



Machine Learning Competitions: a Meta-Learning Perspective

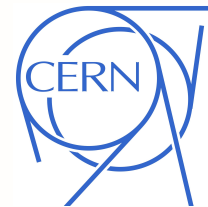
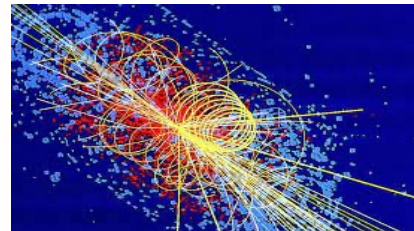
Isabelle Guyon

Adrien Pavao & Zhengying Liu

UPSaclay / INRIA



CodaLab competitions (aka challenges)



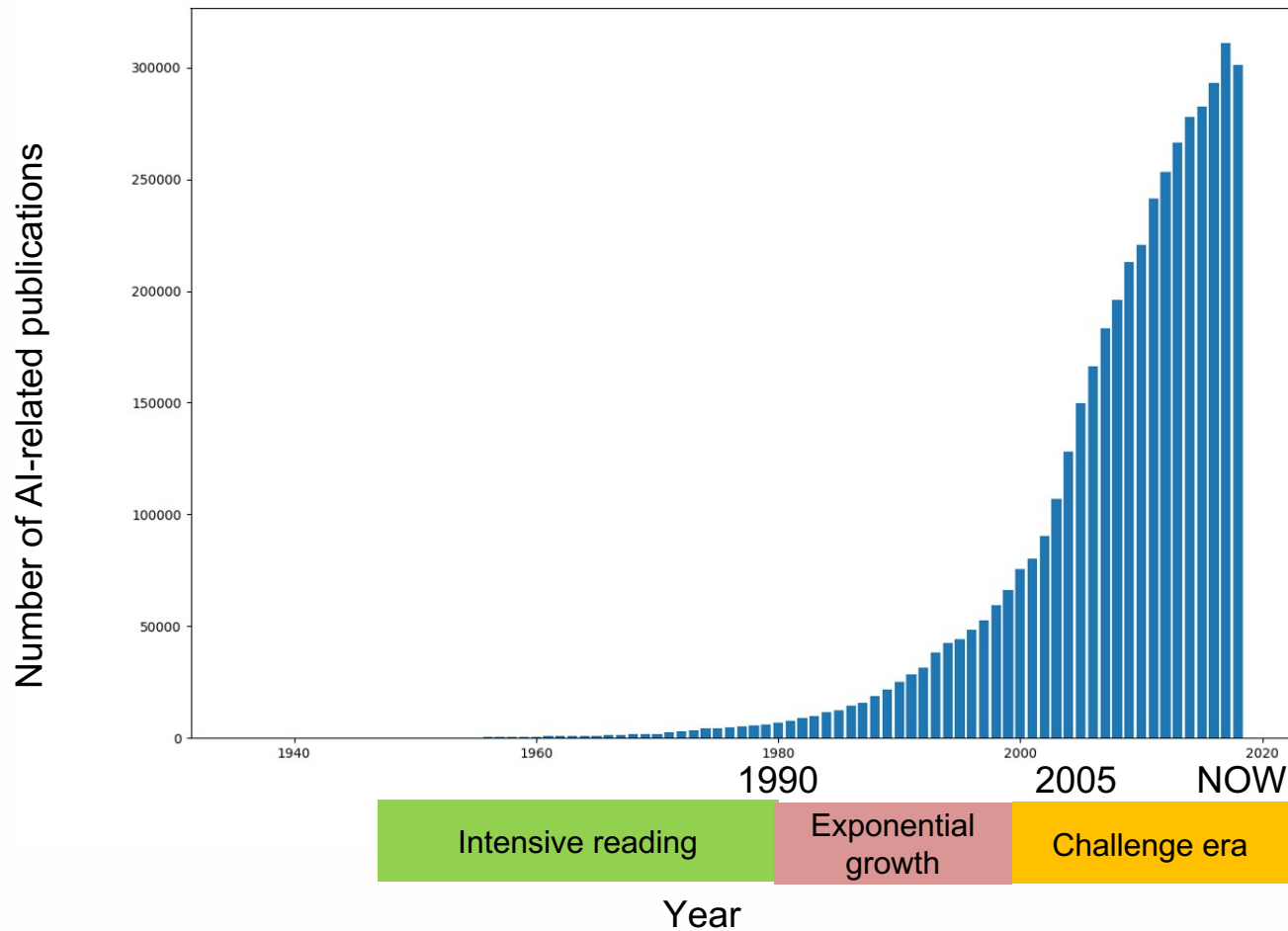
Crowdsourcing
Data Science



Large Scale Applications

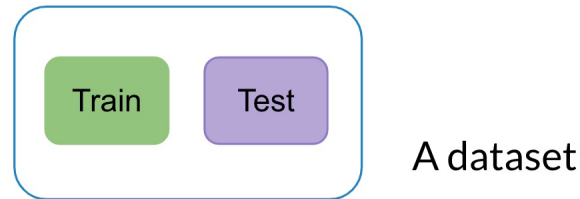


Challenge's ambition: algorithm recommendation

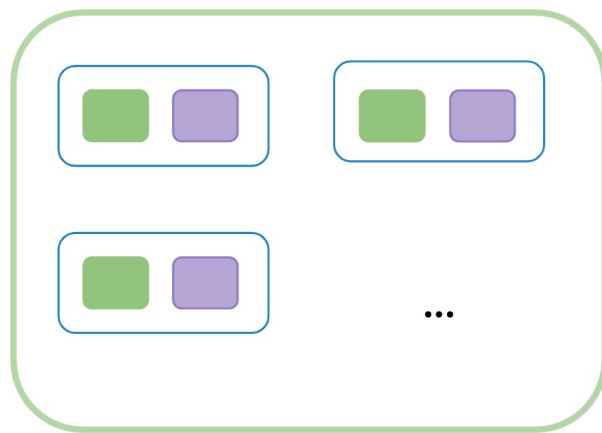


Algorithm recommendation: a meta-learning problem

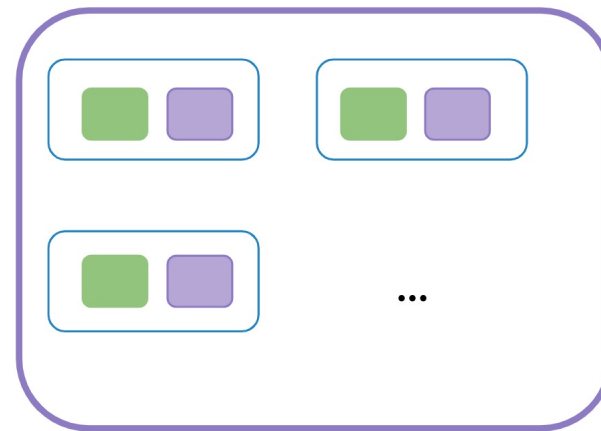
Classical ML setup



Meta-learning setup



Meta-train datasets



Meta-test datasets



Challenge organization paradox

Challenges aim at recommending algorithms

Algorithm recommendation is a meta-learning problem

Hence challenges perform meta-learning

ML can overfit training data

Could meta-learning overfit data too?

Well, yes, of course!

Challenge organization paradox:
popular competitions may
yield worse recommendations...



Challenge overfitting avoidance

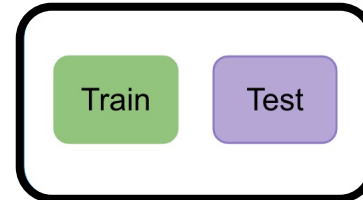
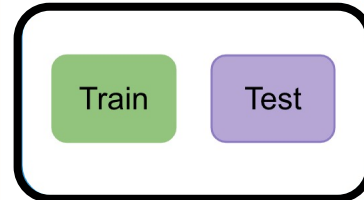
Participants should not overfit

Organizers neither!



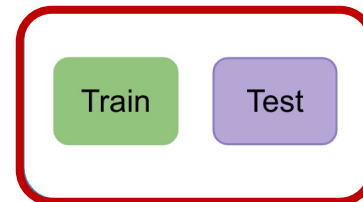
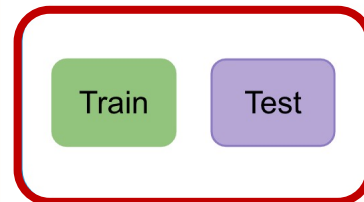
Participant overfitting avoidance

PHASE 1: Development



One (or several) datasets;
multiple submissions

PHASE 2: Final test

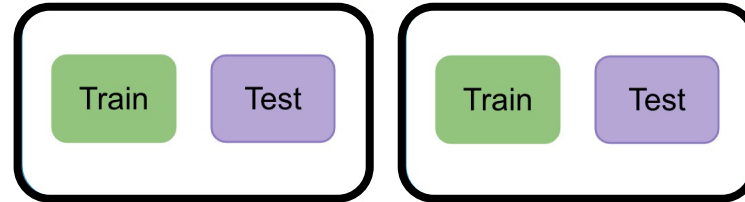


Other datasets;
single submission

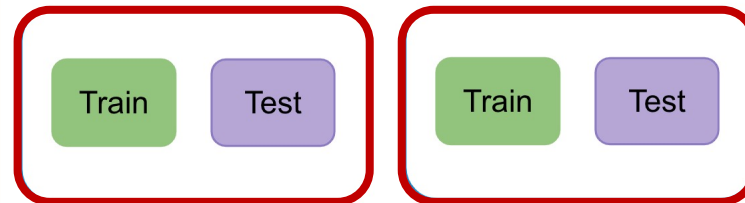


Organizer overfitting avoidance

PHASE 1: Development



PHASE 2: Final test

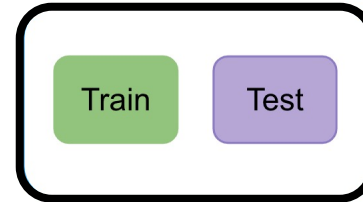
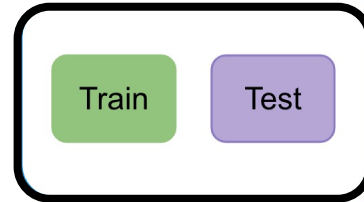


[Training phase for the organizers]



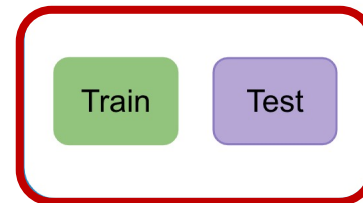
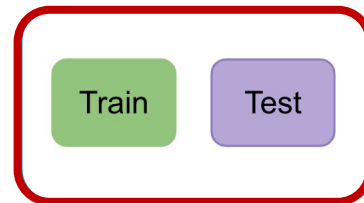
Organizer overfitting avoidance

PHASE 1: Development



One (or several) datasets

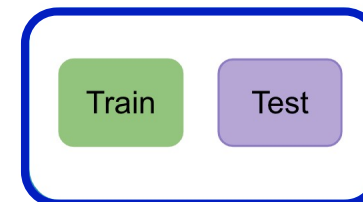
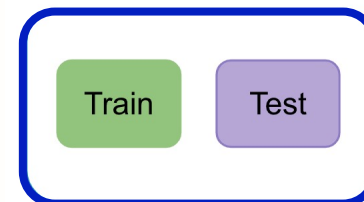
PHASE 2: Final test



Other datasets

[Training phase for the organizers]

POST CHALLENGE



Yet other datasets

[Test phase for the organizers]

Regularization?



Top-k algorithm

Select the top-k participants in the development phase [**Prior**]
[**Tom Jerry Titi Grosminet Laurel Hardy**]

Select the winner in the final phase in this subset [**Meta-training**]
[**Laurel Jerry Grosminet Tom Titi Hardy**]



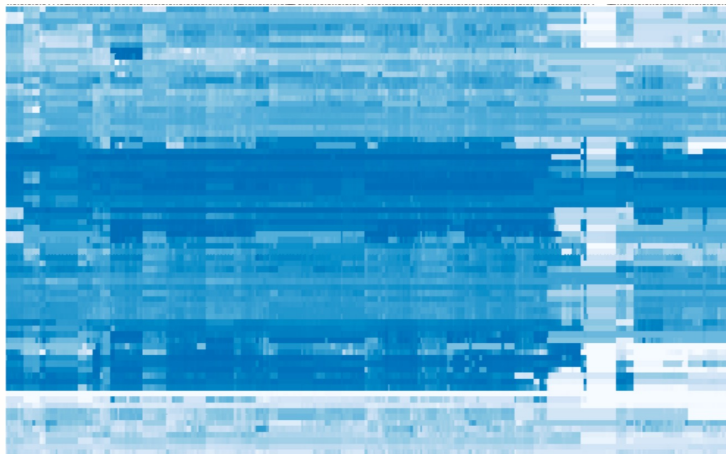
Check winner performance w. post-challenge data [**Meta-test**]

Questions:

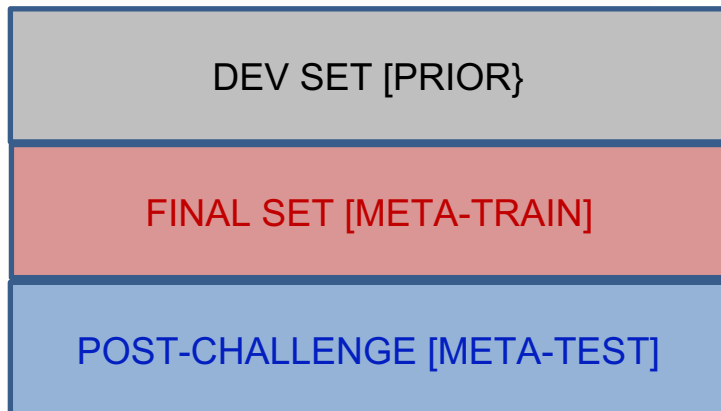
- Do we get better (meta-)generalization?
- Is there an optimal value of k?



Meta-generalization with top-k algorithm



ALGORITHMS

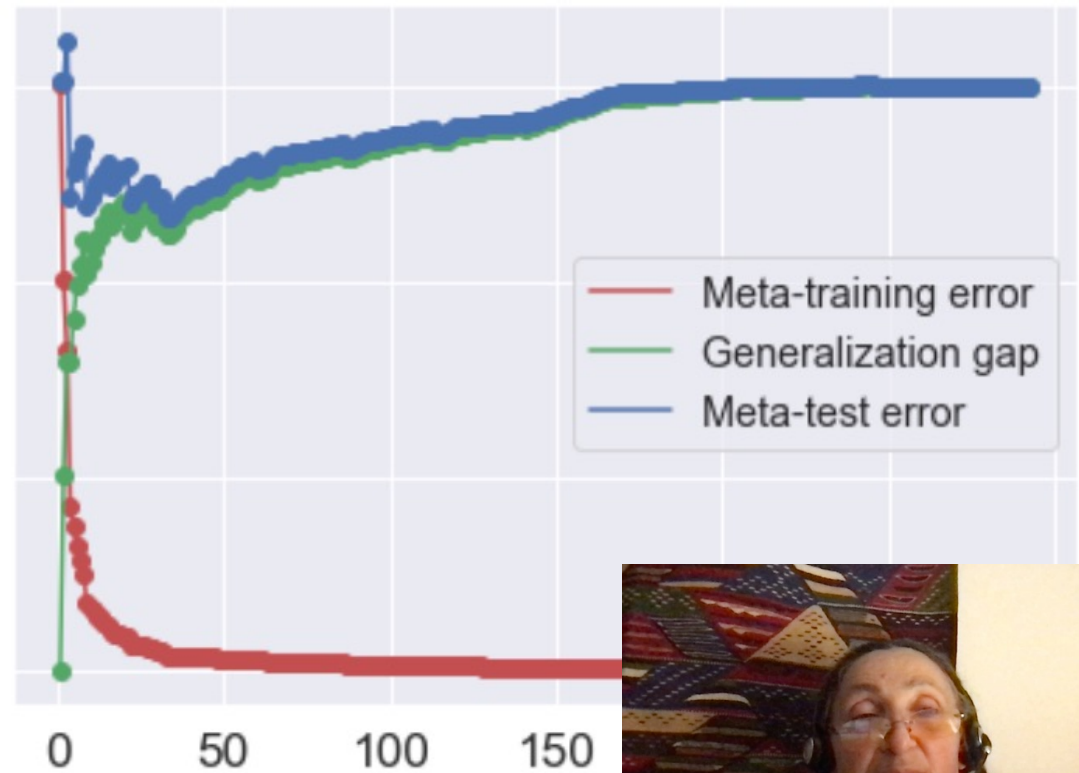


ALGORITHMS

DATASETS

DATASETS

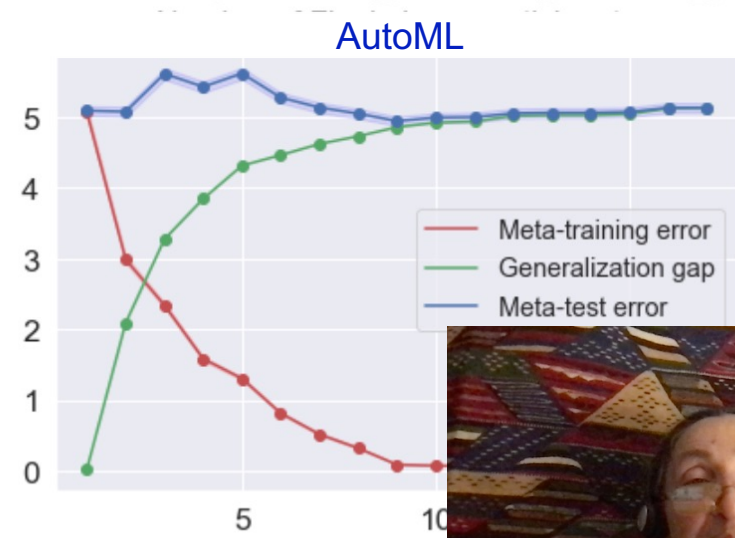
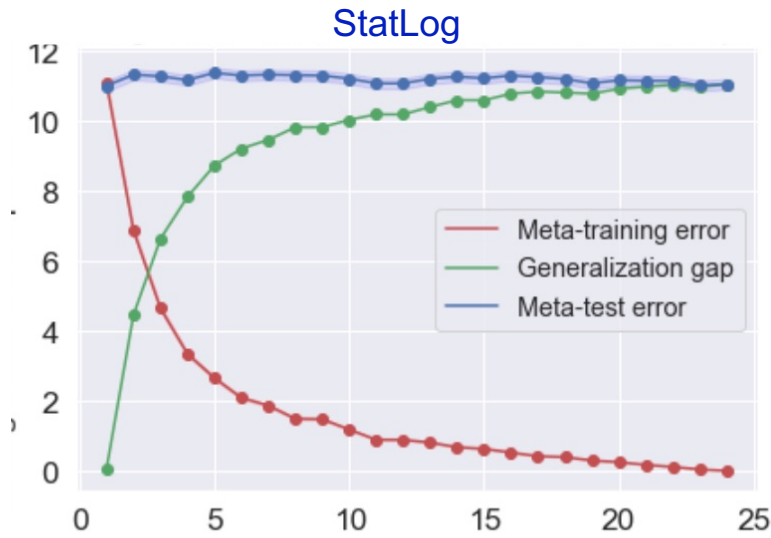
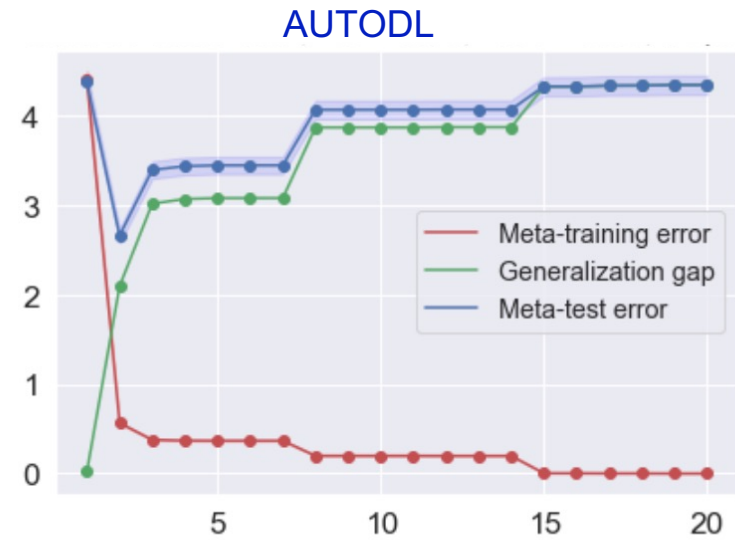
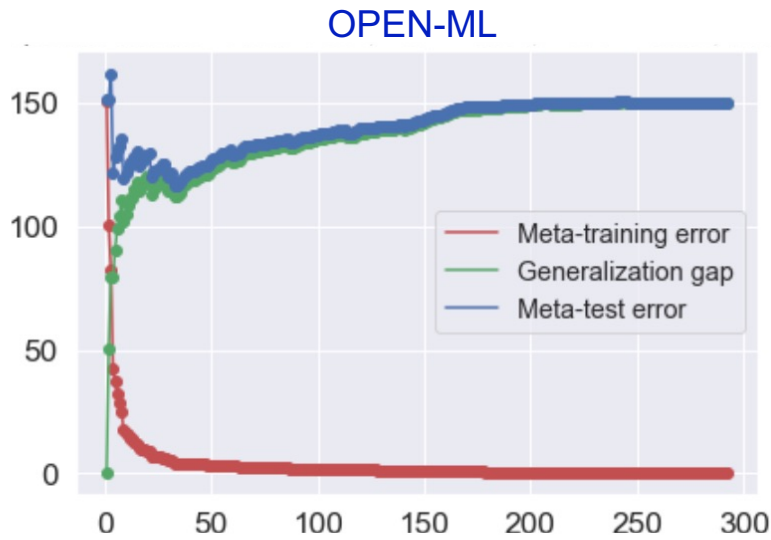
OPEN-ML



k



Top-k does not always work




k

k



Synthetic data generation

- Ideal “true” ranking: 1 2 3  4 5 6 ... n
- Choose 1 position at random in 1:n-1, and swap i and i+1.
- Repeat N time

Generate D, F and P this way:

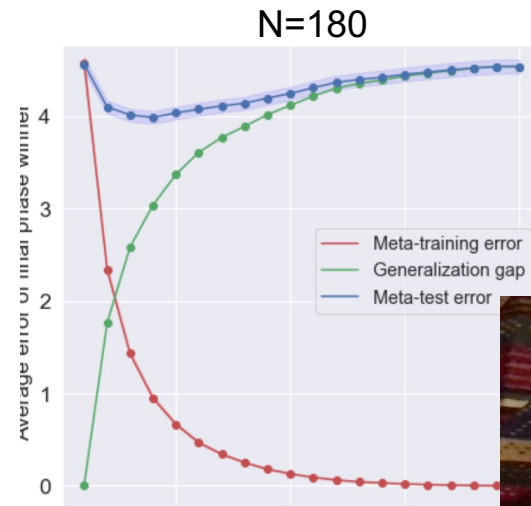
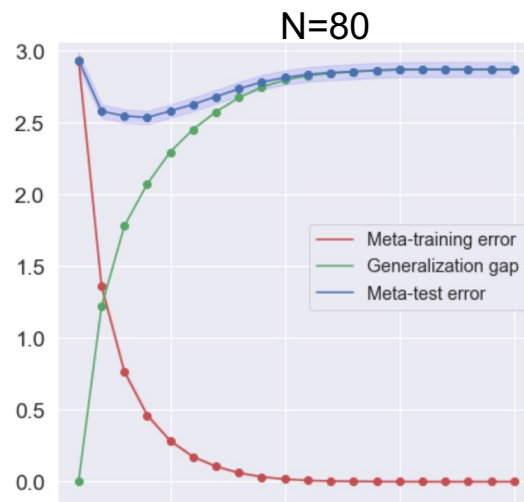
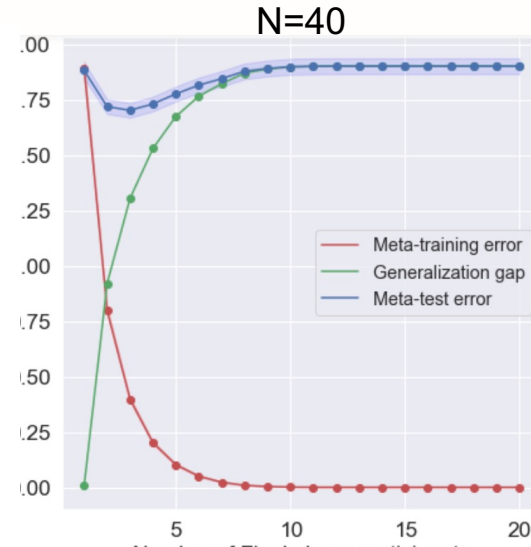
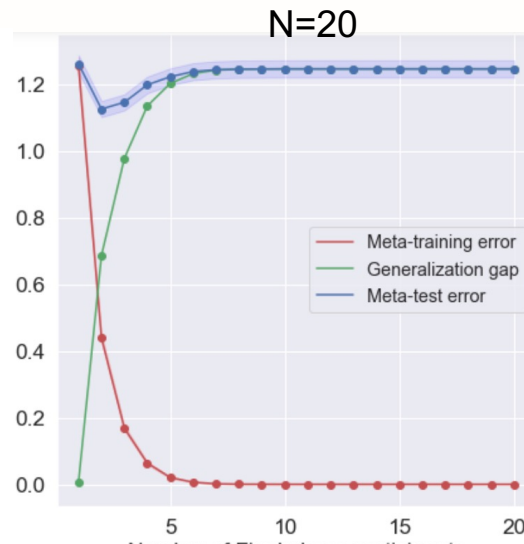
D => prior

F => (meta-)training

P => (meta-)test



Synthetic data results



k

k



Theoretical setting

- Ideal “true” ranking: 1 2 3 ↔ 4 5 6 ... n
- Development phase: 1 2 4 3 5 6 ... n
D(3)=4

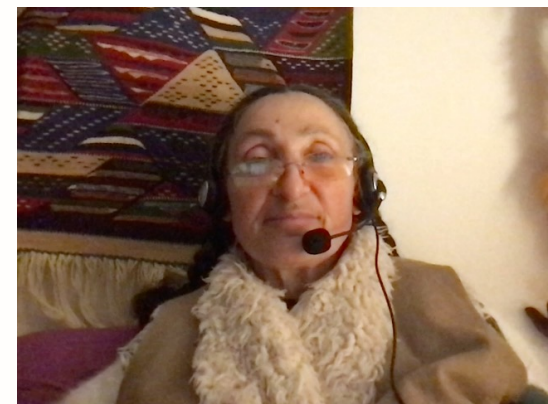
$$D(j) = i$$

$$D^{-1}(i) = j$$

$$i^* = \arg \min_{D^{-1}(i) \leq k} F^{-1}(i).$$

$$P(k) = \text{Proba}[\arg \min_{D^{-1}(i) \leq k} F^{-1}(i) = 1]$$

$$k^* = \arg \max_k P(k)$$



Optimal k value

$n \gg 1$ and $N \ll n$:

$$P(k = 1) = P(k = n) \simeq 1 - \phi$$

$$\phi = \frac{N}{n},$$

$$P(k) = \text{Proba}[\arg \min_{D^{-1}(i) \leq k} F^{-1}(i) = 1 \mid D^{-1}(1) \leq k] \text{Proba}[D^{-1}(1) \leq k]$$

$$P(1) (1 - (2\phi)^k) + \frac{3}{2k} (2\phi)^k$$

$$1 - \phi^k$$

$$\frac{dP(k)}{dk} = 0$$

$$k^* \simeq 1 - \frac{1}{\ln \phi}$$



Experimental validation: k^*

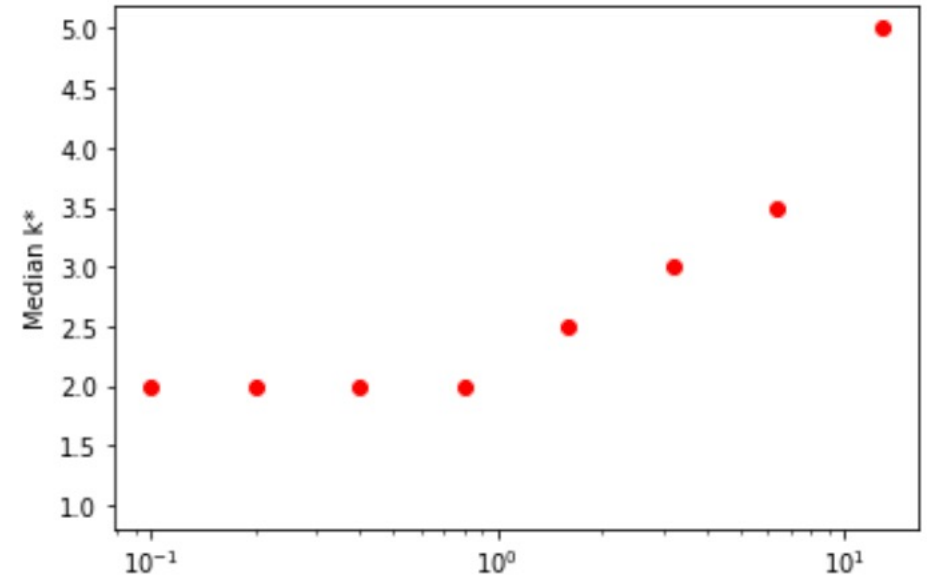
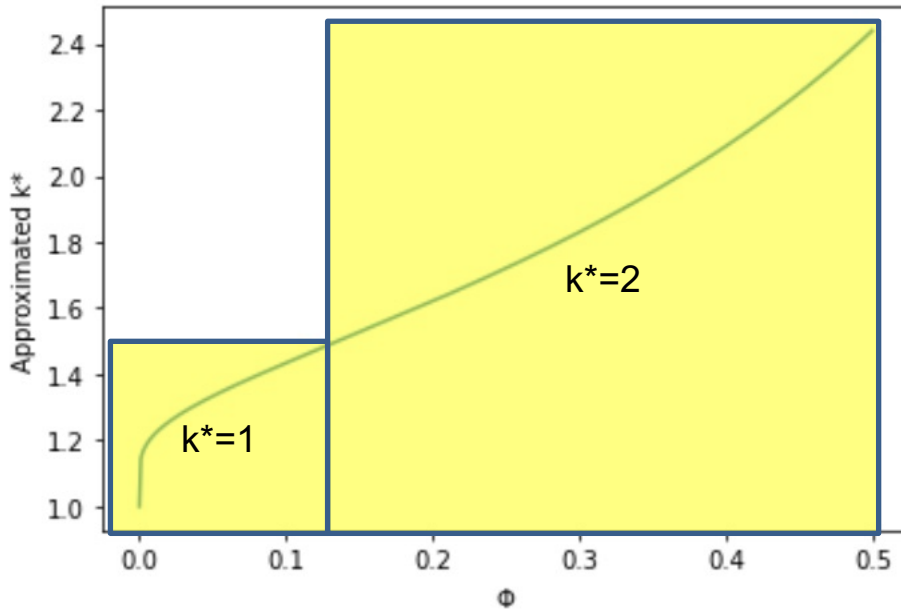
Theory for small ϕ

$$k^* \approx \text{round}(1 - 1/\ln(\phi))$$

ϕ in 0, .1, .2, .4, .8, 1.6, 3.2, 6.4, 12.8

n in 10, 20, 40, 80, 160

k^* does not depend on n , it depends on ϕ



Discussion

Organizers of challenges perform “meta-learning” to select winning algorithm.

Problem: They may overfit algorithm selection in the final phase.

Solution: Prior on participant ranking (top-k method).

This is also computationally advantageous.

But: Is using the development phase as a “prior” dangerous?

No if we assume the participants do not overfit the development phase.

But, if we use it to rank them: they have an incentive to do so!

E.g. cheating with multiple accounts, making many submissions.

In practice: Check teams, limit submission, select participants above baseline.



Conclusion

We presented the top-k method to alleviate overfitting in challenge winner selection.

- **Main result:**

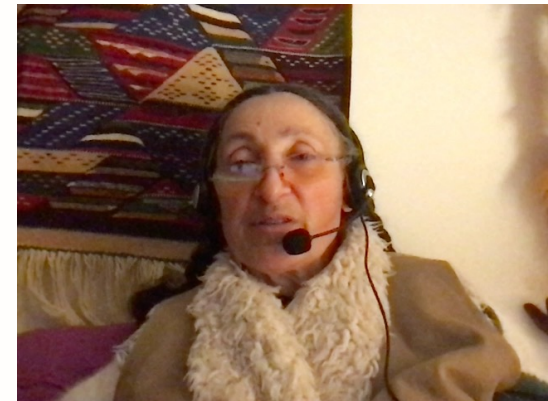
- $\phi = N/n$
- With current assumptions: $n \gg 1$ and $\phi \ll 1$
- k^* (ϕ , ~~#participant~~)
- $k^* \sim 1 - 1/\ln\phi$

- **In practice:**

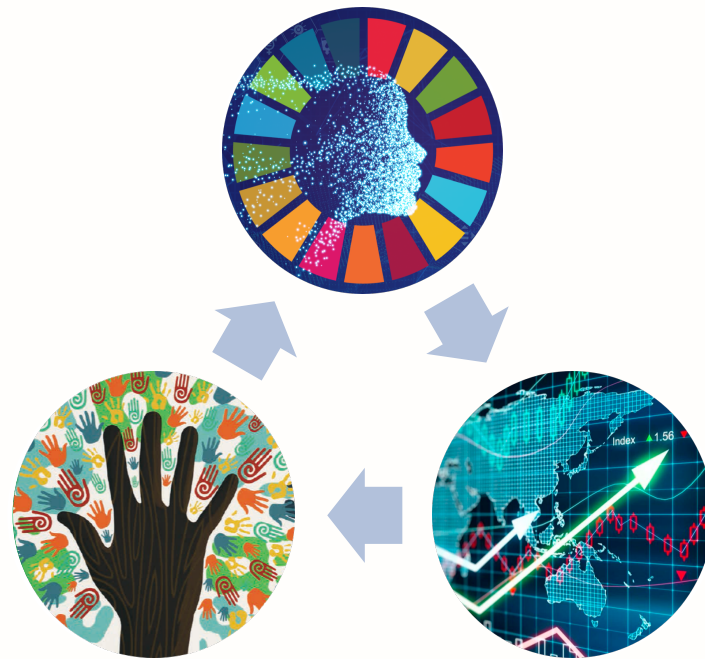
- k^* predicted very small: this may encourage dev phase overfitting or cheating.
- Just keep participants outperforming the baseline in development phase.

- **Further work:**

- 3-best selection (instead of winner).
- Handling ranking with ties.
- Selecting optimal k with meta-CV.
- Other ways of regularizing winner selection.
- Other ways of combining results of various phases.



HUMANIA



<http://guyon.chalearn.org/projects/humania>

