Machine Learning Optimization of Di-Higgs Signal-Background Separation in Collider Physics

Santiago Ampudia Castelazo (SUID: ampudia)
Sheng Moua (SUID: smoua)
Shinnosuke Yagi (SUID: syagi2)
Department of Computer Science, Stanford University
CS229 – Machine Learning
March 17, 2025

1 Introduction

Future high-energy colliders aim to test the Standard Model (SM) and search for new physics with unprecedented precision. A key measurement is the di-Higgs production cross-section, probing Higgs self-coupling and deviations from SM predictions.

Accurate measurement requires effective signal-background separation. Traditional cut-based methods are suboptimal, making machine learning (ML) a powerful alternative. ML techniques have been widely used in high-energy physics, but the optimal ML approach remains unclear, with different architectures and tuning strategies leading to varying classification performances.

This work evaluates ML methods—Boosted Decision Trees (BDTs), Neural Networks (NNs), and stacking models for signal-background separation at the XFEL Compton Collider (XCC) for the $\gamma\gamma \to HH \to b\bar{b}b\bar{b}$ channel, where dominant backgrounds arise from the different interaction points. Optimizing these approaches enhances di-Higgs cross-section precision, strengthening XCC's and similar future colliders' physics potential.

2 Related Work

Barklow et al. in [3] discussed the challenges of background rejection at the XCC, reinforcing the need for advanced ML approaches. Various strategies have been explored for signal-background separation in Higgs self-coupling studies. Tian in [9] and Dürig in [7] independently trained separate BDTs for each background and manually optimized selection cuts at the ILC.

A more automated method was used in the XCC study in [2], where BDTGs were optimized using a genetic algorithm.

Stacking techniques were explored by Alves in [1], who combined multiple classifiers using a meta-learner to enhance Higgs identification at the LHC.

Our work builds on these efforts by systematically comparing multiple ML methodologies to determine the most effective approach for improving the precision of Higgs self-coupling measurements and similar studies at future colliders.

3 Dataset and Features

The dataset consists of Monte Carlo (MC) samples generated by Dr. Tim Barlow for XCC di-Higgs studies such as [2] and [4]. These samples include the signal, $\gamma\gamma \to HH \to b\bar{b}b\bar{b}$, and twelve backgrounds: $\gamma\gamma \to q\bar{q}$, $t\bar{t}$, ZZ, ZH, W^+W^-; $e\gamma \to q\bar{q}$, $q\bar{q}q\bar{q}$, $q\bar{q}H$; $e^+e^- \to b\bar{b}$, $b\bar{b}q\bar{q}$, ZH, $t\bar{t}$.

A fast detector simulation was performed using Delphes [7]. Then a preselection was applied to retain only relevant events, *i.e.*, those kinematically similar to the signal. Afterwards, relevant physical observables were reconstructed to serve as input features for machine learning models. The selected features include jet kinematics, angular variables, event shape characteristics, and reconstructed invariant masses. These features, as well as the preselection and detector simulation settings, were determined to be the most relevant for signal-background discrimination based on prior XCC studies ([2], [4]).

4 Procedure for Model Comparison

For the Higgs self-coupling at future colliders, the main point of comparison is the measurement of the di-Higgs production cross-section σ_{HH} at the XCC and its associated uncertainty. Following [9], a likelihood ratio test based on the chi-square (χ^2) function is used:

$$\chi^2 \equiv -2\ln\frac{L_{s+b}}{L_b}, \quad L_{s+b} = \prod_i \frac{e^{-(s_i+b_i)}(s_i+b_i)^{n_i}}{n_i!}, \quad L_b = \prod_i \frac{e^{-b_i}b_i^{n_i}}{n_i!} \tag{1}$$

where b_i is the expected background events, n_i is the number of observed events, and s_i is the unique parameter, which is related to σ_{HH} by:

$$s_i = \sigma_{HH} \cdot Lumi \cdot Eff_i \cdot BR_i, \tag{2}$$

where $Lumi$ is the 10-year luminosity at XCC 4900 fb^{-1}, Eff_i is the signal efficiency, and BR_i is the branching ratio $\gamma\gamma \to HH \to b\bar{b}b\bar{b}$ (0.5824^2).

Because of the direct relationship between σ_{HH} and the number of expected signal and background events, another commonly used metric is the signal significance, defined as:

$$S = \frac{S}{\sqrt{S+B}}, \tag{3}$$

where S and B are the expected signal and background event counts. This metric quantifies the distinguishability of the signal from background fluctuations, ensuring the robustness of our measurement. It is inversely proportional to the uncertainty of the measurement, and thus often used as point of comparison. We will use it here to optimize our models and then quantify and compare their performances.

5 Description of chosen ML methods

We test Boosted Decision Trees (BDTs) and eXtreme Gradient Boosting (XGB) with two optimization strategies: neural network (MLP) combination and genetic algorithm-based cut selection. Stacking is also evaluated.

BDTs use adaptive boosting to iteratively refine decision boundaries, while XGB improves this with gradient-based optimization and regularization. We train separate classifiers for each background versus the signal and use their outputs for further optimization.

To combine them, an MLP with two hidden layers takes BDT/XGB outputs as inputs, learning a non-linear decision boundary. It consists of fully connected layers with ReLU activation, trained via backpropagation to minimize cross-entropy loss. The optimal threshold is set to maximize significance.

Another approach is a genetic algorithm (GA), which automates threshold selection for BDT/XGB outputs. It evolves a population of candidate cuts through selection, crossover, and mutation, optimizing for maximum signal significance.

Stacking combines multiple (base) classifiers through a meta-model trained on their predictions. The meta-model is trained using cross-validation to prevent overfitting, and the final selection is done to maximize signal significance.

These methods are evaluated to identify the most effective strategy for maximizing significance and minimizing measurement uncertainty.

6 Training and application for BDTs and XGBs

We trained Boosted Decision Trees (BDTs) using `AdaBoostClassifier` with `DecisionTreeClassifier` as the base model. XGBoost (`xgb.XGBClassifier`) was also trained. The datasets were preprocessed by splitting into training, validation, and test sets using `train_test_split`, ensuring stratification to maintain class balance. Cross-validation for 10 trials and reweighting were also performed. The following hyperparameters were used: **BDT:** *Estimators*: 100; *Learning rate*: 0.1; *Max depth*: 3. **XGBoost:** *Estimators*: 100; *Learning rate*: 0.1; *Max depth*: 2.

6.1 NN to combine outputs

A multi-layer perceptron (MLP) neural network was trained to combine the outputs of BDT and XGB classifiers. The NN consists of two fully connected layers; *Input:* outputs from BDT and XGB; *Hidden layers:* 64 and 32 neurons, with ReLU activation. *Output:* sigmoid activation for binary classification.

Classification treshold was then placed to maximize (3). Results for are found in 1.

6.2 Genetic algorithm to combine outputs

A genetic algorithm (GA) was implemented to optimize classification thresholds for BDT and XGB outputs, maximizing (3). The GA evolved a population of 12-dimensional threshold vectors, where each element corresponded to a cutoff value for one of the 12 background models.

The fitness function was defined as (3), computed by applying the candidate threshold vector to classifier outputs. The GA selected individuals based on tournament selection and iteratively improved thresholding through crossover and mutation. Key GA settings determined given the size and features of input dataset: *Population size*: 100; *Number of generations*: 300; *Crossover rate*: 0.5; *Mutation rate*: 0.2; *Selection method*: Tournament selection (size = 3); *Crossover method*: Blend crossover ($\alpha = 0.5$); *Mutation*: Gaussian mutation with mean=0 and standard deviation=0.1; *Convergence criterion*: Stops early if no improvement in significance for 20 consecutive generations.

To improve optimization stability, a spread control mechanism was applied. If the number of individuals showing improvement fell below a threshold, the mutation rate was reduced to prevent excessive exploration; if frequent, increased to accelerate convergence.

Separate GAs were trained for BDT and XGB outputs. The best individuals from each GA run for (3) maximization were extracted as the final classification thresholds. Results for are found in 1.

Table 1: NN and GA models results after 10 trials.

	BTD Threshold	**BTD Significance**	**XGB Threshold**	**XGB Significance**
NN Average	N/A	8.4631	N/A	9.5802
NN Variance	N/A	0.1264	N/A	0.0705
Average	0.86	8.007	0.901	8.008
Variance	0.00266	0.40115	0.00311	0.18733

7 Stacking

On the same data, we trained four base models selected for their diversity: XGBoost (strong performance on structured data), LightGBM (fast and efficient), LightGBM Random Forest model (reduced variance and improved generalization),

and MLP (non-linear relationships captured). We optimized the hyperparameters for each one separately using the optimization framework *Optuna* to maximize (3).

The chosen hyperparameters account for class imbalance and overfitting prevention, since signals are rare among background events. Specifically, we used L2 regularization in XGBoost and LightGBM, dropout layers in the MLP, and bagging-based regularization in the LightGBM Random Forest model. The decision tree models were also trained on subsamples of the training data and features (*subsample = 0.7, subsample = 0.799, feature_fraction = 0.865*) to improve generalization. Key hyperparameters optimized for each model: **XGBoost**: *Subsample*: 0.7; *Scale_pos_weight*: 3.648; *Reg_lambda*: 10; *N_estimators*: 300; *Max_depth*: 7; *Learning_rate*: 0.1; *Gamma*: 0; *Colsample_bytree*: 0.7. **LightGBM**: *N_estimators*: 174; *Max_depth*: 0; *Learning_rate*: 0.0823; *Num_leaves*: 69; *Colsample_bytree*: 0.894; *Subsample*: 0.799; *Reg_lambda*: 7.77; *Boosting_type*: gbdt. **LightGBM (RF)**: *N_estimators*: 163; *Max_depth*: -1; *Num_leaves*: 100; *Feature_fraction*: 0.865; *Bagging_fraction*: 0.523; *Bagging_freq*: 8; *Learning_rate*: 0.0421; *Reg_lambda*: 0.435. **MLP**: *Hidden1*: 317; *Hidden2*: 122; *Alpha*: 3.07e-5; *Learning_rate*: 0.00084.

For the meta-model, we evaluated a Logistic Regression classifier and an XGBoost classifier and chose to use the XGBoost classifier as the meta-model since it maximized (3). We additionally tested different combinations of base models and concluded that using all four yielded the best performance. Using the base models' predictions as inputs, the meta-model was trained, and a probability threshold that maximized (3) was chosen. Results in 2.

Table 2: Results for Base Models and Final Stacking Model after 10 trials

	XGBoost	**LightGBM**	**LightGBM (RF)**	**MLP**	**Final Stacking Model**
Average significance	7.5244	7.9323	6.1561	6.7131	8.1269
Variance	0.0951	0.1221	0.0668	0.0776	0.0223

8 Conclusion and Future Work

The GA with XGB achieved the highest significance. This superiority stems from XGBoost's tree-based learning, through gradient boosting and regularization, effectively captures feature interactions and handles class imbalance. The GA's adaptive threshold optimization accounts for the combination of models train and tested on completely different event topologies. All performance-related plots can be found in [10].

Future work could explore different architectures for BDT combination or stacking base models, deeper hyperparameter tuning, and more advanced meta-learning strategies such as transformer-based approaches.

9 Contributions

Santiago: obtained, cleaned and preprocessed the datasets; researched how to measure di-Higgs self-coupling; derived the optimization parameter for the models and the comparison method. Wrote and formatted the paper. Shinnosuke: Trained, optimized, and tested the approaches involving using GAs and NNs to combine the outputs of BDTs. Delivered the results for comparison. Sheng: Trained, optimized, and tested the stacking approach. Delievered the results for comparison. The three of us did extensive research on several methods and hyperparameters in order to come up with the scope of the paper.

References

[1] Alves, A. *Stacking machine learning classifiers to identify higgs bosons at the LHC.* Journal of Instrumentation, , vol. 12, no. 05, 30 May 2017.

[2] Ampudia, S. et al. *Higgs Self-Coupling Measurement with the XCC (XFEL Compton Collider) Concept.* Higgs STudies I APS Global Physics Summit 2025, March 2025.

[3] Barklow, T. et al. *XCC: An X-ray FEL-based $\gamma\gamma$ Compton collider Higgs factory.* Journal of Instrumentation, DOI: 10.1088/1748-0221/18/07/p07028.

[4] Barklow, T. et al. *XFEL Compton Collider (XCC) $\gamma\gamma$ Higgs Factory.* LCWS 2024, July 2024.

[5] Celik, A. *Deep Learning Approaches for BSM physics: Evaluating DNN and GNN performance in particle collision event classification.* Acta Physica Polonica B, DOI: 10.5506/aphyspolb.55.10-a2.

[6] de Favereau, J. et al. *"DELPHES 3: A modular framework for fast simulation of a generic collider experiment.* arXiv: 1307.6346v3, March 2014.

[7] Dürig, C. F. *Measuring the Higgs Self-coupling at the International Linear Collider.* DESY-THESIS-2016-027. DOI: 10.3204/PUBDB-2016-04283. October 2016.

[8] Particle Physics Project Prioritization Panel. *Pathways to Innovation and Discovery in Particle Physics: Report of the 2023 Particle Physics Project Prioritization Panel.* U.S. Department of Energy. December 2023.

[9] Tian, J. *Study of Higgs self-coupling at the ILC based on the full detector simulation at 500 GeV and 1 TeV.* Helmholtz Alliance Linear Collider Forum: Proceedings of the Workshops Hamburg, Munich, Hamburg 2010-2012, Germany, January 2013.

[10] Performance-Related Plots. Available at: Google Drive.