Deep Learning applied to hit classification for the BESIII drift chamber

Yao Zhang, Ye Yuan, Qiumei Ma
Institute of high energy physics, Beijing, China
zhangyao@ihep.ac.cn, yuany@ihep.ac.cn, maqm@ihep.ac.cn

Abstract. Drift chamber is the main tracking detector for high energy physics experiment BESIII. Due to the high luminosity and high beam intensity, the BESIII drift chamber is suffered from the beam background and electronics noise which represent a computing challenge to the reconstruction software. Deep learning developments in the last few years have shown tremendous improvements in the analysis of data especially for object classification. Here we present a first study of deep learning architectures applied to the real data of BESIII drift chamber to accomplish the hit classification of the background and signal.

1. Introduction
BESIII [1] is a multi-purpose spectrometer works on tau-charm region of BEPCII. The drift chamber is located in the inner most detector on BESIII which is the main tracking detector. The drift chamber is suffered with the background hits from beam background and electronic noise. The elimination of the background hits from the signal can improve the track finding efficiency, improve the track fitting performance and decrease the computing time.

Machine Learning is a kind of key technology on Artificial Intelligence (AI), and deep learning is an important research field and technology in Machine Learning. The base of deep learning - neural network [2] - was introduced in 1980s. Limited to computing capacity, neural network has little development after just being introduced. With the development of computer technology, the computing capacity has a large improvement in the past few decades, which it possible to construct complex model and process large number of data in computer. With high computing capacity and large amount of data, the deep neural network shows it can improve the performance in many fields.

Introducing machine learning technology into data processing is an important development trend of high-energy physics experiment. This paper focuses on applying the deep learning technology to the task of hits classification for the BESIII drift chamber.

2. Classification of signal and background hits
Classification of signal and background hits is a basic and important task of high energy physics tracking. It is a problem of a binary classification, so it is simple and suitable as the beginning of the research of deep learning technology applying to the high-energy physics data analysis.
Unlike general binary classification, there is an important limit in signal-background classification. In this study we want to keep the signal accuracy rate as high as possible at given false positive rate.

2.1. Data sample preparation
The event sample used in our study is the Bhabha event from real data of one run. In order to get data labeled by the signal and background, we reconstructed training and cross-validation data by taking the following steps:

(1) For each Bhabha event which have two reconstructed tracks, tag the hits associated with reconstructed tracks as signal hits and discard non-signal hits.

(2) Take the hits from an event without track of random trigger as background hits.

(3) Mix one event of signal hits from (1) and one event of background hits from (2) as one event. The average number of the signal hits is 90 and the distribution of the number of background hits is related with the beam situation. The typical noise hit number of the background is about 200.

(4) The number of events in data set from (3) is about 60,000. Divide it into 6 parts as training data set and cross-validation data set.

2.2. Traditional method on hit classification
Figure 1 shows the time and charge distribution of signal hits and background hits. Event start time is reconstructed in advance. We can see that the distribution of signal and background has obvious distinction. So a simple classification algorithm is to apply a cut on the time distribution (T-CUT model). Take hits whose time satisfied $-25\text{ ns} < t < 700\text{ ns}$ as a signal, and otherwise, as a background. Using above data in this configuration, we find that using the time cut, the background removal rate is 63.9% with signal reserve rate 99.1% which is not satisfied for this data set.

![Figure 1](image)

**Figure 1.** Charge distribution (left) and time distribution (right) of the data set, blue lines represent signal, and orange lines represent background.

2.3. Deep Learning on hit classification

2.3.1. Features
We take drift time, charge and position (radius and azimuth angle) of end-point of a hit as features for the classification. To take account in the track segment information as features for the classification, we introduce the neighbor array for each hit. An array including the status of neighbor hits for each hit will be constructed with 20 neighbor hits around it. The neighbor hits are defined based on the relative position on the end point of the wire which is shown in Figure 2. The element of array will be marked with one if the hit have time and charge information, otherwise it will be marked with zero. There are 24 features in total for one hit which including drift time, charge, radius and azimuth angle as local features, and a neighbor array with 20 elements.
2.3.2. Training model
We use a neural network model to do hit classification in this study. The neural network contains an input layer, an output layer and 2 inner full-connected layers. Input layer contains 24 nodes corresponding to 24 input parameters. Output layer contains 2 nodes corresponding to the classification results. There are 256 nodes in each inner layer.

Our training algorithm uses Softmax as activation function, which maps a group of real numbers to [0,1], and the output can represent probability. We use default optimizer – ADAM (Adaptive Moment Estimation) [3] – to train our model, and the loss function of optimizer should be consider carefully. With the limit that signal being misclassified as background about 1%, we should consider the loss function carefully. As for binary classification problem, the most frequently-used loss function is so-called Cross Entropy Loss Function. The formula of it is:

$$\text{Loss} = -[y \log \hat{y} + (1 - y) \log(1 - \hat{y})]$$

Because signal being misclassified as background is unacceptable, it is obvious that we should increase the loss of it. The first term of cross entropy loss function represents the loss of signal being misclassified as background, and the second term represents the loss of background being misclassified as signal. We should modify the formula of cross entropy loss function:

$$\text{Loss} = -[k y \log \hat{y} + (1 - y) \log(1 - \hat{y})]$$

Here k > 1, and k is determined by probing on a small size subset of data set. In our research, we find that k = 9 is a good choice for loss function.

We set epoch = 1, which means each training sample will be used just one time. And we set batch size = 128, which means each iteration in training will use 128 samples.

2.3.3. Architecture of prediction framework
We use TensorFlow [4], a famous and wide-used machine learning framework, to construct neural networks, and use Python as programming language. Due to the conflict of the software requirements, TensorFlow cannot be installed on the analysis node. We set up another standalone server to run TensorFlow service. The architecture is shown in Figure 3.
2.3.4. Test
We use data getting from section 2.1 to do training and cross-validating, and the fold number of cross validation is 6. Because T-CUT model gets a satisfied result, we combine the neural network model with T-CUT model, using AND logic, just as Figure 4 shows. And in this combined model, we set T-CUT parameter that if time information satisfies \(-25 < t < 800\) ns, return signal, and otherwise, background.

Figure 4. Combining neural network model with T-CUT model

In our study we find that using combined model, the background removal rate is 84.8% with signal reserve rate 99.0%, which has obvious improvement compared with using T-CUT model only. Figure 5 shows the cross validation result of data set with the statistic on each event.

Figure 5. The cross validation result of our data set. The abscissa is signal reserve rate and the ordinate is background reserve rate on each event. Each blue point represents an event in our data set, and the orange point represents the average value.
Figure 6 shows a background removal result for an event with event display.

Figure 6. The results on the prediction of the validation data set are shown above. Left: A labeled original event. Blue point is the tagged signal and red point is the tagged background; Right: After background removal. Blue point is the predicted signal and red point is the predicted background.

3. Summary
We have done the primary work of deep learning applied to hit classification for the BESIII drift chamber. From our work, we have seen the great improvement of deep learning in hits classification than traditional time cut method. We use 24 local and neighbor features and achieved considerable performance improvement.

However, our work is just a primary work. There are still a lot of problem need to be researched and a lot of methods worth to try in this field. First, We just research one type of event – Bhabha. Whether we can use the same idea in other type of events is an important problem to research. Second, we can try to use other types of machine learning models, such as GBDT(Gradient Boosting Decision Tree), and find which model is the best for this problem. Last, how to simplify our deep learning framework is also important.

4. Acknowledgments
This work is in part supported by National Key Basic Research Program of China under Contract No. 2015CB856706 and National Natural Science Foundation of China (NSFC) under Contracts Nos Y911U801K5 and 11911530090.

References
[1] BESIII Collaboration, Nuclear Instruments and Methods in Physics Research Section A (2010)
[2] Haykin, S. (1998). Neural Networks: A Comprehensive Foundation, 2nd edition. Prentice-Hall, Upper Saddle River, NJ.
[3] Diederik P. Kingma, Jimmy Lei Ba. ADAM: A Method For Stochastic Optimization, arXiv:1412.6980v9 (2017)
[4] TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, URL: https://www.tensorflow.org