Deep Learning Methods for Particle Reconstruction in the HGCal

Ofir Arzi

The Hebrew University of Jerusalem
Supervisors: Clemens Lange, Predrag Milenovic, Jan Kieseler

December 7, 2017

1 Introduction

The High Granularity end-cap Calorimeter is part of the phase-2 CMS upgrade (see Figure 1)[1]. Its goal it to provide measurements of high resolution in time, space and energy. Given such measurements, the purpose of this work is to discuss the use of Deep Neural Networks for the task of particle and trajectory reconstruction, identification and energy estimation, during my participation in the CERN Summer Students Program.

![Figure 1: The HGCal future position is within the green triangles](image)

2 Neural networks

The problem that lies ahead of us is of classifying the content of a shower, meaning to determine which are the particles that compose it, and to decide the amount of energy each of the particles had, namely a regression problem. One good technique that proved great performance in those problems is neural networks, and since our problem may viewed as a computer vision problem (2, since the hits and the shower maybe plotted as an image, we are naturally drawn to use convolutional neural network (CNN), a key tool in the field.

A neural network is a computing algorithm that maybe viewed as as a directed graph3. It is compute of an input layer (denoted by red nodes in the figure), an output layer (green nodes) and sometimes one or more hidden layers (blue nodes). The input from each node is multiplied by a weight and then summed over all the inputs to a node, a non-linear activation function is then applied over the sum to give the nodes output.
An important and useful property in many problems of computer is spatial invariance, meaning that when looking at an image of a shower, the shower may be found at any location within the image, and not necessarily in the center or top right corner. CNN use this property by convolving a small filter with the image, the filter will output the likelihood to find critical features in the image that imply the existence and identity of a particle. By stacking several convolutional layers upon each other, the network can detect features of increasing complexity, which helps handle more difficult problem, or a larger variety of classes to identify.

3 Data preprocessing

The High Granularity Calorimeter (HGCal) is composed out of 52 layers (Figure 4), for the first 40 layers each layer is made of hexagonal wafers that are composed of even smaller hexagonal sensors, which define the spatial resolution of the calorimeter.
The basic unit of data in this work is a hit, a localized deposit of energy, meaning a reading from a sensor that in some coordinates \((x,y)\) a certain amount of energy \(E\) was found. An example is seen in Figure 5. The hexagonal structure and ordering of the sensors is clearly visible.

The data is read out from the sensors in the form of a list of hits and energy measurements. In order to utilize deep learning techniques from the field of computer vision, we need to represent our data as an image. The hexagonal layout of the sensors poses a problem to the NN, since it does not have the rectangular pixel structure of an image. Therefore we attempted several methods to handle this difficulty, as explained in the following subsections.

After transforming the data, it is required to convert the files from ROOT format to numpy format, so it can be fed to a keras\(^2\) (with tensorflow backend \(^3\)) model.

3.1 Pixel Shifting

Simply trying to impose a Cartesian will force to split readings between pixels that did not share them. Thus, before the data is fed to a NN, we apply a transformation on the hits’ coordinates in order to get new coordinates, relative to a seed, which is a particle based in truth information from a simulation. First we decide whether the hit lies on an even or an odd row, by looking at \(\text{round}(\frac{y-s_y}{dy}) \mod(2)\), where \(s_y\) is the seed’s \(y\) coordinate and \(dy\) is the pixel’s height. Then we set

\[
y' = y; \quad x' = \begin{cases} x, & \text{row is even} \\ x - dx, & \text{row is odd} \end{cases}
\]

where \(dx\) is the pixel’s width. After this, the hits are shifted by the distance from the seed. The transformed hits are seen in Figure 5 as the red crosses, with the desired Cartesian structure. The distances \(dx, dy\) were found by looking at common values for the distance between hits in each direction, and then taking the minimal non-zero values, but once the final HGCal architecture is determined those values will be parameters to be given to the algorithm.

![Figure 5: Hits from a photon simulation, before the shift in blue dots and after it in red crosses.](image)

The hexagon pixels size is not a constant, but depends on the distance from the center of the layer. Therefore \(dx, dy\) are not constant so the shifting will not be consistent and the structure of the jets will be disturbed. However, this work is important since it allows us to determine a characteristic size for a sensor.

3.2 Pixel sampling

In this method, a Cartesian structure is forced by overlaying a grid of rectangles, roughly the size of the hexagon sensors, upon the hexagons, See Fig. 6. An hexagon sensor \(h\) that contains an energy deposit \(E\), will contribute to an overlapping rectangle \(p\) an energy deposit \(\frac{\text{area of overlap}}{\text{area of h}} \cdot E\). This will be summed over all of the hexagon overlapped by the pixel. Geometric calculations were done using Clipper\(^4\).
4 Network Structure

With the Data in the correct format, a DNN is trained over data from simulated photons, where we chose to use hits at the 15th layer as a representative for the entire shower as it contains a significant number of interactions.

The network is fed with 3 inputs - a \(13 \times 13 \times 3\) matrix where each entry contains three channels with the hit's energy deposit in this location, time information and layer, the sum of energies in the window, and eta. The networks output is a prediction for the particle's energy, and a \((1,5)\) vector where each entry stands for the particles likelihood to belong to either class of particles (photon, electron, etc). The hidden layers are one convolutional layer on the image with a \((5,5)\) kernel and stride \((1,1)\), fololowed by a 5 dropout layer and then concatenation with the rest of the input followed by a 300 nodes fully-connected layer.

5 Analysis

The trained network was tested on a test data set that beside photons contained also electrons, pions and muons. The results are shown in the following figures. Figure 7 shows a histogram of response, \(\text{response} = \frac{\text{predicted energy}}{\text{true energy}}\) for the photons. For energies above 150\(\text{GeV}\) the plots is centered around 1, meaning the network has no tendency to overestimate or underestimate the energy, and for energies above 300\(\text{GeV}\) the plot shows most predicted energies are about 10\% below or above the true energy.

Figure 7: Photon response histogram

Figure 8 shows the class profile, meaning the response as a function of the true energy, for the four test classes. For most of the energy spectrum the photons response is around as expected, but quite remarkably it is also the case for the electron. Such a result is not too surprising as
the electron is a very small particle. The muons and pions show much worse results at they are much heavier, and the muon is also known to be very elusive. When the energy is low prediction becomes much harder because background noises become more significant, and so we observe that the errors in this region are very large.

![Graph of energy vs. response for different particle types](image)

Figure 8: The dependence on energy of the response for several test classes

Figure 9 presents the absolute error, predicted energy − true energy, as a function of the true energy for the four test classes. As before we observe the predictions for the photons and electrons are very good, while for the muons and pions are very bad. Since the error was not normalized against the true energy, the plot deviates more and more from 0 as the energies go higher, but even so error does not go above 10% or so. We conclude that the network is mostly accurate for high energies.

![Graph of absolute error vs. energy for different particle types](image)

Figure 9: The absolute error of the response as a function of energy, for various test classes

6 Conclusion

In this paper we discussed the use of neural networks for particle reconstruction and energy regression. We explained the steps to be taken in order for it to be possible to apply DNN techniques from computer vision to this problem, by implementing two kinds of data transformation - one is the pixel shifting method and the other being the pixel sampling method, which due to the structure of the sensors the later is more useful. The paper also discussed the experiments that
were carried out base on a single model of convolutional neural network, and the promising results this model had shown i.e. good energy predictions for energy for photons and the ability to successfully identify photons and electrons.

Finally, I would like to thank my supervisors, Clemens Lange, Predrag Milenovic and Jan kiseler for thiere help and Guidance. I am also very thankful to CERN for hosting me here and giving the opportunity to take part in this work, and to the Summer Students team who also worked very hard to let this happen.

References


