DivBO: Diversity-aware CASH for Ensemble Learning

A Appendix

A.1 Ensemble Selection

We provide the pseudo-code in Algorithm 1. In our paper, the $Perf$ metric is the classification error based on the mean predictions of learners in the ensemble.

Algorithm 1: Procedure of ensemble selection.

```
Input: The ensemble size $E$, the configuration observations $D$, and the validation set $\mathcal{D}_{val}$.
Initialize the ensemble as $B = \varnothing$;
for $i = 1, ..., E$ do
    $a \leftarrow \arg\min_{a \in D} Perf(B \cup \{a\}, \mathcal{D}_{val})$;
    $B \leftarrow B \cup \{a\}$;
return the final ensemble $B$.
```

A.2 Dataset Details

In Table 1, we provide the details of the datasets used in our experiment. While all the datasets are collected from OpenML [6], we provide the OpenML ID as the identification of the dataset.

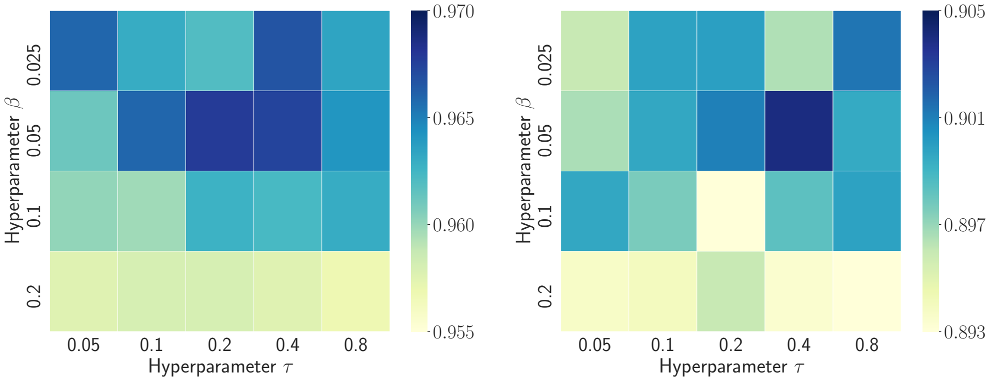
Table 1: Basic dataset information.

Datasets	OpenML ID	Classes	Samples	Features
amazone_employee	43900	2	32769	9
bank32nh	833	2	8192	32
cpu_act	761	2	8192	21
cpu_small	735	2	8192	12
eeg	1471	2	14980	14
elevators	846	2	16599	18
house_8L	843	2	22784	8
pol	722	2	15000	48
pollen	871	2	3848	5
puma32H	752	2	8192	32
quake	772	2	2178	3
satimage	182	6	6430	36
spambase	44	2	4600	57
wind	847	2	6574	14
2dplanes	727	2	40768	10

A.3 Search Space

The search space for algorithms and feature engineering operators are presented in Tables 2 and 3, respectively.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) spambase (b) house_8L

Figure 1: Sensitivity analysis on spambase and house_8L.

Table 2: Search space for algorithms. We distinguish categorical (cat) hyperparameters from numerical (cont) ones. The numbers in the brackets are conditional hyperparameters.

Type of Classifier	#λ	cat (cond)	cont (cond)
AdaBoost	4	1 (-)	3 (-)
Random forest	5	2 (-)	3 (-)
Extra trees	5	2 (-)	3 (-)
Gradient boosting	7	1 (-)	6 (-)
KNN	2	1 (-)	1 (-)
LDA	4	1 (-)	3 (1)
QDA	1	-	1 (-)
Logistic regression	4	2 (-)	2 (-)
Liblinear SVC	5	2 (2)	3 (-)
LibSVM SVC	7	2 (2)	5 (-)
LightGBM	6	-	6 (-)

Table 3: Search space of feature engineering operators.

Type of Operator	#λ	cat (cond)	cont (cond)
Minmax	0	-	-
Normalizer	0	-	-
Quantile	2	1 (-)	1 (-)
Robust	2	-	2 (-)
Standard	0	-	-
Cross features	1	-	1 (-)
Fast ICA	4	3 (1)	1 (1)
Feature agglomeration	4	3 (2)	1 (-)
Kernel PCA	5	1 (1)	4 (3)
Rand. kitchen sinks	2	-	2 (-)
LDA decomposer	1	1 (-)	-
Nystroem sampler	5	1 (1)	4 (3)
PCA	2	1 (-)	1 (-)
Polynomial	2	1 (-)	1 (-)
Random trees embed.	5	1 (-)	4 (-)
SVD	1	-	1 (-)
Select percentile	2	1 (-)	1 (-)
Select generic univariate	3	2 (-)	1 (-)
Extra trees preprocessing	5	2 (-)	3 (-)
Linear SVM preprocessing	5	3 (3)	2 (-)

A.4 Implementation Details

We implement the performance surrogate of Bayesian optimization based on OpenBox [4], a toolkit for black-box optimization. The other baselines are implemented following the open-source version or original papers. For NES, the population is set to 30; for EO, the ensemble size is set to 12; for RB, α and trial per action are set to 3 and 5; for BO and DivBO, we sample 4950 and 50 candidates from global and local sampling, respectively; for all post-hoc designs, we set the ensemble size of ensemble selection to 25; for DivBO, we set β and τ to 0.05 and 0.2. All the experiments are conducted on a machine with 64 'AMD EPYC 7702P' CPU cores.

A.5 Additional Results

Sensitivity Analysis. We first provide the hyperparameter sensitivity analysis on the dataset spambase. The choices for β are $\{0.025, 0.05, 0.1, 0.2\}$, and the choices for τ are $\{0.05, 0.1, 0.2, 0.4\}$. The validation accuracy of 20 combinations of hyperparameters are shown in Figure 1. Remind that β is the maximum of diversity importance and τ controls the speed of approaching saturation. We observe that a large β (0.2) leads to a clear accuracy drop, and we suggest using a $\beta = 0.05$. However, we need to tune τ to achieve the best results on different datasets. The reason may be that the difficulty for different datasets to find good configurations are different. As DivBO builds on the intuition that we need to focus on accuracy rather than diversity in early iterations, a smaller τ is required if it's difficult to find accurate learners in early iterations. The suggested region for tuning τ is [0.1,0.8]. In our paper, we use 0.2 by default, but a tuned τ may achieve better results.

Ablation Study. In this part, we compare DivBO without weight scheduling by setting $w = 0.1$. The results on five datasets are shown in Table 4. The results show that DivBO with weight schedule (Equation 5) performs much better than fixing the weight for diversity (not significant on quake, but significant on other 4 datasets). It fits the intuition that motivates the weight schedule design in Section 3.2.

Table 4: Ablation study on weight scheduling.

Method	elevators	house_8L	pol	quake	wind
DivBO (fixed)	9.59 ± 0.30	11.51 ± 0.28	1.66 ± 0.19	45.63 ± 1.45	14.27 ± 0.36
DivBO	$\mathbf{9.40 \pm 0.28}$	$\mathbf{10.80 \pm 0.22}$	$\mathbf{1.34 \pm 0.17}$	$\mathbf{45.55 \pm 1.37}$	$\mathbf{13.93 \pm 0.42}$

Analysis on Ensemble Strategy. While previous study [1] claims that ensemble selection performs well with CASH using Bayesian optimization, we also evaluate the influence of different strategies on DivBO. We compare ensemble selection with weighted bagging and stacking. We pick 5 learners with the best validation errors from observations to build ensemble for bagging and stacking following EO [3]. The test results on 5 datasets are shown in Table 5. Among three compared strategies, ensemble selection performs the best. In addition, the cost of bagging and ensemble selection can almost be ignored, since the intermediate predictions are stored during optimization. However, stacking is a time-consuming strategy, as the learners are re-trained multiple times after optimization. For example, if we pick 5 base learners for stacking and 5-fold cross validation is used, the cost of stacking equals to that of 25 configuration evaluations, while the budget for the entire CASH process is only 250 evaluations. Therefore, we apply ensemble selection as the default post-hoc ensemble strategy in our paper.

Table 5: Test error (%) of different ensemble strategies on different datasets.

Method	elevators	house_8L	pol	quake	wind
Bagging	10.98 ± 1.70	11.12 ± 0.33	1.85 ± 0.31	47.22 ± 1.85	14.21 ± 2.82
Stacking	9.53 ± 1.13	11.04 ± 0.23	1.54 ± 0.28	46.58 ± 1.13	14.02 ± 1.43
Ensemble selection	$\mathbf{9.40 \pm 0.28}$	$\mathbf{10.80 \pm 0.22}$	$\mathbf{1.34 \pm 0.17}$	$\mathbf{45.55 \pm 1.37}$	$\mathbf{13.93 \pm 0.42}$

Analysis on Ensemble Size. We also provide the results if we set a larger ensemble size. As ensemble selection directly optimizes the performance on the validation set, the validation performance is definitely no worse than using a smaller ensemble size due to the greedy mechanism. However, as pointed out by [2], if we optimize the validation set too much (i.e., setting a too large ensemble size for ensemble selection), the test results may deteriorate, which is referred to as the overfitting issue in AutoML. The results when setting the ensemble size to 100 for BO-ES are presented in Table 6.

Table 6: Test error (%) of different ensemble sizes on different datasets.

Method	elevators	house_8L	pol	quake	wind
BO-ES (ens_size=100)	9.98 ± 0.30	11.52 ± 0.26	1.45 ± 0.33	47.43 ± 1.62	14.04 ± 0.47
BO-ES (ens_size=25)	$\mathbf{9.61 \pm 0.36}$	$\mathbf{11.06 \pm 0.33}$	$\mathbf{1.35 \pm 0.18}$	$\mathbf{46.10 \pm 2.52}$	14.04 ± 0.53

In main paper, the ensemble size is set to 25 following VolcanoML [5], which shows good empirical results across different datasets. We observe that when we set the ensemble size to 100 for BO-ES, the test results are generally worse than setting the ensemble size to 25 due to the overfitting issue (not significant on wind but significant on the other four). We have also mentioned this risk of overfitting in the limitation.

Comparison with Intuitive Designs. In this part, we evaluate another intuitive CASH design, which tunes each algorithm for the same budget and then builds a post-hoc ensemble. In fact, it is a simplified version of the baseline RB-ES, in which RB-ES eliminates some of the algorithms after several iterations. We name it kBO-ES and present the results in Table 7. We observe that the results of kBO-ES are quite similar to RS-ES (Random search with ensemble selection). The reason is that the search space contains a lot of algorithms while the budget is quite limited (250 iterations). Each algorithm can only be tuned about 22 times. For each algorithm, we also need to tune the feature engineering operators (>50 HPs in auto-sklearn search space), and thus the BO surrogate for each algorithm is under-fitted. Therefore, Bayesian optimization for each algorithm performs like random search. kBO-ES is an intuitive method but seems to perform not competitively when the search space is very large.

Table 7: Test error (%) compared with another intuitive design.

Method	elevators	house_8L	pol	quake	wind
RS-ES	9.51 ± 0.28	11.21 ± 0.38	1.39 ± 0.15	46.79 ± 1.57	14.34 ± 0.47
kBO-ES	9.55 ± 0.32	11.18 ± 0.34	1.39 ± 0.16	46.81 ± 1.48	14.29 ± 0.45
DivBO	$\mathbf{9.40 \pm 0.28}$	$\mathbf{10.80 \pm 0.22}$	$\mathbf{1.34 \pm 0.17}$	$\mathbf{45.55 \pm 1.37}$	$\mathbf{13.93 \pm 0.42}$

Comparison with AutoGluon. The search space plays a significant role in CASH optimization. As DivBO is an algorithm framework rather than a system, we compare it with other baselines by using the same search space (i.e., auto-sklearn space). However, AutoGluon applies a more compact space than auto-sklearn, and it's not fair to directly compare DivBO on auto-sklearn search space with AutoGluon. To make a relatively fair comparison, we reproduce a similar search space of AutoGluon except for the specified neural networks due to implementation difficulty. The results on five datasets are in Table 8.

Table 8: Test error (%) compared with AutoGluon.

Method	elevators	house_8L	pol	quake	wind
AutoGluon Tabular	9.10 ± 0.00	$\mathbf{9.98 \pm 0.00}$	1.23 ± 0.00	44.72 ± 0.00	14.37 ± 0.00
DivBO (AutoGluon)	$\mathbf{9.01 \pm 0.11}$	10.06 ± 0.17	$\mathbf{1.18 \pm 0.07}$	44.75 ± 0.60	$\mathbf{14.24 \pm 0.18}$

Note that, the search space affects the results a lot. For example, AutoGluon's results on wind are worse than RS-ES using the auto-sklearn space. However, AutoGluon's results on the other four datasets are better than most of the results using the auto-sklearn space, which is consistent with the observation that AutoGluon often outperforms auto-sklearn. The reason may be that AutoGluon is equipped with a well-designed search space, which kicks out less reliable algorithms on modern datasets (e.g., Naive Bayes) and adds strong ones (e.g., Catboost). The variance of AutoGluon's results is zero because it fixes the random seed in its inner design. In addition, we observe an error decrease when using DivBO in this search space. Concretely, the improvement is statistically significant on three datasets, not significant on one (quake), and slightly worse on the other one (house_8L).

Further Analysis on Diversity. In this part, we analyze the influence of removing the most diverse learners from the final ensemble. While DivBO is extended from naive BO, we evaluate the mean influence of removing the top-3 diverse models from the final ensemble. As the ensemble is built on the validation set, we present the validation error gaps on five datasets in Table 9. (A positive gap means removing the models leads to an error increase.) We observe that generally, removing the most diverse learners from DivBO leads to a larger error increase than BO-ES. As it's easy to learn good learners on the dataset pol (the accuracy of almost all the learners in the ensemble is above 98%), removing learners affects quite little on the ensemble performance.

Table 9: Error gap (%) when removing the top-3 diverse models.

Gap	elevators	house_8L	pol	quake	wind
BO-ES	+0.21	-0.04	+0.02	+0.98	+0.38
DivBO	+0.38	+0.15	+0.01	+1.28	+0.61

References

[1] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and robust automated machine learning. In *Advances in neural information processing systems*, pages 2962–2970, 2015.

[2] F. Hutter, L. Kotthoff, and J. Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[3] J.-C. Lévesque, C. Gagné, and R. Sabourin. Bayesian hyperparameter optimization for ensemble learning. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 437–446, 2016.

[4] Y. Li, Y. Shen, W. Zhang, Y. Chen, H. Jiang, M. Liu, J. Jiang, J. Gao, W. Wu, Z. Yang, C. Zhang, and B. Cui. Openbox: A generalized black-box optimization service. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021.

[5] Y. Li, Y. Shen, W. Zhang, J. Jiang, B. Ding, Y. Li, J. Zhou, Z. Yang, W. Wu, C. Zhang, et al. Volcanoml: speeding up end-to-end automl via scalable search space decomposition. *Proceedings of the VLDB Endowment*, 14(11):2167–2176, 2021.

[6] J. Vanschoren, J. N. Van Rijn, B. Bischl, and L. Torgo. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.