Outline

Outline of talk:
• In the beginning: **b-jet tagging in ALEPH**
• Going large scale: **Electrons and photons in ATLAS**
  – Data samples, variables, and selections
  – Electron PID and Energy Reconstruction (ER)
  – Discussion of performance measures (loss functions)
• Looking at the future: **ν-reconstruction in IceCube**

Purpose of talk:
• Show the possibilities with Machine Learning
• Discuss conceptual approach and performance measures
• Open up for possible inspiration/collaboration

In the following, all numbers and plots are “Not Even Preliminary”, and should in not be used elsewhere.
In the beginning: b-jet tagging in ALEPH
25 years ago, particle physics was actually at the forefront of Machine Learning. We had large computers and much data fit for ML usage.

At the time, LEP was searching for the Higgs boson at lower masses, where its decay was almost always to b-quarks.

For this reason, many resources were used to get the best possible b-jet tagging in place.
25 years ago, particle physics was actually at the forefront of Machine Learning. We had large computers and much data fit for ML usage.

At the time, LEP was searching for the Higgs boson at lower masses, where its decay was almost always to b-quarks.

For this reason, many resources were used to get the best possible b-jet tagging in place.

Both lifetime (displaced vertices), jet shape, and lepton pT was used, but none of these by themselves provide a good way to select b-jets.
However, using one of the very first ML algorithms (JetNet 3.4), six variables were put together in a neural network with two hidden layers each with 10 neurons:

- Light quark (uds) jet probability from track impact parameter significance.
- Difference in Chi2 from search for secondary vertices in jet.
- Transverse momentum of (possible) electron/muon in jet.
- Boosted sphericity of jet.
- Energy flow multiplicity (scaled by jet energy).
- Sum of transverse momenta (with respect to the jet axis) squared.

The neural network was trained on 400,000 simulated events, and though I haven’t been able to find the exact time used for this training, colleagues have told me “many hours, sometime days”.

Interestingly, my students now code the setup in about an hour, and get results in minutes.
ALEPH b-jet tagging

The result of these labours was a very nice b-jet tagging variable, which allowed ALEPH to get the most out of their data.
ALEPH b-jet tagging

A more “modern” plot could look like this:

Btags of signal and different types of backgrounds

Count

$10^2$ $10^3$ $10^4$ $10^5$ $10^6$

Btag

0.0 0.2 0.4 0.6 0.8 1.0

- gluons
- b quarks
- c quarks
- l quarks
- Signal
Going large scale: e/γ PID & ER in ATLAS
Overview

Currently in ATLAS, electrons and photons are identified using a likelihood approach:

\[ d_L = \frac{L_S}{L_S + L_B}, \quad L_S(x) = \prod_{i=1}^{n} P_{s,i}(x_i) \]

The likelihood is composed of 22 variables, for which 1D histograms are used to compute the likelihood value.

To minimise correlations, the likelihood is divided into regions of $E_T$ and $\eta$.

This makes for a very transparent approach, which at the same time performs well.

The question is, if there is more information to be gained, and thus a more powerful PID to be gotten.

Enter Machine Learning (ML)…
Electron PID - on MC
Electron & Photon samples

- \( Z \rightarrow ee \) (Zee)
  
  \[
  \text{mc16}_\text{13TeV.361106.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{Zee.merge.AOD.e3601_s3126_r10201_r10210}
  \]

- \( Z / \gamma^* \rightarrow ee \) (DYee)
  
  \[
  \text{mc16}_\text{13TeV.301000.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{DYee_120M180.merge.AOD.e3649_s3126_r10201_r10210}
  \]

- \( Z \rightarrow ee\gamma \) (eegamma)
  
  \[
  \text{mc16}_\text{13TeV.301535.Sherpa_C10_eegammaPt10_35.merge.AOD.e3952_s3126_r10201_r10210}
  \]

- \( Z \rightarrow \mu\mu\gamma \) (mumugamma)
  
  \[
  \text{mc16}_\text{13TeV.301536.Sherpa_C10_mumugammaPt10_35.merge.AOD.e3952_s3126_r10201_r10210}
  \]

- \( Z \rightarrow \mu\mu \) (Zmumu)
  
  \[
  \text{mc16}_\text{13TeV.361107.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{Zmumu.merge.AOD.e3601_s3126_r10201_r10210}
  \]

- \( W^\pm \rightarrow e^\pm \nu \) (Wenu)
  
  \[
  \text{mc16}_\text{13TeV.361103.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{Wminusenu.merge.AOD.e3601_s3126_r10201_r10210}
  \text{mc16}_\text{13TeV.361100.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{Wplusenu.merge.AOD.e3601_s3126_r10201_r10210}
  \]

- \( W^\pm \rightarrow \mu^\pm \nu \) (Wmunu)
  
  \[
  \text{mc16}_\text{13TeV.361104.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{Wminusmunu.merge.AOD.e3601_s3126_r10201_r10210}
  \text{mc16}_\text{13TeV.361101.PowhegPythia8EvtGen_AZNLOCTEQ6L1}_\text{Wplusmunu.merge.AOD.e3601_s3126_r10201_r10210}
  \]

- Dijet (JF)
  
  \[
  \text{mc16}_\text{13TeV.423300.Pythia8EvtGen_A14NNPDF23LO_perf_JF17.merge.AOD.e3848_s3126_r10201_r10210}
  \]

- \( \gamma + \text{jet} \) (gammajet)
  
  \[
  \text{mc16}_\text{13TeV.423099.Pythia8EvtGen_A14NNPDF23LO_gammajet_DP8_17.merge.AOD.e4453_s3126_r10201_r10210}
  \]
Zee candidates are selected with Tag&Probe (T&P).

**Purity: 45-95%**
Signal/background selection

- DAOD production: mixture of EGAM1, EGAM3, EGAM7, EGAM8 and EGAM9 including cells and bunch crossing information
- $Z \rightarrow ee$ Tag and Probe
  - Tag
    - $p_T > 24.5$ GeV
    - Tight ID
    - Loose Isolation
    - Crack veto and Central
    - Pass HLT_e26_lhtight_nod0_ivarloose
    - Has track particle and vertex
    - Pass object quality cut
  - Probe
    - $p_T > 4.5$ GeV
    - Pass object quality cut
    - $|\eta| < 4.9$
  - Combined
    - $dR > 0.4$ for tag and probe
    - $M_{ee} > 50$ GeV
- Dijet, $W^{\pm} \rightarrow \ell^\pm \nu$ ($\ell = e, \mu$), $Z \rightarrow \mu\mu$ samples background selection
  - Missing transverse energy (MET) < 25 GeV
  - $p_T > 4.5$ GeV
  - pass Object Quality
  - Z veto: Match with any other medium electron, $|M_{ee} - m_{Z^0}| < 20$ GeV
  - W veto: Match with MET, transverse mass < 40 GeV
- $Z(\gamma) \rightarrow ee$ (Drell-Yan), $Z \rightarrow \ell\ell\gamma, \gamma +$ jet
  - EGamma truth particles
- More complete documentation https://twiki.cern.ch/twiki/bin/viewauth/AtlasProtected/EgammaMachineLearning
The MC signal and background samples have very different distributions in energy (and also a bit in eta and $<\mu>$).

For this reason, we reweigh the samples…
Reweighing
Reweighing

The background is reweighed to look like signal in $E_T$, $\eta$ and $\langle \mu \rangle$ using GBReweighter (https://arogozhnikov.github.io/hep_ml/reweight.html)
Electron PID performance

The electron PID performance is generally much improved with ML:

We train the Machine Learning (ML) algorithm (LightGBM) with a mix of backgrounds, and then see how well it performs on each. We compare to the current ATLAS LH, not to boast our results, but as a solid reference, which helps us getting the most performant & general results.
Electron PID performance

The electron PID performance is generally much improved with ML:

The ML performance clearly improves with number of variables. From the 18 (LLH) variables to 26 and 29 variables, performance increases a lot… after that it only grows very slowly.

Q: Should we aim at 26-29 variables?
Where do we improve (most)?

The improvements are NOT uniform in energy and angle. We gain most in the “crack” and forward direction.

<table>
<thead>
<tr>
<th></th>
<th>10.0 - 0.66</th>
<th>10.6 - 0.88</th>
<th>0.8 - 1.15</th>
<th>1.15 - 1.37</th>
<th>1.37 - 1.52</th>
<th>1.52 - 1.81</th>
<th>1.81 - 2.01</th>
<th>2.01 - 2.37</th>
<th>2.37 - 2.47</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>$E_T$ [GeV]</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.0 - 20</td>
<td>1.979</td>
<td>3.014</td>
<td>2.453</td>
<td>3.91</td>
<td>9.407</td>
<td>7.43</td>
<td>6.122</td>
<td>5.483</td>
<td></td>
</tr>
<tr>
<td>20.0 - 30</td>
<td>1.617</td>
<td>1.698</td>
<td>2.07</td>
<td>4.191</td>
<td>6.053</td>
<td>3.057</td>
<td>6.934</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30.0 - 200</td>
<td>1.411</td>
<td>1.995</td>
<td>4.322</td>
<td>2.424</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Hadron acceptance LHValue / LGBM at 98% electron efficiency**
Electron PID feature importance

The importance of each PID input variable is shown below (https://github.com/slundberg/shap).
Using Angular Variables to disentangle $H \rightarrow ZZ^*$ → $eeee$?

The importance of each PID input variable is shown below (https://github.com/slundberg/shap).

The ML approach can easily incorporate more input variables, also those which describe environment more than PID in itself (e.g. energy, direction, pile-up, etc.).
Impact in data
ML electron PID on probe side yields **in data** more Zee events (same background):

While the gain is modest (4.5%), it is doubled, when also applied to the tag side. Better energy reconstruction can also contribute…
Electron Energy Reconstruction - on MC
Electron ER - BDT vs. CNN

We started to work on electron energy reconstruction (ER) using scalar variables combined with a BDT approach, just like ATLAS does. However, we are now exploring to use a Convoluted Neural Network (CNN) for the task, as this “naturally” fits the problem, when considering the calorimeter cells as images. Naturally, there are still scalar variables to add to the regression:

### BDT scalar variables
- p_eAccCluster
- p_f0Cluster
- p_R12
- p_etaCluster
- p_cellIndexCluster
- p_etaModCalo
- p_phiModCalo
- p_fTG3
- p_dPhiTH3
- p_pt_track
- averageInteractionsPerCrossing
- NvtxReco

### CNN scalar variables
- p_eta
- p_deltaPhiRescaled2
- pX_deltaPhiFromLastMeasurement
- pX_deltaPhiRescaled0
- pX_deltaEta2
- pX_deltaEta3
- p_charge
- BC_distanceFromFront
- BC_filledBunches
- p_pt_track
- averageInteractionsPerCrossing
- NvtxReco

- **Blue** are used by the current $E$ calib
- **No ECAL variables**
The photon energy reconstruction performance is shown here (for $Z \rightarrow ee\gamma$ sample):
Photon ER performance

The photon energy reconstruction performance is shown here (for $Z \rightarrow ee \gamma$ sample):

<table>
<thead>
<tr>
<th>Key</th>
<th>MAE(Z)</th>
<th>MSE(Z)</th>
<th>ICE(5)</th>
<th>ICE(25)</th>
<th>Lower5</th>
<th>Upper5</th>
<th>Median</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS</td>
<td>0.0501</td>
<td>0.0075</td>
<td>0.0726</td>
<td>0.0384</td>
<td>-0.117</td>
<td>0.122</td>
<td>0.0019</td>
<td>0.0021</td>
</tr>
<tr>
<td>LGBM126</td>
<td>0.0410</td>
<td>0.0071</td>
<td>0.0560</td>
<td>0.0320</td>
<td>-0.093</td>
<td>0.091</td>
<td>-0.0000</td>
<td>0.0012</td>
</tr>
<tr>
<td>LGBM100</td>
<td>0.0420</td>
<td>0.0077</td>
<td>0.0571</td>
<td>0.0325</td>
<td>-0.095</td>
<td>0.093</td>
<td>0.0000</td>
<td>0.0014</td>
</tr>
<tr>
<td>LGBM50</td>
<td>0.0421</td>
<td>0.0073</td>
<td>0.0581</td>
<td>0.0326</td>
<td>-0.096</td>
<td>0.095</td>
<td>-0.0001</td>
<td>0.0011</td>
</tr>
<tr>
<td>LGBM35</td>
<td>0.0428</td>
<td>0.0069</td>
<td>0.0595</td>
<td>0.0332</td>
<td>-0.099</td>
<td>0.097</td>
<td>-0.0000</td>
<td>0.0011</td>
</tr>
<tr>
<td>LGBM15</td>
<td>0.0506</td>
<td>0.0095</td>
<td>0.0715</td>
<td>0.0393</td>
<td>-0.117</td>
<td>0.118</td>
<td>-0.0000</td>
<td>0.0021</td>
</tr>
</tbody>
</table>
What is a CNN?

CNNs are a type of neural network, which works well with spatially dependent data (typically images). CNNs use parameter/weight sharing.

Multi-layered images (e.g. RGB or ATLAS calorimeter) are handled naturally.

A CNN works by sliding (small) filter across the image, outputting the convolution (inner product) of the filter and pixels covered.
What is a CNN?

CNNs are a type of neural network, which works well with spatially dependent data (typically images). CNNs use parameter/weight sharing.

Multi-layered images (e.g. RGB or ATLAS calorimeter) are handled naturally.

A CNN works by sliding (small) filter across the image, outputting the convolution (inner product) of the filter and pixels covered.
The great thing is that for each cell we don’t just have the energy, but also the time (rejecting out-of-time pile-up), gain, and cell noise level (gauging the energy precision).
However, these are not same units, so combined with gate (not concatenation).
**CNN architecture**

We use a $3 \times 3$ convolution matrices for all layers.

Each convolution layer is followed by a batch normalisation and activation.

For all $i > 1$, block begins with downsampling and the number of feature maps is doubled.

A worry is, that the scalar variables “drown” in the many feature map outputs. To be investigated further. However, we know that scalar variables improve performance as it is!

Images containing time are treated differently…

Frederik Faye
Using Angular Variables
to disentangle

\[ H \rightarrow \mathcal{Z} \rightarrow \mathcal{E} \]

CNN results

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>IQR(Z)</th>
<th>rMAE</th>
<th>rIQR(Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E calib.</td>
<td>1.753</td>
<td>0.041</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>LGBM(9)</td>
<td>1.726</td>
<td>0.040</td>
<td>1.016</td>
<td>1.014</td>
</tr>
<tr>
<td>LGBM(12)</td>
<td>1.685</td>
<td>0.039</td>
<td>1.040</td>
<td>1.047</td>
</tr>
<tr>
<td>CNN</td>
<td>1.562</td>
<td>0.037</td>
<td>1.122</td>
<td>1.100</td>
</tr>
<tr>
<td>CNN(s)</td>
<td>1.548</td>
<td>0.036</td>
<td>1.132</td>
<td>1.124</td>
</tr>
<tr>
<td>CNN(s,t)</td>
<td>1.533</td>
<td>0.036</td>
<td>1.144</td>
<td>1.138</td>
</tr>
</tbody>
</table>

\[ Z = \frac{(E_{\text{pred}} - E_{\text{truth}})}{E_{\text{truth}}} \]
CNN results

Frederik Faye

<table>
<thead>
<tr>
<th>Model</th>
<th>MAE</th>
<th>IQR(Z)</th>
<th>rMAE</th>
<th>rIQR(Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E calib.</td>
<td>1.753</td>
<td>0.041</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>LGBM(12)</td>
<td>1.685</td>
<td>0.039</td>
<td>1.040</td>
<td>1.047</td>
</tr>
<tr>
<td>CNN(s)</td>
<td>1.548</td>
<td>0.036</td>
<td>1.132</td>
<td>1.124</td>
</tr>
</tbody>
</table>

\[
Z = \frac{E_{\text{pred}} - E_{\text{truth}}}{E_{\text{truth}}}
\]
Using Angular Variables to disentangle $H \rightarrow ZZ^* \rightarrow eee e$.
Using Angular Variables
to disentangle

CNN results

Frederik Faye
Status of efforts - DATA
Applying ML PID trained on MC to data naturally gives lesser results.

Also, the shown improvement is a lower bound, as signal in the background lowers (apparent) performance.

Stefan Hasselgren
Master thesis finished (link below)
(defend end of December 2018)

Performance is best in the forward region at high energies. However, the later statement might be a result of determining performance with impure data (more so at lower energies).
Looking at the future: ν-reconstruction in IceCube
Ideas for the future

The IceCube detector is a less “classic” particle physics detector. Here, 86 strings with about 5000 Digital Optical Modules (DOMs) in total are put in the ice at the South Pole, and used to detector neutrinos (and involuntarily cosmic muons) interact in the ice.

The detector is triggered by coincidences of several adjacent DOMs, and then read out.

Each DOM provides a measurement in time and size of signal. However, there is a significant amount of noise and also effects such as after-pulses, which makes the data less clean.
Ideas for the future

The IceCube detector is a less “classic” particle physics detector. Here, 86 strings with about 5000 Digital Optical Modules (DOMs) in total are put in the ice at the South Pole, and used to detector neutrinos (and involuntarily cosmic muons) interact in the ice.

The detector is triggered by coincidences of several adjacent DOMs, and then read out.

Each DOM provides a measurement in time and size of signal. However, there is a significant amount of noise and also effects such as after-pulses, which makes the data less clean.

The **bottleneck** is the event reconstruction!

This is based on the minimisation of a likelihood including ice properties.
Neutrinos and cosmic muons interact in the ice, and leaves signals to be reconstructed.
Ideas for the future

Neutrinos and cosmic muons interact in the ice, and leaves signals to be reconstructed.

Problem 1:
Which hits belong to the event and which are noise?
Ideas for the future

Neutrinos and cosmic muons interact in the ice, and leaves signals to be reconstructed.

Problem 1: Which hits belong to the event and which are noise?

Problem 2: Given a list of hits, how to determine the direction, energy, type, etc.? And... how to do it in a "reasonable" amount of time? Currently $t(\text{reco}) = 30$ min.
Neutrinos and cosmic muons interact in the ice, and leaves signals to be reconstructed.

Problem 1: Which hits belong to the event and which are noise?

Problem 2: Given a list of hits, how to determine the direction, energy, type, etc.? And... how to do it in a "reasonable" amount of time?

Currently t(reco) = 30 min.

A student of mine (Andreas Søgaard) tried to see, if he could get an ML algorithm to do the reconstruction. It didn’t perform very well (yet!), but t(reco) = 0.01 sec.
Conclusions

I think that there is a lot of prospect in Machine Learning for physics.

• New algorithms see the light of day almost daily.
• In some cases, it may simply give a more performant data analysis.
• However, in some cases, it makes all the difference.

I’ve been surprised by the speed with which students “pick up” ML, once you give them an introduction to it. The challenge is often to find data “suitable” for the algorithms given.

However, these are getting more and more diverse. So if you have (possibly dirty, flawed, etc.) data from your experiment, I suggest that you try to hand it to students as a project.

Meanwhile, I’m preparing a course (at University of Copenhagen), which to some extend covers “Applied Machine Learning”. I was surprised by the turnout…
Bonus slides
Purities after T&P

The electron probe purities for both signal and background are far from ideal!

Signal sample purities

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 2.01-2.47$</td>
<td>58.8%</td>
<td>83.2%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 1.52-2.01$</td>
<td>52.5%</td>
<td>80.1%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 1.37-1.52$</td>
<td>45.7%</td>
<td>75.8%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 0.8-1.37$</td>
<td>47.9%</td>
<td>78.9%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 0.0-0.8$</td>
<td>47.5%</td>
<td>79.9%</td>
</tr>
</tbody>
</table>

Background sample purities

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 2.01-2.47$</td>
<td>70.0%</td>
<td>69.3%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 1.52-2.01$</td>
<td>75.2%</td>
<td>77.5%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 1.37-1.52$</td>
<td>66.1%</td>
<td>71.3%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 0.8-1.37$</td>
<td>74.4%</td>
<td>80.8%</td>
</tr>
<tr>
<td>$</td>
<td>\eta</td>
<td>: 0.0-0.8$</td>
<td>80.6%</td>
<td>83.0%</td>
</tr>
</tbody>
</table>
Comparison of Combination

Eff(bkg) @ 92% Eff(sig):
ATLAS Likelihood: 0.40%
ML(Calo)+ML(Trk): 0.12%
ML(Calo+Trk): 0.09%

Note that the ML(Calo+Trk) can not be trained on real data, as one quantity (ML(Calo) or ML(Trk)) is required in order to get a clean sample for the other to be trained on.
Performance of all methods

Eff(bkg) @ 92% Eff(sig):
- ATLAS Likelihood: 2.2%
- ML(Calo)+ML(Trk): 0.78%

Improvement by factor 2.8 (in DATA)
Tag & Probe

Zee candidates are selected with Tag&Probe (T&P).

Purity: 30-90%

Probe electron

Tag electron
The Idea: Extended Tag&Probe

1) Zee candidates are selected with Tag&Probe (T&P).
   Purity: 30-90%

2) The probe electron is “divided” into three independent (?) parts: Track, Calo, Isolation
The Idea: Extended Tag&Probe

1) Zee candidates are selected with Tag&Probe (T&P).
   Purity: 30-90%

2) The probe electron is “divided” into three independent (?) parts:
   Track, Calo, Isolation

3) When considering one part of the probe electron, the other two can be used to further purify probes:
   Purity: 99-99.9%