Foundations of Trustworthy AI – Integrating Reasoning, Learning and Optimization

TAILOR

Grant Agreement Number 952215

Challenge Guidelines

Document type (nature)	Report
Deliverable No	2.2
Work package number	2
Date	Due 28 February 2021
Responsible Beneficiary	INRIA, ID 3
Authors	Sébastien Treguer & Marc Schoenauer
Publicity level	Public
Short description	General guidelines for organizing and setting up a challenge, either academic or industrial, with an amphasis on the Codalab platform.

History & revision plan (month 6)			
Revision	Date	Modification	Author
0.9	24 February 2021	First Complete Draft	Sébastien Treguer
0.91	25 February 2021	+ few edits	+ M. Schoenauer
1.0	28 February 2021	Implemented internal reviewers' suggestions	ST + MS

Document Review		
Reviewer	Partner ID / Aconym	Date of approval
Fredrik Heintz	1 / LU	23/02/2021
Silke Balzert-Walter	26 / DFKI	25/02/2021

Contents

1 Introduction

Whereas the Imagenet Large Scale Visual Recognition Challenge (ILSVRC) launched in 2010 have helped assessing the Deep Learning revolution (DL methods won the 2012 edition, and became the only competitors soon afterward), challenges, competitions and benchmarking have long been an extremely popular format for solving AI problems of all flavors, including Learning, Optimization and Reasoning, and beyond. On the purely academic side, the SAT competitions started in 1992, and adopted a yearly format in 2002 [JLBRS12]; The AI-planning competitions started in 1998 and one of their indirect outcome was the adoption and regular improvement of the PDDL language (The Planning Domain Definition Language); Black-box continuous optimization benchmarking were initiated in 1995 to promote a fair comparison between stochastic black-box optimization algorithms, and became regular benchmarking events in ACM-GECCO conferences in 2009 [HAR+10], and are now including deterministic mathematical programming methods; The MiniZinc challenges are held every year since 2008 along with the International Conference on Principles and Practice of Constraint Programming, allowing the comparison of Constraint Programming solvers; and Machine Learning challenges existed long before Kaggle (see Section 5.1), within the PASCAL European Network of Excellence (2003-2008), including the Visual Object Classes (VOC) challenge [EVGW+10], pioneering the image recognition challenges; the Interspeech Computational Paralinguistics ChallengE (ComParE) addresses since 2009 issues related to states and traits of speakers as manifested in their speech signal's properties. The DARPA challenges are also well-known for boosting research on more applied topics, like the autonomous vehicle in desert surrounding in 2004 and 2005, and in Urban context in 2009, and now going to subterranean tasks – explicitly advertising their quest for ”out of the box” solutions. Another well-known challenge in the field of robotics is the Robocup series, associated with the IJCAI and IROS international conferences, that started with the world-wide well-known football cup in 1996, but now concern more useful tasks like the Robocup Rescue in challenging environments. Last but not least, large industrial companies also realized the benefits they could gather from organizing challenges to solve their particular problems: The Netflix 1M$ prize in 2009 increased the accuracy of Netflix recommendation system by more than 5%; Total aims at improving their autonomous exploration robot in gas and oil environments with the ARGOS challenge; Microsoft organized the Malmo Collaborative AI Challenge, targeted to multi-agent systems.

However designing and organizing an AI and data science challenge is a challenging project in itself, and its success is a multi-criterion objective, requiring many different skills, more human resources than one might think, and incentives for participants that generally can materialize in money prizes, but definitely not limited to it. Before launching such a project, it is recommended to leverage on the experience learned from previous challenges and for new organizers to get advice from experienced AI practitioners as well as organizers of past challenges. As for many scientific projects, there are no absolute rules that can guarantee the success of an

AI competition challenge. However, based on the experience of past challenges, it is possible to extract some guidelines to avoid many common pitfalls, and setup the proper conditions for success, i.e. minimizing mistakes and maximizing positive outcomes, inline with the initial objectives, starting by a clear definition of what are these objectives.

This document aims at providing some guidelines to help AI challenges organizers in their task, and is more specifically targeted toward challenges related to TAILOR program. In particular, it will assume that the challenge to be organized is a purely software challenge, focusing on the Codalab platform (see Section 5.2) to support its organization, though most material in this text applies whatever the platform used. But issues regarding hardware setups (e.g. for robotics challenges) are out of the scope of this document.

Several challenges will be organized during the course of the TAILOR project: Academic challenges will be proposed in the Research Work Packages 3-7 while applied challenges will arise from Work Package 8 and the Theme Development Workshops that will be organized therein. All should deal with TAILOR topics, i.e., should address the combination of Learning, Optimizing and Reasoning. This should impact the definition of the different tasks in the challenge (Section 2), the type of data needed, and in particular the biases to avoid (Section 3) and should drive the way the candidates will be ranked (Section 4), possibly requiring some multi-objective measures of success to assess the different dimensions of the challenge. Potential challenge organizers are welcome to read and comment this document, and to propose improvements after they have experienced on their own the complete organization of a challenge.

2 Challenge Design

This Section surveys the questions that challenge organizers must ask themselves before even starting to organize a challenge, then focuses on the main goal and objectives in organizing this challenge, the target audience, the scientific questions addressed by the challenge, and the issues related to the data.

2.1 Important Questions

Designing an AI competition challenge involves many choices over many dimensions, among which (this list does not pretend to be exhaustive):

- What are the appropriate AI tasks corresponding to the main questions to be answered? Machine Learning tasks (binary classification, multi-class or multi-labels classification, regression, recommendation, policy optimization), Optimization tasks (continuous, combinatorial or mixed; single- or multi-objective), Reasoning (logic or probabilistic, planning, constraint programming, . . .), or, within TAILOR, combinations thereof.

- What are the application domains, if any, of the data and tasks? Medicine, neuroscience, environment, economy, finance, transport, visual arts, education, music, cross-domain, etc.

- What are the data types or modalities ? Tabular, univariate or multivariate time-series, image, video, text, speech, graph, multi-modal (combining multiple modalities)

- Are the data readily available? Are all data publicly available or should some data be kept secret?

- What are the most appropriate evaluation metrics to assess the performance of the submissions? Quantitative metrics, like final accuracy, area under the curve, area under the learning curve, mean square error for Learning tasks; SSIM, SHIFT, SURF for image similarity; best objective value or time to reach given precision, for Optimization tasks; success or failure, time-to-solution for Reasoning tasks; or qualitative metrics possibly implying human evaluation (i.e., by some expert committee).

- Code submission vs submission of results on test sets: Is it better to ask participants to make code submissions and run all code submissions on the challenge platform, or is it sufficient to simply ask participants to submit results obtained on some holdout samples of a test set?

- Which ranking method of the participants/teams is the most appropriate and fair?

- Which baselines are appropriate? From random to state of the art (SOA), with various degrees of complexity, whether the organizers want to lower barrier to enter or push the limits of SOA forward.

- What should be thought in advance, written and specify in the challenge rules, in order to drive participants in the right direction and give the limits of ”the game” to avoid possible complaints?

- What kind of content, degree of details is required for the documentation? What make a good documentation adapted to the profile of the participants?

- What are the tests procedure and time required to be confident enough before launching the challenge?

- Would the challenge benefit to be conducted in association with a conference workshop, or a dedicated competition tracks of a conference, as it exists now in many AI conferences like NeurIPS, IEEE FG, IJCAI-PRICAI? Then what would be a good time schedule for the different phases of the challenge?

- What kind of communication, targeting which community of potential participants, through which channels, with which schedule to attract the targeted community?

2.2 Main goal and objectives of the challenge?

First of all, the overall goal and objectives to be reached through organizing the challenge should be very clear.

- Is it to drive the research community into a specific direction by addressing a specific research problem and pushing forward the boundaries of the state of the art (SOTA)? This will probably be the main goal of all TAILOR academic challenges.

- Is it to get some insight for the R&D of your company, and transfer some results from research to address some of your applied problems? This will most probably be one of the objectives of TAILOR industrial challenges.

- Is it mainly a communication medium, to promote the research activity of your lab, group or company? This is very often a secondary objective for all challenge organizers.

- Is it to attract the attention of students, and recruit some new talents in your fields of interest? This is likely to be one of the goals of industrial challenges.

- Is it to discover new applied methods and launch them in production with minimal adaptation efforts – here again likely to motivate some industrial challenges.

- Or is it a mix of two or several of the above mentioned motivations?

Whatever the motivations, it is crucial to be very clear on the main scientific questions and objectives for the participants, and rank priorities in case of multiple scientific objectives. Clear answers to these questions will help you define the target audience, and the scientific questions you will ask with your challenge (see Section 4).

2.3 The Target Audience

It is important to define the target audience, in order to design a challenge which is attractive enough and adapt the level of difficulty with a barrier to enter that is not too high. However if the target audience is a mix of beginners and more experienced practitioners in Artificial Intelligence, a crucial issue is to find a sweet spot, to set the barrier low enough to allow for beginners to enter without too much headache, while keeping the competition challenging enough for experienced practitioners. Lowering the barrier to enter can be achieved by providing a good documentation along with a simplified tutorial in a starting kit, providing compute resources to make it accessible to anyone, not only people with own access to farms of GPU or TPU. And at the same time, keeping the problem to solve interesting enough for experienced practitioners might require several levels of difficulty, several phases of the challenge. Choose carefully a subject that is interesting for your targeted audience at the time of the competition. Choose the start date, time length, and time investment required, according to the targeted audience. If you target researchers, make sure that being successful doesn't involve too much engineering time efforts with respect to the scientific contribution, and coordinate with other conferences, workshops and other competitions in the field.

2.4 The Scientific Questions

Wait a minute, what is the main problem we want to address and would like to be solved? Asking the good questions is key to get results inline with the initial goals.

What are the objectives of the challenge? Is our priority to address scientific questions, and which ones precisely, or to get as outcomes models easy to transfer to a production system with all its constraints in terms of robustness, explainability, performance monitoring, maintenance, etc ?

Is the only objective the final accuracy at the end of training without constraints on resources: compute, memory, time, etc? Or should the applicants also take into account limits in training time, computer power, memory size, etc, with the goal to find the sweet-spots for good trade-offs?

The definition of each task to achieve must help to solve a specific question raised by the challenge, but must also carefully take into account all constraints and reflections previously mentioned.

Then what are the constraints in terms of data: volume, balance or unbalance of classes, fairness, privacy, external vs internal, etc ? These questions are related to the tasks that are themselves related to the initial questions to be addressed.

For each scientific question, you will then need to define some metrics allowing to measure how well each participant answers the question: More details in Section 4.1.

3 Data collection and preparation

In the following, dataset will mean an entire collection of data allowing to run the tasks and measure the performance of the participants. For instance, for supervised learning, this amounts to a set of labelled examples; for reasoning, it is generally a set of context instances (e.g., SAT instances); for combinatorial optimization, this could be a set of instances of input data and constraints defining different problem instances (e.g., list of cities for the TSP), whereas for continuous optimization, it generally amounts to a set of functions, either analytically defined, or computed using some dedicated code.

3.1 The three phases

A typical challenge can be split in three phases, each phase being supported by a different dataset:

- Phase 1: local training supported by a public training dataset, that each participant can download or access, and make experiments with in order to tune one's proposed approach on one's own compute resources.
- Phase 2: feedback phase supported by a first test dataset, not seen or known by participants but providing feedback results to submissions from participants and used for ranking all participants on the public leaderboard
- Phase 3: final blind test phase supported by a private test dataset, not seen, not known, ideally never used in any previous AI competition, and not known to the AI community. This final private test dataset is used only once to evaluate the last submission from each team or participant, and make the final ranking.

3.2 Data Collection

Each dataset should be chosen with care in relation to the tasks, and objectives of the challenge. This choice is not neutral. At first it is important to evaluate and set the difficulty of the dataset, with respect to the target task, at an appropriate level. As said, regarding the target audience of participants, the difficulty will be set at intermediate level, not too high to attract enough participation and avoid discouraging too many possible candidates, and not too low to make the challenge interesting enough.

Another point to take into consideration is to be able to differentiate the submissions of the different participants with sufficient significant variations. For that purpose, in addition to choosing a good metric (see Section 4.1), again it is important to set appropriate intermediate levels of difficulty, avoiding too easy or too difficult instances that could lead to very close submission results, whose differences would be not significant enough.

The dataset for the final blind test phase (phase 3) should be kept as secret as possible. If possible not known or accessible to anybody in the target community. Otherwise some participants could figure out by using metadata and/or making extensive experiments, which dataset you are using. Or if you are using a subset or a dataset derived from a known distribution, some participants could by chance, or on purpose, take advantage of existing solutions. for instance in ML challenges, they could benefit from neural networks pretrained on the same data distribution and leverage the feature extraction of such models to transfer it to other kind of tasks. If possible, it is recommended to try to find, or build, datasets that have never been released. If the privacy of these data is critical, this constraint can be turned into an advantage, since these data do not need to be released and can be kept secret even after the challenge has ended. Another benefit is to be able to reuse it for future other challenges.

3.3 Real vs Synthetic Data

In addition to collect data from real dataset, it is also possible to synthesise datasets with the appropriate characteristics.

Choosing real data "into the wild" requires to ensure the legal right of use. Is there a license attached to the whole dataset, or to some samples? Is there any specific law to be respected, regarding data privacy or discrimination issues ? If you collect data from the internet or social media, make sure to check for offensive contents.

Collecting a sufficient amount of "clean" data, e.g., annotated with labels in the case of supervised learning tasks, or representing actual use cases in the case of optimization tasks, with appropriate distributions and difficulty level, that can be used and publicly released for training purpose, can represent a major hurdle, be time consuming and costly, and sometimes is simply not possible. In essence, in case of rare events, like some decease, fraud or anomaly detection, collecting a sufficient amount of appropriate samples can prove to be extremly difficult, if at all possible.

In such case, generating synthetic data, that mimics real data, can be tempting, in order to get enough volume of data for training, public and private test datasets. In addition it allows to keep a closer control of the generated data distribution, without issues related to privacy, licence of use, or offensive content, but it comes with its challenges and drawbacks.

In particular, if you choose to generate synthetic data, it is especially important to ensure that the relative performance of any two algorithms (train and test) on the synthetic dataset is similar to their relative performance (train and test) on some real dataset.

Synthetic data, might be "too perfect", missing edge cases that exist in real data, and having a distribution that is also too smooth. The result might then poorly generalize on real data. Remember for instance that naively randomly generated SAT instances either don't have a solution, or are trivial to solve (the "phase transition" issue [GW94]). This is why there are several tracks in SAT competitions (e.g., random track, real-world track) and different algorithm win different tracks.

3.4 Dataset related to a task

For instance, if the task is about ML classification with generalization ability across domains and/or modalities, the various datasets, the ones provided for training during the development phase, the ones used for public leaderboard and the final private ones for final blind tests, should be designed accordingly, from different domains/modalities, not revealed for the validation and private test set. If the competition is based on results over a test set provided without labels, make sure to format the data in a way that the domain/modality can't be inferred. If the task is about generalization across modalities, the data can be formatted in a specific way or projected to a specific representation of vectors or tensors, making them looks similar whatever the modality in order to hide this information to the participants.

3.5 Common biases

3.5.1 Why should I care?

First it is important to be aware that most, if not all, big datasets, whether collected from digital sources or from the physical world, are biased some way or another. But why is it so critical to be aware of biases in the data, identify and mitigate them? After all, biases will be the same for all competitors.

- For legal reasons: For instance if the challenge consists in matching the best candidates with job positions, the data have to respect laws against gender and ethnicity discrimination. The responsibility of the challenge organizers (and of TAILOR as the umbrella organizing institution) is at risk.

- For fairness and ethical reasons: Obviously nobody wants oneself, or one's organization, being caught in the storm of a scandal for ethical reasons. This should be a universal concern, even more so in TAILOR, where the "T" stands for "Trustworthy".

- To strengthen trained models robustness and generalization capabilities: If the challenge is a classification problem, make sure you have a balanced number of samples from each classes. If the challenge involves to deal with unbalance datasets, like for classification of rare events or detecting anomalies, at least make sure the dataset include some diversity that is representative to the real data. For instance if the competition task is to detect fraudulent financial transactions, make sure the data and metadata include all kinds of possible fraudulent and none fraudulent transactions from a diverse set of user profiles, from a diverse set of institutions, with various amounts, . . .

3.5.2 Data Leakage

Data leakage is a common pitfall to avoid in competitions. It can be defined as ”the creation of unexpected additional information in the training data, allowing an algorithm to obtain unrealistically good results.”

- **Target leakage**

  This leakage takes place when some information present in the training data, and not at the time of prediction in real life, giving the model a exploitable information that is highly correlated with the target to predict.

  For instance, in **supervised learning** competitions, target leakage can happen when some features happen to be highly correlated to the target concept in the training set. One well-known example of such leakage is the tank recognition task, that reached 100% accuracy in the recognition of Russian tanks . . . because on all images of Russian tanks, the background was full of snow.

  In **optimization**, this can happen if all training instances share some characteristics that can be used by the optimization algorithms, e.g., if the minimum of all test function is the origin in continuous optimization, or if the same set of variables is irrelevant in all test functions.

  How to prevent such target leakage? This is of course challenge-dependent. But in case of tabular data, a look at variables that are highly correlated with the target can help to detect it. In case of images, a look at weights associated to each input pixel can help to identify which are the most used area in the images to make predictions and see if the model focuses on the correct information.

- **Out-of-core leakage**

  In ML tasks, this kind of leakage happen when unexpected information, out of the core content of the data, and usually not available in real life, are used to improve predictions on the challenge data. For instance, challenge participants can find some exploitable leakage in the metadata. An example of this happened during The ”Kaggle - ICML 2013 Whale Challenge - Right Whale Redux”, where a competitor found leaks in:

  - The distribution of file lengths
  - The timestamp embedded in the audio clip filename
  - The chronological order of the clips

  For a more detailled explaination of the leaks that have been identified and fixed, take a look at this kaggle post

- **Dataset leakage** In some cases of machine learning competitions, some participants/teams are able to infer the dataset used for the public leaderboard. It is more likely to happen if it is a publicly known dataset in the ML community, or if it has already been used in previous competitions, or if some participant are expert in the specific domain of the challenge. For instance if the challenge is about predicting diagnosis of autism syndrome from fMRI data, some participants with strong expertise in the field or connections with institutions holding such data, could restrict the possible datasets to a very limited number of existing datasets and, by making some targeted test submission, figure out which dataset is used. This would give them an unfair competitive advantage and bias.

Similarly, well-known test suites exist in all fields of optimization (see e.g., the OR-library for combinatorial optimization), and you should not use them in your challenge.

How to prevent such dataset leakage? First, use datasets that have never been publicly released for the validation on public leaderboard, and of course for the final test dataset. Another recommendation is to write clear rules in the ”Terms and Conditions” of the challenge (see Section 7), forbidding the use of other data than the public dataset provided by the organizers. Or to allow external data, under the condition to inform the organizers about it, and release it to all participants.

3.5.3 Preprocessing

Is it better to provide raw data, or data that have already been preprocessed? For instance in neuroscience, EEG data can be collected at various sampling rates, so it is common practice to resample data at a similar sampling rate. Moreover, raw EEG signals comes with artefacts, from motion or eye blinking, etc. So, is it better to release the raw data or a preprocessed version, and which one precisely? There is no easy single answer to this question, that would fit all challenges.

An argument against preprocessing is the potential loss of information induced at this step. An argument in favor of preprocessing is that it gives much more robustness to the submitted code if they only have to handle data that have been preprocessed the same way.

Anyway, it depends again on the main goal of the challenge, is it relevant regarding the tasks designed for this challenge, the expertise of the participants, the amount of time they can dedicate during the duration time of the challenge.

3.5.4 Labeling

In case of an ML challenge based on supervised learning tasks, are the data already available with labels or is it needed to label them? Is it feasible and reasonable to produce the labels internally by the organizing team or is it better to rely on crowdsourced labels? This raises not only the issue of the confidence level you can have in the labels themselves, but also some ethical questions about the labelling process itself.

3.6 Distribution

How to release the access to data? Is it possible, and if yes, is it preferable, to release the data through an API that can be queried by the participants? Or is it possible, and if yes, is it sufficient, to allow the participants to download the complete datasets?

Also, depending on the organization of the challenge, and the possible multiple phases of increasing difficulty, what schedule should be used to release the data?

3.7 Privacy and Ethics

TAILOR network is primarily concerned with trustworthiness, and hence with ethic considerations, especially privacy preservation. Four important deliverables of the project are focused on these issues, that should be read in detail before fetching and releasing data for a challenge: The Ethics Requirements 1, 2, & 3, Deliverables 13.1, 13.2 and 13.3; and the Data Management Plan, Deliverable 1.6. However, a few ideas are worth mentioning in the specific context of challenge organization.

Since the adoption of the Universal Declaration of the Human Rights by the assembly of the United Nations in 1948, Right to privacy (article 12) is a fundamental right of individuals.

In recent years, due to the increase in volume and diversity of the personal data collected and the computational power and technical progress to process them, there has been an increase in privacy and ethical risks, which has lead to the implementation of the European General Data Protection Regulation (GDPR), applied in practice since 2018. As a consequence, the respect of data privacy and more broadly ethics, is a first priority concern, and not just a ”nice to have” feature, for any AI challenge organizer.

One possible concern is that data collected about individuals shouldn't be “re-used” for a different purpose without asking their consent.

Moreover, with modern AI approaches, the possible inferences are getting more and more powerful, fine-grained and accurate. For instance, with social data, in some cases it could be possible to reconstruct the social graph and infer political opinions, religion, sexual orientations, hobbies, . . .

Which inferences are considered ethical enough, or at least acceptable, and which ones are not?

Does the challenge data present a risk of privacy leakage? If so, how to make the data anonymous before releasing it, in order to limit the privacy risks without loosing relevant information, and some of its predictive power?

Here again, in the context of TAILOR, it is recommended in case of any doubt to refer to the Data Management Plan (Deliverable 1.6) to involve your Data Protection Officer, and to consider performing Data Protection Impact Assessment, DPIA.

As a first principle, release only the minimal necessary information to carry out the AI task, with the appropriate performance level, to answer the initial question corresponding to the main goal of the challenge.

Privacy concerns could be an additional motivation to choose a code submission process rather than a submission process based on results over a test set that would have to be released.

Privacy concerns could also be a strong argument in favor of synthetic data rather than real data, but in such a case, be very careful to properly evaluate the quality and diversity of the synthetically generated datasets (see discussion in Section 3.3).

4 Result Assessment and Ranking

We will now take a close look at how to rank the participants, so that the winner actually answers the question asked by the challenge. This goes back to properly and accurately define the main goal of the challenge from a scientific point of view (see Section 2.4).

4.1 Metrics

What are the most appropriate metrics to asses performance of each submissions from participants with respect to the initial question and objectives of the challenge?

What really matters:

- In case of an ML challenge, is the final performance at the end of the training the only metric that matters or does the shape of the learning curve and the speed of convergence also need to be taken into consideration?
- In case of an ML binary classification, regarding the initial question to be addressed what is the relative importance of precision and recall, i.e. true positives/(true positives + false positives) and true positives / (true positives + false negatives)?
- In case of classification tasks, with unbalanced classes representation, like for anomaly detection or fraud detection, accuracy is obviously inappropriate. Consider instead using an F-score.
- Even for pure optimization challenges, there are several possible points of view balancing precision of the result and computing time: An overall computing time has to be given in any case. It is usually measured in number of function evaluations, to be hardware-independent, assuming all evaluations have the same cost. But should precision be favored, measuring only the best objective function value obtained in the total available time, but making it difficult to compare algorithms on different test functions, or should also the time to reach a given precision be considered (that can be different for functions of different difficulties) [AH05] ?
- How important is the explainability of the proposed approach? And how to evaluate an explaination? In particular, can it be done automatically, or does it require human evaluation?
- Combining multiple criteria, multiple datasets, or multiple judges: as already mentioned, but something that is probably even more prominent for TAILOR challenges, that will involve several aspects of AI, many challenges in fact involve several criteria – and this is particularly true in the context of TAILOR, that should have at least two dimensions among Learning, Optimization and Reasoning. But should they then be aggregated into a single number, or should some kind of Pareto front be considered as ex-aequo winners?

Whatever choice you make here, it is important to describe it clearly in the rules of the challenge, ideally unveiling the piece of code that will actually be used to compute the ranking.

However, there are general recommendations to follow for an unambiguous ranking.

4.2 Variance

You must pay attention to the variance in the results. It is not only important to determine clear winners in the final phase, when the last submission of each participant/team is used to assess the final ranking on the private test set. It is also important in the context of TAILOR, as a small variance over the test set demonstrate a high robustness of the results, i.e., a larger trustworthiness.

In order to evaluate this variance, organizers can repeat the scoring of the last submission of each participant/team over several runs. At this stage it is crucial to ensure that the difference between top participants is statistically significant. Otherwise a clear rule to fairly separate them should be written in the competition rules.

It is important to determine the source of the variance: does it come from the code of the organizing team? From the data ingestion program? from the scoring program? or from the code of the participants/teams in case of a code submission based competition. One way to reduce the impact of this variance is to announce that statistical tests will be performed between the results of the different candidates. The organizers can choose to take for final ranking the average of n evaluations, n being greater or equal to 3, of the last submission for each participant/team, and to compute p-values for different confidence levels. Once the variance from ingestion and scoring program are eliminated or limited as much as possible, it is possible to require from participants/teams to also reduce the variance of their code submissions, by also setting each random seeds in their code. However it can be tedious to check for this in the participants/teams code. Another possible avenue for organizers is to add an incentive to reduce the variance. For instance organizers can decide that they will take for final ranking the performance of some bad quantile (e.g., the 75% quartile or even 90% decile), from n evaluations of the last submission. Whatever the chosen solution, it should be clearly disclosed before the competition starts, and written in the competition rules.

4.3 Ties

Ties are not really an issue during the public feedback phase, but become important for the final ranking, as their might be some equal rewards for the top ranked participants/teams. In case of similar performances, or if the difference between two participants/teams is not statistically significant for the chosen statistical test, it is important to set a clear rule, written in the competition rules, so as to avoid complaints from the participants. If two participants have the same score, w.r.t.

the evaluation metrics, and if they rank similarly, and are eligible for prize, either the prize is shared, or an additional rule is applied, that was written in the ”terms and conditions” of the challenge. For instance, the advantage can be given to the one who submitted first, or the relative ranking could also take into account the computational cost of running the code if it is a code based submission process, or some additional challenging instances could be kept holdout only for the purpose of breaking ties.

4.4 The public leaderboard

In many AI competition, one of the key elements is the leaderboard that ranks participants/teams during the feedback phase, and the final evaluation phase. During the feedback phase each participant/team gets a feedback for its submissions through a publicly displayed learderboard. During this phase, which can involve code submissions or results submission against a test set, submissions are generally evaluated against some hold out test instances unknown from participant. However it is possible for participants to integrate repeated feedback information they receive from the learderboard in the design of their solution and start to overfit the public leaderboard. In such a situation, overfitted submissions will most probably generalize poorly on the private test set used for final ranking and as a consequence the final ranking can be significantly different from the ranking displayed by the public leaderboard. However, organizers should try to minimize this behavior and effect, and make the public leaderboard to better reflect the final ranking of participants/teams. One way to deal with this is to limit the number of re-submissions per participant/team; another workaround is to limit the precision of the given public score. However, this does not provide any theoretical guarantee. A recently proposed solution is to display for each participant/team only submissions that beat their last best solution, by at least some fixed margin unknown to participants/team . . . or according to some given statistical test. this limits the information provided by the public leaderboard and therefore limits the possibility of overfitting the public leaderboard.

4.5 Prizes

Last, but that should in fact come first, it is important to decide and advertise the prizes that will be distributed to the top ranked participants. Scientific glory can be the only reward, but unless the challenge is already well-known and advertised worldwide, this might not be enough to motivate the participants to spend enough time to fulfill your goals (Section 2.2). A straightforward incentive is money. You do not need to try to match Netflix or DARPA 1M$ prizes, but if you have enough budget (private sponsors, supporting European project, . . .), be sure to distribute more than one prize, and balance the amounts between the 3 or 5 top ranked participants. If you are supported by a conference, you can offer some free registrations, though most people do not pay registration fees from their own pocket. You can also offer visits of research centers, of industrial plants that are not generally open to the public, etc. It is a matter of imagination, but can tremendously raise the interest in your challenge, and bring in many participants.

5 Platforms

5.1 Kaggle

Born as an independent company in 2010, Kaggle is owned since 2017 by Google LLC. Hence it can be seen as a communication medium for Google tools and especially Google Cloud Platform (GCP). Nevertheless, it is a great platform not only to organize a competition challenges, but also to communicate, as it has a very large and diverse community of participants, and users with various goals: learning, experimenting, gaining visibility, etc. In June 2017, Kaggle announced that it passed 1 million registered users, or Kagglers, with a community spanning over 194 countries. The competitions have become very fierce, and the level needed to reach top positions is very high. As a consequence, some participants/team are dedicating a lot of human and computing resources, in order to gain a tiny advantage over other competitors: This is often leading to overfitting the public learderboard (e.g., by submitting many slight variants of a given approach to get more feedbacks and use it for optimization) and/or proposing solutions made of large ensembles of many models, which are difficult to transfer later into production.

5.2 Codalab

Codalab is an Open Source framework and platform, with a web interface, hosting hundreds of data science competitions each year. Codalab takes its roots in the scientific ML research community, and has been extensively used to run data science and all sort of AI competitions since its creation in 2013 as a join effort between Microsoft and Stanford University. It has an emphasis on research, providing a flexible tool to ease collaboration through competitions, gathering a wide range of contributions on the same framework, making solutions more easily comparable and reproducible. Even if the hosted competitions are more scientific and research oriented due to its origins and community, its flexibility allows to setup a wide range of data science competitions, dealing with machine learning, optimization, reasoning or other advanced computational approaches, helpful to address challenges from many fields, whether in academia or in industry, using any kind of data modalities, images, videos, tabular, text, speech, graph, . . . Moreover, being an open source framework, it offers the organizers two possibilities, either to setup their own instance and host the competition on their own computing resources, or to host the competition directly on Codalab platform. Codalab offers two ways to set up a new challenge, either by using a web interface for ease of use for less technical users, or by using a command line interface for more powerful functionalities and more flexibility, at the price of a more complex interaction with the platform.

Codalab a great tool to organize any kind of data science competition. But CodaLab is more than that, it is also a great tool to help making research more reproducible, as it offers to any researcher to keep track of one's own experiments, to share it with others, to re-run it and reproduce the original results, to display them on a web interface dashboard, to link these results to the files used to produce them, to look at the code, and eventually the associated paper, all this in a few click. For a demo of how Codalab works, see this 2 minutes video.

For a basic tutorial and more technical details go to Appendix A

6 Tests

Before releasing any competition to the public, it is important to make extensive tests, at different levels and scales. And once again, the human effort that is needed for these tests should not be underestimated.

6.1 Different Type of Tests

- **Technical tests**
  These are the classical bug hunting tests, that should be first performed within the organizing team, at least the technical members of the organizing team. As usual, all branches of the code should be tested, i.e., all levels of participants should be mimicked, and if possible several variants of actual submissions (e.g., different versions of the baselines) should be submitted successfully.

  As it is difficult to predict the number of participants/teams and their level of activity, some tests should be made to tests the capacity of the infrastructure to handle the workload. If time permits, it is a good idea to test the limits of the infrastructure in terms of workload in order to be able to set the limits with a bit of leeway. For instance the number with a limit of submissions per day for each team, or a limit of compute time per team per day.

- **Scoring program**
  Ensure that the scoring program provides appropriate measure in any possible cases, including edge cases. Ensure that scores are consistent over several repeated experiments with similar configurations. At first start with toy examples to check the validity of results, then progressively increase the complexity of methods to ensure the results of the scoring program is coherent.

  In case of code submission challenge, if several frameworks are supported, for instance Tensorflow and Pytorch for deep learning, it is needed to evaluate how implementations of similar classic methods or baselines are performing in both framework, and repeat tests to evaluate their variance even after fixing the seeds, as they might make use of the hardware resources differently.

- **Data and baselines** As mention earlier it is import to check the data don't include any offensive content, especially for, images, text, speech.

  With respect of the task(s) of the challenge, some datasets can be more challenging than others, so it is recommended to test the performance of classic methods, that could be released to participants as baselines, or not, in order to evaluate the difficulty of the task on these data and define the right balance between difficulty and compute time. This will allow to set time bounds per submission, taking into account the task, the data, the compute resources, the expertise of the targeted participants and the duration of the challenge.

- **Functional and documentation tests**
  These should be made first by the less technical members of the organizers,

then opened to the circle of friends that are not familiar with this specific competition.

6.2 Some Best Practices

- Breaking tests in small fractions. It helps to isolate issues faster and fix it also faster. It also make it possible to distribute tests over a larger number of testers and accelerate the iterative process of tests and fixes.

- Designing and writing tests cases as soon as possible, ideally from the very beginning. It is beneficial to think about tests early in the development of the competition, and write test plans accordingly, so that tests can start early in development cycle of the challenge, along the challenge implementation without waiting for every pieces to be finished

- Writing tests with maximum coverage.
  It seems obvious to write tests cases for valid conditions of usage, but it can be also valuable to write tests for borderline conditions, and even to think about unexpected conditions/behaviors (not all participants will always be rational).

- Making notes and reports for all tests performed.

- Using testers from highly diverse origins: Internal and external, experts and beginners, technical and functional.

7 Documentation

Documentation is often the last but should definitely not be the least part on which organizers dedicate time and efforts. It may sounds obvious but whatever the type of challenge, and the level of expertise of the targeted participants, it is important not to neglect the documentation, nor the time and effort required to produce a good one, since the first interaction participants will probably have with the challenge is by reading the documentation, and decide if it is worth their time and efforts. As currently said, "there is only one chance to make a positive first impression."

But what makes a good documentation? The goal of the documentation is to help with the on-boarding of all participants, lower the barrier to enter the challenge, and motivate the potential users to actively participate. This can only be achieved by providing all the useful information and elements of understanding to enter in the challenge and make relevant submissions. It has to be well structured and adapted to all targeted participants profile, whatever their kinds and levels of expertise.

What should a documentation contain a minima?

- **An overview** of the challenge.
  It should help anyone to quickly get the big picture and make his/her mind to dive deeper or not.

- **The schedule**
  It should provide the start and end date of each phase if the challenge is made of several phases, but also the time window for teams to merge if relevant and allowed.

- **The task** It must provide a concise description of the task to be solved, with a bit more details than in the overview to understand the specificity of this challenge, pointing out the difficulties to overcome to reach the main goal of the challenge. It can be interesting to position the challenge relatively to previous works in the field, including previous challenges, scientific papers and benchmarks.

- **The description of the data**
  Usually it would describe the type of data, their format, how the public data for the first phase is provided – should it be downloaded and where, or is by making requests on an API, and how?

  Be careful however not to disclose any information that could be a leakage about the test datasets whether the one used for the public leaderboard or the private test dataset used for final evaluation (see Section 3.5.2).

- **The evaluation metric(s)**
  The documentation should provide the mathematical formulation of all quantitative metrics used for the final ranking, as well as their textual explanations, especially if they are not standard ones, in order to provide a better

understanding of the specificity of the challenge and to minimize wrong interpretations. The best way to avoid misunderstanding is to provide the code of scoring/ranking program used.

- A description of **baselines** (Optional: if baselines are provided)
  Not all participant might be expert in the field of the challenge. Some might even take advantage of the challenge as a way to learn more about the problem and existing solutions before experimenting some of their ideas. Therefore, if some baselines are provided they should be clearly detailed, even if they are trivial, and a link to the corresponding research paper or resources should be provided for more details.

- **How to make a submission**
  Whether it is a code based submission process, or results over a test dataset, the process of making a submission should be well explained and ideally an example of a well formatted submission should be provided in the starting kit.

- **Terms and conditions**
  It can be based on a General Rule Terms, as a basis for most AI competitions, and add some specific rules appropriate for each specific challenge competition.

  As an example, some of the specific topics to be covered are:

  - Conditions of participation
  - Registration
  - Anonymity
  - Submission method
  - Reproducibility
  - Prizes
  - Dissemination

  To make sure it is complete, protective enough for the organizers, but not without liable commitments that will prove difficult to fulfil, it is recommended to ask professional lawyers for advice.

- **Starting kit** Even if a piece of code that is explicit enough can in some cases be seen as documentation, a good practice is to properly document the starting kit, with a textual description, usually provided as a README.md file, if the starting kit is made as a shared git repository, complemented with additional comments in the code (see Section 8 below).

8 Starting Kit

Even for participants with a strong expertise and experience in AI competitions, a starting kit is valuable to give them a better idea of how things are implemented and guide them through the submission process and make a first valid one. It is a key element to ease the understanding and accelerate the on-boarding of participants from any expertise levels.

It is often implemented and shared in the form of a git repository, as in the example provided below.

The starting kit can be structured with several nested levels of directories covering the following parts:

- **Overview**
  The starting kit should provide a short overview of the competition, its goal, task(s), data, eventually with a bit of context, and links to additional resources for more detailed information. At least a link to the competition website. It is also a good practice to describe the structure and content of the git repository.

- **Installation process**
  The starting kit should guide participants to setup their own environment ready to run the code of the starting kit, which ideally includes a tutorial. In case of code submission challenge, it is crucial to ensure that every participant and the organizers are on the same page in terms of code environment, and able to run the code of the starting kit whatever their hardware platform and operating system.

- **Requirements**
  In case of code submission challenge, the starting kit should cover all the libraries supported and dependencies with the required versions. In practice, it is often provided in a requirements.txt file, so that all dependencies can be installed with one command line, e.g., in Python

  ```
  pip install −r requirements.txt
  ```

- **Data**
  At first, the starting kit should describe the format of the data, where and how to access it and how to fetch them. It can be provided in the form of an ingestion program, with the loading functions already implemented, or left at the responsibility of the participants, but anyway it should be well described to ease the access and loading, ideally illustrated with code in a tutorial.

- **Scoring program**
  An implementation of the evaluation metrics should be included in the starting kit, so that participants/teams can test and evaluate their approaches

- **A tutorial**
  There should also be in the staring kit a clear understanding of the competition

workflow. A good idea is also to provide a first data exploration analysis. It is often made of a Jupyter notebook, in order to mix executable code, textual explanations and graphics, or code files but in such a case make sure the associated documentation and code comments are complete and explicit enough.

An example of a starting kit is available in the github repository of the MetaDL challenge, run between October and December 2020.

9 Schedule and Promotion

It is important to plan a proper schedule, suitable for the scientific/technical tasks to be properly addressed, and suitable for the targeted communities of participants to dedicate enough time. For instance if organizers are targeting students in a particular region, it is obviously recommended to avoid their exam periods, as well as their holiday time. If the targeted participants are researchers/engineers in a specific field, make sure to avoid conflicting with deadlines of papers submissions of the main research conferences in the field. Whoever are the targeted participants, it is a good idea to go and talk to some of them to ensure to avoid such schedule conflicts.

As mentioned in Section 2, a good way to promote an AI challenge is to register it as an official competition of some important conference in the field, or to associate it with a conference workshop, for conference that don't have a dedicated competition tracks. It can be valuable for the challenge organizers, the participants and the whole community, academic researchers as well as practitioners in industry, and in some cases help to build links and bridges between research and industry. In such a case, be careful of the deadlines of the "call for proposal" of the targeted conference workshops and competitions.

Then plan to dedicate enough time to get ready for each phase. One way to address this is to backpropagate from the end date. Then organizers should ask themselves relevant questions related to the organization schedule and dependencies, among which:

- If relevant, when is the conference competition track, or associated workshop?
- How much time is required between the end of the competition and the conference, to make final evaluations with all checks, and officially disclose the final results? Inviting participants to submit papers to an associated workshop, eventually inviting top participants to an oral presentation of their solutions at the conference workshop, requires to take into account the schedule of the associated conference workshop/competition track.
- How much time should be allocated to run the final phase? In case of test results submission, it simply requires to run the scoring program on the tests results submitted by participants. In case of code submissions, it requires more time for the organizers, as they need to run the complete submitted codes (e.g., including training for machine learning challenges).
- How much time should be given to participants for the development phase with feedback on the public leaderboard, in order to tackle the tasks of the challenge?

10 Conclusion

Now after reading this guide you can go ahead, better prepared to take up the challenge of designing and launching trustworthy AI, Learning, Optimization and Reasoning challenge competitions.

However keep in mind:

- Organizing the challenge you have been dreaming of will most likely require more time than initially planned, even after having read this, as organizers regularly have to face unexpected hurdles that are likely to happen on the way. In order to get prepared it is recommend to make sure to have enough resources, starting with skilled human resources to be able to cope with expected and unexpected difficulties. In particular, for TAILOR challenges, the help you will get from TAILOR technical staff will be limited to technical implementation, and will not concern the specifics of the challenge, that will remain your responsibilities.
- Clearly state the main goal and objectives of the challenge, this lays the foundation, from which everything else will depend and can be adequately defined and built, like the targetted communities of participants, to the tasks, data, evaluation metrics, ranking method, etc.
- Make sure to have the proper data available at hand, and check for any potential legal, ethical and privacy issues.
- Carefully design adequate combination of AI tasks-datasets to answer the main question to be solved.
- Make sure the evaluation metrics gives the right incentives, and that the ranking method is fair and robust
- Carefully write the terms and conditions to prevent any possibility of ”cheating” the metrics and ranking and also to cover against edge cases and behaviors as well as legal issues.

References

[AH05] Anne Auger and Nikolaus Hansen. Performance evaluation of an advanced local search evolutionary algorithm. In Proc. IEEE Congress on Evolutionary Computation, volume 2, pages 1777–1784, 2005.

[EVGW+10] Mark Everingham, Luc Van Gool, Christopher Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. International Journal of Computer Vision, 88:303–338, 06 2010.

[GW94] Ian P. Gent and Toby Walsh. The SAT Phase Transition. In A. Cohn, editor, Proc. 11th European Conference on Artificial Intelligence. John Wiley & Sons, Ltd, 1994.

[HAR+10] Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In Proc. ACM-GECCO 2010, page 1689, Portland, Oregon, USA, 2010. ACM Press.

[JLBRS12] Matti Järvisalo, Daniel Le Berre, Olivier Roussel, and Laurent Simon. The International SAT Solver Competitions. AI Magazine, 33(1):89–92, March 2012.

Appendices

A Codalab

As mention in section 6, about competition platforms, Codalab is an open source framework that can help anyone to setup a data science competition either on its own servers, cloud provider, or on Codalab ones and providing the tools to make research more easily reproducible.

A.1 basic tutorial

There are two ways one can interact with Codalab platform, either using the web interface, for ease of use, or the command line for more powerful functionalities. For each competition, the first step is to upload the required material in a zip archive, named a **Bundle**.

This Bundle should contain files/directories with data, for training, validation and test, code with the scoring program, eventually some baselines, and appropriate documentation.

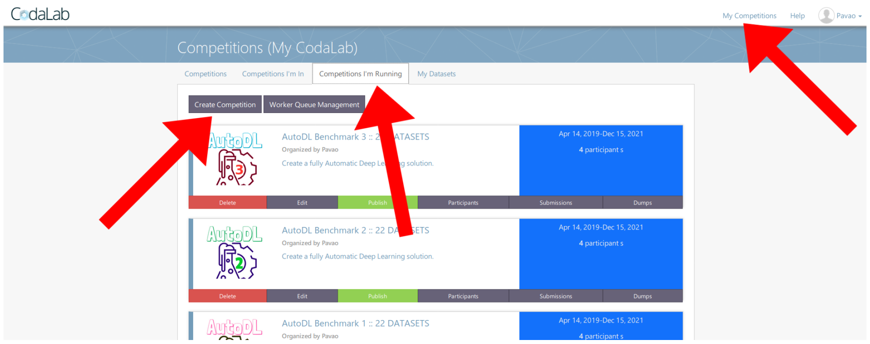
Here is a step by step procedure to upload a competition Bundle on Codalab through the web interface.

1. Users without an account yet should create one with the sign up button on Codalab competition platform (top right)

2. Sign in

3. Click on ”My Competitions”, ”Competitions I’m Running”, ”Create Competition”

4. Upload the archive ”competition_bundle.zip”.



AutoDL Benchmark 3 :: 2 ATASETS
Organized by Pavao
Create a fully Automatic Deep Le g solution.
Apr 14, 2019-Dec 15, 2021
4 participant s

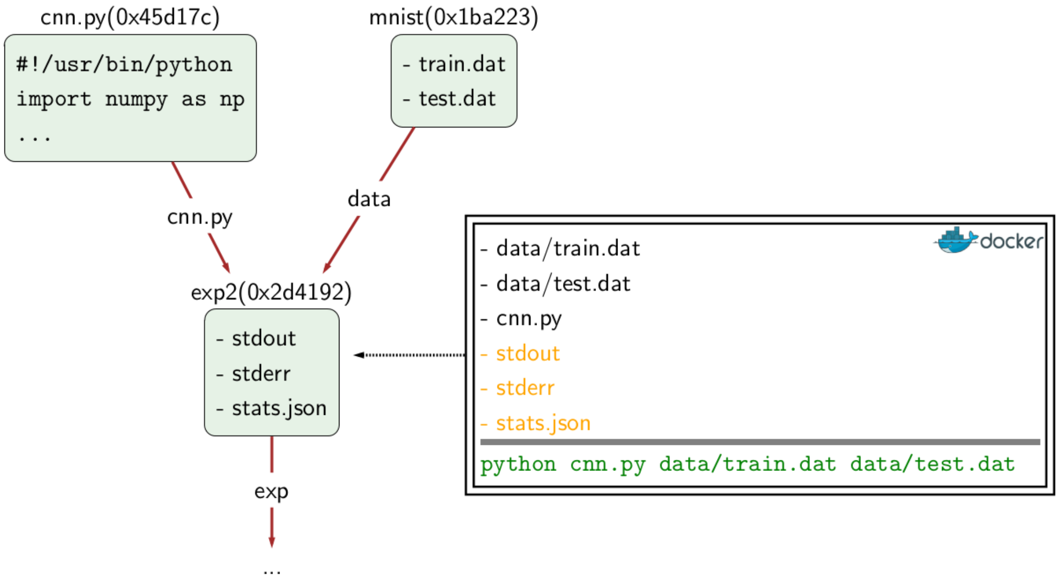
AutoDL Benchmark 2 :: 22 DA SETS
Organized by Pavao
Create a fully Automatic Deep Learning solution.
Apr 14, 2019-Dec 15, 2021
4 participant s

AutoDL Benchmark 1 :: 22 DATASETS
Organized by Pavao
Apr 14, 2019-Dec 15, 2021
4 participant s

A.2 technical structure

There are two important concepts in CodaLab: bundles and worksheets.

Bundles are immutable files/directories that represent the code, data, and results of an experimental pipeline

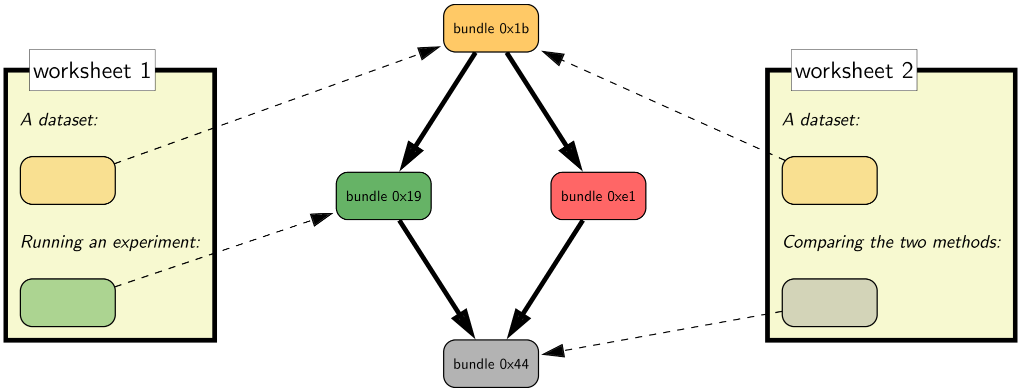


```
#!/usr/bin/python
import numpy as np
...
```

```
python cnn.py data/train.dat data/test.dat
```

Above, each rounded rectangle represents a bundle, and arrows represent dependencies between bundles. There are two bundles which are uploaded by the user: the top left bundle is a single script cnn.py containing the training code, and the top right bundle mnist contains the dataset. Then there is a run bundle exp2, which depends on cnn.py and mnist. CodaLab creates a Docker container and executes the shell command (bottom of box in green). Running exp2 produces new files stdout, stderr, and stats.json.

Worksheets organize and present an experimental pipeline in a comprehensible way, and can be used as a lab notebook, a tutorial, or an executable paper. Worksheets contain references to bundles, and are written in a custom markdown language.



For more details you can visit codalab documentation in readthedocs format