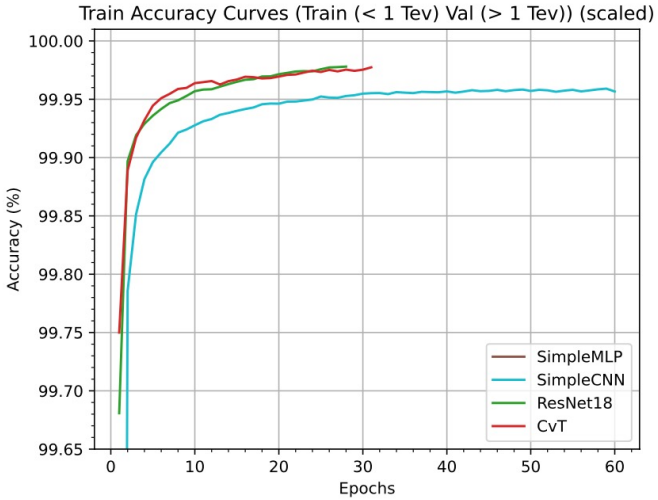



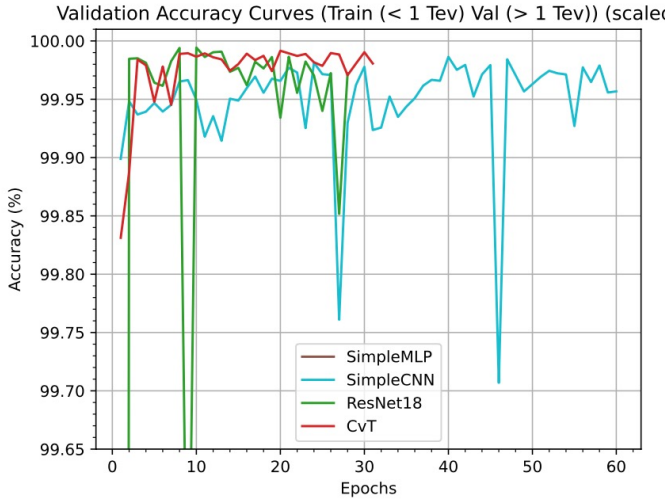
# Exp 2 – Comparison on 1st Reconstructed Energy Dataset

- Plotted loss and accuracy curves (zoomed and scaled) ➔ CvT performs the best on val set.

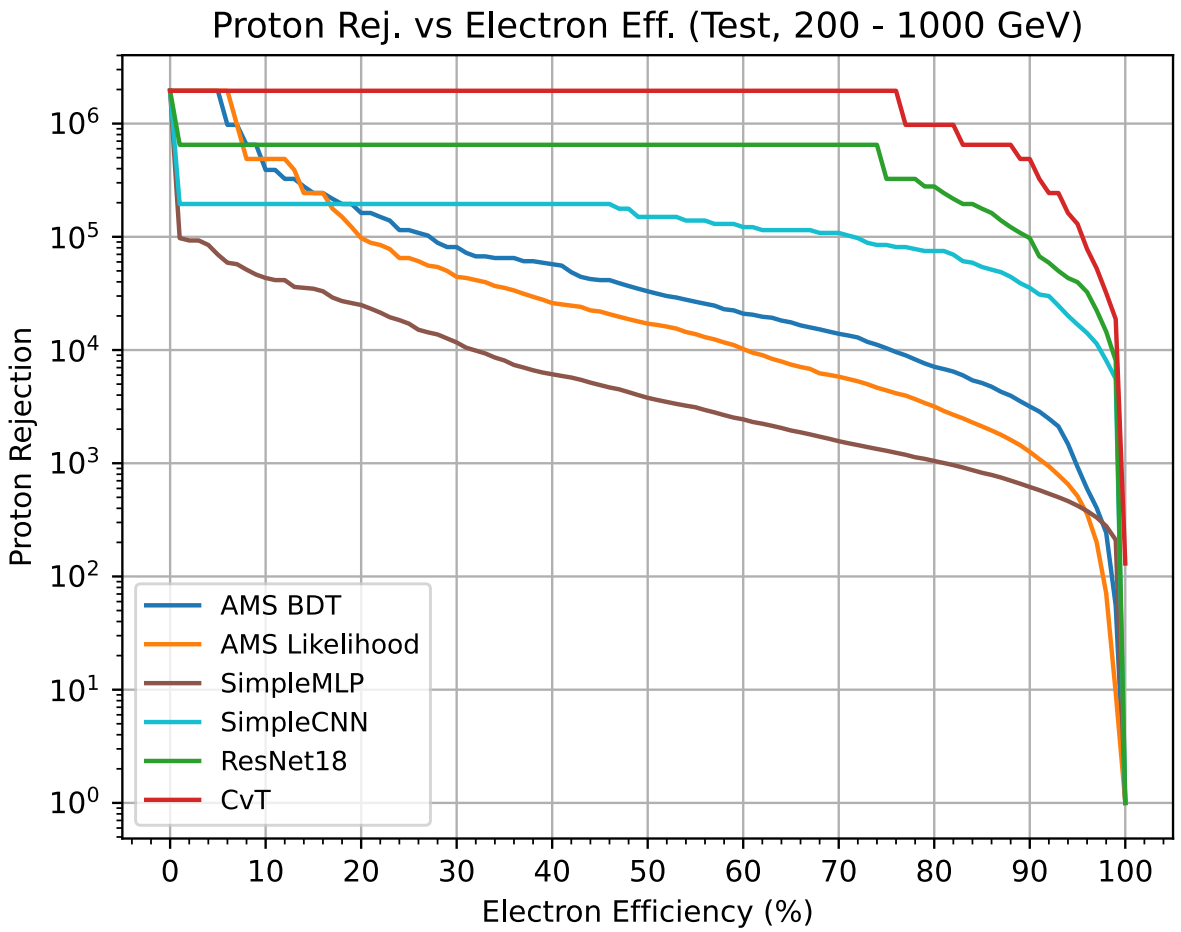


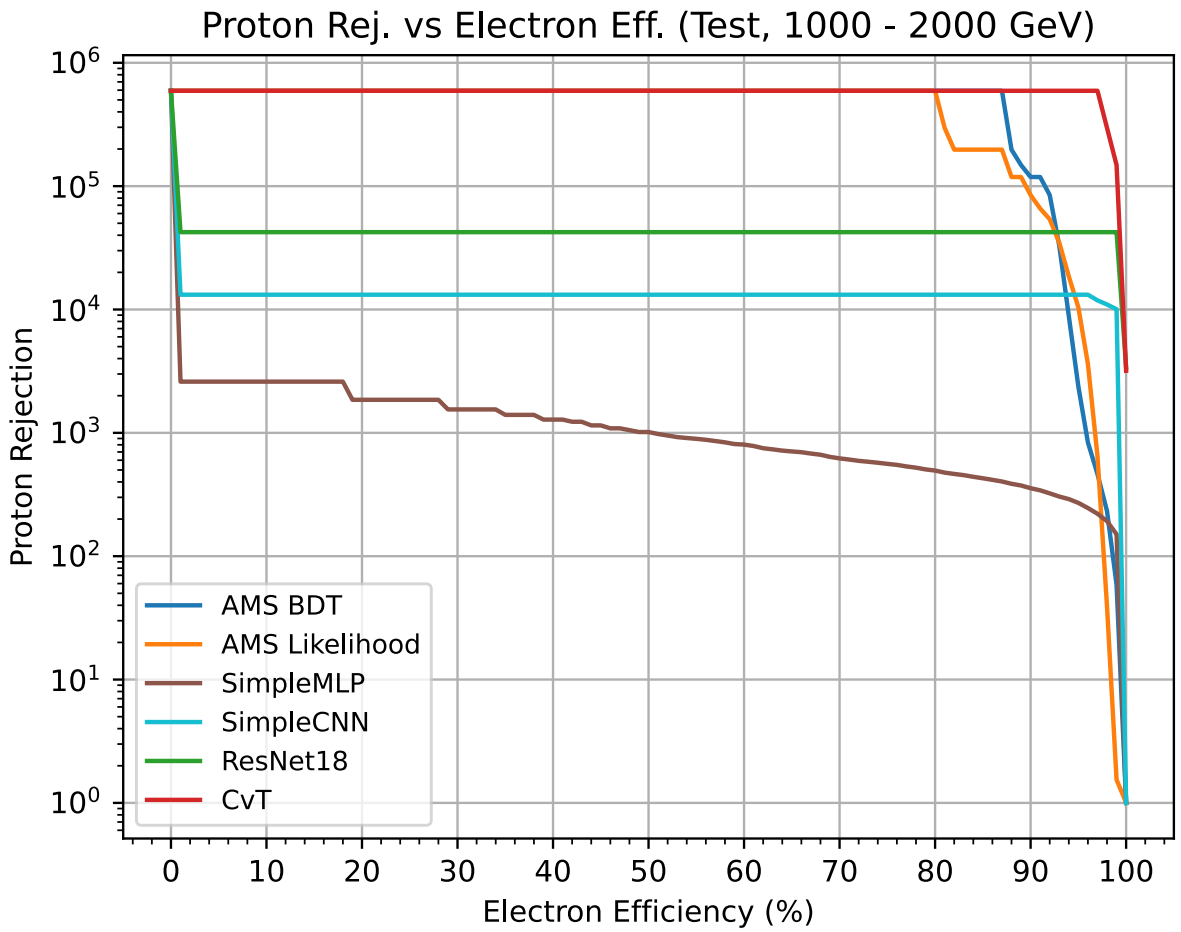



# Exp 2 – Proton Rej. vs. Electron Efficiency

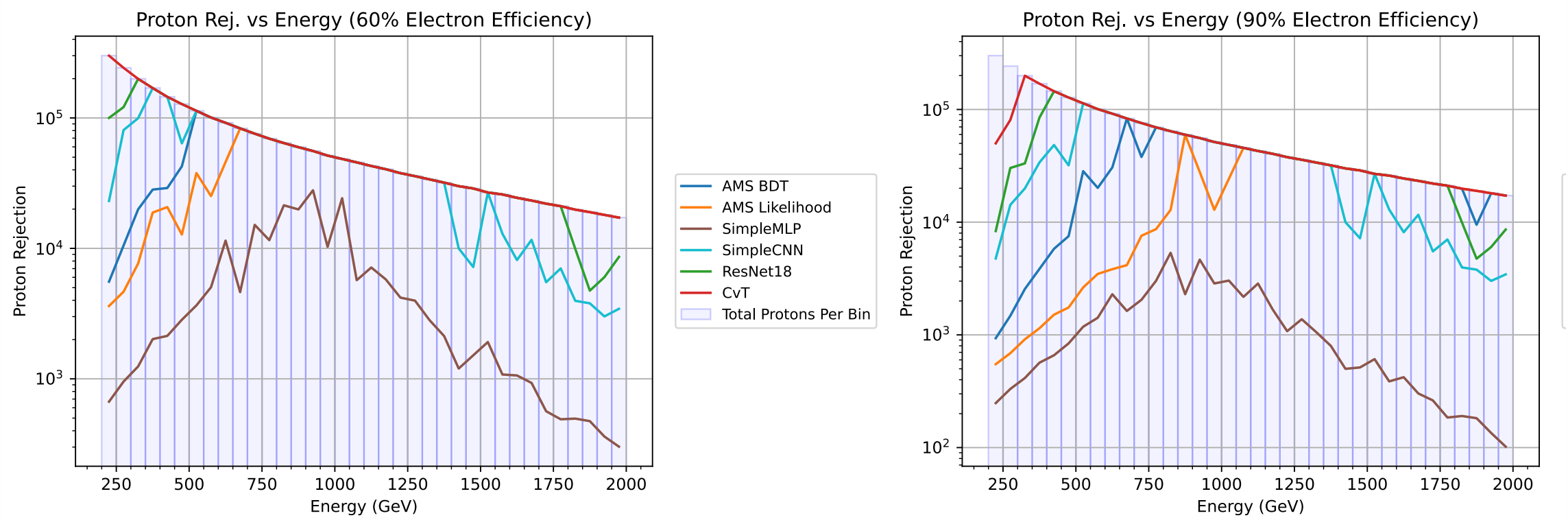



At 90% electron efficiency, CvT outperforms the ResNet18, SimpleCNN, AMS BDT, AMS Likelihood, and SimpleMLP models by factors of 5, 14, 153, 386, and 789, respectively




At 90% electron efficiency, CvT outperforms the AMS BDT, AMS Likelihood, ResNet18, SimpleCNN, and SimpleMLP models by factors of 5, 7, 14, 45, and 1666, respectively.

# Exp 2 – Proton Rej. vs. Energy



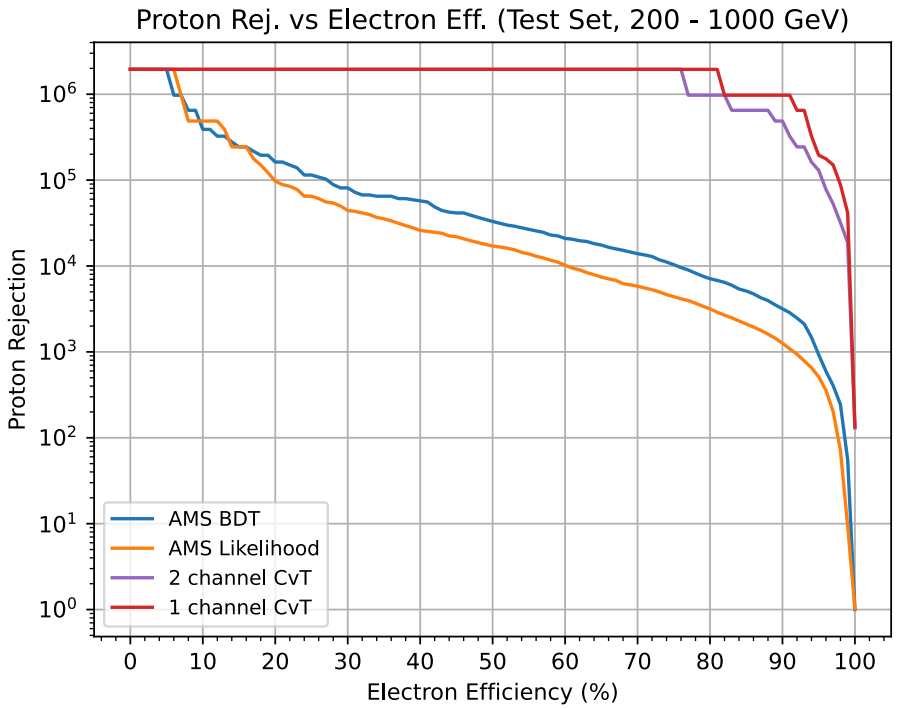

The CvT completely completely rejects all available protons.

The CvT rejects all protons above 350 GeV.

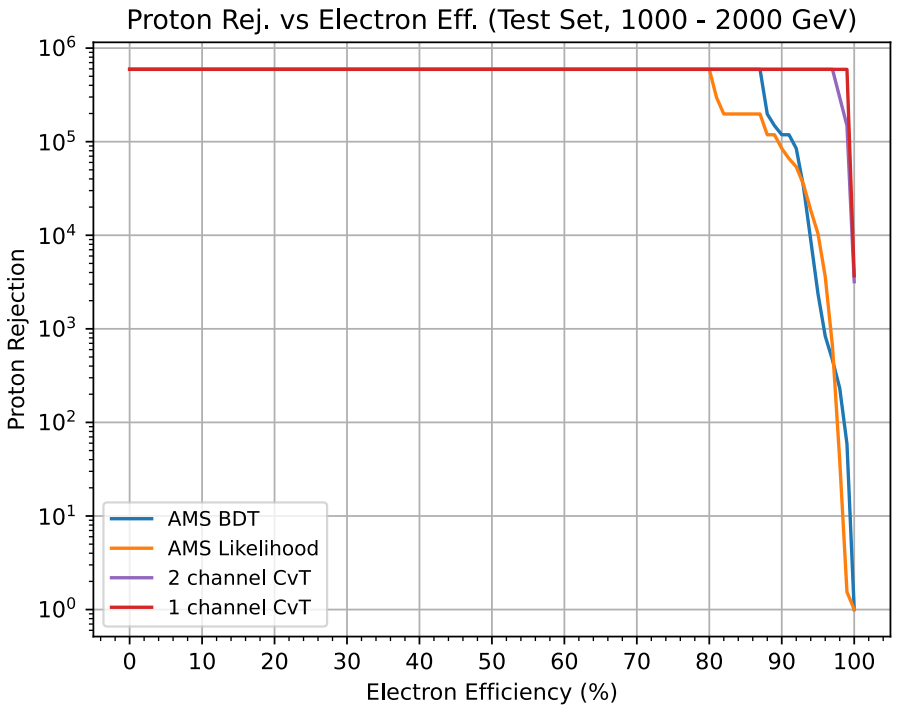
**Conclusion**: CvT performs the best, generalizes from below 1 TeV to above 1 TeV the best.

# Exp 3 – 2 Channels vs. 1 Channel Representation

- 2 x 18 x 72 vs. 1 x 18 x 72
- Results could be due to stochastic factors (initial random weight initialization).
- 1 channel has fewer input values and slightly reduced number of training parameters.
- **Conclusion**: We switch to 1 channel representation for the remainder of the experiments.




At 90% electron efficiency, the 1 channel CvT outperforms the 2 channel CvT by a factor of 2.

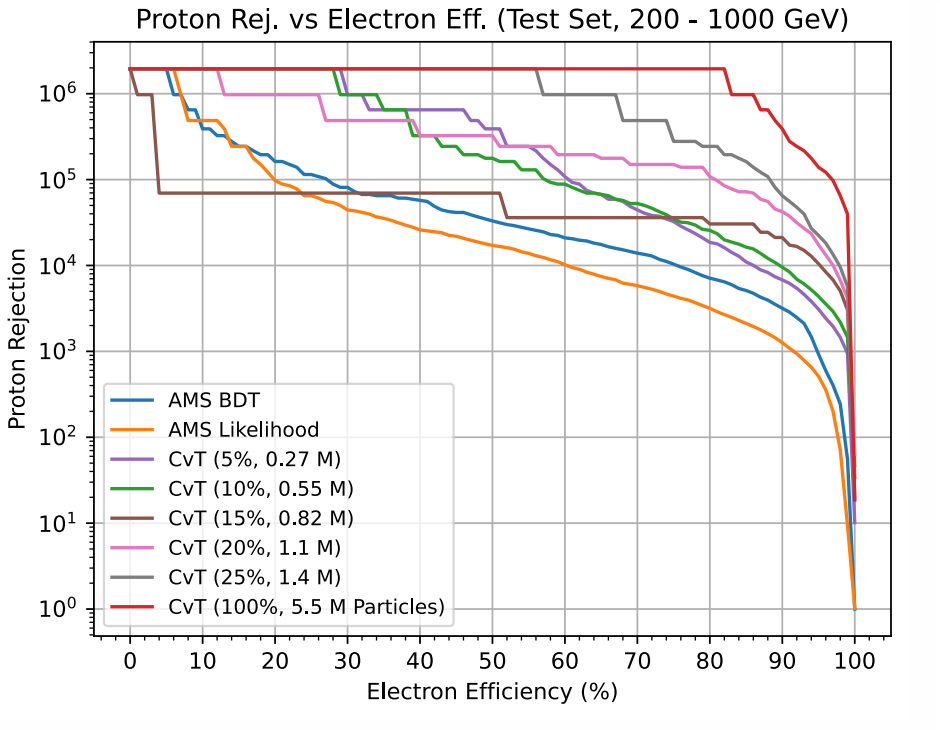


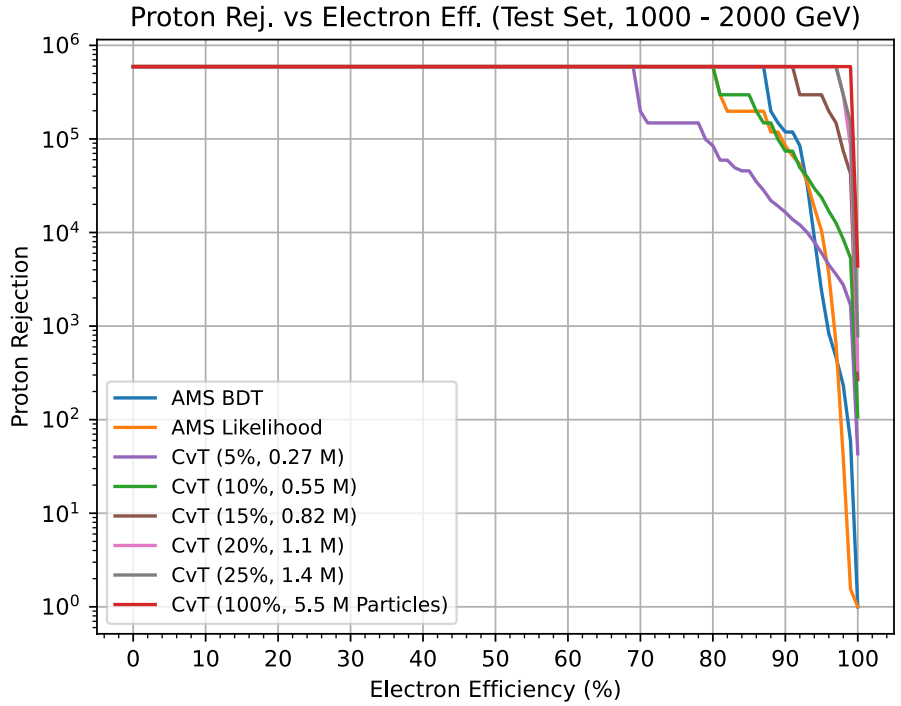

At 90% electron efficiency, both the 1 channel CvT and 2 channel variants perform equally.

# How much data do we need to train the CvT?

# Exp 4 – CvT Performance on Smaller Training Sets

- Amount of ISS Data < Amount of MC Data.
- Question: How much data is needed to effectively train a CvT for our case?
- Experiment: Train on smaller amounts of train data, validate and test on the same val and test sets.
- Conclusion: The more, the merrier.

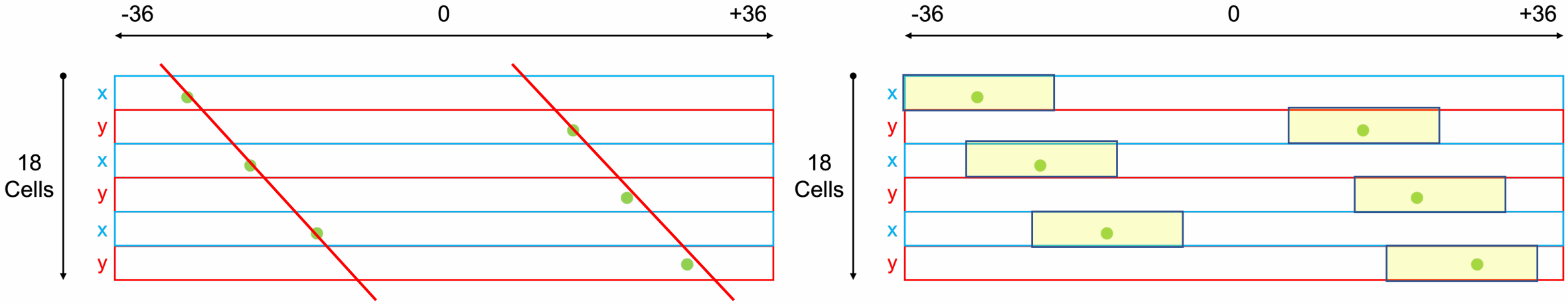


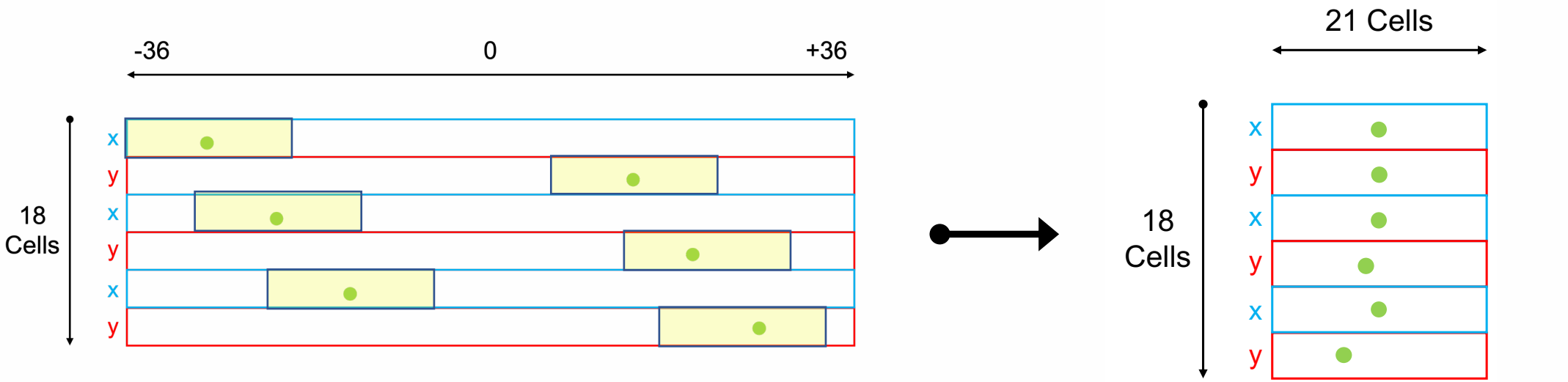

Steady increase in performance as more training data is used. Additionally, at least 0.82 million training events needed to completely outperform AMS models.

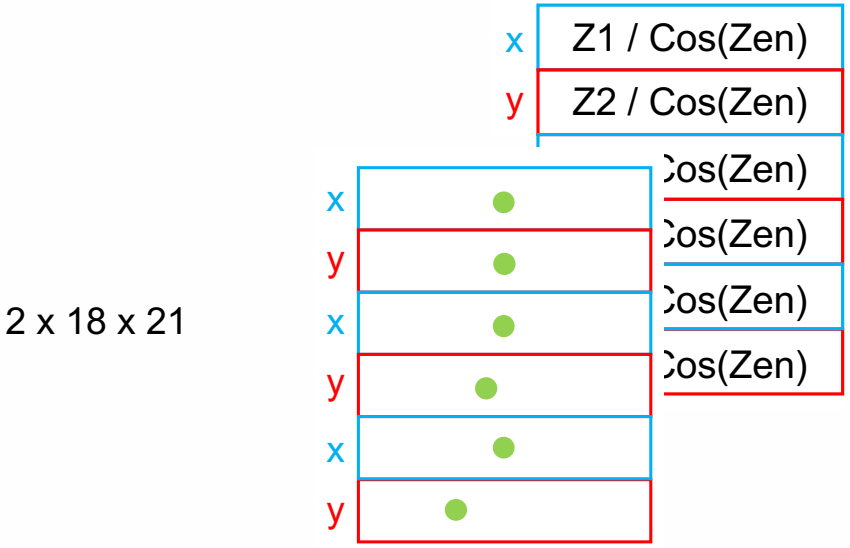
# Developing the Physics-based Feature Engineering

- Results of previous experiment motivated the need to improve learning efficiency.
- Since showers develop in a similar way ➔ Map their trajectory and only extrapolate important pixels.
- Use 5 variables from the Silicon Tracker: X-, Y-, Z-coordinates, zenith, azimuth.
- Use spherical coordinates to map trajectory in 3 dimensions.



# Developing the Physics-based Feature Engineering




Add Depth Information

Actual distance covered by particle $= \frac{Distance\ along\ Z}{Cos(Zenith)}$

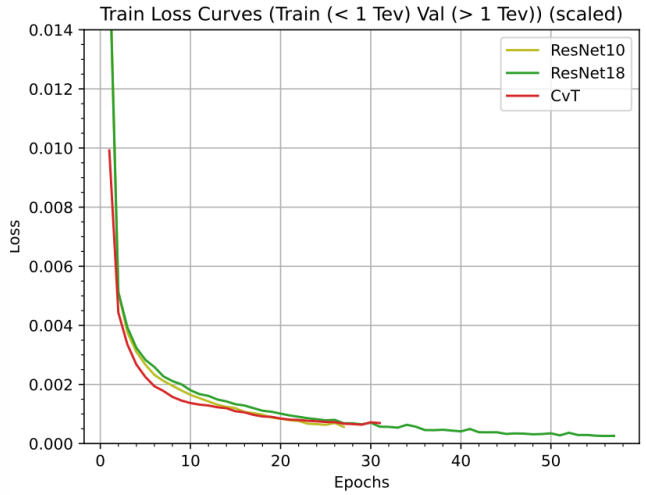
# Exp 5 – Comparison on 2nd Reconstructed Energy Dataset

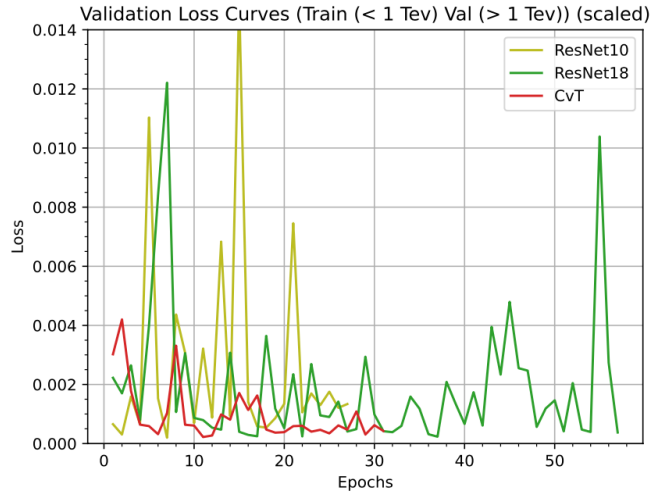
- Performed on Dataset 3 (MC particles with reconstructed energy between 200 – 2000 GeV + Tracker Variables).
- Added Phys+CvT and Phys+CvT (small kernels)
- Dropped SimpleMLP as performance was consistently bad.
- Added ResNet10 ➔ Maybe ResNet18 was overfitting.

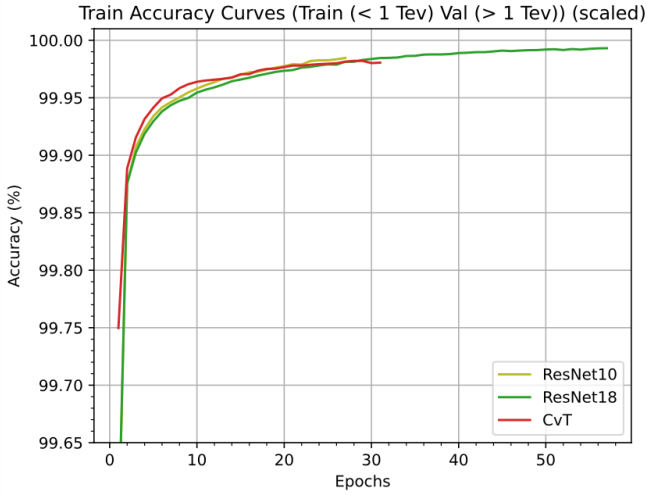
| Model (1 Channel) | Trainable Parameters |
|---|---|
| SimpleMLP | 192,001 |
| SimpleCNN | 106,317 |
| ResNet10 | 4,900,033 |
| ResNet18 | 11,170,753 |
| CvT | 4,895,873 |
| Phys+CvT | 4,896,641 |
| Phys+CvT (Small Kernels) | 4,895,489 |

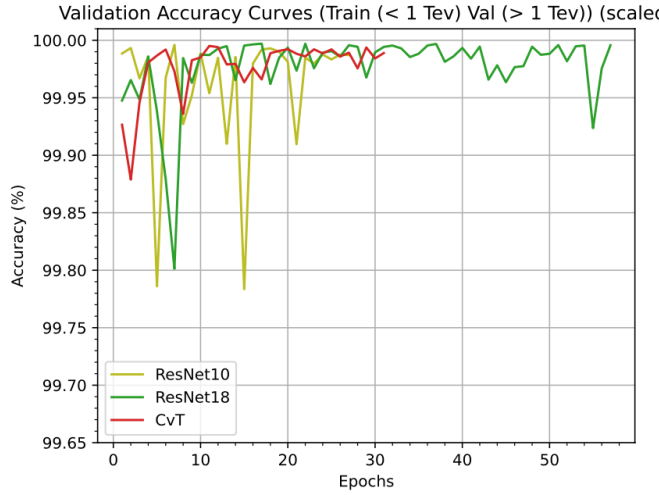
# Exp 5 – Comparison on 2nd Reconstructed Energy Dataset

- Plotted loss and accuracy curves for ResNets ➔ No significant difference between ResNets



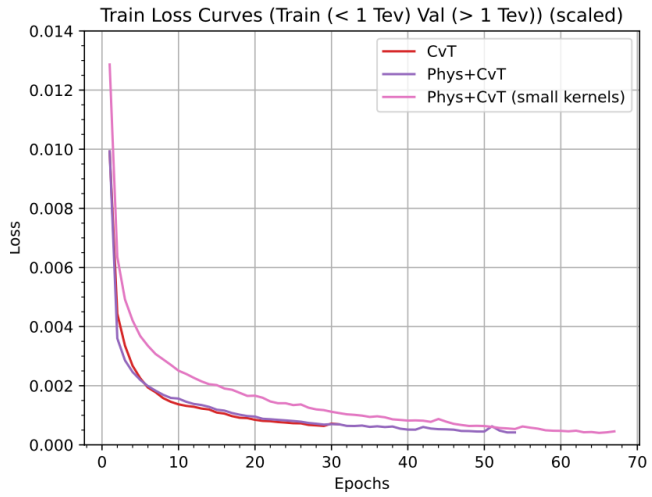



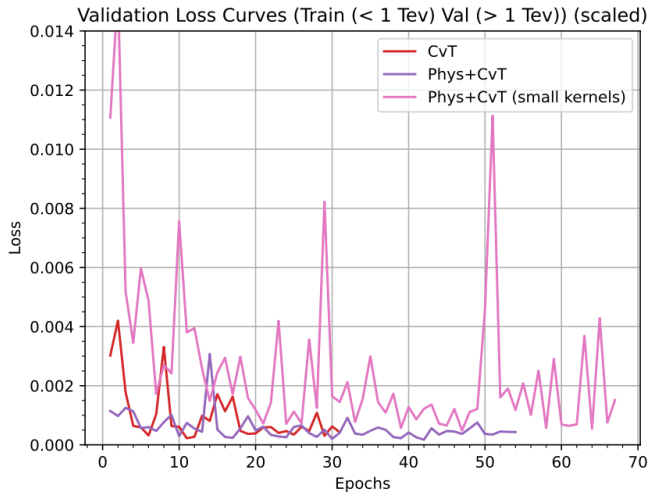



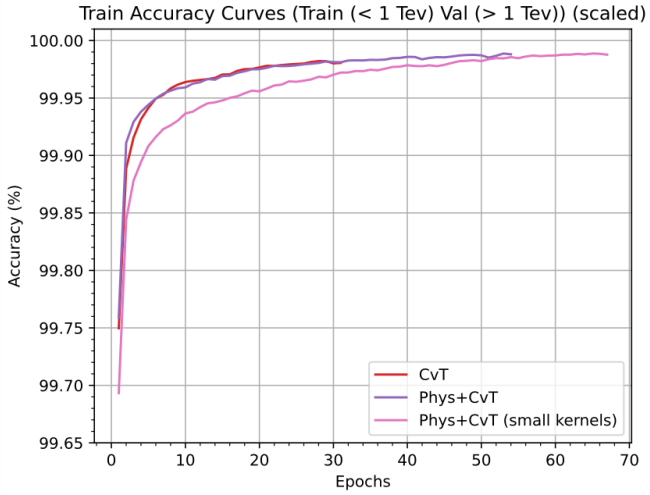



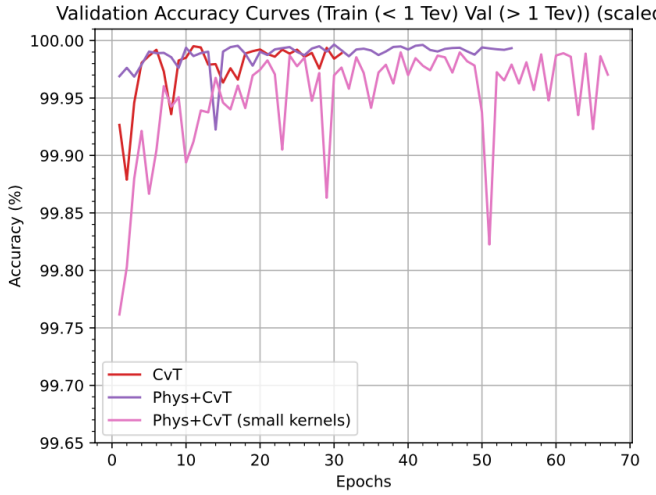
# Exp 5 – Comparison on 2nd Reconstructed Energy Dataset

- Plotted loss and accuracy curves for PhysCvTs ➔ CvT & Phys+CvT perform similarly.

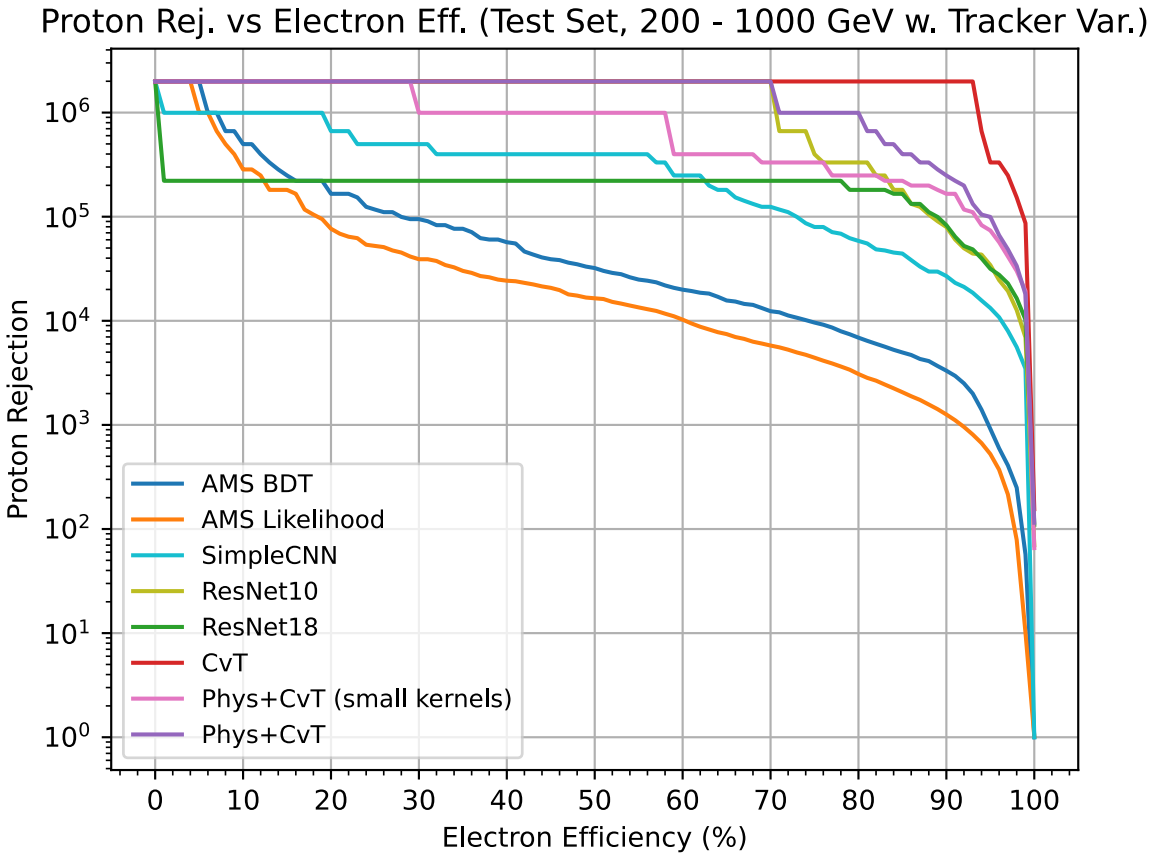


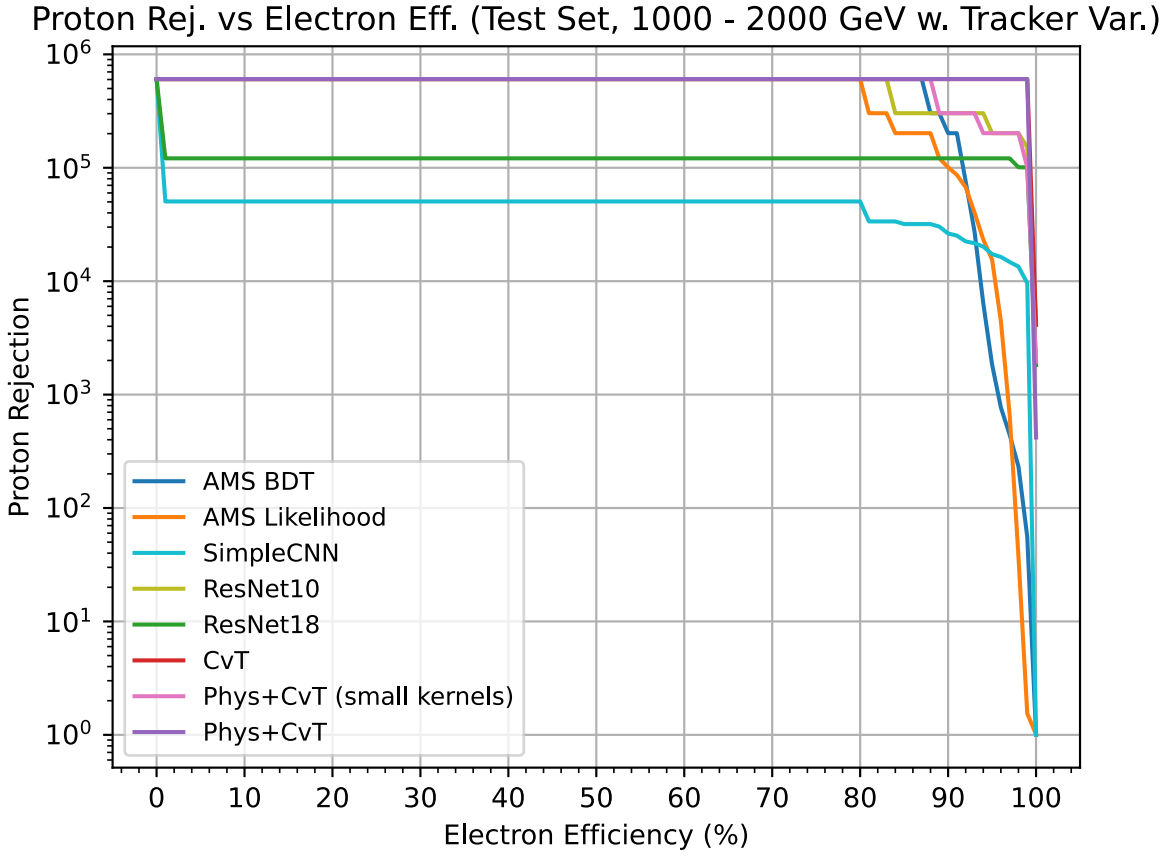



# Exp 5 – Proton Rej. vs. Electron Efficiency




At 90% electron efficiency, CvT outperforms the Phys+CvT, Phys+CvT (small kernels), ResNet18, ResNet10, SimpleCNN, AMS BDT, and AMS Likelihood models by factors of 8, 12, 24, 25, 74, 600, and 1581, respectively.

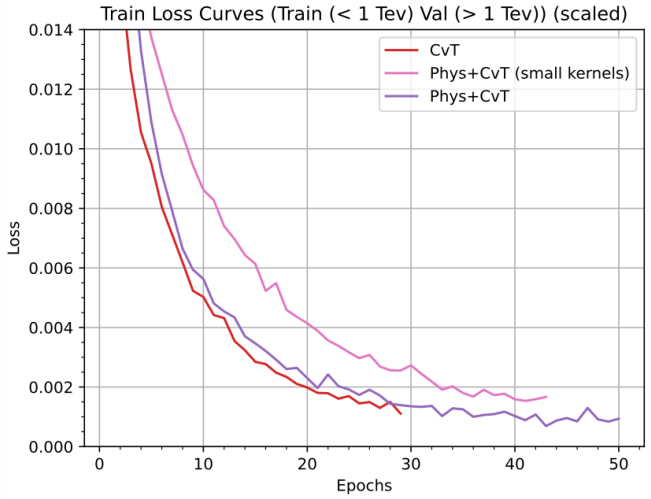
At 90% electron efficiency, CvT performs equally with the Phys+CvT, outperforms the ResNet10 and Phys+CvT (small kernels), which perform equally, by a factor of 2 and outperforms the AMS BDT, RestNet18, AMS Likelihood, and SimpleCNN models by factors of 3, 5, 6, and 23, respectively.

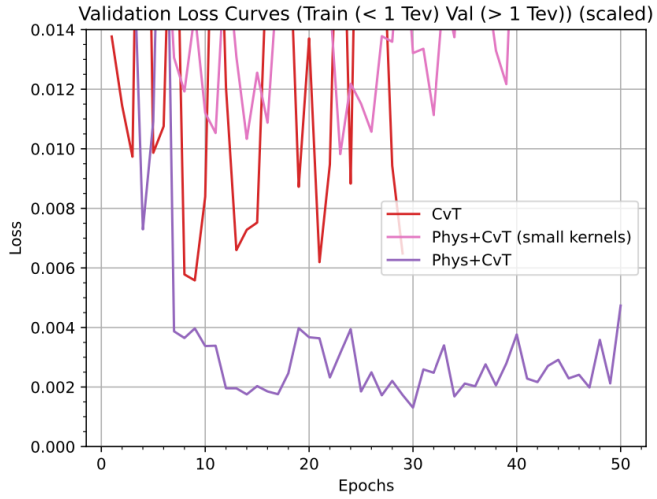
**Conclusion**: ResNet10 comparable to ResNet18. Phys+CvT does not improve over CvT.

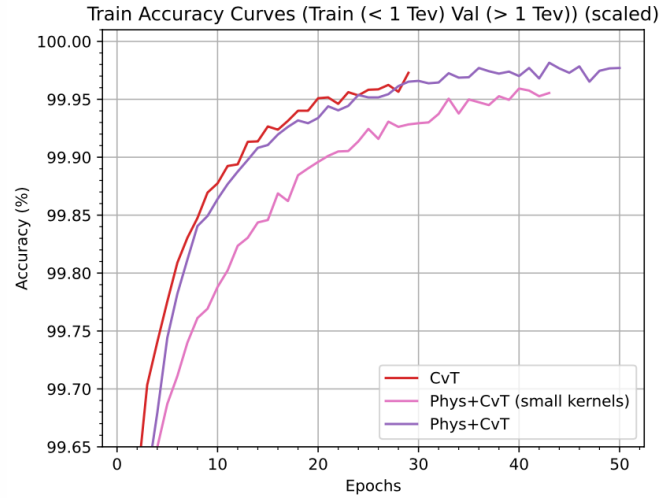
# Does Phys+CvT perform better on limited amounts of MC?

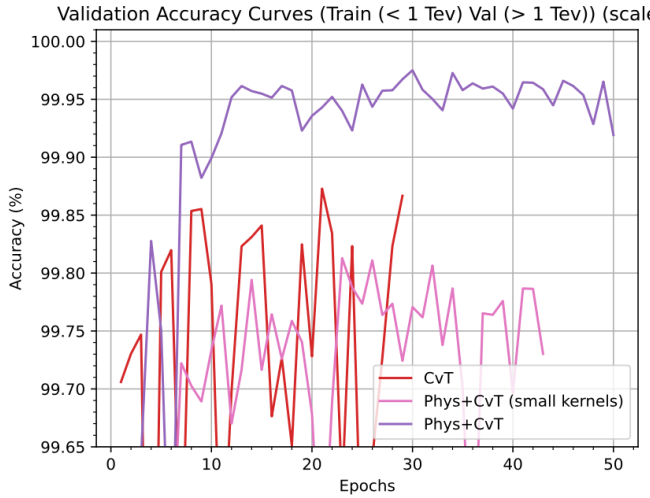
# Exp 6 – Phys+CvT Performance on a Smaller Training Set

- Plotted loss and accuracy curves after training on 0.29 million particles ➔ Phys+CvT performs the best on Val set.

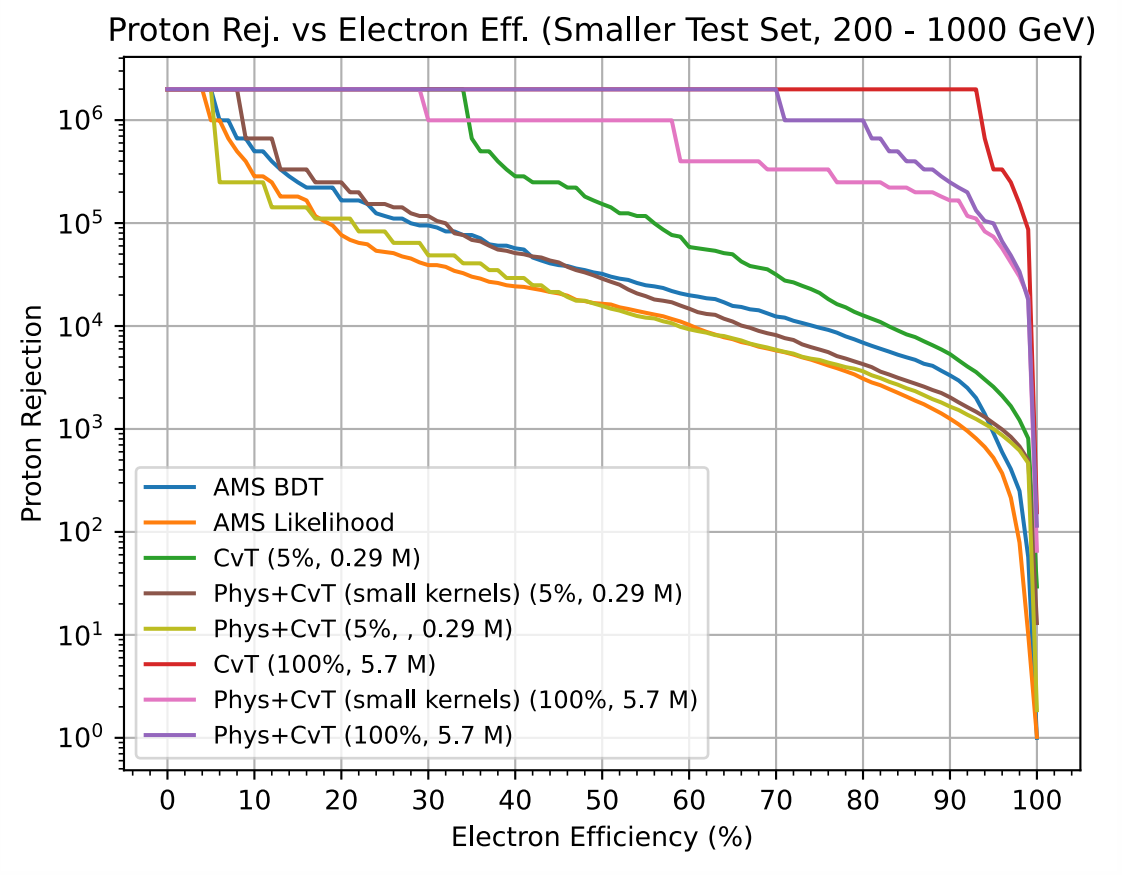


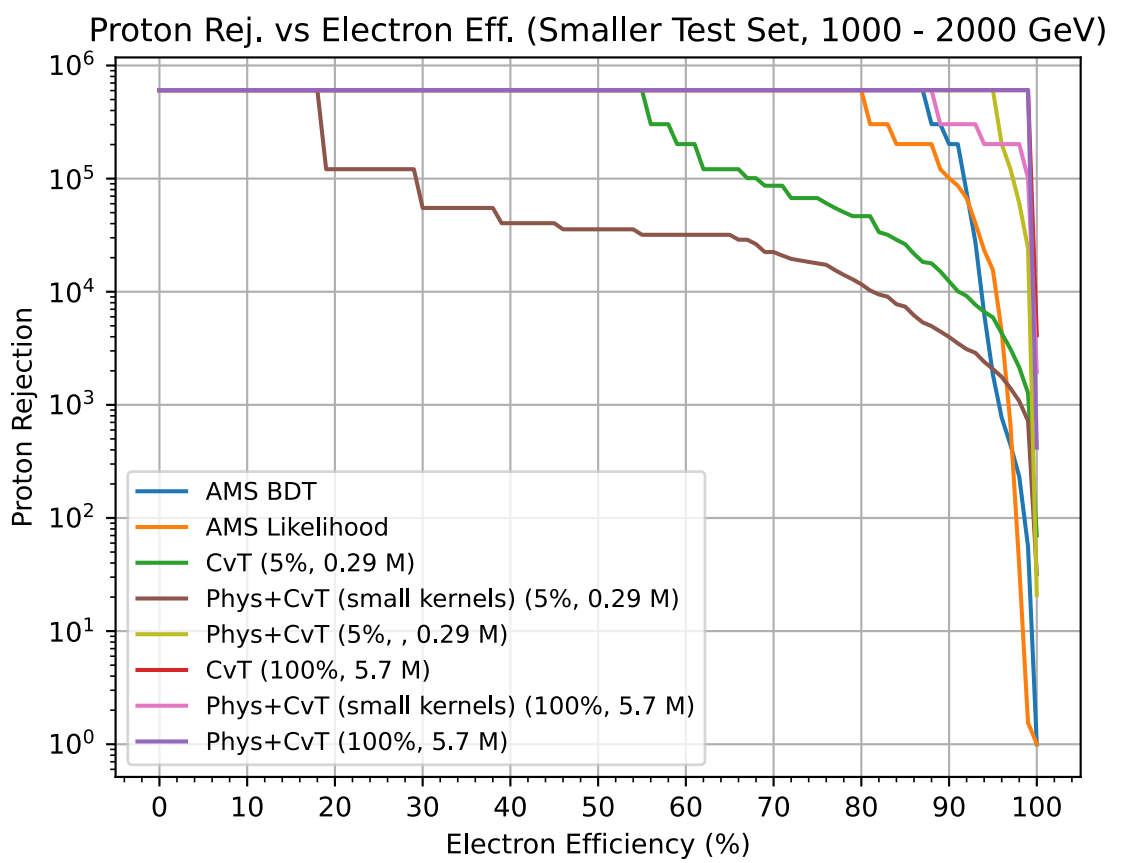



# Exp 6 – Proton Rej. vs. Electron Efficiency



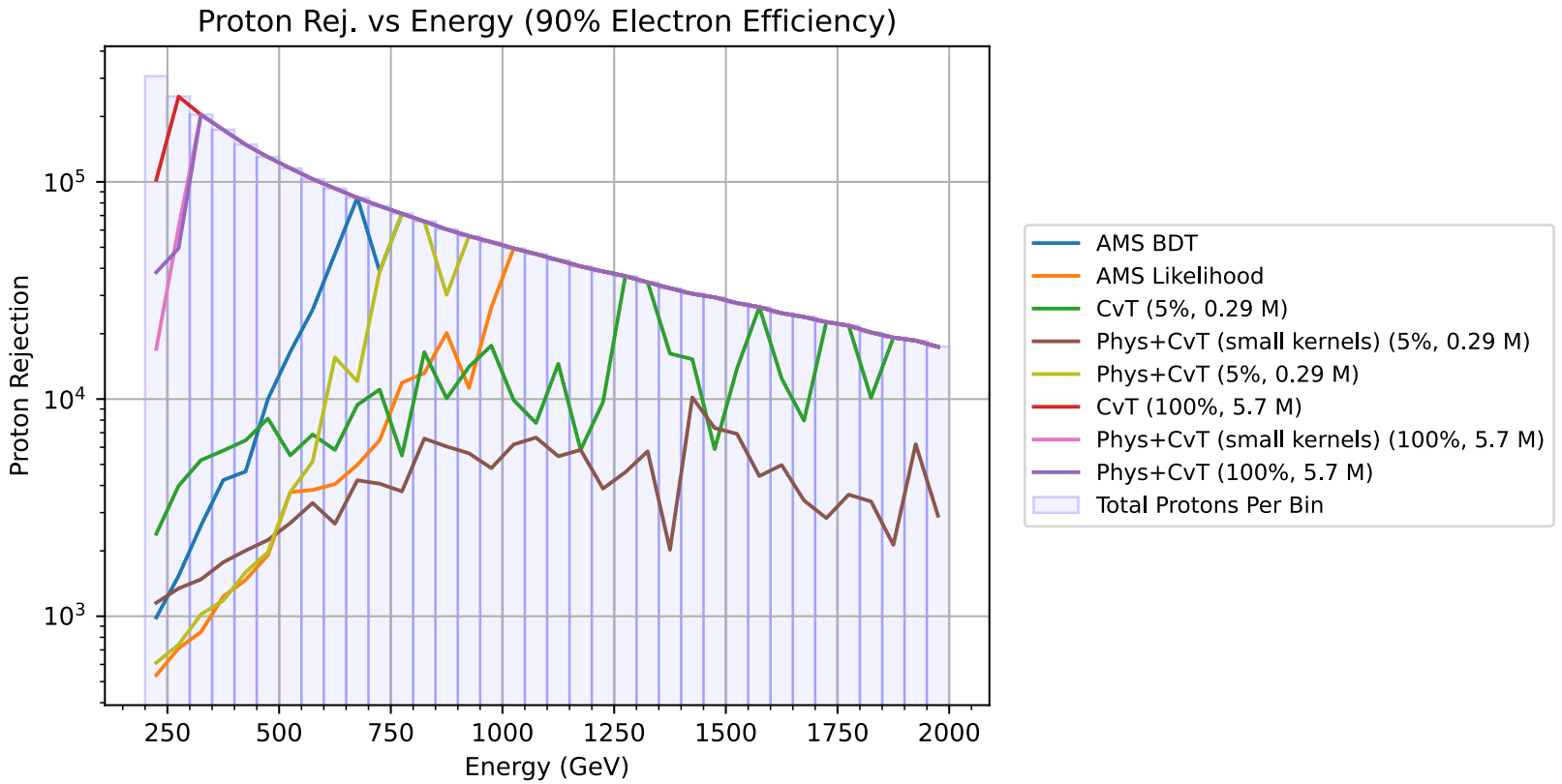

None of the models trained on 5% of the train set compare to the DL models trained on 100% of the data. Phys+CvT (5%) did worse than CvT (5%).




The Phys+CvT trained on 5% of the train set outperforms the CvT trained on the same amount of data and is close in performance to the DL models trained on 100% of the train set.

**Conclusion**: Phys+CvT did not improve efficiency on MC data.

# Exp 6 – Proton Rej. vs. Electron Efficiency




At lower energies, the Phys+CvT model trained on 0.29 M particles falls rapidly below 800 GeV.

**Conclusion**: Phys+CvT did not improve efficiency on MC data.

# Does Phys+CvT perform better on limited amounts of ISS data?

# ECAL – Datasets

We extract 4 datasets from AMS ROOT files to be used in Python using a script with defined cuts:

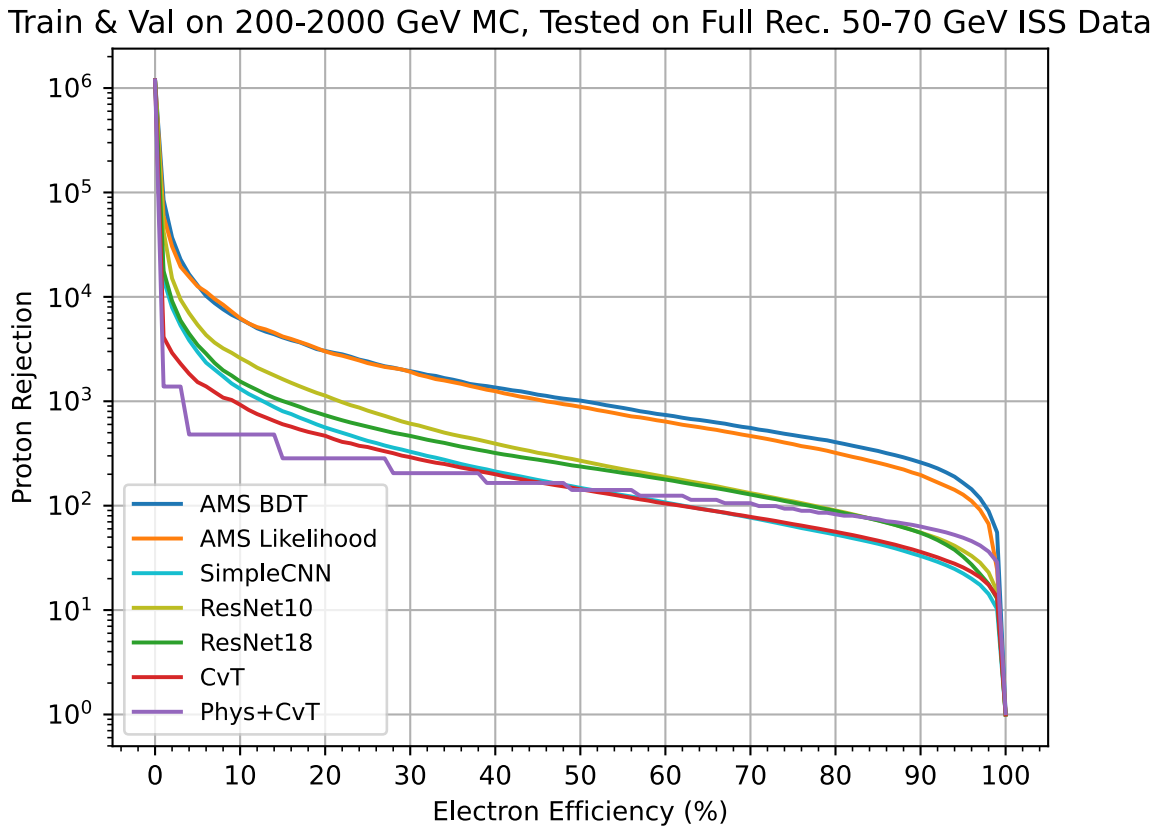
1. MC with a generated energy between 200 – 600 GeV.
2. MC with a reconstructed energy between 200 – 2000 GeV.
3. MC with a reconstructed energy between 200 – 2000 GeV + Additional variables from Tracker.
4. ISS data with a reconstructed energy between 50 – 70 GeV.
   - Used TRD as an independent method to get electrons and protons.
   - Energy range selected to get a pure (reliable labeling) and large dataset (large amounts of electrons and protons at this energy range in space).

Number of events (i.e. images) in each dataset.

| Source | Dataset | Below 1 TeV (in Millions) | | Above 1 TeV (in Millions) | |
|---|---|---|---|---|---|
| | | Electrons | Protons | Electrons | Protons |
| MC | 200-600 GeV, Generated | 4.60 | 0.16 | 0 | 0 |
| MC | 200-2000 GeV, Reconstructed | 7.03 | 3.90 | 2.69 | 1.19 |
| MC | 200-2000 GeV, Rec. + Tracker Variables | 7.51 | 3.98 | 2.89 | 1.21 |
| ISS | 50-70 GeV, Reconstructed | 0.03 | 1.19 | 0 | 0 |

# Exp 7 – Performance on ISS Data

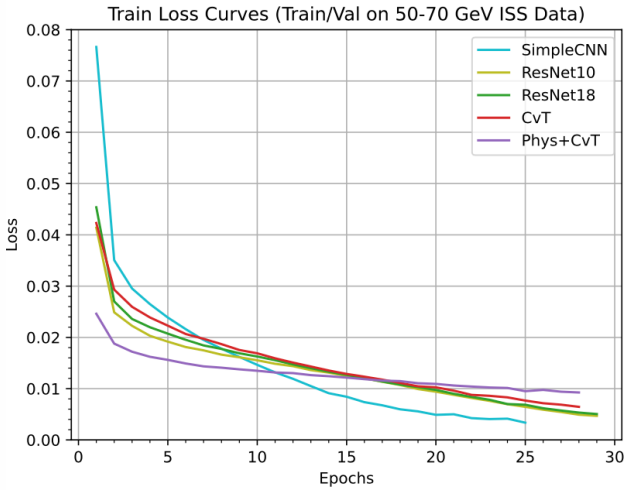
- Tested our MC trained and validated models on full ISS dataset.
- **Conclusion**: AMS models, which were trained on ISS data, perform better.

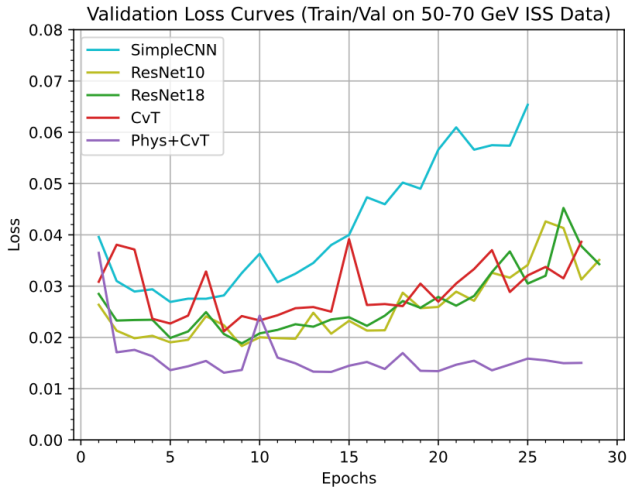


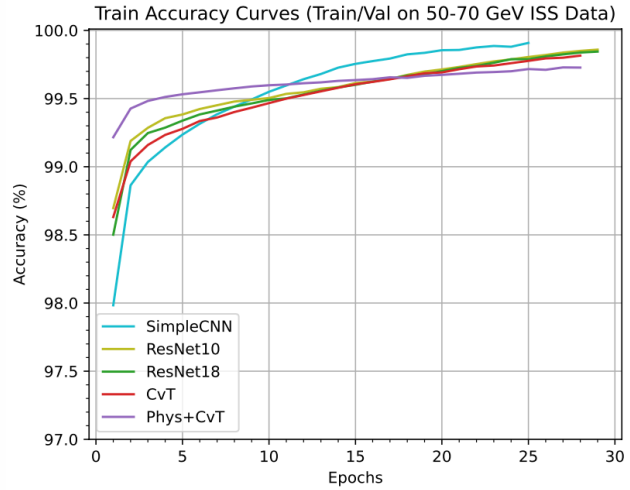

The AMS BDT outperforms the Phys+CvT, ResNet10, ResNet18, SimpleCNN, and CvT models by factors of 4.11, 4.69, 4.72, 7.87, and 9.21, respectively.

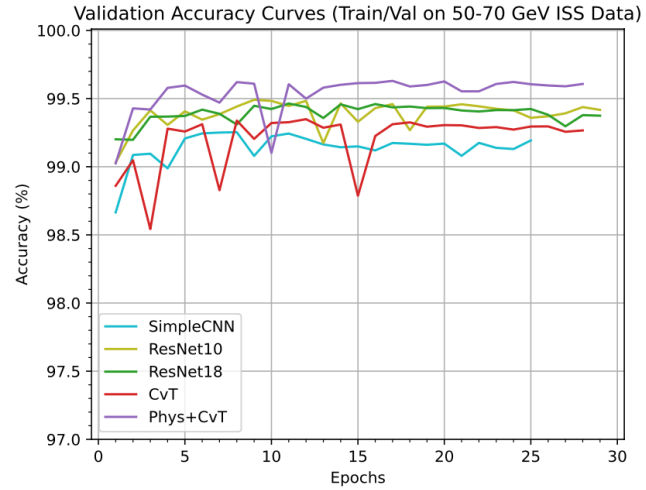
# Exp 7 – Train/Val/Test on ISS Data

- Plotted loss and accuracy curves (zoomed and scaled) ➔ CNN overfits the most, Phys+CvT performs the best on Val set.

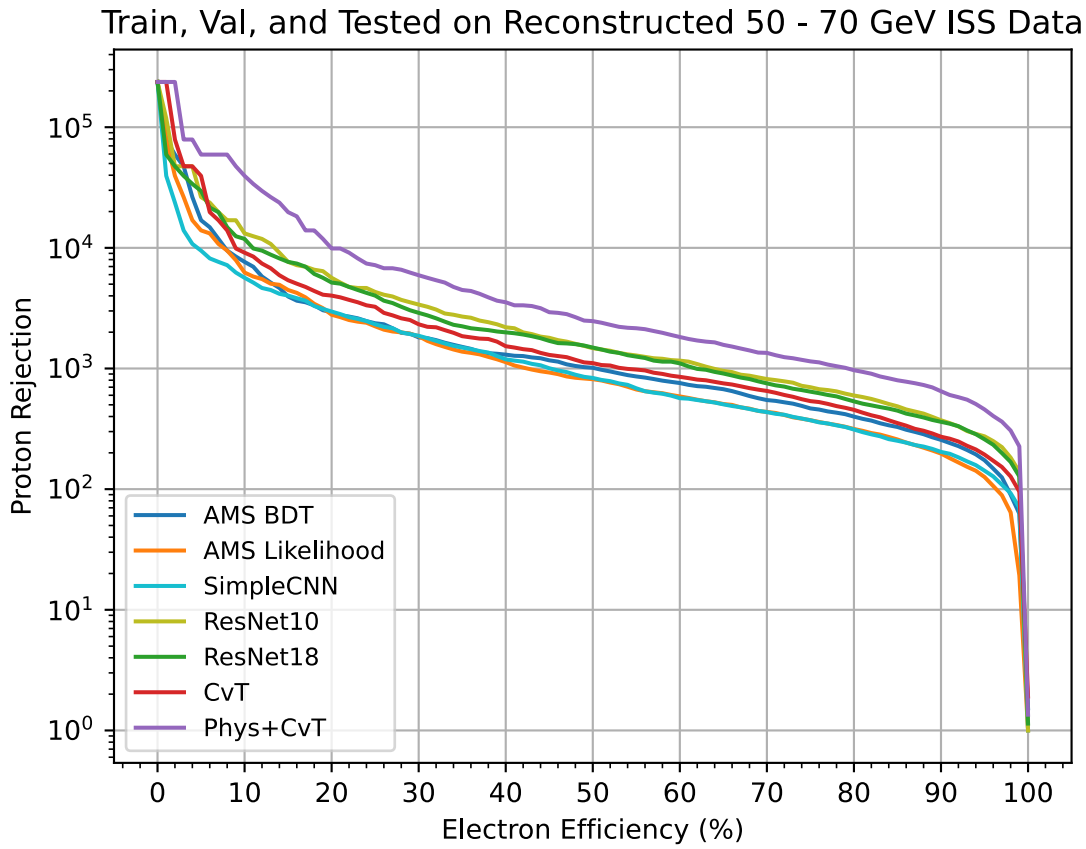


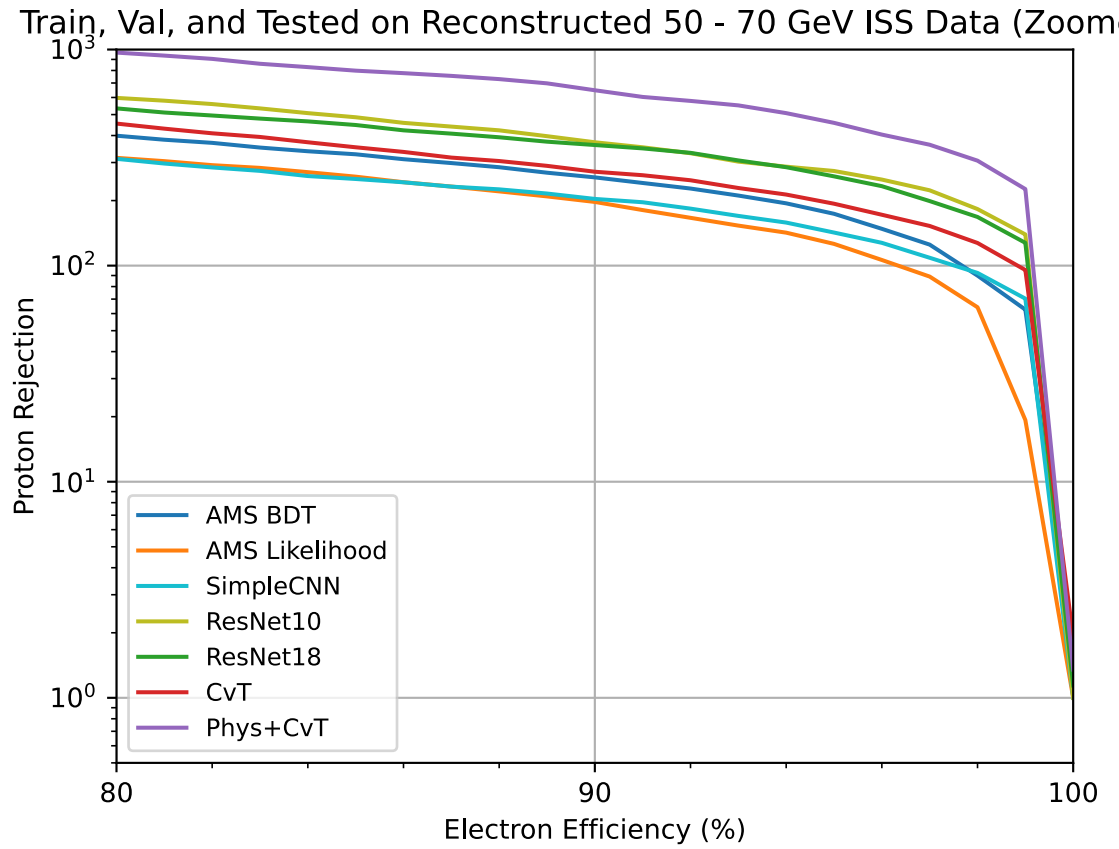



# Exp 7 – Train/Val/Test on ISS Data




The Phys+CvT model outperforms the ResNet10, ResNet18, CvT, AMS BDT, SimpleCNN, and AMS Likelihood models by factors of 1.74, 1.79, 2.38, 2.53, 3.18, and 3.39, respectively

**Conclusion**: Phys+CvT was successful in being more data efficient on ISS Data. All DL models (except SimpleCNN) outperform the current AMS models, albeit on this very, very small range of data.

# Summary



- We evaluated the potential of deep learning to separate electrons (and by extension positrons) from protons.
  - Used an MLP, CNN, two ResNets, a CvT, built a physics-based feature engineering, and evaluated it using two CvT variants.
- With the 1st Dataset (MC 200 – 600 GeV, generated):
  - We showed DL models outperformed Simple ML models.
- With the 2nd Dataset (MC 200 – 2000 GeV, reconstructed):
  - We showed the CvT has an excellent performance on the 0.2 – 2 TeV (while only being trained on 0.2 – 1 TeV)
  - Demonstrated CvT's need for large amounts of training data.

# Summary



- With the 3rd Dataset (MC 200 – 2000 GeV, reconstructed + Tracker variables):
  - We developed a feature engineering, though it was not successful at making the CvT more data efficient on MC data.
  - We also showed the ResNet10 had comparable performance to the ResNet18, but was not comparable to CvT models.
- Finally, with the 4th Dataset (ISS 50 – 70 GeV, reconstructed):
  - We showed further evidence of the discrepancies between MC and ISS data by noting the poor performance of MC-trained models against ISS data-trained models.
  - Outperformed AMS models after training DL models on a small sample of ISS data.
  - For this small range of ISS data, the feature engineering improved learning performance for the CvT.

In conclusion.

We have provided empirical evidence of newer architectures, such as the Convolutional vision Transformer, being a viable alternative to the commonly used BDTs and CNNs and provided evidence that they show promise for future use in the AMS experiment.

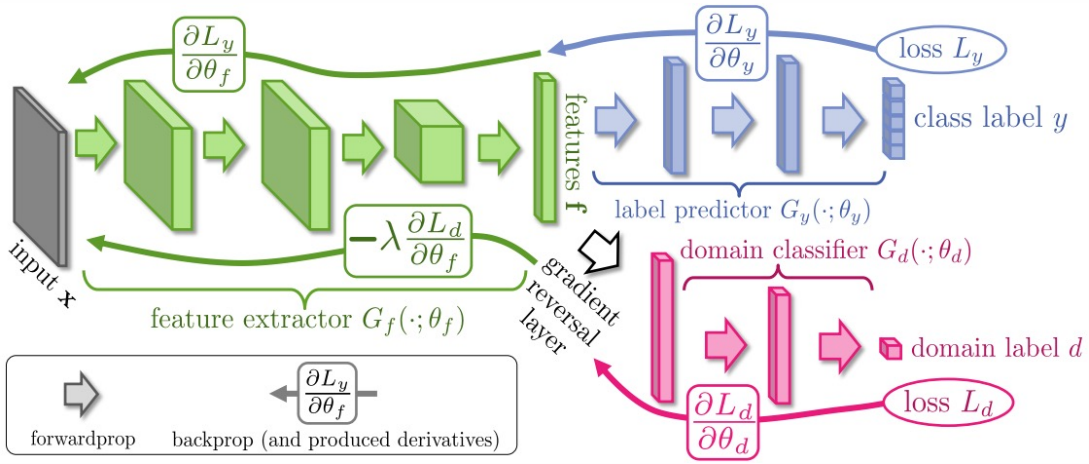
# Future Direction

- Use 2 Additional Datasets: Test Beam and MC of Test Beam.
- Train on Test Beam, Test on MC.
- Train on MC, Test on Test Beam.
- Cross Reference and See Performance.

# Future Direction



- Further hyperparameter optimization.
- Tackling domain shift (from below 1 TeV to above 1 TeV and potentially MC to ISS) directly using an unsupervised domain adaption technique [15]




The loss on the domain is subtracted from the loss on the class label, and therefore, minimizing the total loss results in a balance between reducing the class loss but keeping the domain loss up to get an overall minimal value. This allows the model to improve class label classification while learning domain-invariant features that increase the loss on the domain label

[15] Y. Ganin and V. Lempitsky, 32nd International Conference on Machine Learning, ICML 2015, vol. 2, pp. 1180–1189, Sep. 2014, doi: 10.48550/arxiv.1409.7495

# Future Direction

After extracting and evaluating our models on a larger range of ISS data (which requires more steps), we hope to present our work (again) to IML and the AMS collaboration for potential use in their future physics analyses.

# Acknowledgements

TÜBİTAK

TENMAK
TÜRKİYE ENERJİ, NÜKLEER VE MADEN ARAŞTIRMA KURUMU


Alpha Magnetic Spectrometer
AMS-02
Europe • Asia
North America

Thank You Very Much

# Questions & Answers