Not a Number: Identifying Instance Features for Capability-Oriented Evaluation

Ryan Burnell[1] , **John Burden**[1,2] , **Danaja Rutar**[1] , **Konstantinos Voudouris**[1] , **Lucy Cheke**[1] , **José Hernández-Orallo**[1,2,3]
[1]Leverhulme Centre for the Future of Intelligence, University of Cambridge, UK
[2]Centre for the Study of Existential Risk, University of Cambridge, UK
[3]VRAIN, Universitat Politècnica de València, Spain
rb967@cam.ac.uk, jjb205@cam.ac.uk, dr571@cam.ac.uk, kv301@cam.ac.uk, lgc23@cam.ac.uk, jorallo@upv.es

Abstract

In AI evaluation, performance is often calculated by averaging across various instances. But to fully understand the capabilities of an AI system, we need to understand the factors that cause its pattern of success and failure. In this paper, we present a new methodology to identify and build informative instance features that can provide explanatory and predictive power to analyse the behaviour of AI systems more robustly. The methodology builds on these relevant features that should relate monotonically with success, and represents patterns of performance in a new type of plots known as 'agent characteristic grids'. We illustrate this methodology with the Animal-AI competition as a representative example of how we can revisit existing competitions and benchmarks in AI—even when evaluation data is sparse. Agents with the same average performance can show very different patterns of performance at the instance level. With this methodology, these patterns can be visualised, explained and predicted, progressing towards a capability-oriented evaluation rather than relying on a less informative average performance score.

1 Introduction

The use of a single number to summarise a system's behaviour is at the root of several recurrent problems in AI evaluation and machine learning. For instance, the classical problems of overfitting and generalisation [Sutton, 1996; Neyshabur *et al.*, 2017], especially under situations when there is distribution shift [Koh *et al.*, 2020], illustrate the need for more robust evaluation methods.

Self-driving cars provide a clear demonstration of this problem. Take two cars which have similar success rates across a large number of trips. Based on this simple metric, it appears these two cars have similar capabilities. But a breakdown of the *patterns of performance* for the cars could show a very different picture. One car might perform almost perfectly when visibility is good, but struggle on particularly winding roads when there is heavy fog. By contrast, the other car might be relatively robust to weather conditions, but occasionally crashes seemingly at random. The first car has clear limitations (struggles in poor visibility conditions on winding roads), but performs predictably within those limitations—if visibility is good, we can be confident the car will successfully navigate the route. By contrast, the second car has no obvious limitations (at least in terms of visibility) but is less predictable overall—regardless of the weather, it is unclear if the car will be successful on a given trip. These very different patterns of performance would have different implications for the safety evaluation and deployment of the two cars. Yet these differences are not apparent from a simple metric such as success rate. Instead, a breakdown of performance across different conditions is required for robust analysis of safety (see Fig. 1).

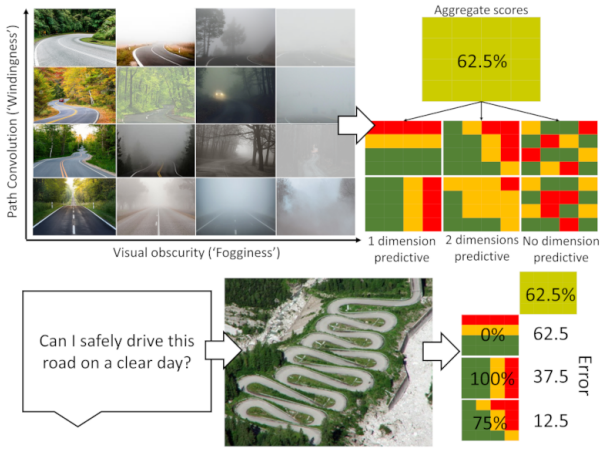

Figure 1: Evaluating performance of a self-driving car on roads varying in straightness and visibility. A single aggregate performance number such as 62.5% (of journeys made safely) is neither informative of current performance nor predictive of performance on new instances.

Unfortunately, AI evaluation today rarely employs such a breakdown. AI systems are trained and tested on large collections of instances, from which an aggregated metric of performance is derived and used to determine which sys-

tems are better than others. Competitions, benchmarks and leader boards are built to examine the progress of the field based on simple curves showing how a few metrics of performance have evolved with time ([Shoham, 2017], aiindex.org, paperswithcode.com). Many problems in AI evaluation derive from this aggregational methodology. Benchmarks are replaced as soon as the performance metric reaches a high value [Schlangen, 2019; Zellers *et al.*, 2019], even if the best systems still fail at some particular instances in the original distribution and fail systematically for other out-of-distribution instances. An enormous effort is put into using distributions that are more diverse or that might contain distribution shifts [Koh *et al.*, 2020]. However, these approaches perpetuate the evaluation paradigm.

This is ultimately a problem of validity, which stems from the fact that aggregate metrics are heavily influenced by the distribution of test instances. For example, imagine we set out to measure the ability of an AI agent to navigate through an environment to reach a reward, and find that across 100 trials the agent has a 50% success rate. If we were then to add 1000 additional easy trials to our test set in which the reward is directly in front of the agent, the success rate of the agent would increase drastically, yet, the actual *capability* of the agent to navigate environments has not improved. For this reason, aggregate metrics are not a robust way of measuring capabilities.

It is hence no surprise that many AI systems solve a task or excel at a particular benchmark, but then fail at other tasks or instances that putatively represent the same capability. This problem becomes clear as performance in many AI benchmarks ramps up rapidly, while concomitant improvement in capability is moderate.

As our self-driving car example demonstrates, two systems with the same overall performance can show a very different distribution of results. If instances are numbers in a dataset, then there is not much structure in the instance space we can use to explain the difference in behaviour. However, if we identify or build into the dataset multiple features that have a value for each instance and represent different challenges that instance poses, we can use these features to explain and predict the results for particular instances. This is a much more powerful and precise philosophy for AI evaluation.

In this paper we present a capability-oriented evaluation methodology for AI that identifies relevant instance features that might influence the behaviour of the system. We demonstrate that this approach can be used to visualise, explain and predict the behaviour of AI systems more robustly than aggregational approaches.

The rest of the paper is distributed as follows. The next section covers related work, Section 3 covers identification of dimensions of interest, Section 4 covers visualising and explaining agent behaviour, Section 5 uses the identified dimensions to predict performance, Section 6 presents guidelines, and Section 7 covers conclusions and discussion. The appendix includes further results and plots, and can be found with all the code and data on Github[1].

[1]https://github.com/RyanBurnell/NotANumber

2 Background

One way of attempting to address several evaluation problems such as generalisation and distribution shift [Sutton, 1996; Neyshabur *et al.*, 2017; Koh *et al.*, 2020] is to introduce training and test variations in the form of benchmark extensions, augmented examples or corruptions. Ideas are taken from software testing, such as [Marijan and Gotlieb, 2020] and [Ribeiro *et al.*, 2020], where directional and invariant transformations are distinguished depending on how they affect performance. The variations can be introduced adversarially [Zellers *et al.*, 2018; Nie *et al.*, 2019; Rozen *et al.*, 2019; Kiela *et al.*, 2021] or through procedural content generation [Risi and Togelius, 2020] or in other ways [Sugawara *et al.*, 2020]. This creates more diverse datasets, but the results for this new distribution end up as a summarised number that depends on that distribution.

This problem is recognised in several engineering applications, in which the performance of a system is tested across a range of 'operating conditions' under which a system needs to operate correctly [Jylhä *et al.*, 2017; Martin-Diaz *et al.*, 2018]. This notion has been extended in machine learning to analyse where a system performs better or worse—for example, the use of ROC analysis in machine learning and statistics to relate operating conditions to the prior distribution and the misclassification costs [Fawcett, 2006]. However, these conditions are associated with the receiver context or the learning parameters of the agent themselves (rather than the instances of the test). Thus the variation is in how the agent weights features or feedback (e.g. misclassification costs) rather than the distribution of instance features themselves.

These problems have also been recognised in other fields. For example, when psychologists assess cognitive performance, they often use large "batteries" of tests that evaluate broad capabilities and identify any changes, developments or deficits. Crucially, test scores are not simply aggregated together. Instead, patterns of performance are examined across different tests and domains. For example, two individuals might score 75% overall, but one has a memory deficit while the other has an executive function problem. Examining patterns of performance allows for better predictions about how an individual will perform under different circumstances (e.g., if an individual has a memory deficit, they will likely perform poorly on memory tests) and also to identify the underlying causes of impairments (e.g., issues with executive functions tasks may indicate frontal-lobe damage).

Some recent attempts have been made to apply these ideas to AI systems by representing several dimensions or categories of performance, such as [Osband and others, 2019], [Ilievski *et al.*, 2021], and [Crosby *et al.*, 2020]. These dimensions are usually partitioned into categories, but are aggregated within the category. While this provides more information as to where an agent succeeds/fails, it is still dependent on which instances are included within the category distribution and thus cannot demonstrate capability. Moreover, partitioning dimensions into categories makes it difficult to evaluate the effects of continuous dimensions. In order to understand the capabilities of a system, we need to be able to evaluate the level of difficulty that system is capable of reach-

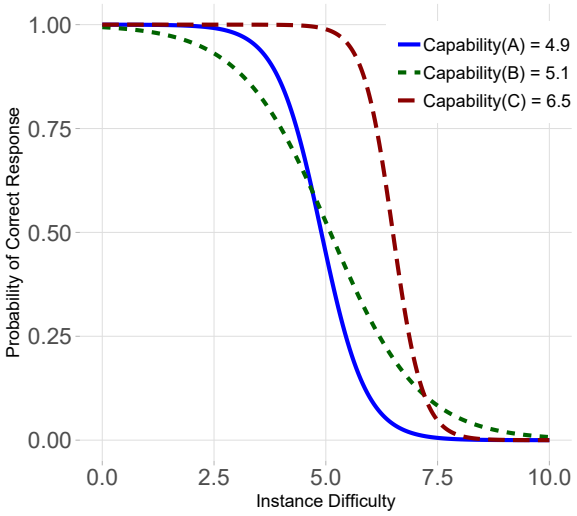

Figure 2: Three *agent characteristic curves*, and their areas (capabilities).

ing [Thurstone, 1937].

Fig. 2 (right) shows the evolution of success of three agents depending on how they cope with instances of increasing difficulty. The point in which the curves goes down is the capability level and it does not change if we modify the specific examples included in the x-axis (so long as difficulty is unchanged). These *agent characteristic curves*, usually following logistic models, can explain behaviour and predict performance, but psychometrics, and its use in AI [Martínez-Plumed *et al.*, 2019], identifies the constructs and the parameters of the curves 'bottom-up' from the subject population. Instead, we want to derive these features top-down from the inherent properties of the task, informed by the cognitive/perceptual challenge they present (e.g. road visibility).

We present a more robust methodology for the evaluation of AI systems based on patterns of performance over several identified features. As a proof of concept, we apply this methodology to the Animal-AI (AAI) Olympics [Crosby *et al.*, 2020], a competition that evaluated AI agents in a 3D environment across a range of task categories, such as spatial memory and causal reasoning. The competition provides a good opportunity for evaluating patterns of performance because it includes instance-level performance data for a range of agents across a variety of tasks.

The AAI Olympics, first run in 2019, asked AI developers to build systems that can solve the Testbed, a set of 300 unseen tasks inspired by animal cognition, using the inventory of objects available in the AAI Environment. Fig. 3 shows some of the tasks within the AAI Testbed. More information in the Appendix.

Here, we investigate how 68 agents (mostly DRL agents) perform on 99 of the simplest object retrieval tasks in the Testbed, in which the agent must obtain a reward in an open arena. Object retrieval requires an agent to be able to perform goal-directed actions, which is the starting point for many more complex behaviours (such as locating occluded objects) [Lind and Vinken, 2021]. Goal-directed action is simple to define and therefore useful for demonstrating the utility of our novel methodology.

Figure 3: Some examples of the 900 instances in the Animal AI Olympics. The first two are included in this paper because they test simple goal-directed behaviour. The rightmost one is not because it tests more complex skills.

3 Identifying Dimensions

In order to understand the patterns of performance for any AI system, the first step is to identify and annotate each tested instance along dimensions of interest. Two types of dimensions are important to identify:

Relevant features: These are features that should meaningfully affect the difficulty of the task. For instance, when evaluating self-driving cars, the presence of fog is a relevant feature because it makes navigating more difficult. These features are important to include in evaluation because they allow us to determine the capability of the system under circumstances of varying difficulty. These features should be identified in a top-down way based on theory or domain-related knowledge. Oftentimes, theory and prior work in a given domain will have already identified features of interest, which can then be used to guide new research in that domain.

Irrelevant features: These are features that, in theory, should not affect performance. For instance, the colour of an obstacle should not affect the probability of colliding with it. These features are important for testing the robustness of the system. Note that a given feature may be relevant or irrelevant depending on the context and capability being assessed (e.g. the colour of a traffic light).

For the AAI competition, we first identified a set of potentially relevant and irrelevant features. We identified the 'size' of, and 'distance' to, the reward, as well whether the reward is in front of or behind the agent ('YPos') as relevant features. We also identified several irrelevant features: 'XPos', the side the reward is on (left vs right) and the 'colour' of the reward (green vs yellow). Reward size and distance were standardised to mean 0 and std deviation 1, and reward size was reversed so larger numbers represent more difficult (smaller) sizes.

Next, we annotated each instance with values for the relevant and irrelevant features. These data were determinable from the original competition information (though not linked with instance performance data), allowing us to manually code for each dimension. While manually annotating is not always feasible (e.g. large testing datasets), it can be largely automated (e.g., from task specification or logs). This approach allows for the re-evaluation of existing datasets and benchmarks without requiring additional data collection.

This process highlights where there is uneven representation of features in a distribution, as was the case with this

dataset. For example, the majority of the instances had rewards between 20-35 from the agent, with no rewards in the 5-10 or 45-50 ranges. Nonetheless, sufficient variation was present to allow for evaluation of how these dimensions affect performance in the AnimalAI Olympics.

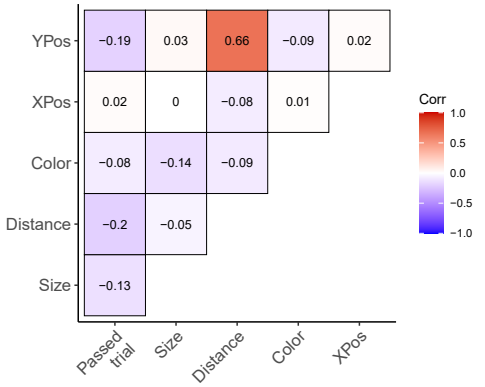

Figure 4: Spearman correlations between trial success and several relevant (reward size, distance to reward, YPos) and irrelevant dimensions (Colour and XPos). Pearson correlations in the appendix.

We scaled some features (e.g., 'size' was reversed so higher scores represent smaller—more difficult—sizes) and used non-linear monotonic transformations (e.g., 'XPos' and 'YPos'). We then analysed the effect of these features on performance for the competition agents. First, we calculated the bivariate correlations between all these features and against the pass/fail result (success rate) for an instance. Fig. 4 shows that the three variables with highest correlation are reward size, distance and YPos. We also noted the high correlation between reward distance and YPos. We nonetheless proceeded with these three features as the relevant *dimensions* in the subsequent steps. Similarly, the XPos and colour of the reward were confirmed to be mostly irrelevant overall.

To better understand the ranges of these three relevant features (size, distance, Ypos) and compare them with an irrelevant feature (XPos), Fig. 5 shows their characteristic curves. One interesting element we will extend multidimensionally is the area of these curves, which can be understood as separate *capabilities* (coping with size, distance and YPos). Another important indicator that can be extracted from these curves is how *conforming* they are with the expected monotonic (decreasing) behaviour, approximated with a Spearman correlation against an ideal decreasing sequence. The shape can follow a sigmoid curve, showing some saturation level that would correspond to the capability. Ideally it should start in almost complete success rate and go down to zero (or random) success rate on the right. In this dataset, the curve that mostly resembles this behaviour is that for reward distance.

4 Visualising and Explaining Behaviour

We can plot one dimension against success rate, and we get a curve, as illustrated in the previous section, but some of these dimensions interact and the correlations may be caused by confounders or mediators. To visualise the effect of the relevant features together we propose a simple visualisation technique placing each feature as a dimension and the success rate (or other performance metric) represented in colours (or shade of grey). For robustness, we also group values into bins of appropriate width in the desired dimensions for analysis. The results (success rate) of agents are then calculated for each bin. Each cell also shows the number of observed instances for a specific combination of values using opacity (and textually in the grid cell). We refer to these plots as *agent characteristic grids*. We show several examples in Fig. 6: for all agents within the Animal-AI Olympics (top-left), a very high scoring agent 'Ironbar' (top-middle) and three other agents with varying patterns of performance. In this case all the plots show reward distance and size, because these dimensions are uncorrelated. See Appendix for grids for other dimensions (e.g. YPos).

Fig. 6 illustrates that 'Ironbar' is successful across a wide range of test instances, while 'y.yang' shows a more mixed performance, particularly struggling with distant rewards. From the grids we can also extract that 'y.yang' is less predictable as reward size diminishes. For larger reward sizes, we see a gradual progression towards lower success as distance to the reward increases, while for smaller rewards there is no clear association between distance and success. Two agents –'Sparklemotion' and 'Juohmaru'– have similar success rates. However, while 'Sparklemotion' is largely unpredictable across the whole feature space, 'Juohmaru' has predictable limitations, performing well when rewards are large and close. In an applied setting, this conformance and predictability may make 'Juohmaru' a more desirable agent for deployment.

Identifying these limits is important for evaluating postrequisite cognitive skills—skills that depend on the skill currently being evaluated. For example, a suite of instances for evaluating whether an agent has object permanence may require the agent to navigate to a certain location in order to make a selection. Known limitations of an agent's navigational skills will predict uniform failure regardless of any additional capabilities. This may mean the agent should not be tested, or that instances are required to assess object permanence that are within the known working limits of the agent (where this is possible and reasonable).

These visualisations can give an immediate overview of an agent's performance, as well as identifying the make-up of the instances and the distribution for each dimension. Additionally a more robust metric of capability can be calculated; taking the mean success rate of each cell with at least one observed instance. Given sufficient coverage, capability is not affected by having more or less cases in some cells over others. In order to get the complete picture, we can calculate the capability of different projections of the instance-space.

Finally, it is also important to analyse the irrelevant features and see how robust agents are to changes in these features. For example, the agent characteristic grids for XPos and YPos showed that the agent 'Sparklemotion' was relatively able to solve instances in which the reward was to the left of the agent, but not when the reward was to the right of the agent. By contrast, 'y.yang' was able to solve instances for both left and right side rewards (see Appendix).

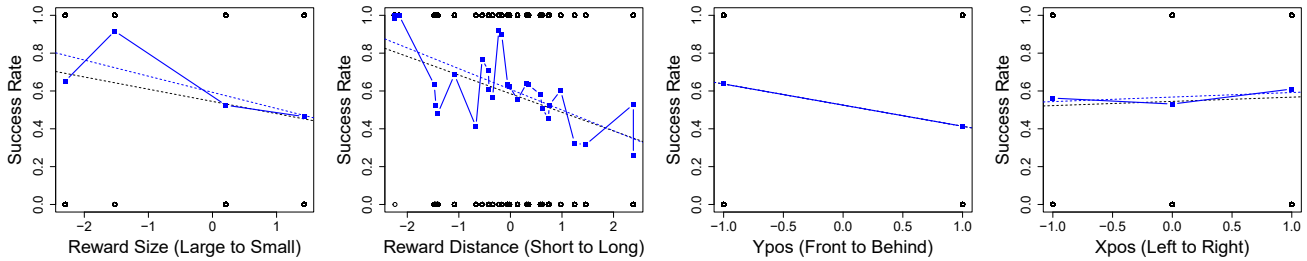

Figure 5: Characteristic curves of all competition entrants (agents) according to three relevant features (size, distance and Ypos) and one irrelevant feature (Xpos). Black dashed lines show the linear regression for the black points (pass/fail), while blue dashed lines interpolate the blue points (binned success rate). The conformances (Spearman correlations against monotonic sequence) are 0.80, 0.60, 1.00 and −0.50, respectively.

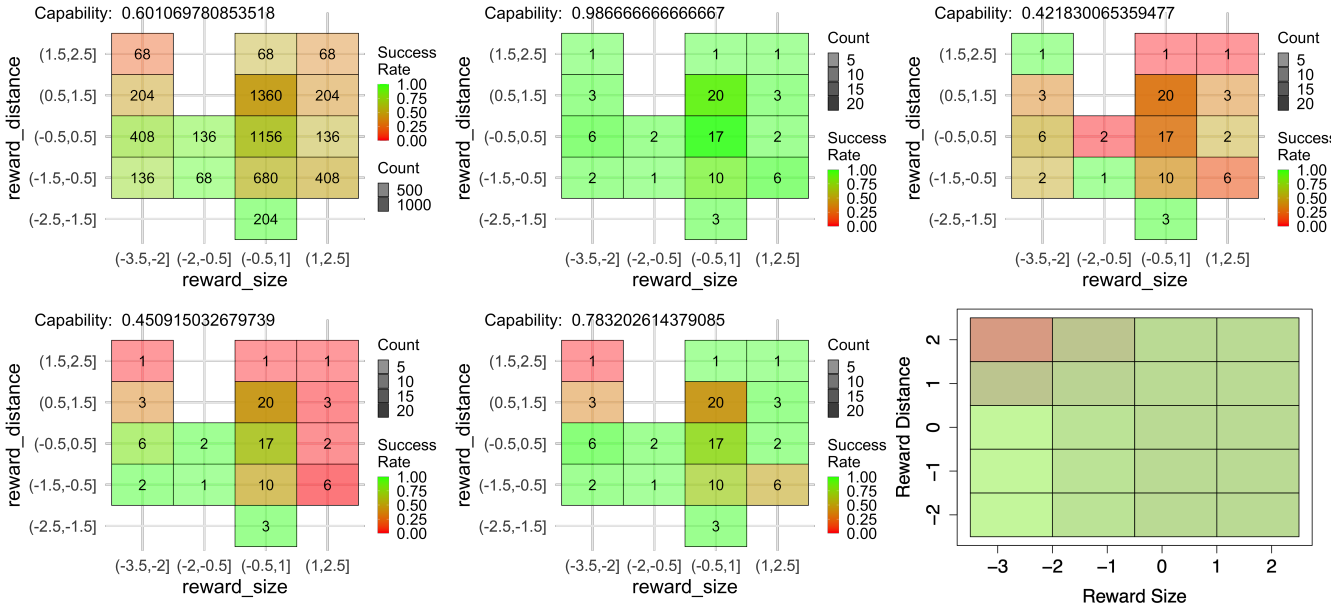

Figure 6: Agent Characteristic Grid displaying results for all agents (top left), 'Ironbar' (a very high scoring agent; middle), 'y.yang' (an agent with a very non-conformant behaviour; top right), 'Sparklemotion' (an unpredictable agent; bottom left), and 'Juohmaru' (a predictable, conformant agent; bottom middle). The bottom right figure shows the outputs of a decision tree model for y.yang based on reward size, distance, and YPos, which we can use to predict performance across the feature space using the test data. For instance, if, during deployment, the y.yang agent is tasked with finding a small reward that is far away (top left bin) the model predicts y.yang will perform poorly.

5 Predicting from the Instance Features

The way in which the grids help visualise agent behaviour may suggest we could use capability as an alternative indicator. Fig. 8 in appendix A.3 shows the relation between Success Rate and Capability for the 68 participants (with 0.96 correlation). This is expected, but some agents diverge from the regression line. These indicators would diverge more significantly if we changed the distribution of instances. If we made a sampling where large reward size and distance are more likely, capability would not be modified (provided the sample is sufficiently large) but success rate would increase.

Despite this advantage, aggregating all dimensions into one single number loses information, and makes comparison for particular instances more difficult. In the unidimensional case, from the characteristic curves for one agent (as those in Fig. 1) we could infer, e.g., that an agent with capability of 1.5 in the distance dimension will likely be successful given an instance where the distance is 0.5. Of course, to refine this *prediction*, we would need more information about the other dimensions as well. We could even draw these curves with a subset of the data assuming all the other dimensions are easy, to avoid confounders. This is all an interesting path for exploration, but here we will take a more powerful one.

One of the main reasons we evaluate AI systems is to determine how they will behave in new situations. The traditional approach is to estimate average performance for a test set and hope that no distribution shift will take place during deployment. For instance, if we take the results of the competition and split the data into 75% test and 25% deployment, and calculate the average performance for all teams, we have that always predicting success gives us an error of 45.3% (see Table 1). A global accuracy extrapolation will make more errors

(48.0%), although MSE is lower. Finally, the usual approach would be simply to extrapolate the accuracy of the model for each agent: this leads to an error of 33.6%.

	Maj. (1)	**G.Acc.**	**T.Acc.**	**∼All+A**	**∼Rel+A**
Error	45.3%	48.0%	33.6%	19.7%	20.6%
MAE	45.3%	49.6%	34.9%	29.3%	30.2%
MSE	45.3%	24.8%	17.6%	14.8%	15.4%

Table 1: Five options to anticipate performance of an AI system for a new instance. From left to right: the majority model always predicting success, the model predicting according to the global accuracy, the model predicting according to the accuracy of each agent, a predictive model (C5.0) when using all variables and agent id as inputs, and when (C5.0) only using the relevant features we chose (reward size, distance and Ypos) and the agent id. The three metrics are crisp Error rate (where models may behave stochastically), MAE (Mean Absolute Error) and MSE (Mean Squared Error).

But we can do much better by the use of a predictive model using all the features as input variables. The error goes down to 19.7%. This possibility is systematically ignored in the literature, as if we could not make things better than just extrapolating the test accuracy for deployment. Interestingly, with the main relevant features (reward size, distance and YPos) and the agent id, we get almost the same error (20.6%). This means that these three variables capture most of the variability that we can explain and predict with the rest.

But the predictive models can also be used to determine which areas are most uncertain and require most testing. For instance, if we compare the actual results for agent 'y.yang' (Fig. 6, top right) with the prediction of the ∼**Rel+A** model (bottom right), we see that some areas that are blank in the original plot (mostly at the bottom) have predictions close to 100%. These are certain areas for which testing is less necessary than other blank areas such size=-1 and distance=-1 where the predictions are more borderline.

6 Guidelines

We have used the AnimalAI Olympics as a proof of concept of a new methodology for the cognitive evaluation of AI systems based on the identification of relevant and irrelevant features. This is a step-by-step summary of this methodology (a more in-depth breakdown can be found in the Appendix):

1. Choose a domain, task or benchmark with instance-level data.
2. Identify features that can be extracted or easily annotated for each instance during testing.
3. Identify which features are relevant (should affect performance) and those that are irrelevant (should not affect performance) based on theory and domain knowledge.
4. Analyse the relationships between features and performance using correlation and other exploratory analyses.
5. Select features of interest, bin them appropriately and build characteristic grids (both global and for individual systems) to evaluate patterns of performance.
6. Build predictive models using extracted features. Compare the results with other ways of predicting performance, such as extrapolating average metrics.
7. Using characteristic grids and predictive models, evaluate capabilities of each system across the distributions of the dimensions of interest.
8. Identify areas of competence for individual systems so that these can later be used for testing more complex or advanced capabilities and skills (where appropriate).
9. Use areas of weakness to inform changes to the benchmark, system models or training.
10. With new insights about which features are relevant, iterate the process to step 2—or to step 1 if more test data is needed (using the models)—for subsequent analyses.

4 Once subareas of competence of individual systems are identified, we can build on these basic capabilities to start testing more complex or advanced capabilities and skills. If no areas of competence exist in these basic capabilities, testing for more complex skills will be uninformative.

In each step, there are many opportunities to use more complex data analysis and machine learning techniques. However, it is important to emphasise that the features are cognitive and should derive from theory rather than being latent factors from a population of systems. In the case of the AAI Olympics, some of the features we identified as relevant had low correlations with performance but we kept them because they should *theoretically* affect it. Ultimately, they were predictive.

7 Discussion

Using the above method, we have demonstrated how patterns of performance of an AI system can be evaluated by identifying dimensions of interest and examining the capability of that system across those dimensions. This leads to a clearer understanding of agent capabilities than a single number such as accuracy. It also allows us to distinguish agents that reliably perform well in only limited situations from agents that are less predictable but have a wider distribution of successes. We have also introduced new visualisations—the 'agent characteristic grids'—which show patterns across dimensions. We then derived capability and conformance scores as a means to understand system behaviour.

In addition, we found that modelling performance using dimensions of interest produces better predictions than the average performance of the agent. This result is not entirely surprising—reducing performance to a simple number for varied instances leads to loss of information. But a key advantage of using patterns of performance is that characteristic grids and capability metrics are independent of the distribution of test instances.

Here, we focused on the ability to navigate to a reward with no obstacles. We did so because the agents included in the AnimalAI Olympics performed poorly on more complex tasks. But this same methodology can be applied to more complex cognitive abilities, considering basic tasks such as navigation as prerequisites for these more complex abilities.

One limitation of the visualisation approach we present here is that the characteristic grids display only two features at a time. Of course, there are likely to be many dimensions of interest for a given system. If these dimensions affect performance independently, grids for various combinations of features can be used to visualise patterns of performance. However, in some situations, performance might depend on interactions between features—for example, rain might only impair performance if it is also foggy and traffic is heavy. In such situations, the patterns of performance might not be clear from separate feature grids. Another limitation is the cost of annotating instances—in larger benchmarks this should rely on full automation or the use of crowd-sourcing.

There are many avenues for future work in the exploration of some of the above issues and the adaptation of the methodology to several domains. This must be part of a collective effort in AI so that evaluation methodologies gain in sophistication and insight, as AI systems display more powerful capabilities, increasingly requiring a cognitive perspective.

A Appendix

Here we include some supplementary material to the paper.

A.1 Animal AI Environment and Competition

The Animal-AI (AAI) environment is built in Unity [Juliani *et al.*, 2018]. Agents are provided with three inputs. First, pixel input derived from the 3D visual arena, whose properties and possible interactions are dictated by a physics engine. Second, an egocentric velocity vector providing velocity information in three dimensions. Third, a points value. Points are gained or lost through contact with rewards of differing size and significance, and punishments of differing severity. Points start at 0 and decrease linearly with each timestep. btaining a green reward is the ultimate goal, providing a positive reward *and* ending the episode. Agents are limited to 4 actions: Move Forwards, Move Backwards, Rotate Left, Rotate Right. In each task, the objective for the agent is to maximise its 'points' within a time-limit.

Developers of AI agents were provided with the test arena and the inventory of objects that could be used to construct tasks. They were instructed that their agents would be tested on problems that animals can solve, but they were not provided with any of the task configurations until after the competition was over. The 2019 Animal-AI Olympics competition used 3 variations of each of 300 individual tasks, almost all adapted directly from the animal cognition literature, ranging from simple object retrieval to complex tool-use tasks. More information can be found at (http://animalaiolympics.com/AAI/), including an online environment for the reader to play the tasks mentioned in this paper and in the competition.

A.2 Pearson Correlations

Fig. 7 contains the Pearson correlations, complementing the Spearman correlations already shown in the paper.

A.3 Capability vs Performance

Fig. 8 shows the correlation between capability and performance for the 68 agents.

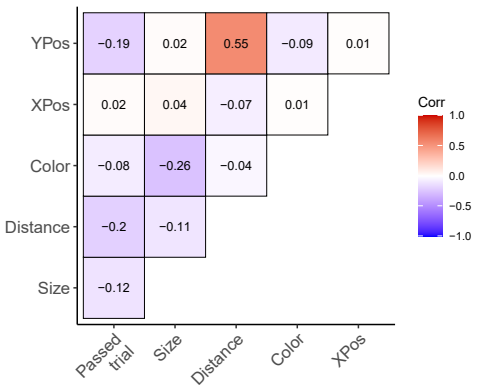

Figure 7: Pearson correlations between trial success and several relevant (reward size, distance to reward, YPos) and irrelevant dimensions (Colour and XPos).

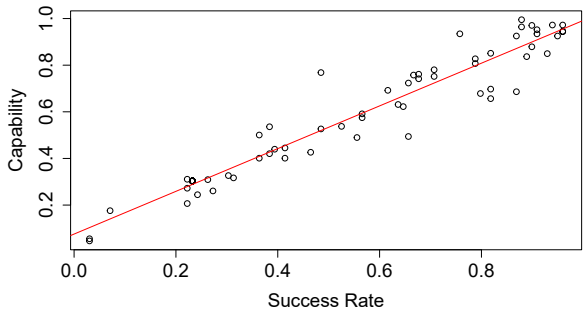

Figure 8: Success Rate vs Capability (correlation 0.96).

A.4 Guidelines

We include a more detailed version of the guidelines in the paper here:

1. Choose a domain, task or benchmark for which there are or we can obtain instance-wise results, i.e., we have the performance of one or more systems for each instance.
2. From an understanding of the domain and what capabilities are being measured, identify a series of features that can be extracted or easily annotated for each instance during testing (and potentially during deployment). These features can be based or derived from the original instance features (e.g., a fog level detector from images).
3. Identify which features are relevant (should affect performance), and those that are irrelevant (should not affect performance). Consider the effect of these features from a cognitive perspective and the contexts in which a feature might be more or less relevant, as well as possible theoretical interactions between these features.
4. Analyse the relationships between features and performance with the existing test data. This can be done by calculating correlation matrices and performing other

exploratory analyses. These analyses can also test hypotheses about which features are relevant and irrelevant. Note that this process is inverse to the extraction of latent variables or constructs in psychometrics, and should not even be confirmatory.

5. Select a subset of features that are of interest (and preferably not highly-correlated), bin them appropriately and use them to build agent characteristic grids (both globally and for individual systems) to evaluate patterns of performance. Depending on the number of features, several grids might be needed for each agent.
6. Build predictive models using extracted features. Compare models that use all features to models using only a subset of relevant features to determine the usefulness of different combinations of features. Also compare the results with other ways of predicting performance, such as extrapolating average metrics.
7. Using characteristic grids and predictive models, evaluate capabilities of each system across the distributions of the dimensions of interest.
8. Identify the subareas of competence of individual systems (e.g., green areas near the origin in the characteristic grids). Within these areas, we can be confident the agent will perform well. Therefore, these areas can later be used for testing more complex or advanced capabilities and skills (where appropriate).
9. Use these models to identify areas of weakness in order to improve the benchmark (including system models, and training) and subsequent analyses.
10. With new insights about which features are relevant, iterate the process to step 2, or to step 1 if more test data is needed (using the models), for subsequent analyses. Without a reasonable distribution of instances on a given dimension, it becomes difficult to evaluate the effect of that dimension on performance.

A.5 Further Characteristic Grids

In Fig. 8, we include some characteristic grids for a dimension that is irrelevant (XPos), showing one agent (top) that performs much better on the right than on the left, and another agent (bottom) whose performance is balanced between sides.

Next, in Fig. 9 we show characteristic grids for the reward size and distance for additional agents: daydayup and 41Animals.

Ethical Statement

There are no ethical issues.

Acknowledgements

We thank the anonymous reviewers for their comments. This work was funded by the Future of Life Institute, FLI, under grant RFP2-152, he EU (FEDER) and Spanish grant RTI2018-094403-B-C32 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way

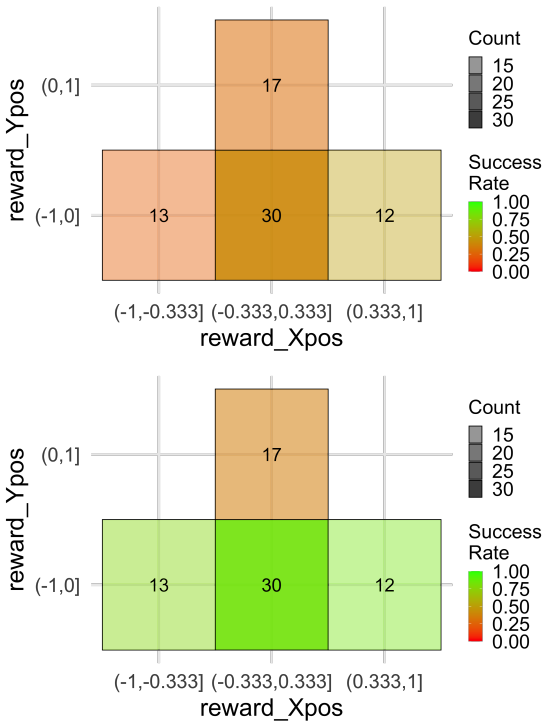

Figure 9: Agent Characteristic Grids displaying XPos and YPos results for 'Sparklemotion' (top) and 'y.yang' (bottom).

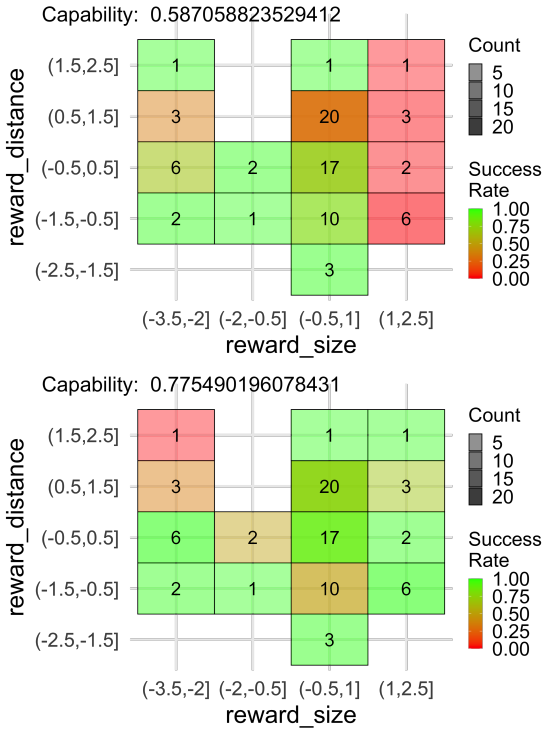

Figure 10: Agent Characteristic Grids displaying Size and Distance results for daydayup (top) and 41Animals (bottom).

of making Europe", Generalitat Valenciana under PROMETEO/2019/098, EU's Horizon 2020 research and innovation programme under grant agreement No. 952215 (TAILOR) and US DARPA HR00112120007 (RECoG-AI).

References

[Crosby *et al.*, 2020] Matthew Crosby, Benjamin Beyret, Murray Shanahan, Jose Hernandez-Orallo, Lucy Cheke, and Marta Halina. The animal-AI testbed and competition. *Proceedings of Machine Learning Research*, (123):164–176, 2020.

[Fawcett, 2006] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[Ilievski *et al.*, 2021] Filip Ilievski, Alessandro Oltramari, Kaixin Ma, Bin Zhang, Deborah L McGuinness, and Pedro Szekely. Dimensions of commonsense knowledge. *arXiv preprint arXiv:2101.04640*, 2021.

[Juliani *et al.*, 2018] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. Unity: A general platform for intelligent agents. *CoRR*, abs/1809.02627, 2018.

[Jylhä *et al.*, 2017] Juha Jylhä, Marja Ruotsalainen, Ville Väisänen, Kai Virtanen, Mikko Harju, and Minna Väilä. A machine learning framework for performance prediction of an air surveillance system. In *2017 European Radar Conference (EURAD)*, pages 187–190, 2017.

[Kiela *et al.*, 2021] Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. Dynabench: Rethinking benchmarking in nlp. *arXiv preprint arXiv:2104.14337*, 2021.

[Koh *et al.*, 2020] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. *CoRR*, abs/2012.07421, 2020.

[Lind and Vinken, 2021] Johan Lind and Vera Vinken. Can associative learning be the general process for intelligent behavior in non-human animals? *bioRxiv*, 2021.

[Marijan and Gotlieb, 2020] Dusica Marijan and Arnaud Gotlieb. Software testing for machine learning. In *AAAI*, volume 34, pages 13576–13582, 2020.

[Martin-Diaz *et al.*, 2018] Ignacio Martin-Diaz, Daniel Morinigo-Sotelo, Oscar Duque-Perez, and Rene J Romero-Troncoso. An experimental comparative evaluation of machine learning techniques for motor fault diagnosis under various operating conditions. *IEEE Transactions on Industry Applications*, 54(3):2215–2224, 2018.

[Martínez-Plumed *et al.*, 2019] Fernando Martínez-Plumed, Ricardo BC Prudêncio, Adolfo Martínez-Usó, and José Hernández-Orallo. Item response theory in AI: Analysing machine learning classifiers at the instance level. *Artificial Intelligence*, 271:18–42, 2019.

[Neyshabur *et al.*, 2017] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *NeurIPS*, pages 5947–5956, 2017.

[Nie *et al.*, 2019] Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*, 2019.

[Osband and others, 2019] Ian Osband et al. Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568*, 2019.

[Ribeiro *et al.*, 2020] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. *ACL2020, arXiv preprint arXiv:2005.04118*, 2020.

[Risi and Togelius, 2020] Sebastian Risi and Julian Togelius. Increasing generality in machine learning through procedural content generation. *Nature Machine Intelligence*, page 428–436, 2020.

[Rozen *et al.*, 2019] Ohad Rozen, Vered Shwartz, Roee Aharoni, and Ido Dagan. Diversify your datasets: Analyzing generalization via controlled variance in adversarial datasets. *arXiv preprint arXiv:1910.09302*, 2019.

[Schlangen, 2019] David Schlangen. Language tasks and language games: On methodology in current natural language processing research. *arXiv preprint arXiv:1908.10747*, 2019.

[Shoham, 2017] Yoav Shoham. Towards the AI index. *AI Magazine*, 38(4):71–77, 2017.

[Sugawara *et al.*, 2020] Saku Sugawara, Pontus Stenetorp, and Akiko Aizawa. Benchmarking machine reading comprehension: A psychological perspective. *arXiv preprint arXiv:2004.01912*, 2020.

[Sutton, 1996] Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *NeurIPS*, pages 1038–1044, 1996.

[Thurstone, 1937] LL Thurstone. Ability, motivation, and speed. *Psychometrika*, 2(4):249–254, 1937.

[Zellers *et al.*, 2018] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*, 2018.

[Zellers *et al.*, 2019] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.