2020

Machine Learning Applications at CMS

Joseph Peetz

THE FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS & SCIENCES

MACHINE LEARNING APPLICATIONS AT CMS

By

JOSEPH PEETZ

A Thesis submitted to the
Department of Physics
in partial fulfillment of the requirements for graduation with
Honors in the Major

Degree Awarded:

Spring, 2020

The members of the Defense Committee approve the thesis of Joseph Peetz defended on April 22, 2020.

Harrison Prosper

Dr. Harrison Prosper
Thesis Director

Ettore Aldrovandi

Dr. Ettore Aldrovandi
Outside Committee Member

Jorge Piekarewicz

Dr. Jorge Piekarewicz
Committee Member

Signature: Jorge Piekarewicz
Jorge Piekarewicz (Apr 23, 2020)
Email: jpiekarewicz@fsu.edu

Signature: Ettore Aldrovandi
Ettore Aldrovandi (Apr 23, 2020)
Email: ealdrovandi@fsu.edu

Signature:
Email: harry@hep.fsu.edu

Machine Learning Applications at CMS

Joe Peetz
Florida State University

Abstract
In this paper, we apply machine learning techniques to challenging problems at CMS. First, we attempt to predict the transverse momenta of jets of particles given their constituents' transverse momentum vectors. Second, we seek to classify types of Higgs boson events in order to extract signal data. We present the results from a variety of pre- and custom-built neural networks. The results for the momentum prediction problem are not successful, whereas the results for the Higgs boson classification problem achieve 90.8% accuracy and an AUC score of 0.959, a significant improvement over benchmark practices.

1 Introduction

At the Large Hadron Collider (LHC) [1], the collision of protons at relativistic speeds results in jets of particles whose properties are measured by experiments such as the Compact Muon Solenoid (CMS) detector [2]. The properties of these jets offer insight into the underlying physical interactions that formed them, thus allowing physicists to test theories and measure quantities of interest, such as cross sections. A primary goal of the LHC and CMS experiment is the discovery of physics Beyond the Standard Model (BSM). The most direct way to do this is by discovering new particles, which has been the focus of scientists at the LHC and several other particle physics experiments for many years. Since the discovery of the Higgs boson in 2012 [3], there have not been any of these direct discoveries. However, the existence of unexplained phenomena, such as neutrinos' non-zero mass, implies the existence of new physics and raises the question of whether a new approach needs to be considered. In particular, it gives credence to the approach of inferring the existence of BSM physics by precisely measuring the properties of discovered particles and their decay modes. This mindset serves as motivation for this thesis, in which two problems of precision measurements are explored. This paper first introduces the context and motivation of these problems, then presents the framework of machine learning needed to address them, and finally discusses the implementations and results.

1.1 Calculating Transverse Momentum of Jets

A key characteristic of jets is their transverse momentum, p_T, defined as the component of momentum perpendicular to the proton-proton beam axis (z). Following standard CMS practice, we define the transverse momentum vector, $\vec{p}_T$, in terms of the azimuthal angle, ϕ, and the pseudorapidity, $\eta = -\ln\tan\frac{\theta}{2}$, where θ is the polar angle. We write,

$$\vec{p}_T = (p_T, \eta, \phi). \tag{1}$$

Given the transverse momentum vectors of the particles comprising the jets, we can determine the overall transverse momentum vector of the jet. However, due to experimental uncertainties, these vectors and thus the p_T of the jet have non-negligible errors. A particular problem of using this method at LHC experiments is that it can mistakenly include as part of the jet particles from separate proton-proton interactions from the interaction of interest. In response, we explore

an alternative method—calculating the jet p_T through machine learning, with the transverse momentum vectors of the jets' particles as input. The hope is that a neural network can discern which particles originate from the correct source and potentially improve the measurement of p_T. Later work may use this process to determine the overall vector $\vec{p}_T$.

We are particularly interested in this algorithm's potential use in the search for contact interactions. In theory, the governing forces of these interactions involve the exchange of heavy bosons with such short decay lifetimes that they travel immeasurably small distances before decaying. Hence, the interacting particles appear to come into "contact" with one another at a single point in space. The cross sections of such interactions are theorized to most significantly affect the overall experimental cross section at high values of p_T. By comparing CMS data to Standard Model predictions, we can search for deviations from the predictions that could be explained by the presence of contact interactions. However, in order to evaluate discrepancies precisely, we must have reliable measurements of p_T, thus providing the motivation for this project. Further, because of the frequent usage of p_T in CMS analyses more generally, an improvement of its measurement would likely have wider applications.

The technical challenge of this task arises from the inherent variability of the input information: Because jets have differing numbers of constituent particles, the dimensionality of the input into the machine learning network will vary. In response, we will use an interaction network [4], a type of graph neural network that can model relationships between inputs of this type. Because we know the true jet p_T in CMS simulation data, we can measure the performance of our network and thus assess the validity of this algorithm in measuring p_T and possibly in more general applications.

1.2 Classification of Higgs Boson Events

Supersymmetry [5] predicts the existence of another class of matter that includes an additional neutral Higgs boson. While the hypothesized Higgs boson may be too massive to create directly at existing colliders, quantum superposition implies that its existence would influence the overall production rate of the discovered Higgs boson. There are many ways in which the Higgs boson can be created, which are referred to as its production modes. Crucially, the existence of an additional Higgs boson would affect the production rates of these modes differently. These differences motivate the desire to measure all Higgs boson production modes as precisely as possible, searching for deviations that suggest the existence of another Higgs boson or other BSM physics.

In order to measure the properties of the Higgs boson, the ATLAS and CMS collaborations have used the $H \to ZZ \to 4l$ decay channel, where Z denotes Z bosons and the leptons l are either electrons or muons. This channel allows for precise reconstruction of events and offers a signal that is relatively large compared to its background [6]. Selected signal events primarily consist of two production modes: gluon-gluon fusion (ggF) and vector-boson fusion (VBF), depicted by their Feynman diagrams in Figure 1. In order to precisely measure the respective amplitudes of these two modes, it is crucial to be able to confidently isolate them from each other. Due to the difficulty of accomplishing this task through traditional algorithms, physicists have been exploring the practicality of using machine learning classifiers instead.

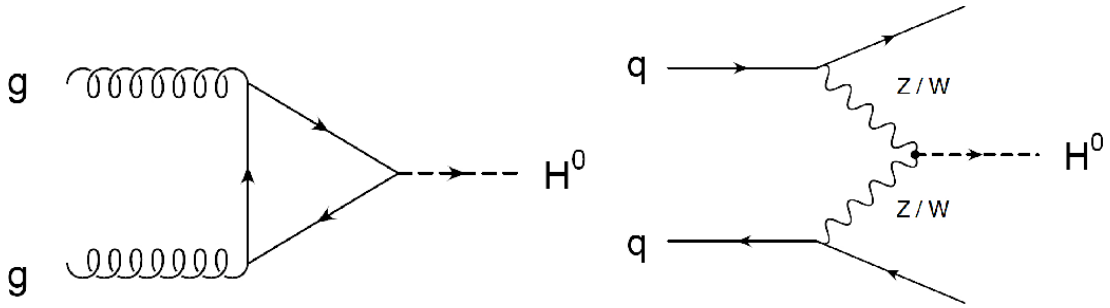

Fig. 1: Feynman diagrams of the Higgs boson production modes of interest. On the left is gluon-gluon fusion (ggF) production, and on the right is vector-boson fusion (VBF) production [7].

Traditional machine learning classifiers between VBF and ggF Higgs boson events require the existence of two or more jets. As input, LHC physicists use the di-jet mass m_{jj} and the absolute pseudorapidity difference $|\Delta\eta|_{jj}$. These variables allow for reasonable discrimination because VBF events typically feature two forward-traveling jets with large rapidity, whereas ggF events do not. The results from this type of classifier are used as a benchmark upon which novel classifiers can improve. This paper specifically explores relaxing the usual restriction on the number of jets and using the transverse momentum vectors $\vec{p}_T$ of the four final leptons as input into a classifier. As is later shown, this results in a higher yield of both VBF and ggF events and improved classification accuracy compared to the benchmark results.

1.3 Machine Learning

In this section, I will start with a brief introduction to machine learning and then discuss interaction networks in greater depth. "Neural network" is an umbrella term for machine learning functions inspired by the structure of neurons in a brain. Each node (or neuron) passes some input through a function with optimizable parameters and then sends the result to either further nodes or to the final output. Despite the common use of vague, abstract words such as "learn" and "train" to describe neural networks, they are ultimately just versatile mathematical functions. By having structures with many variable parameters, neural networks are able to model a wide variety of simple and complex functions.

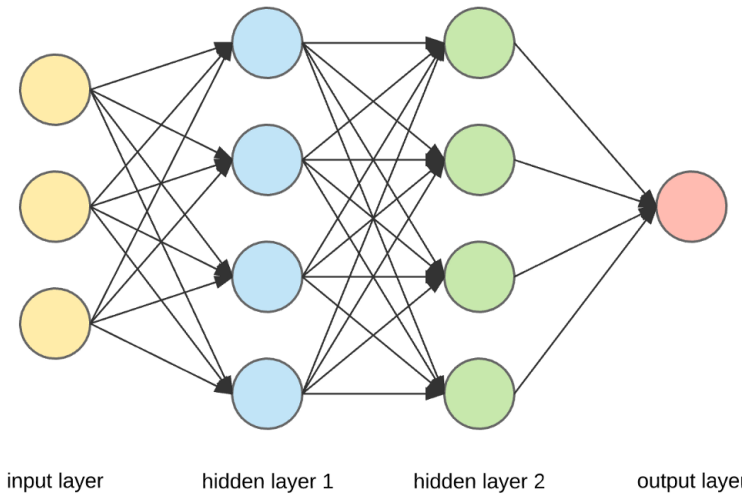

Fig. 2: A simple linear neural network, with three-dimensional input, two hidden layers, and one-dimensional output [8].

As an example, consider the linear neural network in Figure 2 above. The three yellow circles or "nodes" in the first layer correspond to input variables; the eight nodes in the middle two layers serve as intermediary calculations; and the final node in the last layer is the output of the network. The lines connecting nodes represent weights, or coefficients, with convergence of lines indicating a weighted sum of inputs. For example, because each yellow node connects to the uppermost blue node, this blue node is equal to a weighted sum of the yellow input layers. Following this process throughout, the resulting output is then compared to the desired output in order to calculate the error or "loss" of the overall network. A common loss function is the quadratic loss, which is simply the difference between the calculated and desired outputs squared. This loss is then used to incrementally update the weights of each layer in order to improve the accuracy of the output, often through the process of stochastic gradient descent [9]. By increasing the size of layers, adding nonlinear functions, or by designing new network structures altogether, we can build flexible neural networks capable of fitting a wide variety of functions.

Neural networks typically fall into two categories based on the type of output: predictors, which calculate real-valued quantities, and classifiers, which calculate the probability of inputs being members of predefined classes. In Project 1, we use predictor networks to calculate the momentum of jets. In Project 2, we use classifier networks to sort Higgs boson events by production mode. While standard networks have a fixed number of inputs as in Figure 2, in Project 1, we explore relaxing this requirement through the use of interaction networks.

The interaction network was introduced in 2016 in the paper "Interaction Networks for Learning about Objects, Relations and Physics" [4], which will guide our theoretical understanding. The core concept is that within the interaction network I, a flexible function f_R specifically models relationships between input objects O, and a second flexible function f_S accounts for uniform, system-wide effects. Crucially, the parameterization of f_R and f_S will depend on the dimensionality of the individual input objects, but not on the overall number of input objects. Conceptually, while the overall properties of a system (modeled by *I*) can depend on the number of objects within it, the underlying dynamics (modeled by f_R and f_S) will not. Together, our prediction X is given by:

$$X = I(O, f_R, f_S). \tag{2}$$

In order to yield accurate predictions, we seek to optimize the parameters of f_R and f_S with respect to a loss function L, representing the error of a prediction. While varying the function parameters, we pass through our network a set of training data for which we know the correct p_T and can thus calculate the loss of each prediction. We then use the process of stochastic gradient descent to determine the parameters which minimize the loss.

In our case, our objects O are the transverse momenta of the particles $\{\vec{p}_{T,i}\}$ within a jet, f_R models relationships between these, f_S models system-wide effects, and our output X is the transverse momentum of the jet, now denoted as capitalized P_T. That is,

$$P_T = I(\{\vec{p}_{T,i}\}; f_R, f_S), \tag{3}$$

where f_R depends on pairs of particles as well as its own learnable parameters, $\vec{r}$:

$$f_R = f_R(\vec{r}, \{(\vec{p}_{T,i}, \vec{p}_{T,j})\}), \tag{4}$$

and f_S depends on the system of particles as well as its own learnable parameters, $\vec{s}$:

$$f_S = f_S(\vec{s}, \{\vec{p}_{T,i}\}). \tag{5}$$

2 Implementation

We use the Python package PyTorch to implement the interaction network for use in Project 1. PyTorch allows the specification of the exact types of transformations used in f_R and f_S as well as how the overall function I uses the two to yield an output. Compared to a myriad of custom PyTorch network structures, the one found to have the best results is modeled after an event classifier designed for use at the IceCube Neutrino Observatory [10]. The exact network structure and parameters are as follows. Denote the input list of p_T vectors as X, such that X_{ij} denotes the ith value out of (p_T, ϕ, η) of the jth particle in the list. Construct the adjacency matrix A as:

$$A_{ij} = \frac{\exp(d_{ij})}{\sum_k \exp(d_{ik})}, \tag{6}$$

where d_{ij} is given by

$$d_{ij} = \exp(\frac{1}{2\sigma^2}||X_i - Xj||^2), \tag{7}$$

and where σ is an optimizable parameter. Note that the adjacency matrix parallels the relational function f_R discussed previously, and the following operations correspond to f_S. Now, perform the following transformation to the input X, where [C,D] denotes row-wise concatenation of matrices C and D.

$$X' = [\text{ReLU}(Y), Y], \tag{8}$$

where Y is given by:

$$Y = [AX, X]\vec{a_1}^T + b_1\mathbf{1}. \tag{9}$$

Here, $\vec{a_1}$ is a weight vector, and b_1 is a scalar bias. The final output is then given by:

$$P_T = (\sum_j X')\vec{a_2}^T + b_2, \tag{10}$$

where $\vec{a_2}$ is another weight vector, and b_2 is another scalar bias. This output is compared to the true P_T to compute the loss, which is then used to update the various weight parameters through stochastic gradient descent. To clarify, the "true" P_T exists in simulation data but not in experimental data. The results section also compares the accuracy of the network's output value to the accuracy of the "experimental" P_T, as in the transverse momentum calculated using existing, standard algorithms. This serves as a measure of how well the network is performing relative to current benchmarks.

For Project 2, on the other hand, successful results are presented using the Classify[] function from Mathematica. This function tries a wide variety of common network structures before settling on the one with the best results. This approach powerfully addresses the issue of uncertainty in choosing the best type of network for any given machine learning problem. As mentioned before, the benchmark results are determined by a network that uses the di-jet mass m_{jj} and the absolute pseudorapidity difference $|\Delta\eta|_{jj}$ as input. The novel approach presented uses the transverse momentum vectors $\vec{p}_T$ of the four leptons as input, and results in higher signal yield and improved classification accuracy.

Projects 1 and 2 use separate data sets that were both produced using CMS simulations. Following standard machine learning practices, all of these studies involve randomly partitioning the data sets into two samples to separately train the networks and then test their results. In Project 2, each of these sets contains a roughly even split of ggF and VBF events.

3 Results

This section presents and interprets the results, first for Project 1 and then for Project 2.

3.1 Transverse Momentum of Jets

As discussed above, the following results come from a PyTorch implementation of a neural network modeled after IceCube's event classification network. The first two figures below show a variety of indicators of the performance of the network, and the subsequent three figures compare the distributions of events with successfully predicted p_T values to those with unsuccessfully predicted p_T values.

Fig. 3: This figure depicts the performance of the network on the test set as a function of the number of training iterations performed. It uses a number of measures: the percentage of output values within 20 GeV of the true P_T, the percentage within 40% of the true P_T, and the percentage more accurate than standard experimental algorithms.

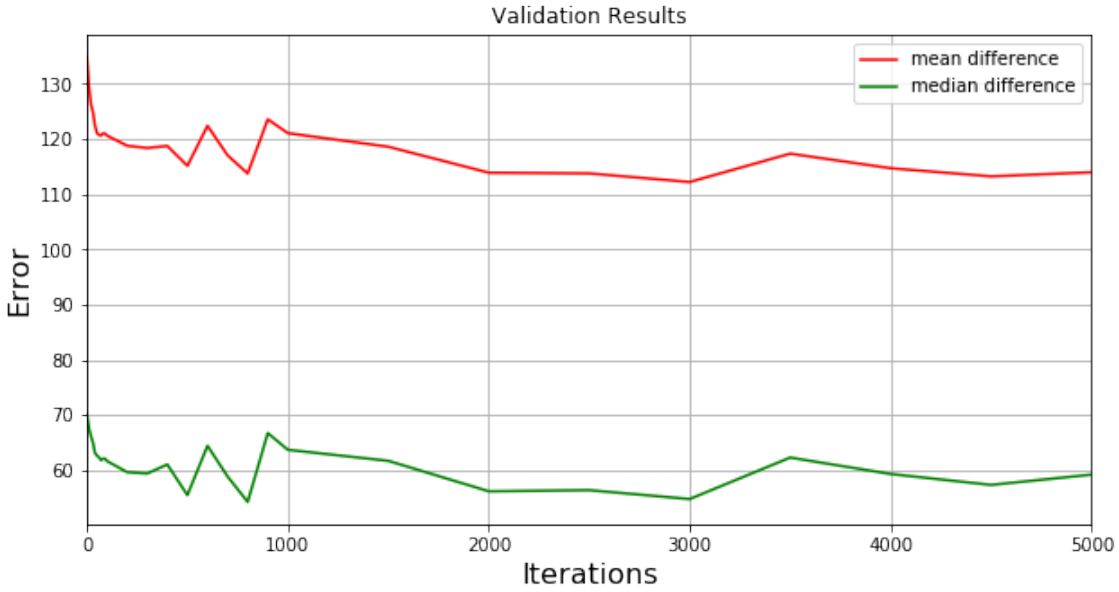

Fig. 4: This figure depicts the mean and median differences between the networks' output values and the true P_T, as a function of the number of training iterations performed. The y axis is in the units GeV.

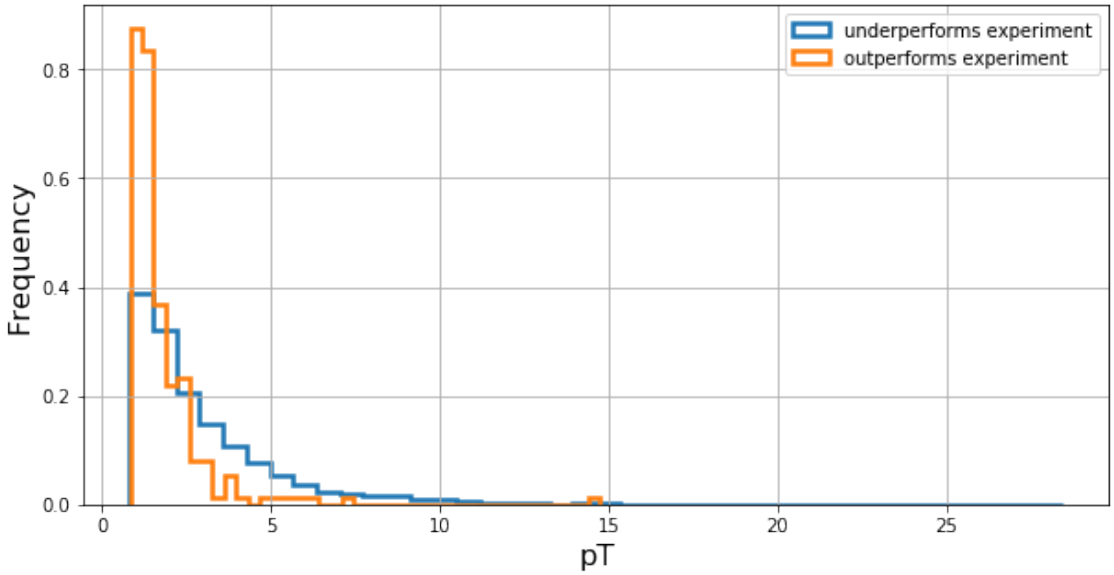

Fig. 5: This histogram plots the average input p_T values of the jets' constituent particles. It compares the distribution of jets with predicted P_T values that are more accurate than standard experimental algorithms to the distribution with less accurate predictions. Both distributions are normalized here.

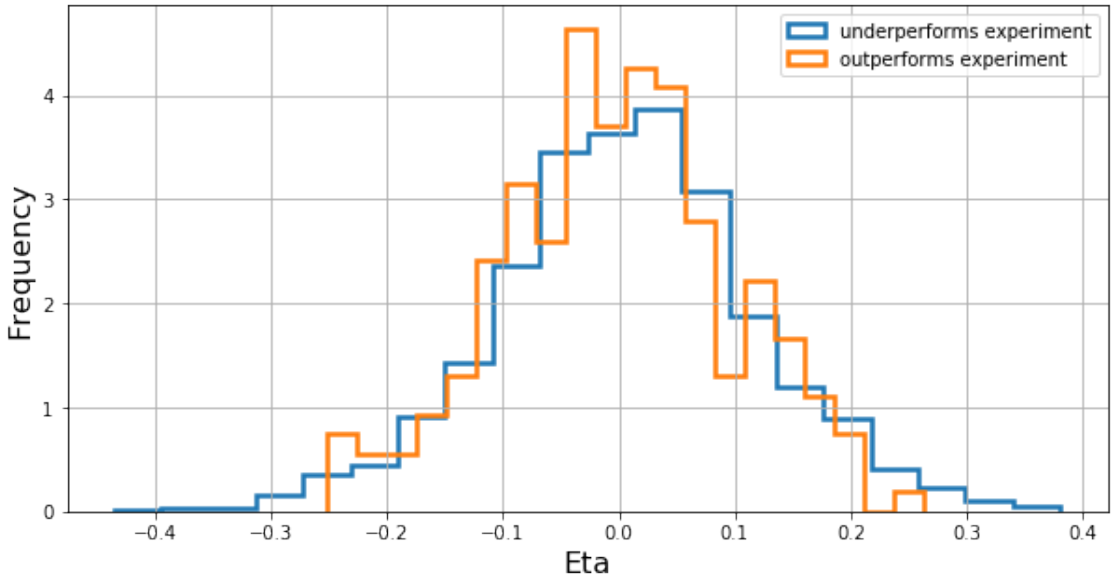

Fig. 6: This histogram plots the average input η values of the jets' constituent particles. It compares the distribution of jets with predicted P_T values that are more accurate than standard experimental algorithms to the distribution with less accurate predictions. Both distributions are normalized here.

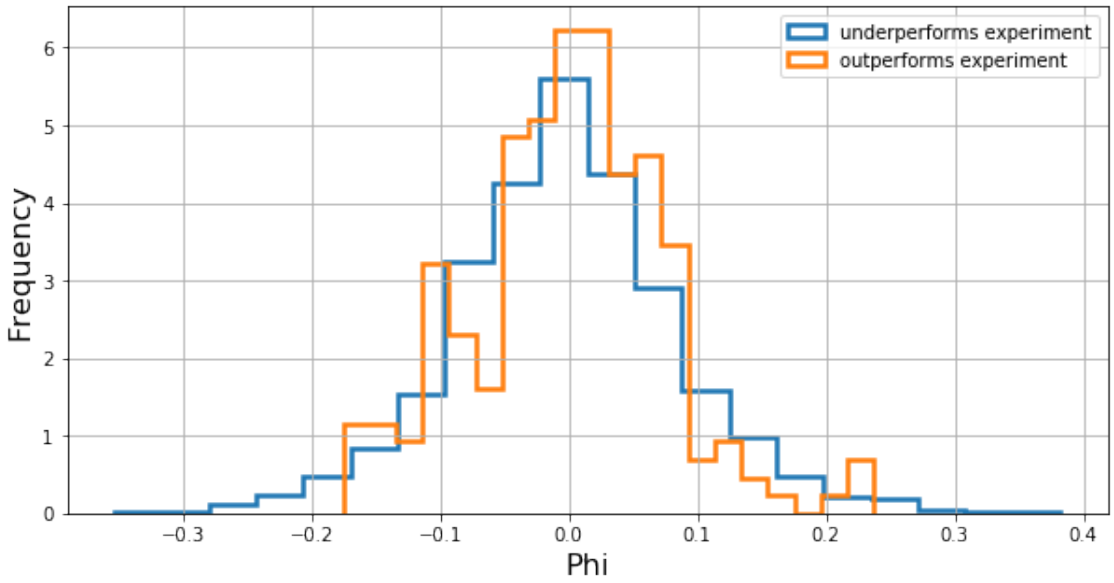

Fig. 7: This histogram plots the average input ϕ values of the jets' constituent particles. It compares the distribution of jets with predicted P_T values that are more accurate than standard experimental algorithms to the distribution with less accurate predictions. Both distributions are normalized here.

After 5000 training iterations, the final performance metrics are: 70.7% of outputs within 40% of the true P_T, 25.6% within 20 GeV, 10.5% more accurate than standard algorithms, a mean difference of 114.0 GeV, and a median difference of 59.2 GeV. Despite signs of these metrics improving during the early training iterations, the end result is ultimately unsuccessful. Additionally, the comparative histograms above do not show obvious, actionable differences between the input jets that outperform standard algorithms and those that underperform. If they

had differing distributions with areas of little overlap, this algorithm may have had useful predictions for a subset of the input data. However, it is clear from these results that this algorithm is ultimately too inaccurate to be useful, especially in trying to output more accurate P_T values than standard experimental algorithms.

3.2 Classification of Higgs Boson Events

The first set of results presented are from a benchmark network with m_{jj} and $|\Delta\eta|_{jj}$ as inputs. These are obtained using Mathematica's Classify[] function as described in the Implementations section. The following four figures depict the benchmark's performance metrics.

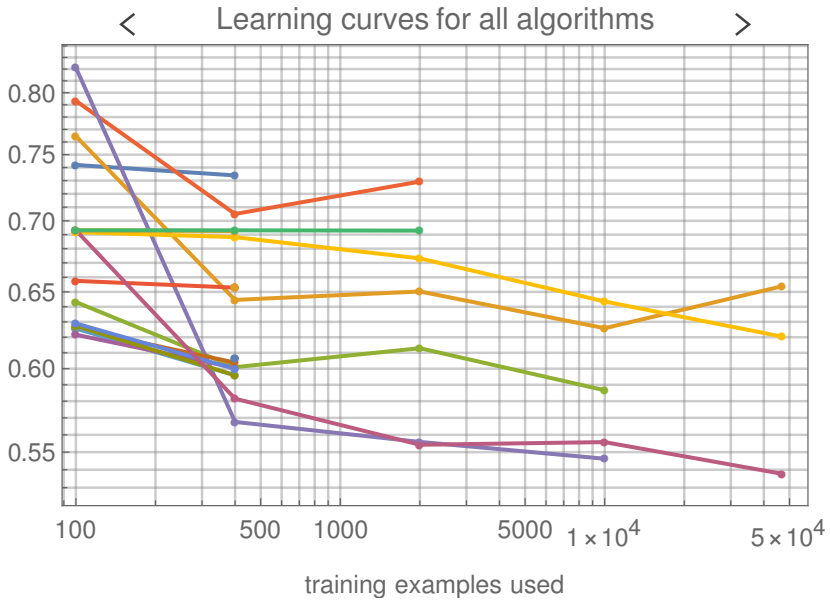

Fig. 8: Pictured here is the mean cross entropy, a measure of loss, as a function of the number of training iterations performed. This figure highlights the power of Mathematica's Classify[] function, which tests several different network structures and outputs the one with the best result. For example, the magenta line represents a gradient-boosted decision tree model, and the violet line represents a nearest neighbor model. The gradient-boosted decision tree has the best performance and is thus used for the other metrics.

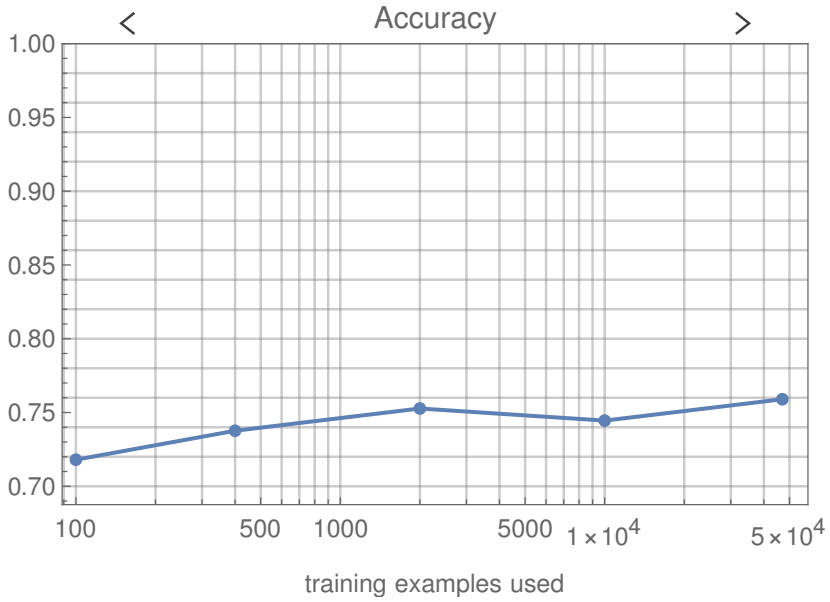

Fig. 9: This graph shows the classification accuracy of the gradient-boosted decision tree as a function of the number of iterations. Note that this accuracy is evaluated on the training data set. The final accuracy attained is 75.9%.

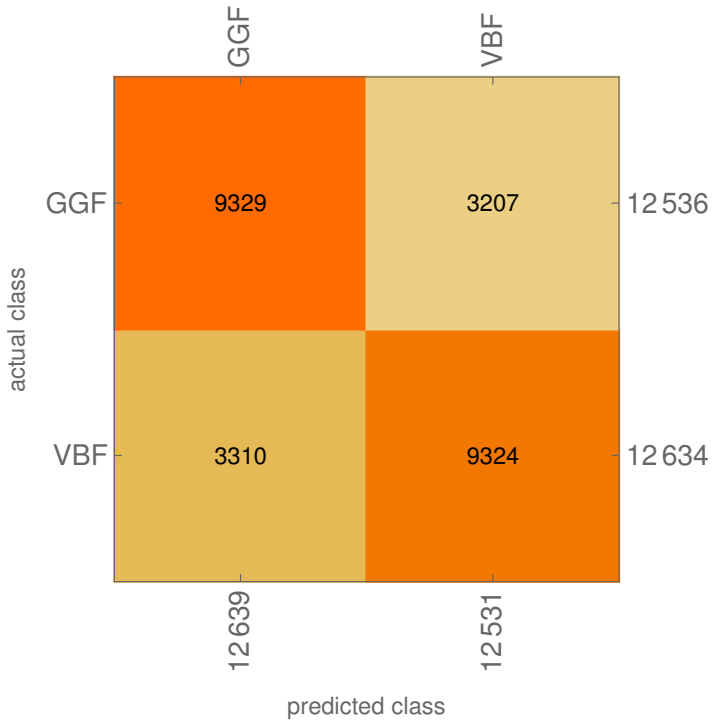

Fig. 10: This is the classifier's "Confusion Matrix," an infographic showing the precise counts of events classified correctly and incorrectly by category. For example, of the 12,531 events predicted to be VBF events, 9324 were actually VBF events and the remaining 3207 events were actually GGF events. Note that these results are obtained using the independent test data set.

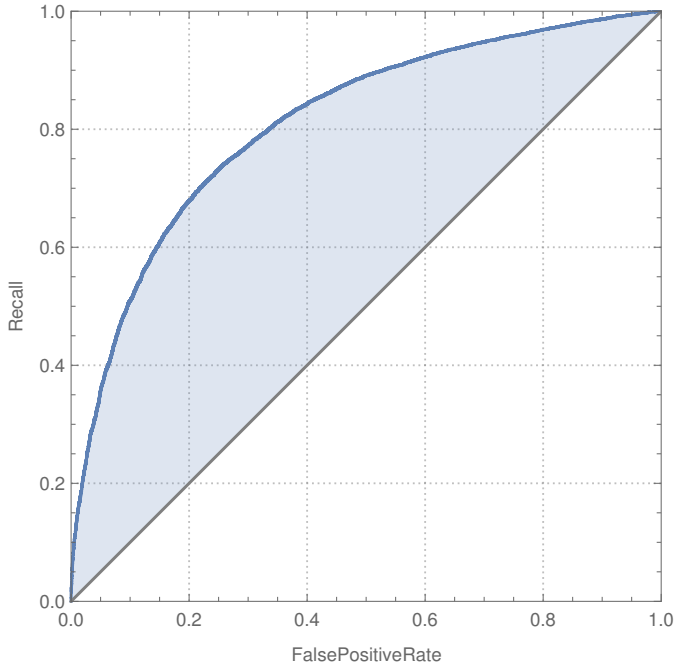

Fig. 11: This figure depicts this classifier's receiver operating characteristic (ROC) curve. While varying the discrimination threshold, this ROC curve plots the true positive rate of VBF classification, i.e. the recall, versus the false positive rate of VBF classification. The area under the curve (AUC) is a commonly used global measure of the discrimination power of a classifier. It includes the area of 0.5 under the diagonal line, which corresponds to no discrimination power. Here, AUC = 0.812.

The second set of results presented are from a novel network with the transverse momentum vectors $\vec{p}_T$ of the four leptons as input. The following four figures depict this network's performance metrics.

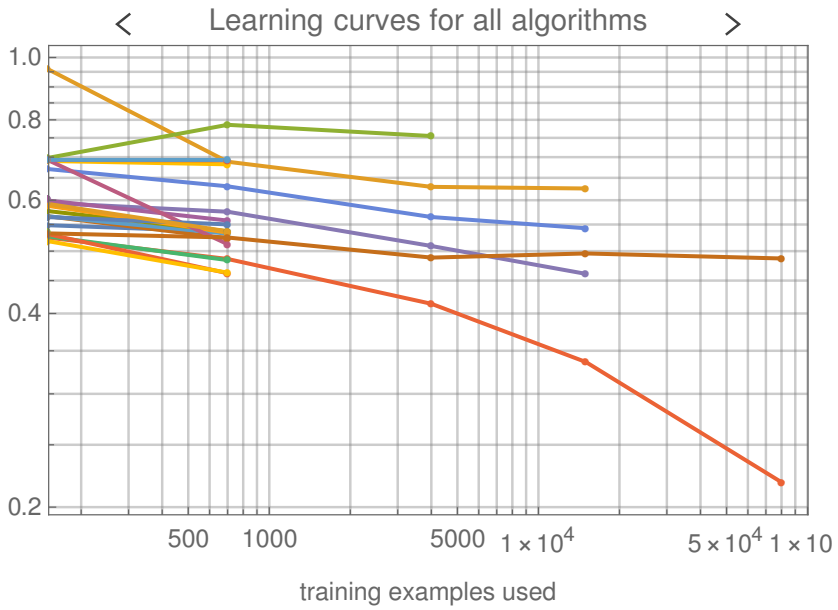

Fig. 12: Pictured here is the mean cross entropy, a measure of loss, as a function of the number of training iterations performed. Mathematica's Classify[] function tests several different network structures and outputs the one with the best result. For example, the red line represents a nearest neighbor model, and the violet line represents a random decision forest model. The nearest neighbor method has the best performance and is thus used for the other metrics.

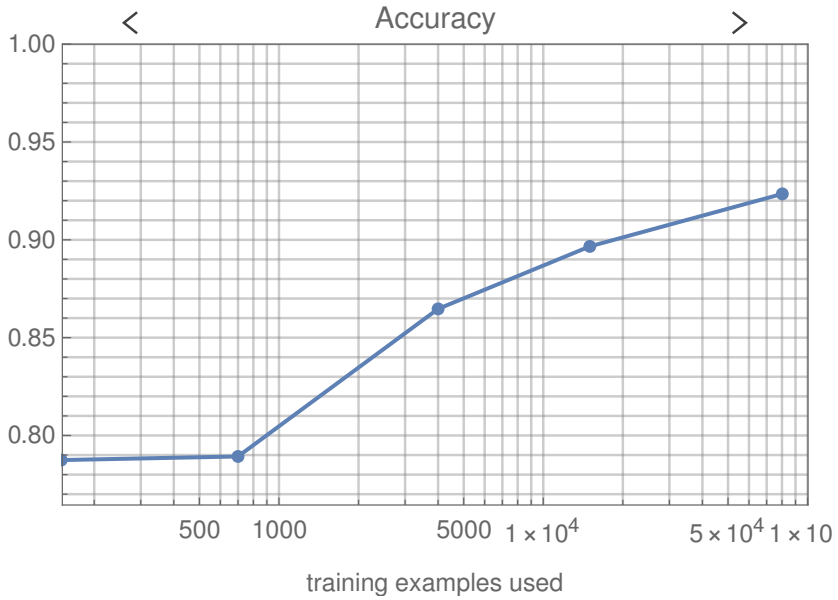

Fig. 13: This graph shows the classification accuracy of the nearest neighbor algorithm as a function of the number of iterations. Note that this accuracy is evaluated on the training data set. The final accuracy attained is 92.3%.

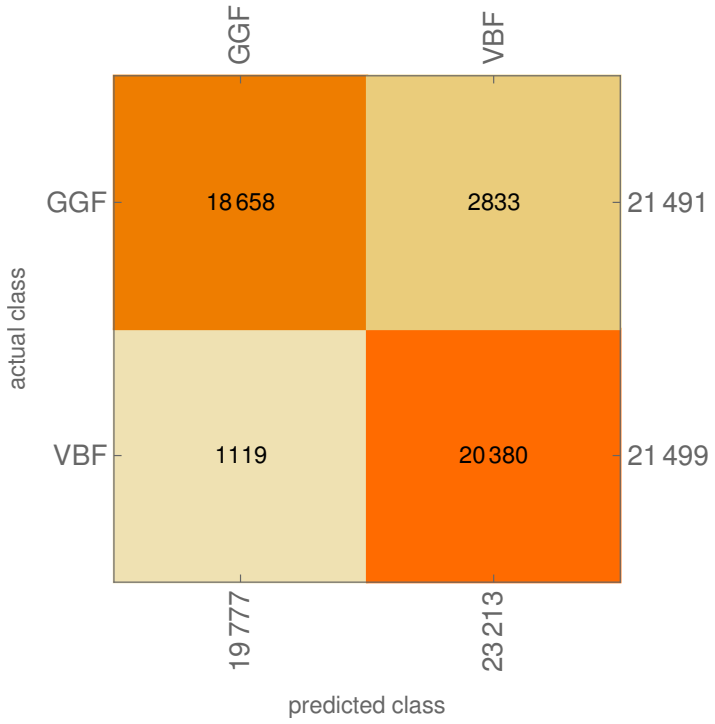

Fig. 14: This Confusion Matrix shows the precise counts of events classified correctly and incorrectly by category. For example, of the 23,213 events predicted to be VBF events, 20,380 were actually VBF events and the remaining 2833 events were actually GGF events. Note that these results are obtained using the independent test data set.

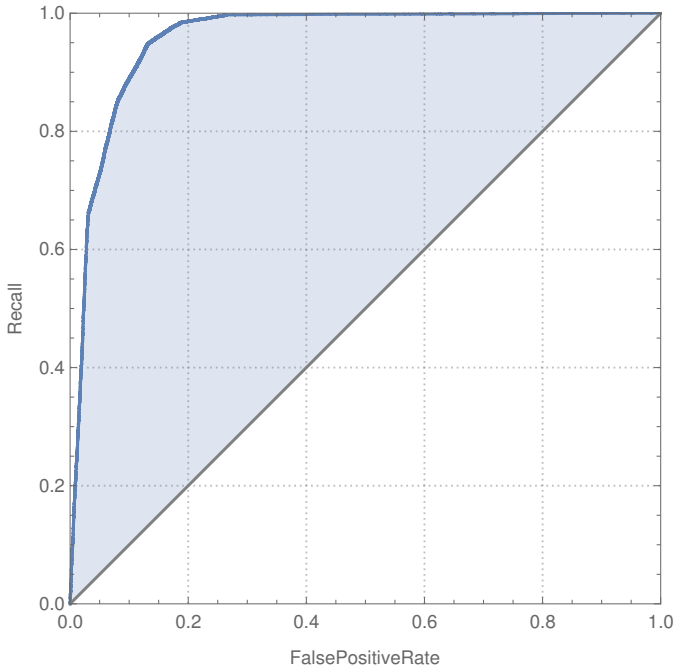

Fig. 15: This figure depicts this classifier's ROC curve, for which AUC = 0.959.

The final results for the benchmark classifier are 74.1% classification accuracy on a test set and an AUC score of 0.812. The corresponding results for the novel classifier are 90.8% classification accuracy on a test set and an AUC score of 0.959. This classifier thus offers significant performance benefits over the benchmark method. Further, the continuing upward trend in accuracy and downward trend in loss suggest that better results may be attainable with larger training data sets. Because the newly proposed method does not restrict the number of jets as in the benchmark method, it also increases the overall yield of signal events, which is discussed next.

In the most recent CMS results on $H \rightarrow ZZ \rightarrow 4l$ [6], simulated Higgs boson events are divided into groups based on the type of production, as depicted in Figure 16 below. According to "Table 2" of this paper, out of the 4.44 total VBF events, 1.14 fall into the VBF-1jet tagged category, 1.08 fall into the Untagged category, and <0.01 fall into the VH-E_T^{miss} category. The remaining categories require at least two jets. Because the proportion of Untagged VBF events with fewer than two jets is not provided, we cannot determine the amount of signal events gained regardless of the number of jets. However, adding the VBF-1jet events alone amounts to a 35% increase in yield of total VBF signal events.

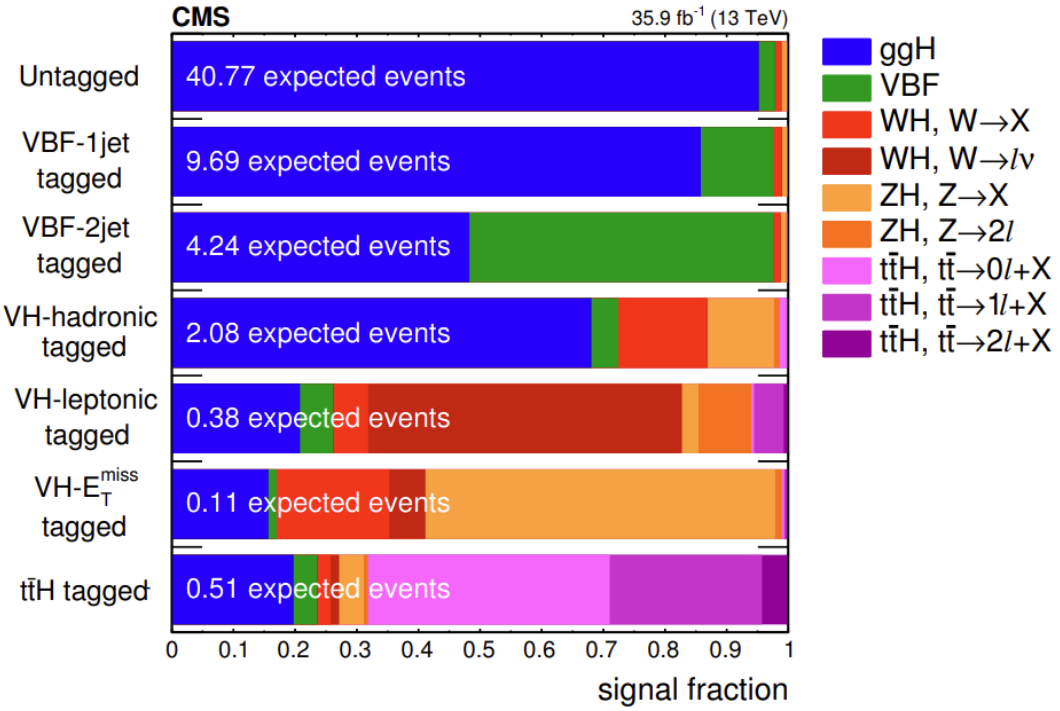

Fig. 16: This chart depicts the amount of events predicted for mutually exclusive selection criteria, broken down by production mode. The absolute numbers of events correspond to an integrated luminosity of 35.9 fb^{-1} [6].

4 Summary

In this thesis, applications of machine learning at CMS are explored in two challenges: 1) calculating the p_T of jets and 2) classifying between VBF and ggF Higgs boson events. The results of Project 1 are ultimately unsuccessful, with the output p_T values too inaccurate to be useful. However, due to their widespread potential, neural networks with variable-length inputs deserve further exploration in additional studies. The results of Project 2 are happily quite successful. By eliminating restrictions on the number of jets in events, the presented

classifier immediately increases the number of VBF events by at least 35%. Additionally, the novel nearest neighbor classifier is 90.8% accurate with a AUC score of 0.959, compared to the benchmark classifier's 74.1% accuracy and AUC score of 0.812. By increasing the yield of Higgs boson events and more accurately distinguishing between the VBF and ggF modes, these modes' production rates can be more precisely measured.

References

[1] L. Evans and P. Bryant (editors), "LHC Machine", JINST 3, S08001 (2008).

[2] CMS Collaboration, "The CMS Experiment at the CERN LHC," JINST 3, S08004 (2008).

[3] CMS Collaboration, "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC," Phys. Lett. B716 (2012).

[4] Battaglia et al., "Interaction Networks for Learning about Objects, Relations and Physics," arXiv:1612.00222 (2016).

[5] Haber, Howard E. and Haskins, Laurel Stephenson, "Supersymmetric Theory and Models," Anticipating the Next Discoveries in Particle Physics (2018).

[6] CMS Collaboration, "Measurements of Properties of the Higgs Boson Decaying into the Four-Lepton Final State in Pp Collisions at $\sqrt{s} = 13$ TeV," Journal of High Energy Physics 2017.11 (2017).

[7] Ilić, Nikolina, "Observation of the Higgs Boson Decaying to WW to leptons and neutrinos," University of Toronto (2015).

[8] https://towardsdatascience.com/everything-you-need-to-know-about-neural-networks-and-backpropagation-machine-learning-made-easy-e5285bc2be3a

[9] Ruder, Sebastian, "An overview of gradient descent optimization algorithms," arXiv:1609.04747 (2016).

[10] Choma et al., "Graph Neural Networks for IceCube Signal Classification," arXiv:1809.06166 (2018).