**Deep Learning Final Report:**

**Using Deep Neural Networks to Detect Higgs Bosons in Simulated Data**

Matthew Boyer

Jonah Goldfine

**Abstract**

Blah blah blah

*Keywords:*

**1 Introduction**

**1.1 History**

The Standard Model is now the most widely accepted particle physics theory that describes three (electromagnetic, weak, and strong interactions) of the four (gravity) fundamental forces and all of the elementary particles. All of the elementary particles can be broken into two categories: fermions and bosons. Generally fermions are associated with matter, whereas bosons are generally considered force carriers. By 1964, physicists had solid theories that could explain and combine the electromagnetic and weak interactions, but this theory failed to explain where mass came from1. That was until six physicists across three different teams proposed a theoretical mechanism and particle as a way for other particles to acquire mass. At the time, all known particles were massless at very high energies, but not at low energies. During the 70s this theory (now known as the Higgs Mechanism and Higgs Boson) would be combined with the electroweak and the Quantum Chromodynamics (the theory that describes the strong interaction) into the full Standard Model2.

By 1977, physicists had experimental proof of all of the elementary particles, except for three: Top Quark, Tau Neutrino, and the Higgs Boson. Though the Top Quark and Tau Neutrino were not experimentally detected until 1994 and 2000 respectively, it was believed that both particles should exist in pairs because of already known particles (Electrons and Muons had Electron and Muon Neutrinos, so Taus should have a Tau Neutrino. Up and charm quarks have down and strange quarks, so Bottom Quark should have a Top Quark)1. This same parity does not exist for the Higgs Boson, making its existence less plausible. That was until 2012 when a subatomic particle with the expected properties of the Higgs Boson was detected by the ATLAS and CMS experiments at the Large Hadron Collider (LHC) at CERN. This evidence was tested and confirmed less than a year later in 20132.

**1.2 Background**

Exotic particles are produced at LHC by colliding two massive particles into each other at speeds near the speed of light (for higgs bosons it is two protons), but in most of the collisions no exotic particles are produced. Even when exotic particles are produced, their detection is not as simple as just watching the collision. Exotic particles decay very quickly, so physicists actually look for signals from the decay constituents. When two protons () collide and produce a neutral Higgs Boson (), the Higgs Boson decays into charged W () and Higgs () Bosons, the charged Higgs Boson decays further into a light Higgs () Boson and another charged W   
() Boson, and, finally, the light Higgs Boson decays into a pair of bottom quarks (). This gives an overall process of 3. The more likely reaction in this proton-proton collision, however, produces two top quarks which also decay into two charged W bosons and two bottom quarks3. For these reasons, the detection of a Higgs Boson is especially difficult.

**1.3 The Original Paper3**

In 2014 computer scientists Pierre Baldi and Peter Sadowski and physicist Daniel Whiteson published a paper describing the standard approach (more basic machine learning techniques) used to resolve Higgs Boson Signals from background signals and proposed the use of Deep Learning methods to better improve detections. This paper also goes further and tests the methodology for the discovery of new supersymmetry particles, but that is beyond the scope of this project.

For the paper, Baldi, Sadowski, and Whiteson created a data set consisting of low-level and high-level features that are related to the proton-proton reaction using monte carlo simulations. The models they tested were a Boosted Decision Tree (as a benchmark), a shallow neural network, and a deep neural network. They also did hyperparameter tuning on the Neural Networks. They tested these models on just the low-level features, just the high-level features, and the complete features, but this project only focuses on the complete features.

**2 The Data**

**2.1 Original Data3,4**

The data set for this project can be downloaded from the UCI Learning Repository4 or downloaded from the TensorFlow Dataset Collection. It consists of 11 million rows, each one representing a different collision and 28 features. Of the 28 features, 16 of the are related to jets (the “shrapnel” after caused by the collisions that contain the decay constituents), three are related to the momentum of Leptons (either electrons or muons), seven different invariant mass distribution peaks, and two related to the missing energy.

* For the 16 jet features, there are four features for each of the four jets:
  + Two represent angle of the jet (jet\_n\_eta and jet\_n\_phi)
  + One represents the momentum of the jet transverse to the beam direction (jet\_n\_pt)
  + One that is a tag that represents the existence of a Bottom Quark in the jet (jet\_n\_b-tag)
* For the three related to the momentum of Leptons:
  + Two representing the angle of the lepton (lepton\_phi and lepton\_eta)
  + One representing the transverse magnitude of the lepton (lepton\_pt)
* For the seven features related to invariant masses:
  + One represents a bottom quark pair (m\_bb)
  + One represents a pair of jets (m\_jj)
  + One represents a triplet of jets (m\_jjj)
  + One represents a jet and a lepton pair (m\_jlv)
  + One represents represents Lepton pair (m\_lv)
  + One represents W boson with bottom quark pair (m\_wbb)
  + One represents a W boson pair and bottom quark pair (m\_wwbb)

**2.2 Data Preparation**

In the original paper, the authors took the simulated data and assumed that each feature has a normal or Gaussian distribution, despite them not necessarily being so (See Figure 1), and fit the data to that3,5. Originally we assumed the same and used StandardScaler to fit the data, but upon further investigation we realized that we could treat some of the features as categorical variables. In some cases this was because they were actual categorical variables (jet\_n\_b-tag), in other cases (lepton\_eta and m\_jj) we were able to treat them as categorical variables because we are not doing anything with them related to physics. Initially we split this data on a 80:20 train:test ratio, however, our methods for training the data further splits the training set into an additional 80:20 ratio. After Yeo-Johnson transformation (normalization), encoding, and splitting, we ended up with 32 features with 11 million rows split on a 64:16:20 training:validation:testing ratio.

**3 Modeling**

| **TECHNIQUE** | **SOURCE** | **AUC** | **ACCURACY** |
| --- | --- | --- | --- |
| Boosted Decision Tree | Baldi Et al. | 0.81 | Unlisted |
| Base DN | Baldi Et al. | 0.885 | Unlisted |
| Best DN | Baldi Et al. | 0.893 | Unlisted |
| Decision Tree | Project | 0.67 | 0.67 |
| Logistic Regression | Project | 0.64 | 0.68 |
| K Nearest Neighbors | Project | 0.63 | 0.621 |
| Table 1: Baseline Models | | | |

**3.1 Baseline Models**

The goal of this project is to develop a Deep Neural Network comparable (and hopefully better) than that created for Baldi Et al., so we, of course, have those models as a baseline, but we also wanted to create our own baselines. We landed on three different standard machine learning methods: Decision Tree Binary Classifier (Supplementary Figures 1 & 2), Logistic Regression (Supplementary Figures 3 & 4), and K Nearest Neighbors (Supplementary Figures 5 & 6). The AUCs and Accuracies for the six baselines are listed in Table 1. Clearly, our baseline models performed worse than the three models from the original paper. Though this was not intentional, it does allow us to better rank any models that we make.

**3.2 Model Designing**

 The Deep Neural Network models5 from the original paper used various Tanh layers from PyLearn26 and ending with a Sigmoid function. Each of the Tanh layers is set up to renormalize data and is also where the number of hidden and visible features are set. In order to emulate this with pytorch, we used pairs of linear and activations layers. Initially we intended on using Tanh for the activation layers to resemble the original paper, however, we also wanted to look at other activation functions (ReLu and LeakyReLy). Furthermore, we were curious about the number of these layer pairs and the input and output sizes for each linear layer. At this time we also looked to tune the learning rate, the batch size, and the maximum epoch size. Initially we used Skorch, a PyTorch Wrapper for Sklearn, to run a random grid search cv on the hyperparameters. This, unfortunately, proved too much to run on a cpu, so we looked for ways to run the models on a GPU. We landed on Optuna, a hyperparameter optimization framework specifically for GPUs.

The best performing model from this was six linear layers witinput output sizes of 32>128>64>32>16>8>1 with LeakyReLu activation functions after each of the first five linear layers and a sigmoid function after the sixth layer, a maximum epoch of 250, a batch size of 200, and learning rate of about 0.001 (See Figure 2. This model did well in terms of accuracy (See Figure 3) and had an AUC of 0.85 (See Figure 4), putting it above all of the other machine learning methods and below the DNNs from the original paper (See Table 1). However, it was slightly more accurate for predicting Higgs Bosons than it was at predicting background signals. Looking at the other models that Optuna tested, we felt that the Tanh activation function was not adequately tested. The original paper also tested dropout, but only on the supersymmetry particle data. For those reasons, we tested three more models (See Supplementary Figures 7, 8, & 9 for training and validation curves).

**3.3 Additional Models**

**3.3.1 Threshold Change**

In an attempt to solve the accuracy imbalance between the Higgs Boson and Background Signal detection of the original model, we adjusted the threshold slightly. This slightly decreased the accuracy for Higgs Boson detection, but increased the accuracy for the background signal detection more (See Supplementary Figure 10). This resulted in an AUC increase to 0.86 (See Supplementary Figure 11)

**3.3.2 LeakyReLu with Dropout**

The next model we tested was the same as the original model, but with the addition of a dropout layer with a probability of 0.25. This model did not have a threshold change, but the Higgs Boson signal and background signal detection accuracies were the closest. The Higgs Boson accuracy was out performed by the original model and the original model with the threshold change, but the opposite is true with the background signals (See Supplementary Figure 12). The AUC for this model at 0.84 is slightly less than the previous two models (See Supplementary Figure 13). This leads us to believe that with further optimization, this model could be the best one.

**3.3.3 Tanh with Dropout**

The final model we tested included the same dropout layer as the last model, but we switched the LeakyReLu layers with Tanh layers. The original paper did not give a reason behind why Tanh was chosen and we wanted to check if our model’s performance could be improved using this Tanh. This model is about as accurate as the original model at classifying background signals, but less accurate at classifying Higgs Bosons (See Supplementary Figure 14). This model had the same AUC as the LeakyReLu with Dropout model at 0.84 (See Supplementary Figure 15), but due to its worse performance in terms of accuracy, we were less hopeful for its potential.

**3.4 Future Discussion**

Future improvements in DNN models hinge on several key enhancements. Tuning more hyperparameters such as the output sigmoid threshold and batch sizes. Raising computing power through more potent GPUs or CPUs to slash the current 20-hour hyperparameter tuning time and the 2+ hours per model fitting. Integrating parallel and distributed systems in a production environment to accelerate processing, while optimizing GPU coding efficiency to ensure maximized resource utilization. Additionally, the guidance of a domain-knowledge expert would become pivotal, as the nuanced insights could amplify the relevance and effectiveness of DNN models.

**4 Conclusion**

The exploration of DNN models in this project has shown how complex the area of model development and optimization is. From data preparation, baseline model creation, to the design and testing of various neural network architectures, the pursuit of accuracy and efficiency has been primary. While our models have shown promise, they still fall short compared to those in the original paper. This discrepancy emphasizes the need for further advancements. Future strides in DNN models should prioritize the tuning of additional hyperparameters or the guidance of a domain–knowledge expert.

**SUPPLEMENTARY FIGURES**

****

**SUPPLEMENTARY FIGURES CONTINUED**

****

****

**Supplementary Figures Continued**

****

**REFERENCES**

1. Wikipedia Contributors. “Standard Model.” Wikipedia, Wikimedia Foundation, 18 Sept. 2019, en.wikipedia.org/wiki/Standard\_Model.
2. Wikipedia Contributors. “Higgs Boson.” Wikipedia, Wikimedia Foundation, 3 Apr. 2019, en.wikipedia.org/wiki/Higgs\_boson.
3. Baldi, P., et al. “Searching for Exotic Particles in High-Energy Physics with Deep Learning.” Nature Communications, vol. 5, no. 1, July 2014, https://doi.org/10.1038/ncomms5308. Accessed 14 Apr. 2019.
4. Whiteson, Daniel. “UCI Machine Learning Repository.” Archive.ics.uci.edu, 2014, archive.ics.uci.edu/dataset/280/higgs. Accessed 16 Nov. 2023.
5. Baldi, P., et al. “Github for ‘Searching for Exotic Particles in High-Energy Physics with Deep Learning.’” GitHub, 6 Sept. 2023, github.com/uci-igb/higgs-susy/tree/master. Accessed 16 Nov. 2023.
6. Ian J. Goodfellow, David Warde-Farley, Pascal Lamblin, Vincent Dumoulin, Mehdi Mirza, Razvan Pascanu, James Bergstra, Frédéric Bastien, and Yoshua Bengio. "Pylearn2: a machine learning research library". arXiv preprint arXiv:1308.4214