Dual Active Learning for Both Model and Data Selection

Ying-Peng Tang and **Sheng-Jun Huang**[*]
College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
Collaborative Innovation Center of Novel Software Technology and Industrialization
MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing 211106, China
{tangyp, huangsj}@nuaa.edu.cn

Abstract

To learn an effective model with less training examples, existing active learning methods typically assume that there is a given target model, and try to fit it by selecting the most informative examples. However, it is less likely to determine the best target model in prior, and thus may get suboptimal performance even if the data is perfectly selected. To tackle with this practical challenge, this paper proposes a novel framework of dual active learning (DUAL) to simultaneously perform model search and data selection. Specifically, an effective method with truncated importance sampling is proposed for Combined Algorithm Selection and Hyperparameter optimization (CASH), which mitigates the model evaluation bias on the labeled data. Further, we propose an active query strategy to label the most valuable examples. The strategy on one hand favors discriminative data to help CASH search the best model, and on the other hand prefers informative examples to accelerate the convergence of winner models. Extensive experiments are conducted on 12 openML datasets. The results demonstrate the proposed method can effectively learn a superior model with less labeled examples.

1 Introduction

Labeling a large number of examples for model training is expensive in many real applications. Active learning (AL) [Settles, 2009] is a common approach to reduce the labeling cost. It usually requires a target model to evaluate the unlabeled data, and tries to select the most informative instances for querying their labels from the oracle.

A common assumption in the AL literature is that the target model is given as a prior. However, due to the lack of knowledge of the task, one can hardly determine the best model before querying in practice. A naive solution for active learning is simply applying a commonly used model for data selection. Unfortunately, this simple method will significantly jeopardize the performance of AL. It has been reported that, if applying different models for data evaluation and task learning, AL tends to be noneffective (i.e., comparable to the random strategy) [Lowell *et al.*, 2019].

One possible remedy is introducing the model search (i.e., model selection) [Elshawi *et al.*, 2019] to active learning to discover the applicable target model [Sugiyama and Rubens, 2008; Ali *et al.*, 2014; Geifman and El-Yaniv, 2019]. However, most of existing works simply conduct data querying and model selection separately, they neglect the vulnerability of either of these two parts in such a challenging setting. Specifically, i) The distribution of the labeled data is skewed with the active querying, the model evaluation on the training set can be highly biased. ii) The target model varies during the AL process due to the model search. Greedily sampling with the winner model in the current iteration could be myopic. iii) Model search is usually time-consuming. It is impractical to perform model selection after each query, which will seriously affect the labeling efficiency. Although Ali *et al.* [2014] try to refrain from the first problem by maintaining an unbiased validation set by random sampling, this simple strategy may lead to extra labeling cost. As a result, it is still a challenging problem to reach the optimal performance with least cost when the target model prior is unavailable.

In this paper, we propose a novel DUal Active Learning (DUAL) framework to cope with the practical challenges by simultaneously performing model search and data selection. It well addresses all the concerns raised above, and deeply exploits the characteristics of both active querying and Combined Algorithm Selection and Hyperparameter optimization (CASH) [Thornton *et al.*, 2013] to improve the final performance. Specifically, we propose an effective CASH method to search model configurations (i.e., algorithm and hyperparameters) along with the data querying process. It is equipped with the truncated importance sampling and a successive function to better accommodate to the framework. The former ameliorates the evaluation bias on the skewed labeled set, and the latter dramatically lowers the computational complexity of the algorithm. Based on the results of CASH, an effective active data querying strategy is proposed. It on one hand tries to help CASH identify the models with high potential to achieve the best performance by selecting the most discriminative examples, and on the other hand queries informa-

[*]This research was supported by the National Key R&D Program of China (2020AAA0107000), NSFC (62076128) and the China University S&T Innovation Plan Guided by the Ministry of Education. Sheng-Jun Huang is the corresponding author.

tive instances to accelerate the convergence of winner models of CASH. Consequently, the performance on the given task will be significantly improved with a few queries. The experimental results on 12 openML datasets demonstrate the superiority of the proposed framework.

2 Related Work

Active learning has been extensively studied in many tasks with expensive labeling cost, such as multi-label learning [Huang *et al.*, 2017], multi-modality learning [Gao *et al.*, 2018], deep learning [Shui *et al.*, 2020], and so on. Many criteria are proposed to evaluate the informativeness of unlabeled data. There are mainly two groups of methods, namely uncertainty-based [Yan and Huang, 2018; Kirsch *et al.*, 2019] and representativeness-based [Sener and Savarese, 2018; Li *et al.*, 2020a]. The former prefers the uncertain data to the target model, while the latter favors the examples which can represent most patterns in the dataset. It has also been reported that a mixture of multiple criteria can usually achieve a better performance [Du *et al.*, 2015; Huang and Chen, 2016; Tang and Huang, 2019]. Recently, several works are proposed to learn a query strategy rather than designing one heuristically [Konyushkova *et al.*, 2017; Casanova *et al.*, 2020]. Most of these works require the target model prior, which can hardly be satisfied in practice.

CASH [Thornton *et al.*, 2013] and Neural Architecture Search (NAS) [Elsken *et al.*, 2019] are popular topics in automatic machine learning [Elshawi *et al.*, 2019]. The former mainly focuses on the traditional models, while the latter considers deep models, which typically lead to much higher cost for model evaluation. In this paper, we mainly consider the CASH setting. CASH methods can be roughly divided into black-box optimization methods and multi-fidelity optimization methods. The former is mainly implemented by Bayesian optimization [Hutter *et al.*, 2011; Bergstra *et al.*, 2011; Falkner *et al.*, 2018], and the latter tries to dynamically allocate time resource to different configurations, which is usually implemented by bandit algorithms [Jamieson and Talwalkar, 2016; Li *et al.*, 2017; Hu *et al.*, 2019; Huang *et al.*, 2020]. All these methods require an unbiased evaluation, such information may not be directly obtained in active learning framework.

Recently, several works try to combine the active learning and model selection for better performance. However, they are significantly different from our work. Specifically, ALMS [Ali *et al.*, 2014] queries points for either model training or model selection. If the maximum score of all unlabeled data for model training is higher than model selection, a point will be selected by the expectation of improvements of the candidate models, and added to the training set. Otherwise, a randomly selected point will be added to the validation set for model selection. However, the algorithm performs leave-2-out cross-validation to calculate the expected model improvement. It could be extremely expensive when the number of candidate models and unlabeled data are large, which limits its practicability. Moreover, the method maintains an unbiased validation set for model selection by random sampling. It may lower the utilization of the labeling cost.

Geifman and El-Yaniv [2019] propose Active-iNAS to search for the proper network architecture along with the AL process. It is different from DUAL with the following aspects. Firstly, the authors apply off-the-shelf data selection methods to query data based on the winner architecture searched by NAS. The query strategy does not consider the model selection. While in our approach, the data querying and CASH are well supported by each other. Secondly, the authors believe that the model capacity should monotonically increase with the size of labeled data. They neglect the characteristics of the dataset. While in our approach, the search space will be adjusted accordingly for different tasks. Lastly, they focus on NAS while we consider the CASH problem.

3 The Proposed Framework

We denote the dataset with n instances by $\mathcal{D}$, which includes a small labeled set $\mathcal{L} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n_l}$ with n_l instances, and a large unlabeled set $\mathcal{U} = \{\boldsymbol{x}_i\}_{i=n_l+1}^{n_l+n_u}$ with n_u instances, where $n_l \ll n_u$ and $n = n_l + n_u$. Let $\mathcal{A} = \{A^1, \ldots, A^K\}$ be the set of candidate algorithms (e.g., SVM, decision tree, etc.). The corresponding hyperparameter space of A^j is Λ_j. A configuration A^j_λ is a specific algorithm with hyperparameters λ where $A^j \in \mathcal{A}, \lambda \in \Lambda_j$. The objective of CASH is searching for the best configuration based on a quality metric $V(\cdot)$ (e.g., the cross-validation accuracy), which can be formulated as

$$\begin{aligned} A^*_{\lambda^*} &= \underset{A^j_\lambda}{\arg\max}\, V(A^j_\lambda, \mathcal{L}) \\ s.t. \quad & A^j \in \mathcal{A}, \lambda \in \Lambda_j \quad \forall j \in \{1, \ldots, K\}\,. \end{aligned} \tag{1}$$

3.1 Model Selection

CASH module responses for discovering the best model configurations for the current task. One challenge here is the distribution of labeled data is not identical to the test distribution due to the active querying, while the model evaluation usually requires an unbiased estimation of the performance for an accurate selection. For this reason, directly conducting cross validation on the labeled set may be suboptimal.

To cope with this problem, we employ the importance sampling [Tokdar and Kass, 2010] to alleviate the bias of evaluation. As discussed in [Sugiyama *et al.*, 2007], the importance weighted validation error is an unbiased estimation of generalization error.

$$W_{\boldsymbol{x}_i \sim p_q}(A^j_\lambda, \mathcal{L}) = \frac{1}{n_l} \sum_{i=1}^{n_l} \frac{p_d(\boldsymbol{x}_i)}{p_q(\boldsymbol{x}_i)} \ell(A^j_\lambda, \boldsymbol{x}_i)\,, \tag{2}$$

where p_q and p_d are the distributions of queried data and the whole dataset, $\ell(A^j_\lambda, \boldsymbol{x}_i)$ is the loss of the trained model A^j_λ on example $\boldsymbol{x}_i$.

In order to get the importance weights of the validation data (sampled randomly from the labeled set), inspired by the recent works [Zhang *et al.*, 2018], we train a domain discriminator to distinguish the data sampled from labeled set and data pool. The objective of the discriminator is

$$\min_D \; \mathbb{E}_{\boldsymbol{x} \sim p_q}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_d}[\log(1 - D(\boldsymbol{x}))]\,. \tag{3}$$

By setting the partial derivative to zero as in [Goodfellow *et al.*, 2014], the optimal D^* can be represented by p_q and p_d. In this way, the importance weights w for the validation data can be approximated by D^*

$$D^* = \frac{p_q}{p_d + p_q} \rightarrow w = \frac{p_d}{p_q} = \frac{1}{D^*} - 1 \,. \tag{4}$$

However, sometimes the discriminator D may output a very small value. It will lead to an extremely large importance weight to overemphasize a specific example and may mislead the model selection. Here we further apply the truncated importance sampling [Ionides, 2008] to mitigate this practical problem. Let $w' = w \wedge \tau$ be the truncated importance weight, where $\wedge$ is the minimum of w and τ, and τ is a hyperparameter. Next, by applying the results from [Ionides, 2008], we show that the validation error with truncated importance weights is consistent with the generalization error.

Theorem 1. *Supposing that $\ell(A^j_\lambda, \boldsymbol{x}_i) < \infty$ and $p_d(\boldsymbol{x}_i)\ell(A^j_\lambda, \boldsymbol{x}_i) = 0$ whenever $p_q(\boldsymbol{x}_i) = 0$. Denote $w'_i = w'(\boldsymbol{x}_i)$, the bias and variance of $\frac{1}{n_l}\sum_{i=1}^{n_l} w'_i \ell(A^j_\lambda, \boldsymbol{x}_i)$ will converge to zero, as long as $\tau \rightarrow \infty$ and $\tau/n_l \rightarrow 0$.*

Proof. The bias is calculated as

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{x}\sim p_q}\left[w'(\boldsymbol{x})\ell(A^j_\lambda, \boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim p_q}\left[w(\boldsymbol{x})\ell(A^j_\lambda, \boldsymbol{x})\right] \\
&= \int_{x:p_q(x)>0} \ell(A^j_\lambda, \boldsymbol{x})\left((w(\boldsymbol{x}) \wedge \tau) - w(\boldsymbol{x})\right) p_q(\boldsymbol{x})dx \\
&= \int_{x:p_d(\boldsymbol{x})>\tau p_q(\boldsymbol{x}), p_q(\boldsymbol{x})>0} \ell(A^j_\lambda, \boldsymbol{x})\left(\tau p_q(\boldsymbol{x}) - p_d(\boldsymbol{x})\right) dx \,.
\end{aligned} \tag{5}$$

Since $\left|\ell(A^j_\lambda, \boldsymbol{x})\left(\tau p_q(\boldsymbol{x}) - p_d(\boldsymbol{x})\right)\right| \le \left|\ell(A^j_\lambda, \boldsymbol{x}) p_d(\boldsymbol{x})\right|$, the bias $\rightarrow 0$ when $\tau \rightarrow \infty$. To bound the variance,

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{x}\sim p_q}\left[\left(w'(\boldsymbol{x})\ell(A^j_\lambda, \boldsymbol{x})\right)^2\right] \\
&= \int_{x:p_q(x)>0} \ell(A^j_\lambda, \boldsymbol{x})^2 \left(w(\boldsymbol{x}) \wedge \tau\right)^2 p_q(\boldsymbol{x})dx \\
&\le \tau \int_{x:p_q(x)>0} \ell(A^j_\lambda, \boldsymbol{x})^2 w(\boldsymbol{x}) p_q(\boldsymbol{x})dx \\
&\le \tau \mathbb{E}_{\boldsymbol{x}\sim p_d}\left[\ell(A^j_\lambda, \boldsymbol{x})^2\right] \,.
\end{aligned} \tag{6}$$

Since

$$Var_{\boldsymbol{x}_i\sim p_q}\left[\frac{1}{n_l}\sum_{i=1}^{n_l} w'_i \ell(A^j_\lambda, \boldsymbol{x}_i)\right] \le \tau \mathbb{E}_{\boldsymbol{x}\sim p_d}\left[\ell(A^j_\lambda, \boldsymbol{x})^2\right]/n_l, \tag{7}$$

when $\tau/n_l \rightarrow 0$, the variance $\rightarrow 0$. □

Letting $V(A^j_\lambda, \mathcal{L})$ be the truncated importance weighted performance on validation set, it is a consistent estimation of the generalization performance. In this case, the off-the-shelf CASH method for model search can be applied. However, simply performing CASH after each query is unacceptable due to the computational complexity. To this end, inspired by [Li *et al.*, 2020b], we further propose a successive-SMAC method to better accommodate the problem setting. It is based the SMAC [Hutter *et al.*, 2011] method, with adopting an elimination function to filter less effective candidates algorithms along with the query process for efficiency.

Specifically, we maintain a set of well-performed candidate algorithms $\mathcal{A}_G = \{A^1, \ldots, A^g\}, \mathcal{A}_G \subseteq \mathcal{A}$, and perform SMAC for each of them separately with the identical time budget after each query. At iteration t, the configuration $A^j_{\lambda^*_t}$ with the highest importance weighted validation performance r^j_t (e.g., accuracy) for each algorithm j in the candidate set will be returned. In this way, each algorithm j has a history performance vector $R^j = \{r^j_1, \ldots, r^j_t\}$. In order to remove those less effective algorithms according to the past performances, we calculate the mean and standard deviation of the performances in the last C iterations $\{r^j_t, r^j_{t-1}, \ldots, r^j_{t-C}\}$, which are denoted by m^j_t, v^j_t. Then we remove the less effective algorithms, which are defined as $\{A^j | m^j_t + v^j_t < m^k_t - v^k_t, j, k \in [1, \ldots, g], k \neq j\}$.

3.2 Data Selection

To maximize the performance on the target task, we believe that the active querying strategy should on one hand help CASH promptly discover the high potential algorithms for the current task, and on the other hand try to improve the winner model to accelerate its convergence. We term these two characteristics as "exploration" and "exploitation", respectively. By well balancing these two aspects, we expect that the size of candidate algorithms should decrease rapidly at early stage. After the best algorithm is identified, the data selection tends to focus more on improving the winner model.

For the exploration, we try to query examples to help CASH in model search. Before we elaborate the selection criterion, we first introduce a lemma to explain our motivation. Remember that we search for the best configuration of each candidate algorithm after each query, and we would like to discover the best algorithm with least iterations. If we regard the performance $\{r^j_t\}^T_{t=1}$ as reward, it is accord with the non-stochastic multi-armed bandit problem [Jamieson and Talwalkar, 2016], where each algorithm is an arm, each arm will be pulled once time at each iteration, the received rewards are non-stochastic (i.e., not sampled i.i.d from a probability distribution supported on $[0, 1]$). The goal is to identify the best arm with least trials.

Formally, define $e^j_t = \max\{r^j_b | b = 1, \ldots, t\}$, we can have that $\{e^j_t\}$ is bounded and non-decreasing. Assume $\lim_{k\rightarrow\infty} e^j_k = \nu^j$, and $|\lim_{k\rightarrow\infty} e^j_k - e^j_t| \le \gamma_j(t)$, $\lim_{t\rightarrow\infty} \gamma_j(t) = 0$. We introduce the following lemma to deduce our data querying strategy.

Lemma 1. *[Jamieson and Talwalkar, 2016] Assume $\nu^1 > \nu^2 \ge \cdots \ge \nu^g$. Define ${\gamma_j}^{-1}(\alpha) = \min\{t \in \mathbb{N} : \gamma_j(t) \le \alpha\}$. If $t_j > {\gamma_j}^{-1}(\frac{\nu^1 - \nu^j}{2})$ and $t_1 > {\gamma_1}^{-1}(\frac{\nu^1 - \nu^j}{2})$, then*

$$\begin{aligned}
e^1_{t_1} - e^j_{t_j} &= (e^1_{t_1} - \nu^1) + (\nu^j - e^j_{t_j}) + 2(\frac{\nu^1 - \nu^j}{2}) \\
&\ge -\gamma_1(t_1) - \gamma_j(t_j) + 2(\frac{\nu^1 - \nu^j}{2}) > 0 \,.
\end{aligned} \tag{8}$$

The lemma indicates that when the assumptions are met, it is guaranteed that $e^1_{t_1} > e^j_{t_j}$ with $\nu^1 \ge \nu^j$. On the contrary, if we do not know which ν is bigger, comparing $e^1_{t_1}$ and $e^j_{t_j}$ suffices to identify, which can be proofed by contradiction. Moreover, the larger the gap between algorithms $\nu^1 - \nu^j$, the earlier we can distinguish the better one. Since the distribution of training examples is controlled by the data querying strategy, it is reasonable to select the instances which lead to significant divergence of candidate algorithms. For this reason, we query the data based on the disagreement between the promising configurations of the candidates $\mathcal{A}_G$, in order to make the superior and inferior algorithms distinguishable as soon as possible. The implementation is deferred to Eq. (9).

For the exploitation, we directly utilize the winner model returned by CASH, and query by uncertainty to accelerate its convergence. Formally, Supposing that there are g candidate algorithms at iteration t, and $r^1_t > \cdots > r^g_t$, we have the following selection criterion $S(\boldsymbol{x})$ for data $\boldsymbol{x}$

$$S(\boldsymbol{x}) = -\sum_i \Big[P_{A^1_{\lambda^*_t}}(y_i|\boldsymbol{x}) \log P_{A^1_{\lambda^*_t}}(y_i|\boldsymbol{x}) + \beta \cdot c(\mathcal{A}_G, \boldsymbol{x}, y_i)/g \log\left(c(\mathcal{A}_G, \boldsymbol{x}, y_i)/g\right) \Big], \tag{9}$$

where i ranges over all possible labels, $P_{A^1_{\lambda^*_t}}(y_i|\boldsymbol{x})$ is the probability prediction of model trained with configuration $A^1_{\lambda^*_t}$ for example $\boldsymbol{x}$ on class i, β is the tradeoff, and

$$c(\mathcal{A}_G, \boldsymbol{x}, y_i) = \textstyle\sum_{j=1}^{g} \mathbb{I}(A^j_{\lambda^*_t}(\boldsymbol{x}) = y_i), \tag{10}$$

where $\mathbb{I}(\cdot)$ is the indicator function, $A^j_{\lambda^*_t}(\boldsymbol{x})$ is the prediction of the configuration of data $\boldsymbol{x}$. The first term of $S(\cdot)$ is the prediction entropy of data according to the winner model, the second term is the vote entropy for estimating the disagreement between the best configurations of each candidate algorithm in $\mathcal{A}_G$. We summarize the proposed DUAL framework in Algorithm 1.

Algorithm 1 The DUAL Algorithm

Input: Candidate algorithms $\mathcal{A}_G$, performance history R^j, iteration t, truncated threshold τ, tradeoff parameter β, unlabeled pool $\mathcal{U}$, labeled set $\mathcal{L}$.
Output: Selected data $\boldsymbol{x}_q$, best configuration $A^b_{\lambda^*_t}$.

```
1: $\mathcal{L}_{tr}, \mathcal{L}_{val} \leftarrow split(\mathcal{L})$   ▷ Split labeled data into training set $\mathcal{L}_{tr}$ and validation set $\mathcal{L}_{val}$.
2: $D \leftarrow trainD(\mathcal{L}_{tr}, \mathcal{L} \cup \mathcal{U})$ ▷ Train domain discriminator $D$ from labeled data and the whole dataset.
3: $\boldsymbol{y}_v \leftarrow predict(D, \mathcal{L}_{val})$
4: $\boldsymbol{w}'_v \leftarrow (\frac{1}{\boldsymbol{y}_v} - 1) \wedge \tau$ ▷ Get truncated importance weights.
5: for $j \leftarrow 1$ to $g$ do   ▷ Perform CASH for candidates.
6:     $r^j_t, A^j_{\lambda^*_t} \leftarrow SMAC(A^j, \Lambda_j, \mathcal{L}_{tr}, \mathcal{L}_{val}, \boldsymbol{w}'_v, V)$
7:     $R^j \leftarrow R^j \cup r^j_t$
8:     $m^j_t \leftarrow mean(R^j, C); v^j_t \leftarrow var(R^j, C)$ ▷ Calculate $m^j_t$ and $v^j_t$ with the last C elements of $R^j$.
9: end for
10: $b \leftarrow \arg\max_j(r^j_t)$
11: $B \leftarrow \{A^j | m^j_t + v^j_t < m^k_t - v^k_t, j, k \in [1, \ldots, g], k \neq j\}$
12: $\mathcal{A}_G \leftarrow \mathcal{A}_G \setminus B$   ▷ Remove less effective algorithms.
13: $Z \leftarrow \{S(\boldsymbol{x}_i) | \boldsymbol{x}_i \in \mathcal{U}\}$   ▷ Scoring unlabeled data with Eq. (9).
14: $q \leftarrow \arg\max_i(Z)$
15: $\mathcal{U} \leftarrow \mathcal{U} \setminus \{\boldsymbol{x}_q\}; \mathcal{L} \leftarrow \mathcal{L} \cup \{\boldsymbol{x}_q\}$
16: Update and test model with configuration $A^b_{\lambda^*_t}$
```

4 Experiment

4.1 Empirical Settings

We compare the proposed DUAL method with the following approaches in the experiments.

- **Coreset**: [Sener and Savarese, 2018] A representativeness-based method which selects the data point to best cover the unlabeled pool.
- **QUIRE**: [Huang *et al.*, 2014] A query strategy that considers both uncertainty and representativeness in AL.
- **Entropy**: An uncertainty-based method to select the data point with the highest entropy value according to the prediction of the target model.
- **Random**: Randomly selecting examples to query.
- **CASH**: Performing successive CASH for model search, along with random querying.
- **ALMS**: [Ali *et al.*, 2014] Selecting model by cross-validating each candidate model, and querying data by expected model improvement.
- **Active-iNAS**: [Geifman and El-Yaniv, 2019] Querying examples by entropy, and searching models from simple to complex gradually.

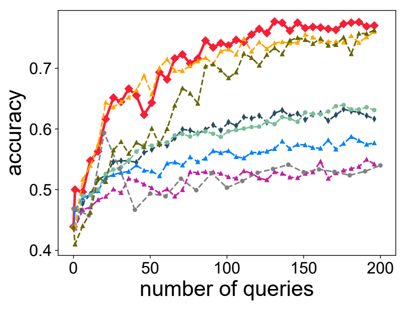
Note that the last 3 methods will conduct model search after each query, and DUAL, CASH, Active-iNAS will use the model search method proposed in this paper for better comparison. ALMS requires a given discrete candidate set for model selection, which will be specified latter. The others employ a fixed target model for data selection, which is searched on the initially labeled set.

We conduct our experiments on 12 openML [Vanschoren *et al.*, 2013] multiclass classification datasets. The basic information is summarized at Table 1. For each case, we randomly sample 40% data as test set, 5% data as initially labeled set, and the rest is treated as the unlabeled pool. The data split is repeated for 10 times with different random seeds.

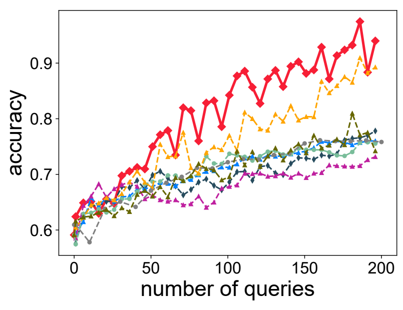
For the initial candidate algorithms in CASH module, we employ 12 commonly used models: Adaboost, Random Forest, Libsvm SVC, SGD, Extra Trees, Decision Tree, K Nearest Neighbors, Passive Aggressive, Gradient Boosting, LDA, QDA, Multi Layer Perceptron. We use the default implemen-

Dataset (ID)	# ins.	Dataset (ID)	# ins.
vehicle (54)	846	tic-tac-toe (50)	958
semeion (1501)	1593	segment (36)	2310
dna (40670)	3186	kr-vs-kp (3)	3196
churn (40701)	5000	phoneme (1489)	5404
optdigits (28)	5620	phishing (4534)	11055
mozilla4 (1046)	15545	letter (6)	20000

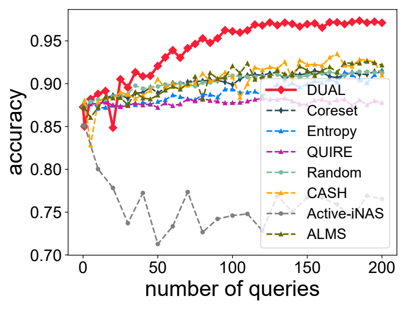
Table 1: The basic information of the datasets in our experiments.



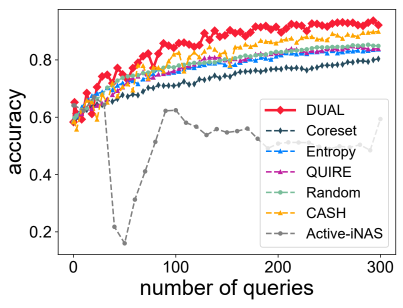
(a) vehicle



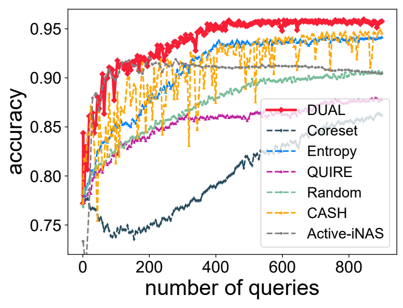
(b) tic-tac-toe



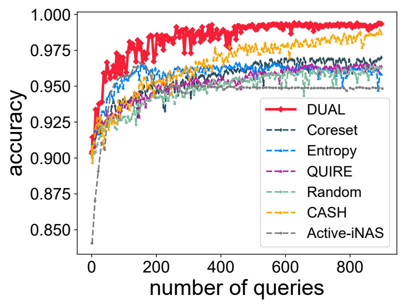
(c) segment



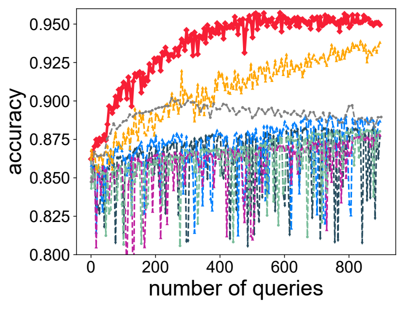
(d) semeion



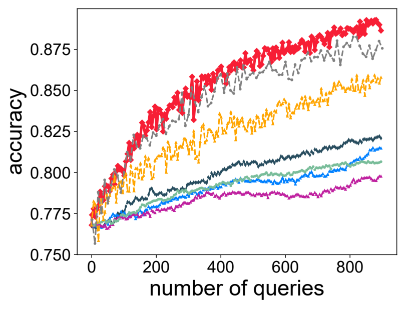
(e) dna



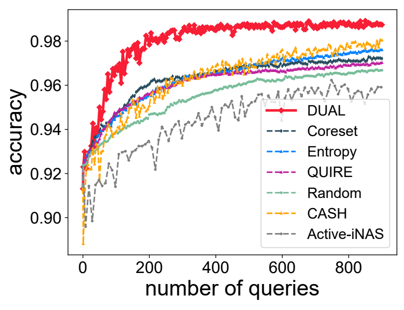
(f) kr-vs-kp



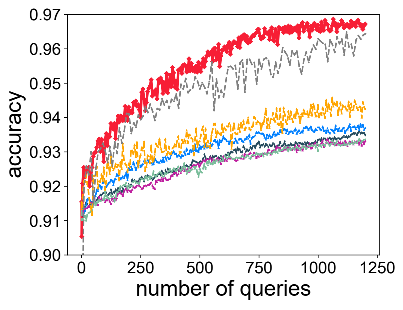
(g) churn



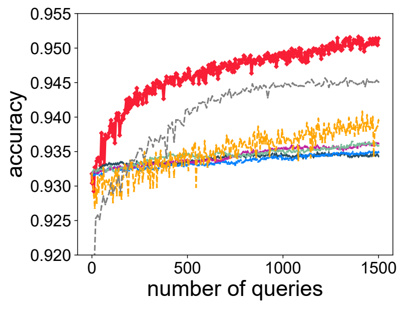
(h) phoneme



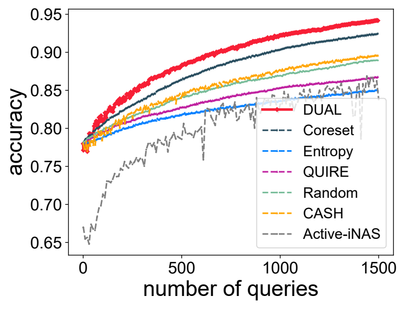
(i) optdigits



(j) phishing



(k) mozilla4



(l) letter

Figure 1: The learning curves of the compared methods.

tation of auto-sklearn [Feurer *et al.*, 2015] for model search.

ALMS can only work with a set of discrete candidate models. We set the candidate models as the default model of each algorithm in $\mathcal{A}$ (i.e., the model with default parameters in scikit-learn). Note that even with this small set, it is still very time-consuming, and can hardly be performed in the datasets with more than 2,000 examples. Thus we only report its results in the relative small datasets.

For Active-iNAS, it requires the complexity rank of candidate algorithms as prior. However, it is hard to compare the complexity of traditional models. To compare with this method, we empirically set the order of candidate algorithms, which is deferred to the supplementary material. Note that once the method selects a complex model, it can never return to a simple one.

For the hyperparameters in our method, we follow the sug-

Datasets	Performance (%)			Running time		
	DUAL	DUAL w/o active	DUAL w/o active & succe.	DUAL	DUAL w/o active	DUAL w/o active & succe.
vehicle	$\mathbf{73.28 \pm 2.23}$	71.80 ± 2.32	$\mathbf{72.50 \pm 1.68}$	$\mathbf{2.71 \pm 0.50}$	$\mathbf{3.30 \pm 0.87}$	12.00 ± 0.00
tic-tac-toe	$\mathbf{86.20 \pm 1.25}$	80.90 ± 1.65	80.53 ± 1.60	$\mathbf{3.49 \pm 0.46}$	5.40 ± 0.71	12.00 ± 0.00
semeion	$\mathbf{84.72 \pm 0.74}$	81.07 ± 1.26	81.51 ± 1.53	$\mathbf{5.53 \pm 0.57}$	7.01 ± 0.74	12.00 ± 0.00
segment	$\mathbf{95.38 \pm 0.31}$	91.52 ± 0.63	91.82 ± 0.84	$\mathbf{4.06 \pm 0.63}$	5.39 ± 0.80	12.00 ± 0.00
dna	$\mathbf{90.28 \pm 1.05}$	86.59 ± 1.30	87.77 ± 1.25	$\mathbf{3.48 \pm 0.42}$	$\mathbf{3.95 \pm 0.91}$	12.00 ± 0.00
kr-vs-kp	$\mathbf{96.95 \pm 0.40}$	94.22 ± 0.76	94.29 ± 0.59	$\mathbf{2.06 \pm 0.44}$	3.25 ± 1.18	12.00 ± 0.00
churn	$\mathbf{91.09 \pm 1.03}$	88.17 ± 1.06	87.87 ± 0.51	$\mathbf{3.89 \pm 0.82}$	7.84 ± 1.74	12.00 ± 0.00
phoneme	$\mathbf{81.65 \pm 0.76}$	79.92 ± 0.64	79.73 ± 0.77	$\mathbf{4.32 \pm 0.83}$	5.86 ± 0.97	12.00 ± 0.00
optdigits	$\mathbf{96.70 \pm 0.30}$	94.45 ± 0.58	94.45 ± 0.65	$\mathbf{3.92 \pm 0.70}$	4.60 ± 0.74	12.00 ± 0.00
phishing	$\mathbf{93.62 \pm 0.21}$	92.50 ± 0.38	92.26 ± 0.34	$\mathbf{3.33 \pm 0.25}$	$\mathbf{3.47 \pm 0.04}$	12.00 ± 0.00
mozilla4	$\mathbf{93.89 \pm 0.26}$	93.07 ± 0.40	93.08 ± 0.45	$\mathbf{2.86 \pm 0.19}$	3.64 ± 0.34	12.00 ± 0.00
letter	$\mathbf{82.20 \pm 0.89}$	80.05 ± 0.88	79.80 ± 0.69	$\mathbf{2.71 \pm 0.70}$	3.05 ± 0.53	12.00 ± 0.00

Table 2: The mean and standard deviation values of the accuracy curves (used to compare the performance) and number of candidate algorithms along with querying process (employed to compare the running time) of each variant method across 10-fold experiments. The best and its comparable performances based on paired t-test at 0.05 significance level are highlighted in boldface.

gestion in [Ionides, 2008] to set the truncated threshold as $\tau = \sqrt{m}$, where m is the number of validation data. For the tradeoff β, we believe that the primary task of DUAL at early stage is to explore the applicable algorithms, the improvement of winner models should be preferred later. To achieve this goal, we set the parameter empirically as $\beta = 1/g$, since the number of candidate algorithms g will decrease along with the query process. Note that the results of parameter sensitivity in the supplementary material indicate that our method is insensitive to β.

4.2 Performance Comparison

We plot the averaged learning curves in Fig. 1. Some legends of the figures are omitted to avoid the occlusion, the figures in the same line have the same legends. It can be observed that, our proposed DUAL method can outperform the others significantly in most cases, which demonstrates its effectiveness. The CASH method with random querying can usually achieve the second best. This phenomenon indicates that an effective target model may bring more benefits than the query strategy, which implicitly reveals the practicability of the proposed framework, since the target model is hard to be decided as a prior in real tasks. ALMS does not perform well. Note that its computational complexity immensely limits the size of candidate models. Due to this defect, the best target model can hardly fall into the candidate set, which impairs the performance. Active-iNAS searches model from the simplest one initially, thus it has a different initial point with the other methods. It works well on some datasets but fails on the others. Note that this method heavily relies on the intuition of the increasing model complexity, which may not be suitable for every dataset. The rest query strategies with the fixed model perform diversely across datasets. It may be caused by the varying target models, which are searched based on the limited initially labeled data. This result also implies the importance of simultaneously considering the model and data selection in active learning, because a query strategy can perform inconsistently with different target models.

4.3 Ablation Study

To further validate the effectiveness of each component in DUAL, we conduct the ablation studies. Due to the space limitation, we report the mean values of the accuracy and number of candidate algorithms along with the querying process on 12 openML datasets instead of plotting them. 'w/o active' means eliminating the active querying part, and 'succe.' represents the successive function in model search. The results are reported in Table 2.

we can observe that DUAL significantly outperforms the methods without active sampling in both effectiveness and efficiency, which demonstrates the superiority of the data selection strategy on both exploration and exploitation aspects. Also, we find that the two ablation methods have comparable performances, but the one with successive function is much faster than another. It reveals that the function could accelerate the model search significantly without impairing the performance. These results on one hand show that the data querying strategy is helpful for both model search and accuracy improvement, on the other hand validate the efficiency of the successive function.

5 Conclusion

In this paper, we propose a novel dual active learning framework DUAL to tackle with the lack of target model prior problem by simultaneously performing model and data selection. On one hand, we propose an efficient CASH method with the truncated importance sampling to search for the best model configuration based on the skewed labeled set. On the other hand, we propose a data querying strategy to adaptively help CASH in model search or improve the winner models by active sampling. Both parts are well supported by each other to improve the final accuracy. Experimental results on 12 openML datasets demonstrate the effectiveness and practicability of the proposed framework. In the future, we would like to introduce the meta learning to deal with the scarcity of prior knowledge problem.

References

[Ali *et al.*, 2014] Alnur Ali, Rich Caruana, and Ashish Kapoor. Active learning with model selection. In *AAAI*, pages 1673–1679, 2014.

[Bergstra *et al.*, 2011] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *NeurIPS*, 24:2546–2554, 2011.

[Casanova *et al.*, 2020] Arantxa Casanova, Pedro O. Pinheiro, Negar Rostamzadeh, and Christopher J. Pal. Reinforced active learning for image segmentation. In *ICLR*, 2020.

[Du *et al.*, 2015] Bo Du, Zengmao Wang, Lefei Zhang, Liangpei Zhang, Wei Liu, Jialie Shen, and Dacheng Tao. Exploring representativeness and informativeness for active learning. *IEEE Transactions on Cybernetics*, 47(1):14–26, 2015.

[Elshawi *et al.*, 2019] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. *arXiv preprint arXiv:1906.02287*, 2019.

[Elsken *et al.*, 2019] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *JMLR*, 20(55):1–21, 2019.

[Falkner *et al.*, 2018] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: robust and efficient hyperparameter optimization at scale. In *ICML*, pages 1436–1445, 2018.

[Feurer *et al.*, 2015] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *NIPS*, pages 2962–2970, 2015.

[Gao *et al.*, 2018] Nengneng Gao, Sheng-Jun Huang, Yifan Yan, and Songcan Chen. Cross modal similarity learning with active queries. *PR*, 75:214–222, 2018.

[Geifman and El-Yaniv, 2019] Yonatan Geifman and Ran El-Yaniv. Deep active learning with a neural architecture search. In *NeurIPS*, pages 5976–5986, 2019.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, pages 2672–2680, 2014.

[Hu *et al.*, 2019] Yi-Qi Hu, Yang Yu, and Jun-Da Liao. Cascaded algorithm-selection and hyper-parameter optimization with extreme-region upper confidence bound bandit. In *IJCAI*, pages 2528–2534, 2019.

[Huang and Chen, 2016] Sheng-Jun Huang and Songcan Chen. Transfer learning with active queries from source domain. In *IJCAI*, pages 1592–1598, 2016.

[Huang *et al.*, 2014] Sheng-Jun Huang, Rong Jin, and Zhi-Hua Zhou. Active learning by querying informative and representative examples. *TPAMI*, 36(10):1936–1949, 2014.

[Huang *et al.*, 2017] Sheng-Jun Huang, Nengneng Gao, and Songcan Chen. Multi-instance multi-label active learning. In *IJCAI*, pages 1886–1892, 2017.

[Huang *et al.*, 2020] Yimin Huang, Yujun Li, Hanrong Ye, Zhenguo Li, and Zhihua Zhang. An asymptotically optimal multi-armed bandit algorithm and hyperparameter optimization. *arXiv preprint arXiv:2007.05670*, 2020.

[Hutter *et al.*, 2011] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pages 507–523. Springer, 2011.

[Ionides, 2008] Edward L Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17(2):295–311, 2008.

[Jamieson and Talwalkar, 2016] Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *AISTATS*, pages 240–248, 2016.

[Kirsch *et al.*, 2019] Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In *NeurIPS*, pages 7026–7037, 2019.

[Konyushkova *et al.*, 2017] Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *NeurIPS*, pages 4225–4235, 2017.

[Li *et al.*, 2017] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *JMLR*, 18(1):6765–6816, 2017.

[Li *et al.*, 2020a] Changsheng Li, Handong Ma, Zhao Kang, Ye Yuan, Xiao-Yu Zhang, and Guoren Wang. On deep unsupervised active learning. In *IJCAI*, pages 2626–2632, 2020.

[Li *et al.*, 2020b] Yang Li, Jiawei Jiang, Jinyang Gao, Yingxia Shao, Ce Zhang, and Bin Cui. Efficient automatic cash via rising bandits. In *AAAI*, pages 4763–4771, 2020.

[Lowell *et al.*, 2019] David Lowell, Zachary C. Lipton, and Byron C. Wallace. Practical obstacles to deploying active learning. In *EMNLP-IJCNLP*, pages 21–30, 2019.

[Sener and Savarese, 2018] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018.

[Settles, 2009] Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison, 2009.

[Shui *et al.*, 2020] Changjian Shui, Fan Zhou, Christian Gagné, and Boyu Wang. Deep active learning: Unified and principled method for query and training. In *AISTATS*, pages 1308–1318, 2020.

[Sugiyama and Rubens, 2008] Masashi Sugiyama and Neil Rubens. Active learning with model selection in linear regression. In *SDM*, pages 518–529. SIAM, 2008.

[Sugiyama *et al.*, 2007] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert MÃžller. Covariate shift adaptation by importance weighted cross validation. *JMLR*, 8:985–1005, 2007.

[Tang and Huang, 2019] Ying-Peng Tang and Sheng-Jun Huang. Self-paced active learning: Query the right thing at the right time. In *AAAI*, volume 33, pages 5117–5124, 2019.

[Thornton *et al.*, 2013] Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *SIGKDD*, pages 847–855, 2013.

[Tokdar and Kass, 2010] Surya T Tokdar and Robert E Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):54–60, 2010.

[Vanschoren *et al.*, 2013] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[Yan and Huang, 2018] Yifan Yan and Sheng-Jun Huang. Cost-effective active learning for hierarchical multi-label classification. In *IJCAI*, pages 2962–2968, 2018.

[Zhang *et al.*, 2018] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *CVPR*, pages 8156–8164, 2018.