Filtering participants improves generalization in competitions and benchmarks

Adrien Pavao[1], Isabelle Guyon[1] and Zhengying Liu[2]

1- LISN/CNRS/INRIA, Université Paris-Saclay, France
2- Huawei, China

Abstract. We address the problem of selecting a winning algorithm in a challenge or benchmark. While evaluations of algorithms carried out by third party organizers eliminate the inventor-evaluator bias, little attention has been paid to the risk of over-fitting the winner's selection by the organizers. In this paper, we carry out an empirical evaluation using the results of several challenges and benchmarks, evidencing this phenomenon. We show that a heuristic commonly used by organizers consisting of pre-filtering participants using a trial run, reduces over-fitting. We formalize this method and derive a semi-empirical formula to determine the optimal number of top k participants to retain from the trial run.

1 Introduction

Competitions and benchmarks in Artificial Intelligence (AI), Machine Learning (ML), and other scientific domains, also called challenges or contests, have been playing an increasing role in research and development. Hundreds of competitions are organized each year on platforms such as Kaggle, AICrowd, CodaLab, and others. Competitions are now part of most major conferences and their protocols are peer reviewed, to increase the rigor of evaluations and check such things as sufficient test set sizes and possible bias in data. The code of the winners is generally open-sourced, to maximize their impact. One example often cited is the start of the "deep learning era" usually traced back to the success of AlexNet in the ImageNet benchmark [5].

The ambition of competitions is generally to *recommend algorithms* that could perform well on new tasks resembling that of the competition. Thus competitions are a problem in which the *organizers* perform a learning task: from the tasks of the challenge, they select an algorithm that should perform well on new future tasks. *Organizer over-fitting* occurs when the number of participants is large and rankings are noisy, increasing the chance of poorly selecting a winner. Competition organizers face a sad paradox: *the larger the number of participants, the more "successful" their competition*, but also *the greater the risk to overfit the particular competition setting.*

A heuristic often employed in sports, chess, and other types of competition is to use eliminatory trial runs to filter participants for the final competition phase. In this paper, we investigate this strategy in the context of 2-phase machine learning competitions. To alleviate over-fitting the challenge setting, we propose to use the *development phase* to filter participants before entering the *final phase.*

2 Related problems and related work

The algorithm selection problem can be traced back at least as far as 1976 [11]. Its relationship with the problem of meta-learning and the risk of over-fitting at the meta-level has been pointed out [7]. However, to the best of our knowledge, this is the first paper that considers *judging competitions* or conducting a *benchmark* as an algorithm selection problem, and considers the risk of organizer over-fitting. This paper addresses the problem of determining *a single winner from rankings already obtained in two-phase competitions*. We do not address here the process of generating a sound and reproducible ranking. However, many related problems have been investigated in the literature: the test set size needed to get good error rate estimations [4], score distributions for stochastic algorithms [10], the selection of the worst run to reduce chance in competitions [2], or the problem of fusing scores from multiple "judges" (multiple tasks and/or multiple metrics) [8].

A question related to the problem we address is that of "out of domain generalization", which has been addressed from a variety of standpoints (although not in the context of judging competitions). For example, recent work has been warning about the danger that meta-learning algorithms may overfit the set of meta-training tasks [13]. Propositions to reduce over-fitting at the "meta-level" include algorithm-specific regularization methods like dropout [15], Bayesian mechanisms [16], and meta-augmentation of datasets [9] (which could easily be applied to challenges). Learning-theoretic bounds on the meta-generalization gap have also been proposed [1], introducing prior knowledge through setting an experience-dependent prior for novel task. This relates to the algorithm proposed in this paper: we use the *development phase* as a prior to select the winner in the *final phase*.

3 Proposed algorithm: top-k

In the sequel, we consider multiple phase competitions of benchmarks, in which participants are ranked multiple times. We assume that such rankings are drawn *i.i.d.* from a distribution of rankings. This abstracts from the need of describing more precisely how such rankings are obtained, which may widely vary from one competition to another. We revisit the *i.i.d.* assumption in Section 5.

In a two-phase competition, the *development phase* is the "trial run" during which participant are filtered, and the subsequent *final phase* is used to determine the winner. The proposed algorithm (Algorithm 1) retains the top k participants in the *development phase* and then selects the winner among them on the basis of the best rank in the *final phase*.

Clearly, keeping only a few participants in the *final phase* may save time and resources and may be logistically necessary if the *final phase* is a live competition. The question addressed in the remainder of the paper is the following: does the top-k algorithm yield as good or better (meta-)generalization than the "vanilla" method of simply selecting the winner in the *final phase*, without pre-filtering?

Algorithm 1: top-k method

Let: $n \in \mathbb{N}, n \geq 1$, be the number of participants,
$k \in \mathbb{N}, 1 \leq k \leq n$, be a chosen quota,
$rankD(i)$ be the rank of participant i in the *development phase*,
$rankF(i)$ be the rank of participant i in the *final phase*, $i = 1 \cdots n$.

Select the winner as: $i^* = \text{argmin}_{rankD(i) \leq k} \, rankF(i)$

4 Empirical results

Although, from the participants' point of view, their algorithms are being tested in the *final phase*, **from the organizers' point of view the *final phase* is used for (meta-)training**[1]. Hence we need a "*post-challenge phase*" for (meta-)testing, to evaluate meta-generalization, if we do not have direct access to the distribution of rankings (which is only possible for synthetic data). We plot meta-learning curves as a function of k, the number of participants pre-selected in the top-k of the *development phase*:

- **Meta-training error**: score in the *final phase* of the declared winner.
- **Meta-test error**: score in the *post-challenge phase* of the declared winner.
- **Generalization gap**: difference between meta-test and meta-training error.

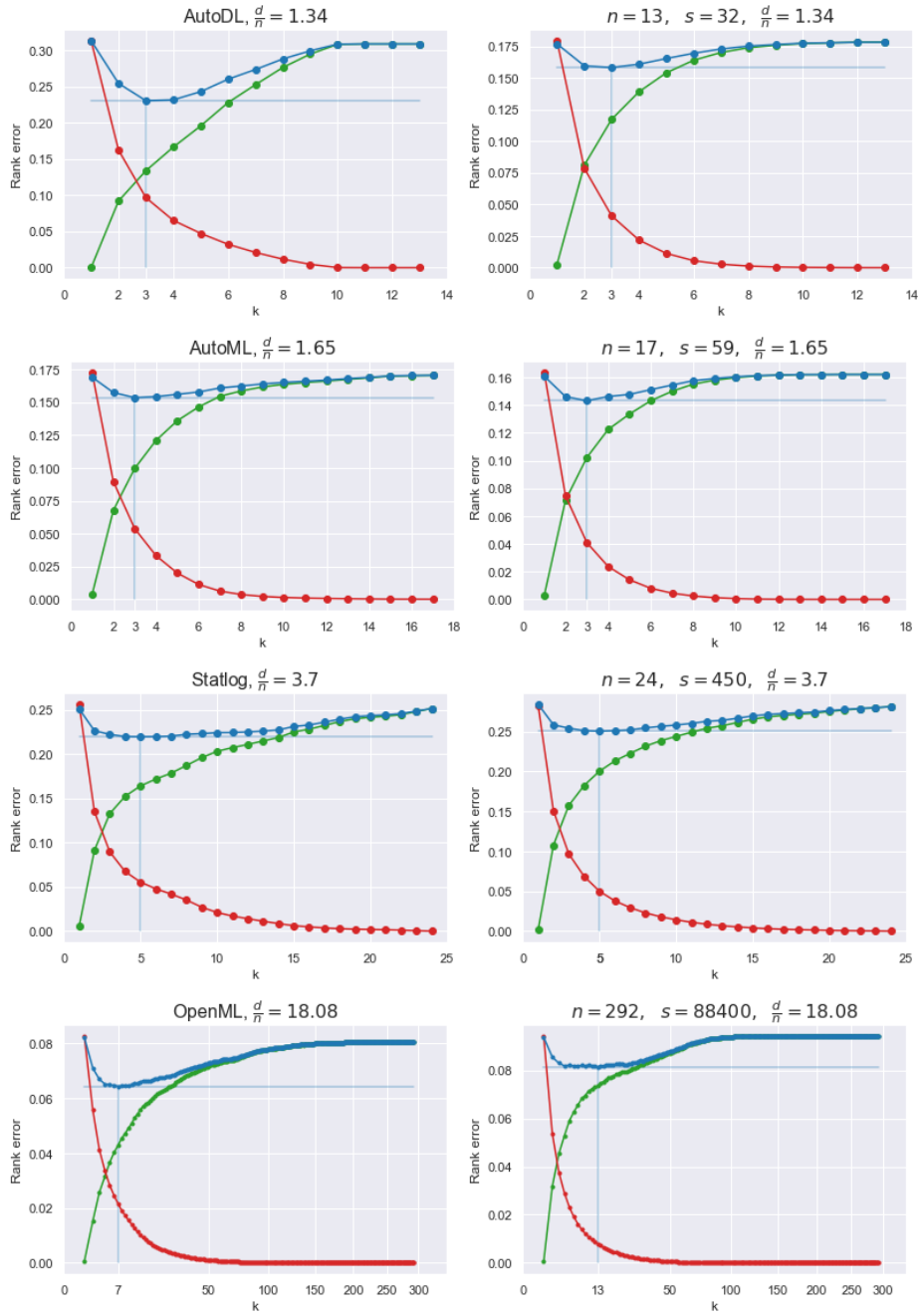
4.1 Experiments on real data

We ran the top-k algorithm on 4 real meta-datasets coming from previously run challenges or benchmarks: AutoDL, AutoML, OpenML and Statlog, provided by [14]. We simulate phases by averaging the scores on sets of tasks. The learning curves shown in Figure 1 were obtained by averaging over 10000 data-split trials. We first observe that the meta-training error monotonically decreases as a function of k. This is to be expected since the meta-training error is the minimum score in the *final phase* of the top-k in the *development phase*: as k increases, one should get smaller and smaller scores by taking the minimum over a larger subset of values. The meta-training error basically varies with $\frac{1}{k}$ [3]. We also observe that the meta-test error goes through a minimum (desired behavior). The last point represents the vanilla method consisting in selecting the candidate with the smallest score in the *final phase*. We remark that the first and last point have similar values. This is not surprising since the first point is analogous to the vanilla method, but using the *development phase* to select the winner instead of the *final phase*.

4.2 Experiments on synthetic data

To gain more insight into the problem, and understand under which condition the top-k method allows us to meta-generalize better than simply selecting the

[1]The *development phase* during which the participants "practice" (i.e. perform a form of training) is used as a prior by the organizers.



(a) REAL data (b) SYNTHETIC data

Fig. 1: **Learning curves with top-k method** (averaged over 10000 trials): Meta-training error, Meta-test error, Generalization gap. Rank error is the rank r of the selected winner in F and P, normalized by $(r-1)/(n-1)$. The horizontal bar marks k^*, and d is the average Kendall τ distance [6] between phase rankings.

winner in the *final phase*, we generated synthetic data, as follows:

Consider a competition with $n \in \mathbb{N}$ participants. Assume that there is an ideal true (but unknown) ranking of participants g (g is for "generalization"). A synthetic empirical ranking obtained in a challenge phase is generated from g by repeated permutations of pairs of neighbors: a position i is drawn at random from $\{1, \cdots, n-1\}$ and the participants i and $i+1$ are exchanged. We repeat this operation s times. The smaller s, the more the empirical rankings will be correlated to the true ranking g (and to one another). We generate in this way 3 fake participant rankings, one for each phase: D, F, and P. We can then perform similar experiments as with real data and compute meta-training error and meta-test error of the top-k method.

We show in Figure 1 the results of experiments on synthetic data next to those on real data. On each row, we took care of matching the distance d between phase rankings (as measured by the Kendall τ). We are pleased to see that qualitatively the real and synthetic data curves remarkably resemble one another: monotonic decrease of meta-training curve; meta-test curve going through an optimum; first and last point of meta-test curve nearly identical (performance of vanilla method). We observe that the optimum value of k is relatively small, even for large numbers of participants (last row). It increases with d, the distance between rankings. While s (the number of swaps applied to the ideal ranking, in synthetic data) can go to infinity, the expected value of d is bounded by half the Kendall τ distance between a ranking and its reverse, i.e. $\frac{n(n-1)}{4}$. Noting that the random process to create artificial rankings is essentially a random walk of the winner, we propose a semi-empirical formula for the optimum of k[2]:

$$k^* \simeq 1 + \frac{d}{n}$$

.

We validate the formula using simulations[3].

5 Discussion and conclusion

The remarkable resemblance between real and synthetic curves when the Kendall τ distance d between phase rankings is matched, allowed us to propose a semi-empirical formula to predict the optimal value of the number of challenge participants k selected during the "trial run": $k^* \simeq 1 + \frac{d}{n}$. However, this requires knowing d *a priori* or estimating it. A more conservative choice of k^* may anyway be preferable in practice, because when rankings are "noisy" (d large), there is a lot of variance (not visible in our experimental results, which are averaged over 10000 trials). So, instead of using k^*, we advocate eliminating participants who do not outperform the baseline methods provided by the organizers with the

[2]Although regular random walk calculations are simple, the boundary conditions imposed in this problem increase the difficulty of obtaining a simple exact formula.

[3]The code to reproduce all experiments and supplemental material can be found at `https://github.com/didayolo/metagen-esann-2022`.

"starting kit" (which may include well performing methods from previous challenges). This should be more acceptable to the participants than setting a hard threshold on the number of entrants of the *final phase*, and will at least eliminate the least serious participants who just submit the "starting kit". The validity of our paradigm rests on the assumptions that rankings in various phases are drawn *i.i.d.* from a distribution of rankings, and without ties. This assumption may be violated if, for instance, the tasks of the *development phase* and *final phase* are of different nature or difficulty, or if the participants overfit the tasks of the *development phase* (e.g. by making many submissions and seeking feed-back from a leaderboard). This last problem does not seem to be severe in practice [12].

Further work include: Extending to 3-best selection (instead of winner selection); Handling ranking with ties; Combining top-k with meta-cross-validation; Other winner selection methods; Considering more than two phases; Finalise the formal analysis of the synthetic problem provided in supplemental material.

References

[1] R. Amit and R. Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pages 205–214. PMLR, 2018.

[2] A. E. Baz, I. Guyon, Z. Liu, J. van Rijn, S. Treguer, and J. Vanschoren. Advances in metadl: Aaai 2021 challenge and workshop. *arXiv preprint arXiv:2202.01890*, 2022.

[3] Y. Gordon, A. Litvak, C. Schütt, and E. Werner. On the minimum of several random variables. *Proceedings of the American Mathematical Society*, 134(12):3665–3675, 2006.

[4] I. Guyon, J. Makhoul, R. Schwartz, and V. Vapnik. What size test set gives good error rate estimates? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[6] R. Kumar and S. Vassilvitskii. Generalized distances between rankings. In *International Conference on World Wide Web, WWW 2010*. ACM, 2010.

[7] Z. Liu. *Automated Deep Learning : Principles and Practice*. Theses, Université Paris-Saclay, Nov. 2021.

[8] A. Pavao, M. Vaccaro, and I. Guyon. Judging competitions and benchmarks: a candidate election approach. In *European Symposium on Artificial Neural Networks*, 2021.

[9] J. Rajendran, A. Irpan, and E. Jang. Meta-learning requires meta-augmentation. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[10] N. Reimers and I. Gurevych. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017.

[11] J. R. Rice. The algorithm selection problem. volume 15 of *Advances in Computers*. 1976.

[12] R. Roelofs et al. A meta-analysis of overfitting in machine learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[13] Z. Shen et al. Towards out-of-distribution generalization: A survey, 2021.

[14] L. Sun-Hosoya, I. Guyon, and M. Sebag. Activmetal: Algorithm recommendation with active meta learning. In *IAL 2018 workshop, ECML PKDD*, 2018.

[15] H.-Y. Tseng et al. Regularizing meta-learning via gradient dropout. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[16] J. Yoon et al. Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.