Ensemble Pruning for OOD Generalization

Scott Merrill
Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC 27514
smerrill@cs.unc.edu

Abstract

Out-of-distribution (OOD) generalization presents a significant challenge in machine learning, especially for tabular data, which is prevalent in critical domains like healthcare and finance. A key obstacle in improving OOD performance is the lack of reliable proxy measures for OOD behavior. While in-distribution (ID) metrics, such as accuracy, are commonly used for model selection, they often fail to correlate well with OOD performance. In this work, we introduce a systematic framework for identifying OOD proxy metrics that can be utilized for both model calibration and selection across a wide range of downstream tasks. We demonstrate how these OOD proxies can be integrated into an evolutionary ensemble pruning algorithm. Our experiments show that optimizing for validation AUC leads to overfitting to ID data, resulting in suboptimal performance in real-world settings. In contrast, optimizing for our proposed OOD metrics results in ensemble models that are more robust to distribution shifts. These findings highlight the need for OOD-aware model selection strategies and provide practical insights for building models that perform well across environments.

1 Introduction

Machine learning (ML) models are trained under the assumption that the training and testing data follow the same distribution. When this assumption is violated, model predictions may become unreliable. OOD generalization has emerged as a critical challenge in ML, and while it has been extensively studied in the contexts of vision and language models, it remains less understood in the domain of tabular data. This gap is especially concerning given the widespread use of tabular data in critical fields such as healthcare, finance, and social sciences.

In ML, a widely used strategy to boost performance is ensembling multiple predictors. By combining models that are both accurate and diverse, ensembling can reduce individual model errors and improve overall predictive performance [8]. However, including too many predictors can backfire, especially if the models are highly correlated or lack sufficient accuracy. To tackle this, ensemble pruning algorithms have been developed to identify and select the most effective subset of models from a larger pool. While ensemble pruning might initially appear to be a niche problem, its significance is rapidly growing. With the increasing availability of AI models on platforms like HuggingFace, practitioners now have access to millions of models and are presented with the challenge of choosing the optimal model combinations. The naive approach of combining all available models is not only computationally infeasible but may fail to deliver the best results. Beyond improving ID and OOD performance, ensemble pruning serves as a powerful regularizer that reduces storage requirements and model complexity while simultaneously enhancing inference speed and interpretability.

Ensemble pruning remains an underexplored area, especially when it comes to OOD generalization. Our research aims to address this gap by investigating metrics that could serve as reliable proxies for OOD performance. Previous studies have highlighted that ID accuracy often fails to correlate

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

with OOD performance and, in some cases, may even exhibit negative correlations [12]. Identifying robust and consistent OOD proxies is crucial as they can be leveraged across various applications. At its simplest, such proxies can guide model selection. Rather than relying on validation accuracy, as is common in practice, models could be chosen based on their performance with respect to an OOD proxy. With a well-designed proxy, this approach ensures the selection of models that are inherently more robust to distribution shifts.

In our work, we first identify informative OOD proxies and then incorporate them as fitness functions within an evolutionary ensemble search algorithm. Our main contributions are as follows:

- Propose a scalable framework for efficiently evaluating a wide range of OOD proxy metrics.
- Establish a systematic methodology to filter and identify high-performing proxy measures.
- Demonstrate the practical value of these measures by leveraging them in an evolutionary algorithm to optimize ensemble performance.

2 Related Work

Ensemble methods such as bagging, boosting, and stacking have long been recognized for their ability to improve generalization in ML. [3] provides a comprehensive overview of these methods, emphasizing their ability to reduce both variance and bias in predictions. Several studies have applied similar techniques to the task of OOD generalization. Much of the existing work in ensembles for OOD generalization focus on training a diverse set of classifiers for predictions on image-based data. [11] for example proposed a method that identifies "hard" training examples and encourages ensemble predictors to disagree on these examples. There have also been some exciting developments in the new field of weight-space ensembles. [14] demonstrated that averaging the weights of multiple fine-tuned models can improve OOD accuracy. To the best of our knowledge there is only one work that investigates ensemble pruning for improving OOD generalization. Similar to the aforementioned studies, it primarily focuses on image data. Specifically, [9] introduce a quadratic program to balance accuracy and diversity in ensemble construction. Although their approach faces scalability challenges, it underscores the importance of both accuracy and diversity for effective OOD ensembles.

3 Identifying Effective OOD Proxies

We organize our approach into two stages. First, we develop a systematic methodology to identify fitness metrics that are robust across diverse datasets and distributions and are strongly correlated with OOD accuracy. Second, we propose a simple evolutionary algorithm to optimize the identified proxies.

3.1 Metrics Evaluated

A common baseline proxy for OOD performance is the in-distribution (ID) validation accuracy. However, this is insufficient, as ID and OOD accuracies are not always strongly correlated [12]. To address this, we conduct a large-scale evaluation of over 10,000 distinct fitness metrics, categorized as follows:

1. **Accuracy-Based Metrics:** Evaluate ensemble performance on training or validation datasets.
2. **Diversity-Based Metrics:** Summarize ensemble diversity by analyzing the prediction vectors of individual models.
3. **Hard Negative Mining Metrics:** Measure an ensemble's ability to handle difficult examples, such as frequently misclassified points, emphasizing robustness on edge cases.
4. **Robustness Measures:** Assess performance on perturbed versions of training/validation data, inspired by research linking noise robustness to distribution shift robustness [13].
5. **Clustering-Based Metrics:** Group data using clustering methods (e.g., KMeans, DBSCAN, environmental partitioning [6, 2]), then compute metrics (1–3) within clusters. These metrics align closely with Distributionally Robust Optimization (DRO) principles which seek to optimize worst-case cluster performance [10].

6. **Outlier-Based Metrics:** Use outlier detection methods (e.g., Isolation Forests, Local Outlier Factor [7, 1]), or logistic regression to identify points far from decision boundaries. Metrics (1–3) are then computed for these outlier points, based on insights from OOD generalization research.

A detailed breakdown of these metrics is presented in Table 1.

Category	Metrics
Accuracy Metrics	Accuracy, Precision, Recall, AUC, F1, Expected Calibration Error, Class Balanced Cross Entropy Loss
Diversity Metrics	Average model agreement, Average brier score, Average Cosine Similarity, Average Kl divergence, Average Hamming Distance, Average Q statistic, Average correlation coefficient
Hard Negative Mining Metrics	Ensemble Focal Loss, Average Model Focal Loss, Uncertainty weighted log loss, Online Hard Example Mining Loss (OHEM)
Robustness Measures	Consistency Loss, Previous Metrics with Gaussian noise, Previous Metrics with Uniform noise, Previous Metrics with Laplace noise, Previous Metrics with Data Corruptions
Clustering Methods	Kmeans, DBSCAN, Decorr
Outlier-Based Metrics	Isolation Forests, Local Outlier Factor, Logistic Regression Outliers, Importance Based Sampling

Table 1: Overview of Fitness Functions by Category

3.2 Large-Scale Metric Evaluation

To test the effectiveness of these metris, we randomly construct 10,000 ensembles of size 100 from a fixed pool of models. For each ensemble, we compute the fitness metrics described above alongside the true OOD metrics we aim to optimize. To filter the best-performing metrics, leverage three data mining metrics:

1. **Correlation Measures:** Both Pearson and Spearman correlation coefficients are computed between each fitness function and true OOD metrics to identify metrics with strong linear or rank-based correlations.
2. **Precision@k:** This metric evaluates how often high-ranking ensembles based on a fitness metric overlap with the top k ensembles by true OOD performance:
$$\text{Precision@}k = \frac{|\{e \in \text{Top-}k(f) \cap \text{Top-}k(\text{true OOD})\}|}{k}$$
3. **Percentile@k:** This metric assesses the average percentile rank of the top k OOD ensembles based on a fitness metric:
$$\text{Percentile@}k = \frac{1}{k}\sum_{i=1}^{k}\text{Percentile}(e_i)$$

We hypothesize that metrics with high Precision@k, Percentile@k, or correlation with the true measure are strong candidates for optimization. However, for evolutionary algorithms, Precision@k and Percentile@k are more relevant because they prioritize the correct ranking of top solutions rather than exact linear relationships. While correlation coefficients capture global trends, evolutionary optimization focuses on the local selection of top models, as only these are chosen for reproduction. For an evolutionary algorithm to be effective, the critical requirement is that high-ranking fitness functions align with high-ranking OOD metrics. As such, a global metric like correlation is unnecessary.

3.3 Combining Fitness Metrics

Many ensemble pruning methods combine both an accuracy and a diversity metric. We similarly attempt to identify effective combinations of fitness metrics by fitting a ridge regression models to predict OOD performance (e.g., OOD AUC) based on pairs of fitness metrics. Ridge regression regularizes the coefficients, preventing overreliance on any single metric. We evaluate combinations using metrics such as Mean Squared Error (MSE) and r-squared to select pairs that exhibit strong predictive power.

4 Optimizing OOD Proxies

4.1 Optimization Framework

After identifying the most promising fitness proxies, we optimize them using a lightweight evolutionary algorithm. Evolutionary algorithms are well-suited for this task due to their gradient-free nature

and ability to simultaneously explore a diverse population of solutions. This flexibility is essential for ensemble optimization, where the inputs are discrete and the objectives may be complex or discontinuous. An evolutionary algorithm operates through a genetic encoding and a set of operations applied to that encoding. In the following subsection, we detail how we encode ensembles and define the mutation and crossover operations for ensemble optimization.

4.2 Genetic Encoding and Operations

Genetic Encoding: The ensembles we optimize are represented as binary arrays, where each element corresponds to a model in the fixed pool of N models (e.g., $N = 10,000$). The encoding is defined as follows:

$$\mathbf{g} = [g_1, g_2, \ldots, g_N] \quad \text{where } g_i \in \{0, 1\}, \quad \sum_{i=1}^{N} g_i = K$$

Here: - $g_i = 1$ if model i is included in the ensemble. - $g_i = 0$ if model i is excluded. - K is the fixed size of the ensemble (e.g., $K = 100$).

Mutation Operation: We define a mutation operation that randomly replaces one model in the ensemble with a model from the model pool. Let $\mathbf{g}$ be the current encoding. The mutation operation generates a new encoding $\mathbf{g}'$ as follows:

$$\mathbf{g}' = \mathbf{g} + \Delta$$

Where Δ is a perturbation satisfying: - $\Delta = -1$ at one randomly selected position i where $g_i = 1$. - $\Delta = +1$ at another randomly selected position j where $g_j = 0$.

Crossover Operation: Crossover combines two parent ensembles, $\mathbf{g}^{(1)}$ and $\mathbf{g}^{(2)}$, to produce two child ensembles. Let $S^{(1)}$ and $S^{(2)}$ denote the sets of indices corresponding to $g_i^{(1)} = 1$ and $g_i^{(2)} = 1$, respectively. The combined set of models is $S = S^{(1)} \cup S^{(2)}$.

To generate child ensembles, we randomly partition S into two subsets S_{child1} and S_{child2}, each of size K. The resulting child encodings are:

$$g_i^{\text{child1}} = \begin{cases} 1 & \text{if } i \in S_{\text{child1}} \\ 0 & \text{otherwise} \end{cases}, \quad g_i^{\text{child2}} = \begin{cases} 1 & \text{if } i \in S_{\text{child2}} \\ 0 & \text{otherwise} \end{cases}$$

5 Experimental Setup

For our experiments, we utilize benchmark datasets from TableShift [5]. Such include a comprehensive collection of real-world OOD splits spanning diverse applications and incorporating both covariate and label shifts. To simplify the analysis while maintaining representativeness, we focus on four datasets from TableShift that are smaller in size yet capture the complexity of OOD generalization. Below, we briefly summarize each dataset; for detailed descriptions, we refer readers to the original TableShift paper [5].

5.1 Datasets

1. **ANES Voter Participation:** This dataset, derived from the American National Election Studies (ANES), examines voter participation in U.S. presidential elections. The task is to predict whether an individual will vote based on social and political attitudes, opinions, and media habits. A domain split by geographic region simulates challenges in modeling voter behavior across diverse regions, reflecting the complexities of domain shifts in voter turnout prediction. The dataset contains 365 categorical features, with a training class balance of 30% positive and 70% negative samples. The OOD test set has a class balance of 41% positive and 59% negative.

2. **BRFSS Hypertension Prediction:** Hypertension is a leading risk factor for cardiovascular diseases, affecting nearly half of Americans. This dataset, constructed using BRFSS survey data, predicts hypertension risk based on factors such as age, genetics, and socioeconomic status. Domain shifts are introduced by varying the body mass index (BMI) distributions, highlighting the challenges of deploying models across populations with different BMI profiles. The dataset includes 99 categorical features and 1 continuous feature. The class balance is consistent across both the training and OOD test sets, with a 59% positive and 41% negative split.
3. **BRFSS Diabetes Prediction:** Diabetes affects over 37 million Americans and incurs significant health and economic costs. Early detection is crucial for effective intervention and care. This dataset, also constructed using BRFSS survey data, predicts diabetes risk based on physical health, lifestyle, and demographic factors. A domain shift is introduced by partitioning the dataset based on race/ethnicity to reflect disparities in diabetes risk modeling between White non-Hispanic individuals and other racial/ethnic groups. The dataset contains 138 categorical features and 4 continuous features. The training set has a class balance of 87% positive and 13% negative, while the OOD test set has a class balance of 83% positive and 17% negative.
4. **Diabetes Readmission Prediction:** Frequent hospital readmissions for diabetic patients indicate unresolved health issues or inadequate treatment, posing significant challenges for healthcare systems. This dataset predicts 30-day readmissions using clinical data from 10 years of U.S. medical records. Domain shifts are introduced by isolating "emergency room" admissions as a distinct and challenging OOD subgroup. The dataset includes 175 categorical features and 8 continuous features. The training class balance is 58% positive and 42% negative, while the OOD test set has an approximately even class balance of 50% positive and 50% negative.

5.2 Model Pool Construction

For each dataset, we pre-train a pool of 10,000 decision trees. Each tree is trained on a distinct subset of the data, using a random selection of features. We apply constraints on the maximum proportion of features and data that each tree can use. Specifically, each tree is trained on a random subset of at most 30% of the available features and no more than 30% of the training data. These constraints help ensure diversity in the model pool and that the models capture different aspects of the data. Further variations in model pool parameters are explored in the ablation studies.

5.3 Evolutionary Algorithm Details

We optimize the identified fitness proxies using an evolutionary algorithm. The process starts with a population of 200 randomly generated ensembles. Each ensemble is evaluated based on the selected fitness metric. We then use tournament selection to pick the best-performing ensembles for reproduction. Selected ensembles undergo crossover, where subsets of models are exchanged between them to create new offspring. To introduce further diversity, mutation is applied with a probability of 0.1, randomly replacing a model in an ensemble with another from the pool. This process is repeated for 1,000 generations. The final output is a single ensemble optimized for robustness to distribution shifts.

5.4 Comparison Benchmarks

To evaluate our approach, we compare the optimized ensembles against the following benchmarks:

1. **XGBoost and CatBoost Classifiers:** As reported in the TableShift paper, these models consistently demonstrate strong OOD performance across all datasets.
2. **Model Pool Baseline:** The performance of the entire model pool is used as a reference. This baseline reflects the lottery ticket hypothesis that a sub-ensemble may be able to outperform a combination of all models [4].
3. **Validation AUC Baseline:** We apply the same evolutionary algorithm to optimize validation AUC. Validation AUC is widely used as a for model calibration and selection. Thus, it

serves as a realistic baseline a practioner who isn't concerned about distribution shifts may use.

6 Experimental Results

In Table 2, we present the five fitness functions with the highest precision@25, while Table 3 lists the top five fitness functions based on percentile@25. Finally, Table 4 shows the results of ridge regression, highlighting the best-performing combinations of two fitness functions. As you may have noticed, our fitness functions are labeled with seemingly arbitrary names. This encoding scheme was essential due to the evaluation of over 10,000 fitness functions, which would have resulted in excessively long names and potential confusion between similar ones. The naming scheme, provides a concise summary of the fitness functions and their respective parameters. In Table 7 in the appendix we provide a brief description of all of the fitness function names used in this paper.

Fitness Function	Pearson Corr	Spearman Corr	Precision 5	Percentile 5	Precision 10	Percentile 10	Precision 25	Percentile 25
10_uwl25_std	0.032962	0.03867	0	0.532493	0.025	0.547884	0.12	0.487272
5_flm5_std	0.002183	0.001407	0.05	0.483483	0.05	0.516409	0.11	0.531997
10_fle3_mean	-0.113903	-0.120179	0	0.325957	0	0.287544	0.09	0.327761
5_ece10_mean	-0.079327	-0.071826	0.1	0.366096	0.05	0.430697	0.09	0.471016
5_fle4_std	0.065773	0.055734	0	0.600317	0.025	0.632382	0.09	0.607983

Table 2: Top Fitness Functions By Precision@25

Fitness Function	Pearson Corr	Spearman Corr	Precision 5	Percentile 5	Precision 10	Percentile 10	Precision 25	Percentile 25
ec_std	0.163034	0.160612	0	0.677033	0	0.718728	0.05	0.681221
ec_mean	0.112251	0.115132	0	0.683453	0	0.719029	0.06	0.677861
10_ec_mean	0.099956	0.108435	0	0.665748	0.025	0.701601	0.06	0.671489
uwl1	0.180362	0.169633	0.05	0.733115	0.05	0.69646	0.07	0.666719
5_ec_mean	0.100303	0.104654	0	0.67316	0.025	0.69807	0.08	0.659497

Table 3: Top Fitness Functions By Percentile@25

Feature 1	Feature 2	Beta 1	Beta 2	R2 Score	MSE
bse	cbcel	-0.776651	-0.223349	0.143167	0.000346
cbcel	uwl1	-0.306398	0.693602	0.140267	0.000346
bse	ece25	-0.581776	0.418224	0.123703	0.000345
bse	ece10	-0.586858	0.413142	0.117072	0.000349
cbcel	uwll5	-0.340557	0.659443	0.116944	0.000357
bse	ece5	-0.611136	0.388864	0.11433	0.000354
ece25	uwl1	0.422093	0.577907	0.106959	0.000352
ece10	uwl1	0.417433	0.582567	0.102184	0.000355
ece5	uwl1	0.39367	0.60633	0.100968	0.00036
bse	ec_std	-0.26212	0.73788	0.090321	0.000371

Table 4: Top Fitness Function Combinations

We begin by diagnosing our method for identifying strong single-metric fitness functions. Table 3 shows that the top fitness function by precision@25 is 10_uwl25_std, which corresponds to the uncertainty-weighted log loss (with the specific parameters provided in the appendix). The 12% precision@25 indicates that there is a 12% overlap between the top 25 models based on OOD AUC and the top 25 models identified by 10_uwl25_std. Given that our random search involved 10,000 ensembles, a 12% overlap between the highest OOD AUC models and those with the highest 10_uwl25_std is quite promising. It is important to note that this 12% overlap is averaged across all datasets. Some datasets exhibited a larger precision@25, but this was averaged out with datasets that had a lower precision@25. In future work, we plan to investigate whether certain dataset characteristics, such as the number of features or the type of feature shift, are linked to higher precision@25 values. Understanding these relationships could help us recommend when specific fitness functions might outperform others.

To further evaluate 10_uwl25_std, we plot the OOD precision-recall curve for each of the four datasets in Figure 1. In each subplot, the red line represents the ensemble formed by the entire model pool,

the colored lines show the three models with the highest 10_uwl25_std, and the grey lines correspond to the other ensembles from the search. On the Anes and Diabetes Readmission datasets, we observe that very few models outperform the model pool. However, when the model pool is beatable—i.e., when there are several grey ensembles outperforming it—10_uwl25_std is able to effectively identify these models. On the BRFSS Diabetes and BRFSS Blood Pressure datasets, the top three models with the highest 10_uwl25_std also exhibit OOD performance exceeding that of the model pool, making them among the best found during the random search. In future work, we plan to investigate what makes some model pools more or less difficult to outperform.

In Figure 2, we plot 10_uwl25_std on the x-axis against OOD AUC at Recall 0.2 on the y-axis. The blue points represent individual models evaluated during the random search. The red points mark the 10 ensembles with the highest 10_uwl25_std, while the orange points highlight the 10 ensembles with the top OOD performance. Although the relationship is not strictly linear, we observe that ensembles with the highest fitness scores are effective at filtering out poor-performing models. The 10 ensembles with the highest fitness generally exhibit an AUPRC@Recall0.2 that is above the population mean. This observation suggests that an evolutionary algorithm could successfully discover high-performing ensembles.

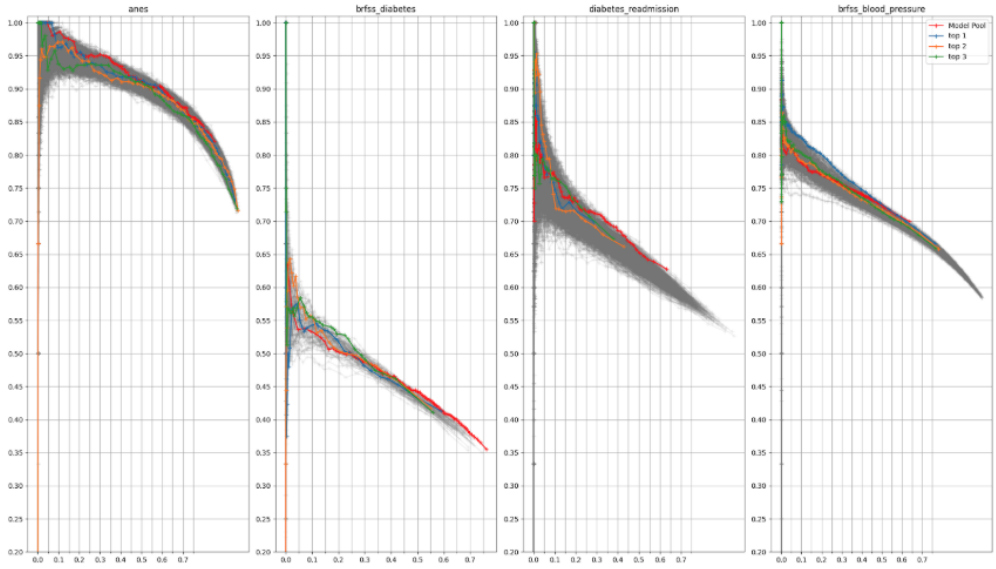

Figure 1: Random Search Results

We also show in Table 4 the top 10 combination of fitness functions of two variables. From the table, we note a steep drop off in the r-squared. The model with the highest r-squared for example has an r-squared value that is more than 55% larger than the model with the 10th highest r-squared. This fact, coupled with the low r-squared values across the board indicates the relationship between OOD AUC and our fitness functions is not linear. Perhaps there is a predictive non-linear relationship between the variables however. Or perhaps, there are some linear relationships within specific datasets, but not a universally linear relationship across all datasets.

Though, as discussed previously, a strictly linear relationship between our proxy metric and OOD performance is not required for an evolutionary algorithm to achieve successful results. We now present the outcomes of the evolutionary algorithm. While we ran the algorithm on the top 10 single-metric fitness functions from Table 2 and Table 3, we only show the results for 6 metrics due to space constraints. Similarly, we show 6 composite metrics from Table 4. In Table 5, we display the OOD AUC for the ensembles resulting from optimizing these metrics across all datasets. The highest OOD AUC in each column is highlighted in bold, and rows outperforming the validation AUC Baseline are indicated in red text. Models that outperform the model pool are highlighted in yellow.

From Table 5, we observe that both XGBoost and CatBoost perform exceptionally well and are difficult to surpass. While these models are often used as benchmarks, it is more appropriate to refer

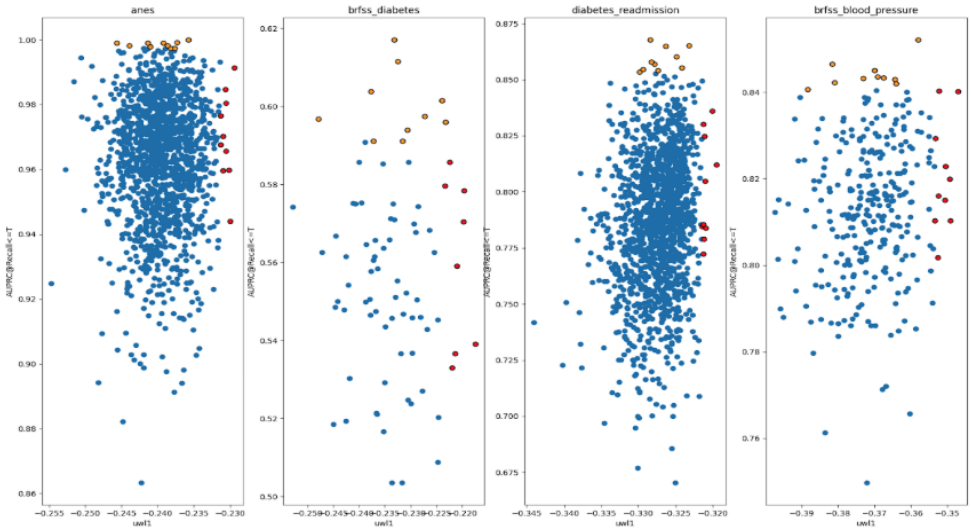

Figure 2: Fitness Function Diagnostic

to them as the gold standard outputs, as they are trained on the entire training dataset, whereas our models are constrained to selecting 100 models from a predefined pool of 10,000 models. Thus, a more fair comparison is with models optimized for validation AUC - validation AUC is used ubiquitously for model selection and provides a realistic alternative to our selection criteria.

When comparing to the validation AUC baseline, we consistently find that our models can identify ensembles with better OOD performance. This is likely explained by Table 6 in the appendix, which shows ID performance for each model. While validation AUC strongly outperforms our models on ID data, our OOD results suggest that optimizing for validation AUC leads to models that overfit to ID data. The result are models that are less robust to distribution shifts.

In Figure 3, we present the OOD precision-recall curves for the final ensembles found during the evolutionary search. To simplify the comparison, we focus on 3 of our fitness functions and compare them to the validation AUC baseline depicted in black. Once again, we see that our fitness functions can identify ensembles that are more robust to OOD shifts than those optimized for validation AUC. In some cases—such as with the Anes and Diabetes Readmission datasets—our ensembles perform significantly better.

When compared to the model pool, we also find that our ensembles occasionally outperform the entire pool's performance, which is surprising given that our ensembles are only 1/100th the size. However, this outperformance is less consistent. On some datasets such as BRFSS Blood Pressure, outperforming the model pool was relatively easy, with many of our models achieving this. In contrast, on other datasets such as the BRFSS Diabetes, none of our models were able to outperform the pool. This discrepancy presents an intriguing area for future work. We aim to understand why certain model pools are more difficult to outperform. Is this due to the models within the pool, how they are trained, or is it related to the characteristics of the underlying dataset? Gaining insights into the factors that contribute to a model pool's OOD performance could help identify which fitness functions are likely to perform better in different scenarios.

7 Analysis

We tested our proposed method across various scenarios with different parameter settings. One key experiment involved testing the effect of ensemble size. Unsurprisingly, we found that larger ensembles generally outperformed smaller ones. Specifically, ensembles of size 250 were more likely to outperform the entire model pool compared to those of size 100. While we found that larger ensembles tend to perform better on average, we also point out that there exists an optimal ensemble size and this need not be the largest size ensemble. This hypothesis is verified by our finding of many

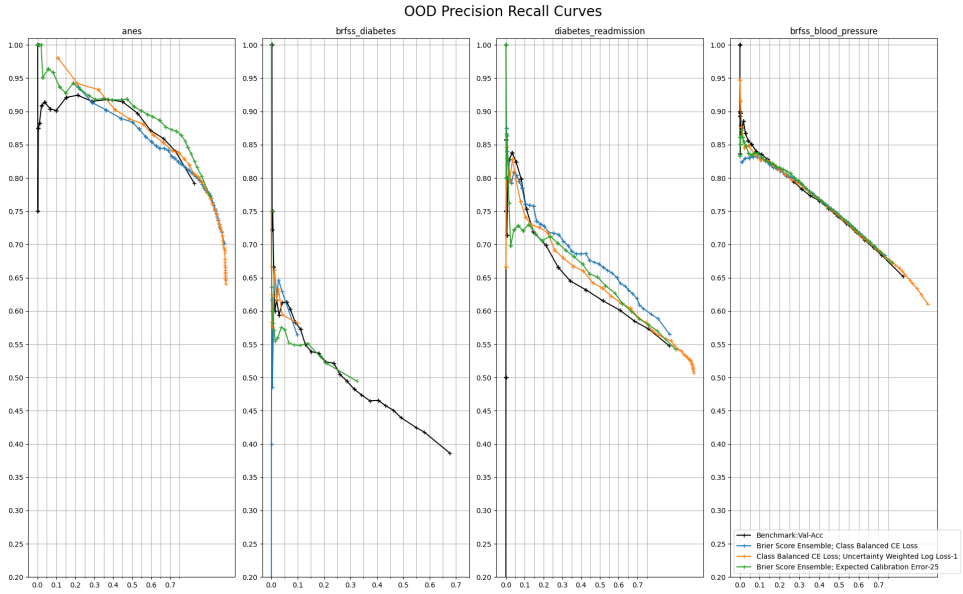

Figure 3: OOD Precision Recall of Final Ensembles

Model	Anes	brfss_diabetes	diabetes_readmission	brfss_blood_pressure
Baselines				
CatBoost	0.8854	**0.8131**	**0.6846**	**0.6886**
XGBoost	**0.8870**	0.8127	0.6850	0.6882
ModelPool	0.8755	0.8065	0.6608	0.6699
Validation AUC	0.8514	0.8015	0.6422	0.6801
Our Models				
Uncertainty Weighted Log Loss-5 (10)	0.8640	0.7964	0.6207	0.6296
Brier Score Ensemble; Class Balanced CE Loss	0.8550	0.8028	0.6765	0.6847
Class Balanced CE Loss; Uncertainty Weighted Log Loss-1	0.8571	0.8036	0.6506	0.6836
Brier Score Ensemble; Expected Calibration Error-25	0.8703	0.8017	0.6507	0.6841
Brier Score Ensemble; Expected Calibration Error-10	0.8744	0.8024	0.6589	0.6847
Class Balanced CE Loss; Uncertainty Weighted Log Loss-5	0.8773	0.7881	0.6652	0.6858
Brier Score Ensemble; Expected Calibration Error-5	0.8709	0.8037	0.6642	0.6858

Table 5: OOD AUCs for all datasets

ensembles of size 100 that can outperform a much larger model pool. In general however, larger ensembles seem to perform better.

We also analyzed the impact of the model pool itself. As previously mentioned, we parameterized the model pool construction with a max feature fraction and max dataset fraction. Specifically, we created model pools for all combinations of 0.3, 0.5, and 0.7 for these two variables, resulting in nine distinct model pools. In each model pool, we then used our evolutionary algorithm to optimize our top 20 fitness functions and the validation AUC baseline. Figure 4 presents the OOD precision-recall curves for all nine model pool configurations, using the Anes dataset as an example. For simplicity, we compare three of our models to the validation AUC baseline shown in black. In every model pool, our fitness metrics outperform the validation AUC optimization. Notably, the level of out-performance increases with the fraction of data and features used in the model pool; this out-performance was greatest when the max feature fraction and max dataset fraction are set to 0.7. We speculate that when the validation AUC is the optimzation objective, more more data and features increases the likelihood of overfitting to the ID distribution. In contrast, our fitness metrics produce consistent results across model pool configuration highlighting their robustness data distribution changes.

8 Conclusion

In this work, we present a systematic approach for identifying fitness functions that perform well OOD and demonstrate how a simple evolutionary algorithm can optimize these functions. We compare our method to the commonly used baseline of validation AUC and reveal a critical issue:

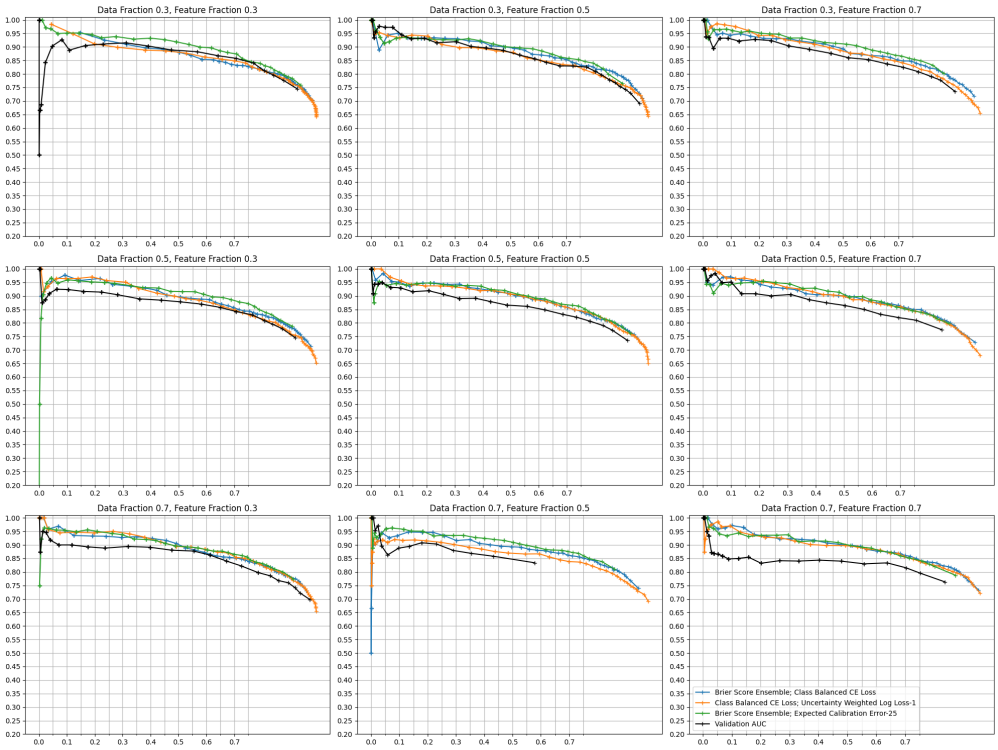

Figure 4: Effect of Model Pool Construction Anes Dataset

while validation AUC selects models that perform well on the same distribution, it fails to generalize to unseen data. Our experiments show that optimizing for validation AUC overfits to ID data and often leads to poor performance on real-world datasets. This indicates that relying on validation AUC for model selection may cause practitioners to unknowingly overfit to the training distribution, which may not reflect the actual conditions encountered when models are deployed in real-world settings.

Practitioners may not anticipate these distribution shifts during model development, particularly when the training and validation data appear to come from the same distribution. However, in many real-world scenarios, distribution shifts are inevitable once the model is deployed. For instance, in the Anes dataset, a domain shift occurs across geographic regions, which introduces variability in voter behavior and attitudes that was not evident during training. Similarly, in the BRFSS Hypertension Prediction dataset, shifts in BMI distributions across different populations challenge model performance when applied to new groups. These shifts, which practitioners might not foresee during model training, can dramatically affect a model's real-world performance when deployed in diverse settings. As a result, models optimized solely for validation AUC may struggle in real world scenarios. Our findings underscore the need for model calibration metrics that account for such distributional shifts.

In conclusion, our analysis reveals notable differences in performance across datasets, highlighting the complexity of optimizing fitness functions for OOD generalization. In future work, we are interested in exploring these inconsitencies to determine if certain fitness functions are more effective with specific datasets. For instance, some fitness functions may perform better on larger datasets, while others might be more suited to high-dimensional datasets. Gaining insights into when and why particular fitness functions excel will be highly valuable for practitioners, allowing them to make more informed decisions during model selection. Additionally, we plan to investigate the structure of the model pool itself—specifically, what factors make certain model pools easier or harder to outperform. This exploration could offer important insights into how to build model pools in an efficient and scalable manner.

References

[1] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, page 93–104, New York, NY, USA, 2000. Association for Computing Machinery.

[2] Elliot Creager, Jörn-Henrik Jacobsen, and Richard S. Zemel. Exchanging lessons between algorithmic fairness and domain generalization. *CoRR*, abs/2010.07249, 2020.

[3] Thomas G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, page 1–15, Berlin, Heidelberg, 2000. Springer-Verlag.

[4] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks, 2019.

[5] Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Benchmarking distribution shift in tabular data with tableshift. *Advances in Neural Information Processing Systems*, 36, 2024.

[6] Yufan Liao, Qi Wu, Zhaodi Wu, and Xing Yan. Decorr: Environment partitioning for invariant learning and ood generalization, 2024.

[7] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008.

[8] Giung Nam, Jongmin Yoon, Yoonho Lee, and Juho Lee. Diversity matters when learning from ensembles. *CoRR*, abs/2110.14149, 2021.

[9] Fengchun Qiao and Xi Peng. Ensemble pruning for out-of-distribution generalization. In *Forty-first International Conference on Machine Learning*, 2024.

[10] Hamed Rahimian and Sanjay Mehrotra. Frameworks and results in distributionally robust optimization. *Open Journal of Mathematical Optimization*, 3:1–85, July 2022.

[11] Alexander Rubinstein, Luca Scimeca, Damien Teney, and Seong Joon Oh. Scalable ensemble diversification for ood generalization and detection, 2024.

[12] Amartya Sanyal, Yaxi Hu, Yaodong Yu, Yian Ma, Yixin Wang, and Bernhard Schölkopf. Accuracy on the wrong line: On the pitfalls of noisy data for out-of-distribution generalisation, 2024.

[13] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification, 2020.

[14] Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, 2022.

A Appendix A: ID Validation AUCs of Models

B Appendix B: Breif Description of Fitness Functions

Fitness Function	Brief Description
bse	Computes the Average Brier Score of the Ensemble
cbcel	Class balanced cross entropy loss. Weights the class balance by the training distribution of classes
ece25	Expected Calibration Error using 25 bins
ece10	Expected Calibration Error using 10 bins
ece5	Expected Calibration Error using 5 bins

Fitness Function	Brief Description
uwl1	Uncertainty weighted log loss with uncertainty weights based on validation accuracy difficulty scores. The difficulty score of a particular example is the fraction of the 10k models in the model pool that incorrectly predicted that example.
uwll5	Uncertainty weighted log loss with difficulty scores scaled by a factor of 5
ec_std	Ensemble Confidence Standard Deviation
10_uwl25_std	The standard deviation of uncertainty weighted log loss with difficulty scores scaled by a factor of 25 across 10 different kmeans bins
5_flm5_std	The standard deviation of model focal loss with alpha set to 5 across 5 different kmeans bins
10_fle3_mean	The average focal loss of the ensemble with alpha set to 3 across 10 different kmeans bins
5_ece10_mean	The average expected calibration error using 10 bins across 5 different kmeans bins
5_fle4_std	The average focal loss of the ensemble with alpha set to 4 across 5 different kmeans bins
ec_mean	The average ensemble confidence
10_ec_mean	The average ensemble confidence across 10 kmeans bins
5_ec_mean	The average ensemble confidence across 5 kmeans bins

Table 7: Description of Fitness Functions referenced in this paper

Model	Anes	brfss_diabetes	diabetes_readmission	brfss_blood_pressure
Baselines				
CatBoost	0.8801	0.8299	0.6997	0.7053
XGBoost	0.8801	**0.8303**	0.7024	0.7043
ModelPool	0.8668	0.8201	0.6748	0.6856
Validation AUC	**0.9217**	0.8290	**0.7837**	**0.7150**
Our Models				
Uncertainty Weighted Log Loss-5 (10)	0.8574	0.8051	0.6461	0.6478
Brier Score Ensemble; Class Balanced CE Loss	0.8656	0.8184	0.7219	0.7036
Class Balanced CE Loss; Uncertainty Weighted Log Loss-1	0.8821	0.8194	0.7348	0.7032
Brier Score Ensemble; Expected Calibration Error-25	0.8753	0.8165	0.6822	0.7079
Brier Score Ensemble; Expected Calibration Error-10	0.8775	0.8185	0.6993	0.7080
Class Balanced CE Loss; Uncertainty Weighted Log Loss-5	0.8912	0.8061	0.7425	0.7088
Brier Score Ensemble; Expected Calibration Error-5	0.8725	0.8168	0.6859	0.7088

Table 6: ID AUCs for all datasets