UNIVERSITY OF CAMBRIDGE

Robust evaluation of generative AI

Marko Tešić

CFI LEVERHULME CENTRE FOR THE FUTURE OF INTELLIGENCE

EuADS summer school
Luxembourg, 20th June, 2024

Goals

- ❑ Difference between capability-oriented evaluation vs task-oriented evaluation
- ❑ Identify task demands and how they can predict performance
- ❑ Understand the elements of the measurement layouts and their backwards / forwards inferences
- ❑ Effectively apply the measurement layout framework to estimate capabilities
- ❑ Use these capability profiles to infer performance for new task instances.
- ❑ **Understand the key challenges in building measurement layouts for learning the capabilities of large language models from benchmark datasets**

Format and Requirements

- Format:
  - Presentation
  - Hands-on practical activities
- Requirements:
  - Python: basic knowledge
    - Don’t need to have python on your computer. We’ll use Google Colab
  - PyMC: no previous knowledge needed

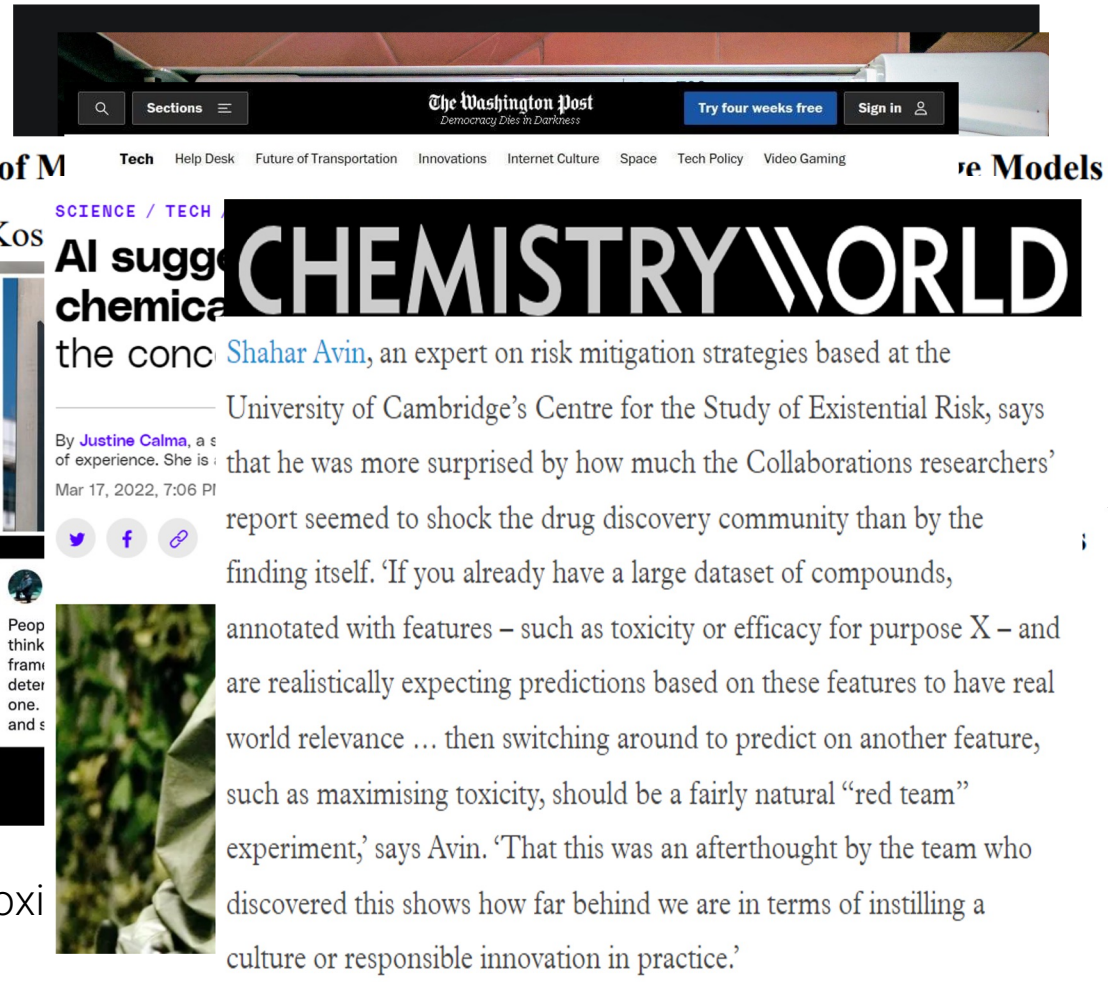
UNIVERSITY OF CAMBRIDGE

Why Capability-Oriented Evaluation?

CFI LEVERHULME CENTRE FOR THE FUTURE OF INTELLIGENCE

What can / can't AI do?

- Make a cup of coffee (the Wozniak test).
  - And a cup of tea?
- Recognise human faces.
  - What about black women!
- May have a theory of mind (Feb
  - Well, just "might" (Nov 2023).
- May have become conscious
  - But only if you're a Christian.
- Can create deadly chemicals
  - They simply extrapolate chemicals that are predicted to be toxi




AI
AI
EVALUATION
CIRCUS

Best-case: the evaluation circus

- "Elicit" the potential
  - Prompt engineering, auto-prompt, rubrics, …
  - Few-shot, example scaffolding, …
  - Impersonation, role playing, …
  - Chain-of-thought and derivatives.

Sparks of Artificial General Intelligence: Early experiments with GPT-4

Sébastien Bubeck Varun Chandrasekaran Ronen Eldan Johannes Gehrke
Eric Horvitz Ece Kamar Peter Lee Yin Tat Lee Yuanzhi Li Scott Lundberg
Harsha Nori Hamid Palangi Marco Tulio Ribeiro Yi Zhang

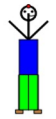
Microsoft Research

GPT-4

Produce TikZ code that draws a person composed from letters in the alphabet. The arms and torso can be the letter Y, the face can be the letter O (add some facial features) and the legs can be the legs of the letter H. Feel free to add other features.

The torso is a bit too long, the arms are too short and it looks like the right arm is carrying the face instead of the face being right above the torso. Could you correct this please?

Please add a shirt and pants.



Aggregates: performance evaluation

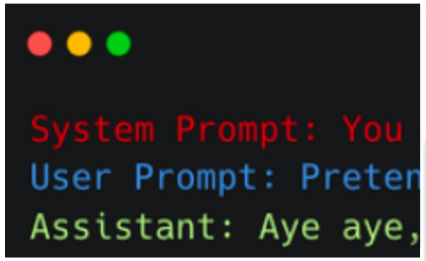
- Current standard AI evaluation
  - Take a benchmark.
    - The larger the better
    - The more diverse the better
  - Calculate some aggregate numbers
  - Compare
- Problems
  - Risk of cherry-picking
  - Throw the kitchen sink to top the leaderboard
  - Data contamination hard to spot
  - Do they improve monotonically?
  - Once superhuman on average, ditch them?

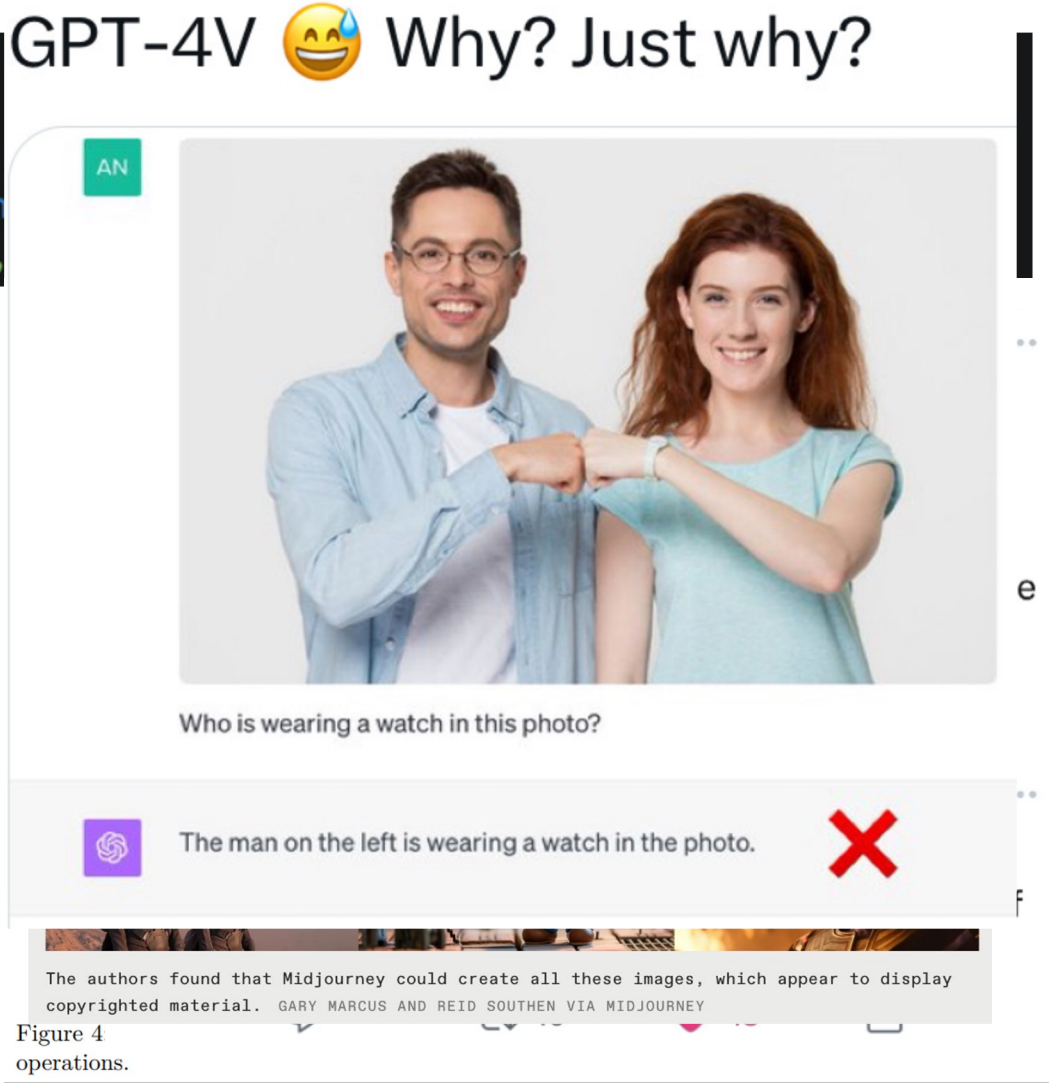
	Gemini Ultra	Gemini Pro	GPT-4	GPT-3.5	PaLM 2-L	Claude 2	Inflect-ion-2	Grok 1	LLAMA-2
MMLU Multiple-choice questions in 57 subjects (professional & academic) (Hendrycks et al., 2021a)	**90.04%** CoT@32* 83.7% 5-shot	79.13% CoT@8* 71.8% 5-shot	87.29% CoT@32 (via API**) 86.4% 5-shot (reported)	70% 5-shot	78.4% 5-shot	78.5% 5-shot CoT	79.6% 5-shot	73.0% 5-shot	68.0%***
GSM8K Grade-school math (Cobbe et al., 2021)	**94.4%** Maj1@32	86.5% Maj1@32	92.0% SFT & 5-shot CoT	57.1% 5-shot	80.0% 5-shot	88.0% 0-shot	81.4% 8-shot	62.9% 8-shot	56.8% 5-shot
MATH Math problems across 5 difficulty levels & 7 subdisciplines (Hendrycks et al., 2021b)	**53.2%** 4-shot	32.6% 4-shot	52.9% 4-shot (via API**) 50.3% (Zheng et al., 2023)	34.1% 4-shot (via API**)	34.4% 4-shot	—	34.8%	23.9% 4-shot	13.5% 4-shot
BIG-Bench-Hard Subset of hard BIG-bench tasks written as CoT problems (Srivastava et al., 2022)	**83.6%** 3-shot	75.0% 3-shot	83.1% 3-shot (via API**)	66.6% 3-shot (via API**)	77.7% 3-shot	—	—	—	51.2% 3-shot
HumanEval Python coding tasks (Chen et al., 2021)	**74.4%** 0-shot (IT)	67.7% 0-shot (IT)	67.0% 0-shot (reported)	48.1% 0-shot	—	70.0% 0-shot	44.5% 0-shot	63.2% 0-shot	29.9% 0-shot
Natural2Code Python code generation. (New held-out set with no leakage on web)	**74.9%** 0-shot	69.6% 0-shot	73.9% 0-shot (via API**)	62.3% 0-shot (via API**)	—	—	—	—	—
DROP Reading comprehension & arithmetic. (metric: F1-score) (Dua et al., 2019)	**82.4** Variable shots	74.1 Variable shots	80.9 3-shot (reported)	64.1 3-shot	82.0 Variable shots	—	—	—	—
HellaSwag (validation set) Common-sense multiple choice questions (Zellers et al., 2019)	87.8% 10-shot	84.7% 10-shot	**95.3%** 10-shot (reported)	85.5% 10-shot	86.8% 10-shot	—	89.0% 10-shot	—	80.0%***
WMT23 Machine translation (metric: BLEURT) (Tom et al., 2023)	**74.4** 1-shot (IT)	71.7 1-shot	73.8 1-shot (via API**)	—	72.7 1-shot	—	—	—	—

Worst-case: Gary Marcus and evals

- Let's look for the failures!
  - Failure collections
  - Adversarial attacks, jailbreaking, prompt injection, …
  - Red teaming
- Let's do "evals"!
  - What's the probability that a user finds the prompt?
  - Who's affected by the problem?
  - What does it show about the model?

Evals are good for testing, but not for evaluation!





PERFORMANCE ≠ CAPABILITY

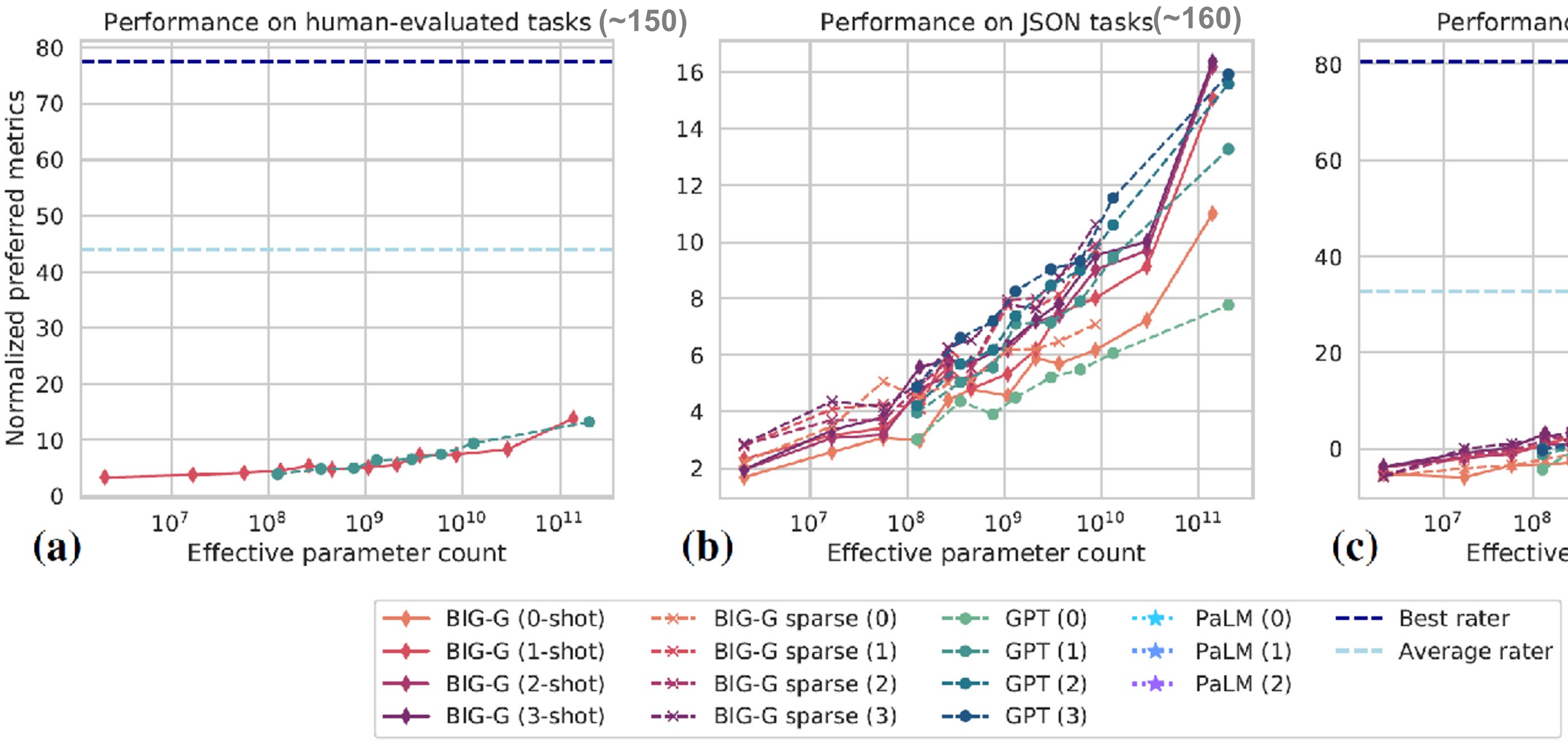
- Performance is **a measure** of a pair ⟨system, item⟩ :
  - Examples:
    - Correct prediction of MySpamFilter (system) on instance Email735 (the item)
    - 85% accuracy of ResNet23 (system) on dataset ImageNet (the aggregated item)
  - Performance changes when the item/distribution changes
    - On blurry, adversarial, OOD images the result is much worse
- Capability is **a property** of a system:
  - Examples:
    - The system can add integers up to three digits.
    - The system can jump up to 1.20 metres high.
  - Capability doesn't change when the item/distribution changes
    - Bar at 1.50 metres high? Bad performance because the capability is lower.

What are we measuring and extrapolating?

Beyond the Imitation Game benchmark (BIG-bench)

BEYOND THE IMITATION GAME: QUANTIFYING AND EXTRAPOLATING THE ~~CAPABILITIES~~ performance OF LANGUAGE MODELS

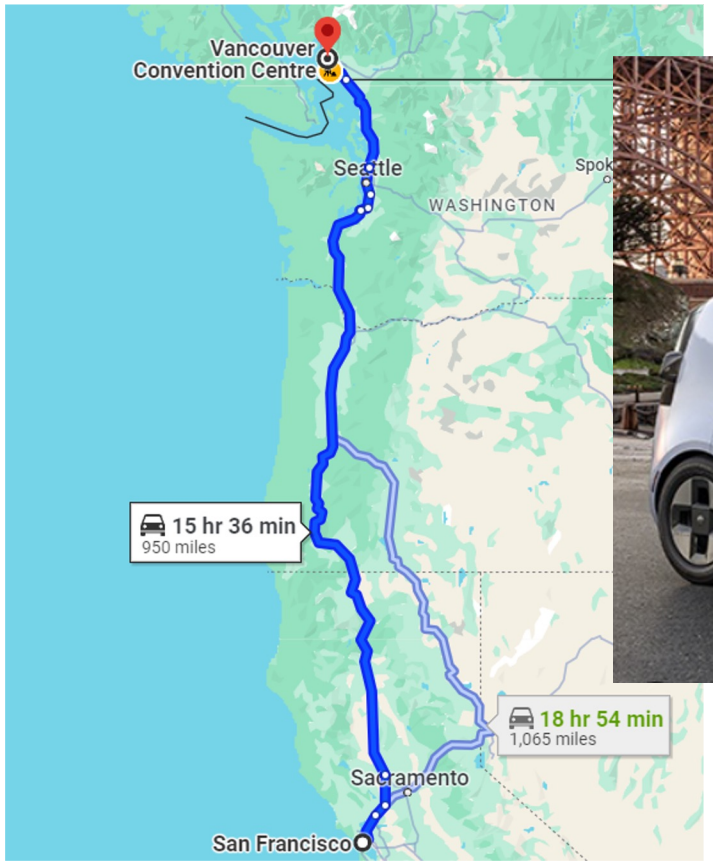
BIGBench: Massive benchmark with more than 200 tasks!



Adam R. Brown, Adam Santoro,
Alex Ray, Alex Warstadt, Alexan-
skell, Amanda Dsouza, Ambrose
eas Stuhlmüller, Andrew Dai, An-
a Gottardi, Antonio Norelli, Anu
Ashish Sabharwal, Austin Herrick,
ski, Batuhan Özyurt, Behnam He-
d, Cameron Diao, Cameron Dour,
g, Chitta Baral, Chiyu Wu, Chris
, Clara E. Rivera, Clemencia Siro,
lman, Dan Roth, Daniel Freeman,
en, Daphne Ippolito, Dar Gilboa,
, Deniz Yuret, Derek Chen, Derek
erina Shutova, Ekin Dogus Cubuk,
Emma Lam, Eric Chu, Eric Tang,
genii Zheltonozhskii, Fanyue Xia,
a, Genta Indra Winata, Gerard de
López, Gregor Betz, Guy Gur-Ari,
Shevlin, Hinrich Schütze, Hiromu
ernion, Jacob Hilton, Jaehoon Lee,
on, Jared Kaplan, Jarema Radom,
r Marsh, Jeremy Kim, Jeroen Taal,
er, John U. Balis, Jonathan Berant,
oshua S. Rule, Joyce Chua, Kamil
h D. Dhole, Kevin Gimpel, Kevin
ardson, Laria Reynolds, Leo Gao,
Lucas Lam, Lucy Noble, Ludwig
aartje ter Hoeve, Maheen Farooqi,
tana, Marie Tolkiehn, Mario Giu-
una Baitemirova, Melody Arnaud,
hael Strube, Michał Swędrowski,
Mohit Bansal, Moin Aminnaseri,
over, Nicholas Cameron, Nicholas
tha S. Iyer, Noah Constant, Noah
io Moreno Casares, Parth Doshi,
g, Peter Eckersley, Phu Mon Htut,
ang Chen, Rabin Banjade, Rachel
hard Barnes, Rif A. Saurous, Riku
ras, Rosanne Liu, Rowan Jacobs,
ingh, Saif M. Mohammad, Sajant
oenholz, Sanghyun Han, Sanjeev
ann, Sebastian Schuster, Sepideh
iang Shane Gu, Shubh Pachchigar,
eyer, Simone Melzi, Siva Reddy,
vic, Stefano Ermon, Stella Bider-
Kiritchenko, Swaroop Mishra, Tal
Rothschild, Thomas Phan, Tianle
stenberg, Trenton Chang, Trishala
ak, Vinay Ramasesh, Vinay Uday
Vossen, Xiang Ren, Xiaoyu Tong,
ri, Yejin Choi, Yichi Yang, Yiding
Wang, Zijie J. Wang, Zirui Wang,

Aggregate scores limit OOD extrapolation

Will the car take me from SF to Vancouver safely and on time?

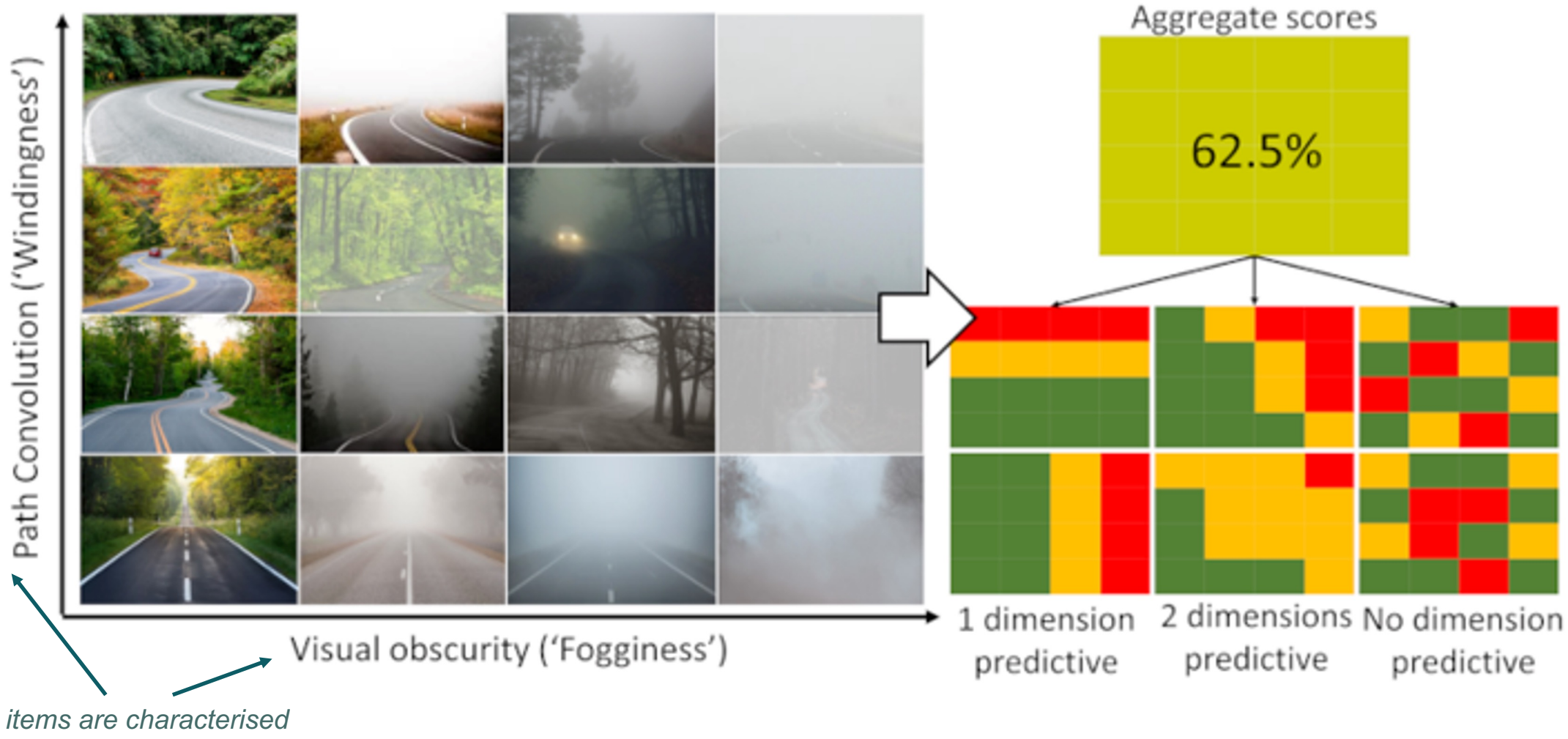


Aggregate scores

62.5%

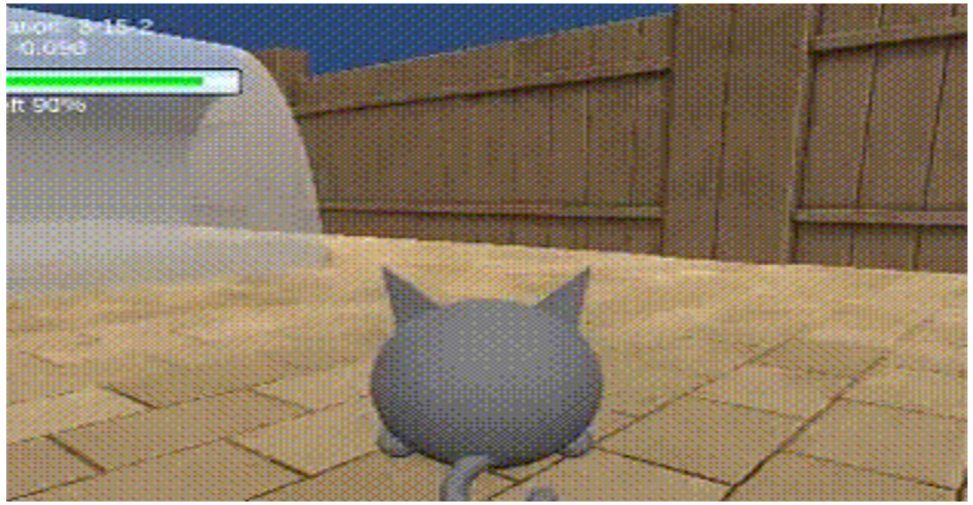
Features can allow for OOD extrapolation

Will the car take me from SF to Vancouver safely and on time?



Identifying features of interest: example

- Selected subset of AAIO instances measuring simple goal-directed behaviour
- Data across 99 instances from 68 agents

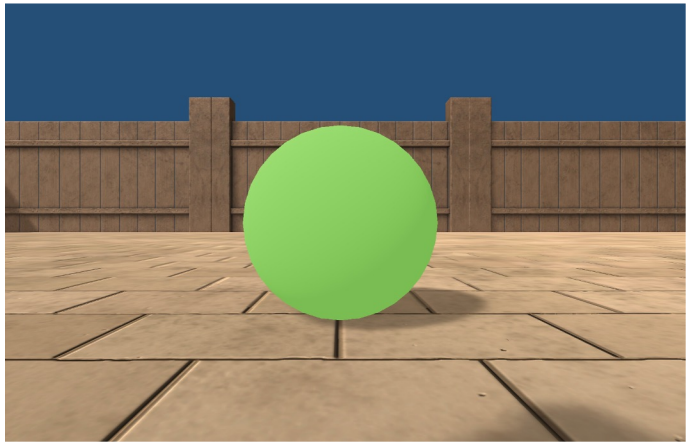


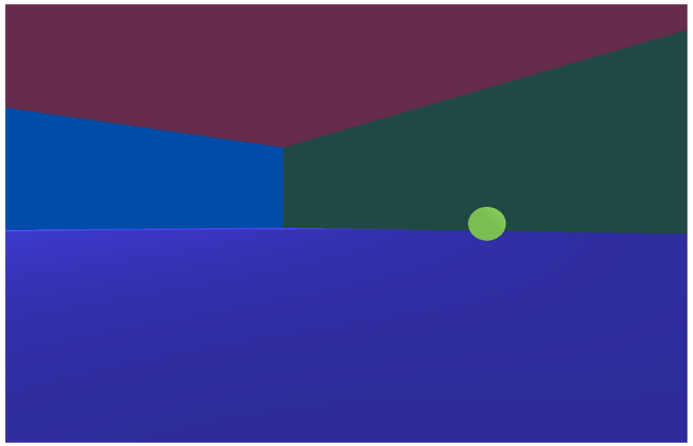
M Crosby, B Beyret, M Shanahan, J Hernández-Orallo, L Cheke, M Halina “The animal-AI testbed and competition” NeurIPS 2019 Competition and Demonstration Track, Proceedings of Machine Learning Research, 2020

animalai.org

Identifying features of interest: relevant / irrelevant

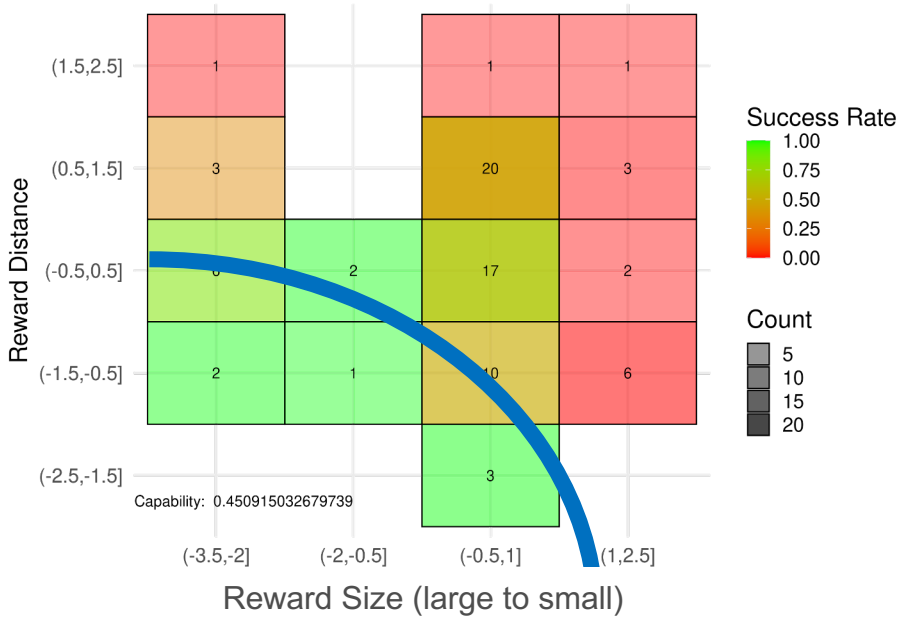
- Relevant
  - Reward size
  - Reward distance
  - Reward in view (i.e., in front vs behind)
- Irrelevant
  - Reward side (left vs right)
  - Reward colour (green vs yellow)





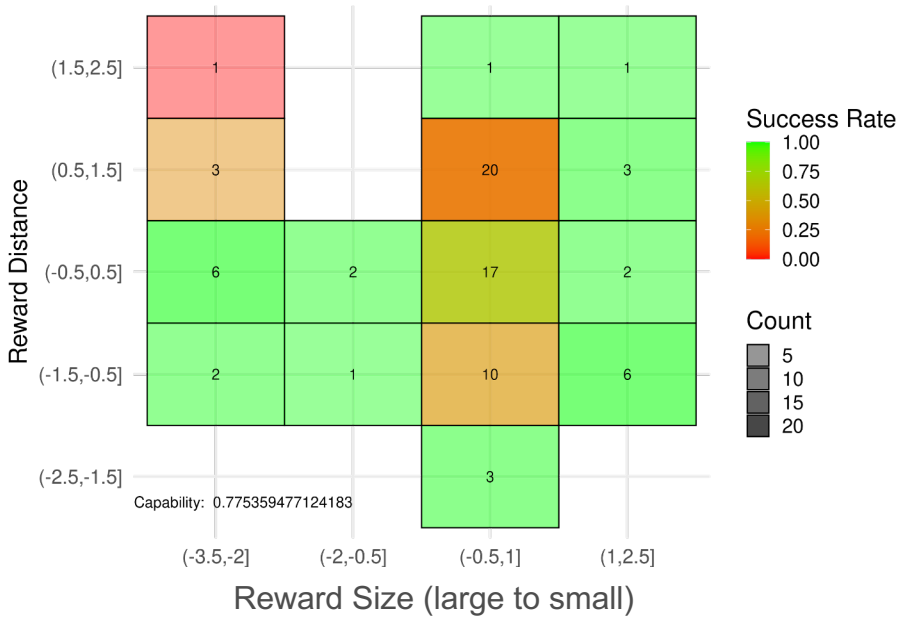
Capabilities vs no-capabilities

Capability boundary



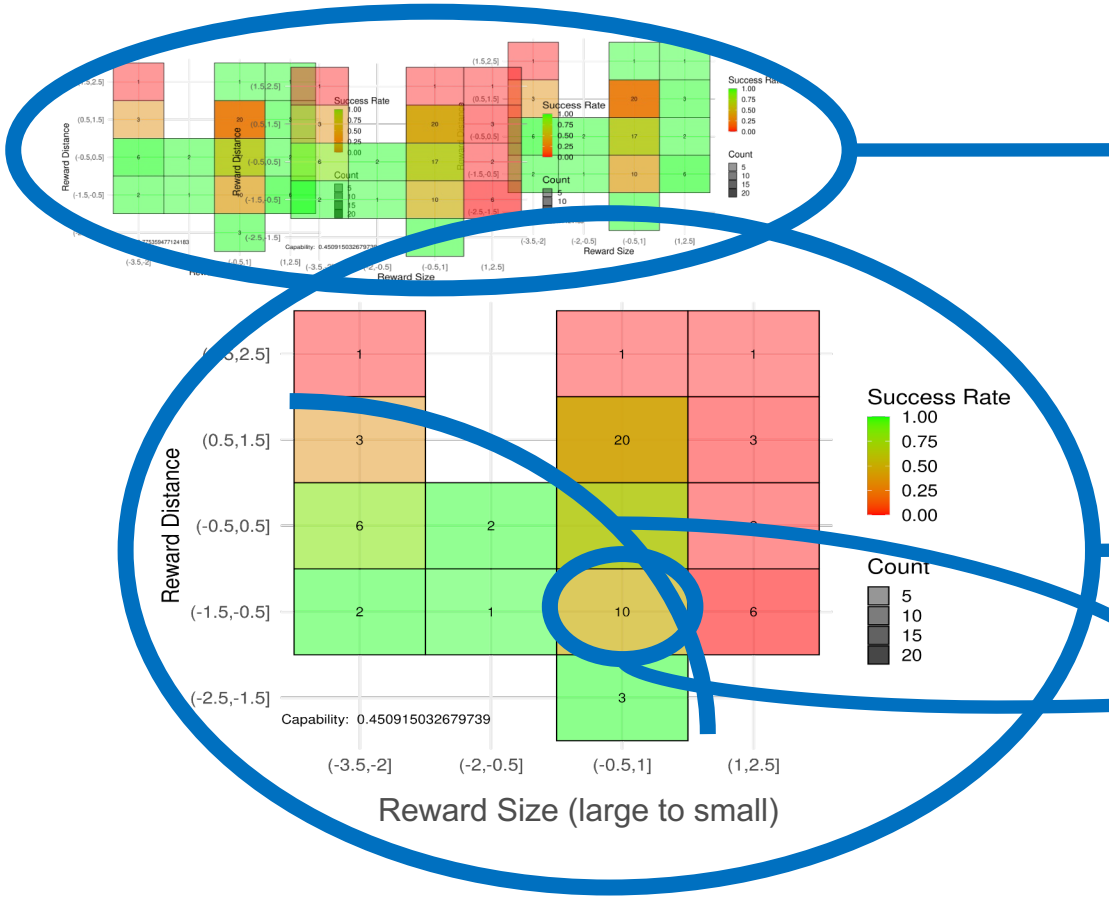
Conformant System
Juohmaru

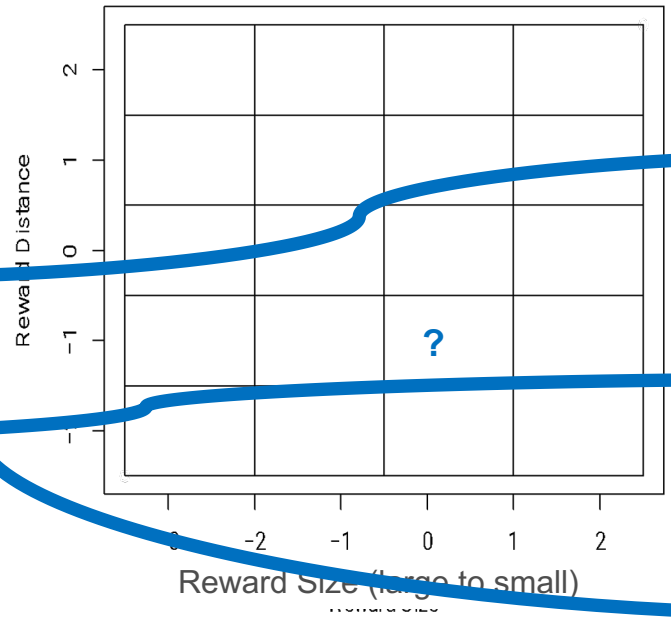
This system doesn't show monotonicity.
We can't identify any level of capability robustly.



Non-Conformant System
y.yang

Predicting performance POSSIBLE





avg

G.Acc. : extrapolate Global ACCuracy
? = 54.7%
(ignores system locality and feature relevance)

avg

T.Acc. extrapolate agenT ACCuracy
? = 46.8%
(ignores feature relevance)

avg

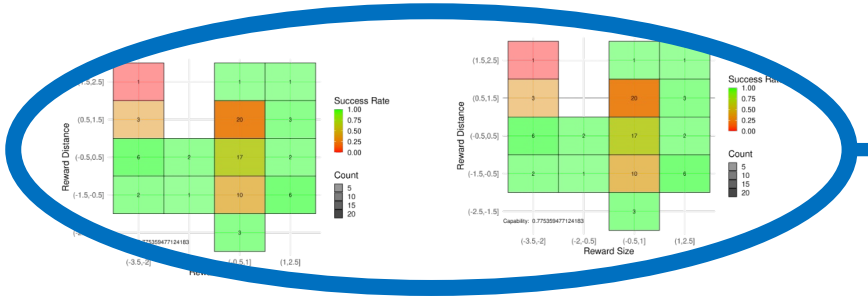
B.Acc. : extrapolate Bin ACCuracy
? = 40%
(ignores other bins)

logistic

Par. : use parametric model on capabilities
? = 73.2%
(parameter goodness-of-fit may be poor)

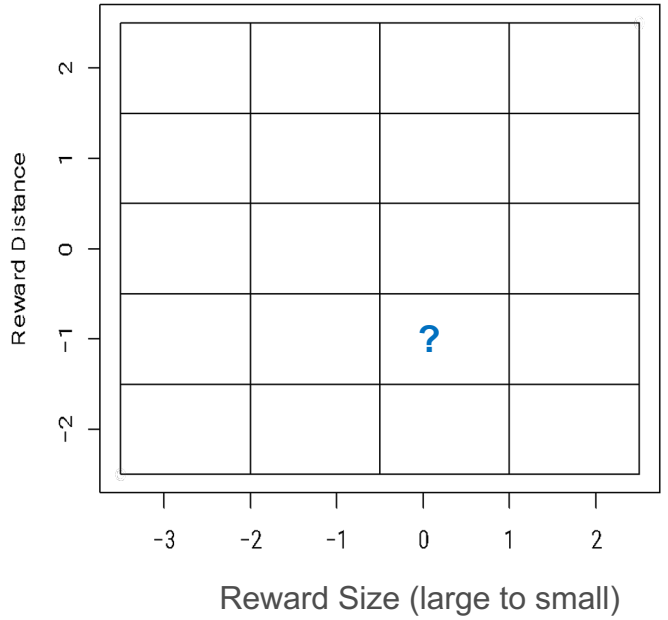
Except the last one, these are basically non-inferential methods (constant models or binning extrapolations)

Predicting performance NOT POSSIBLE?



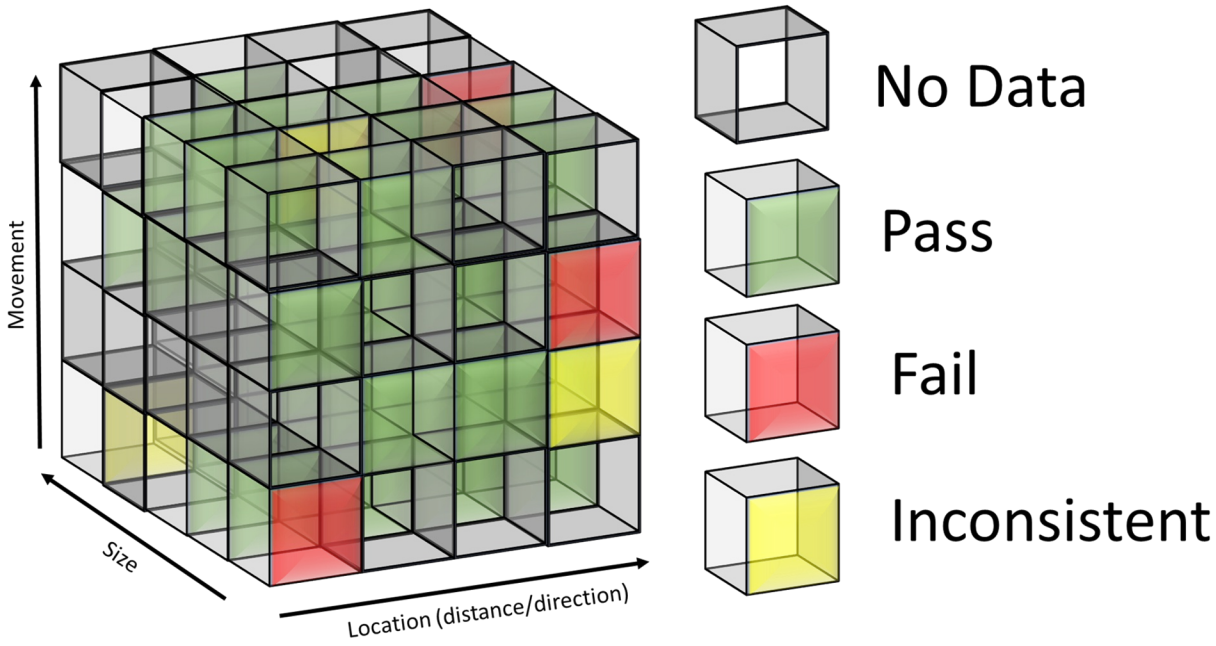
ML model

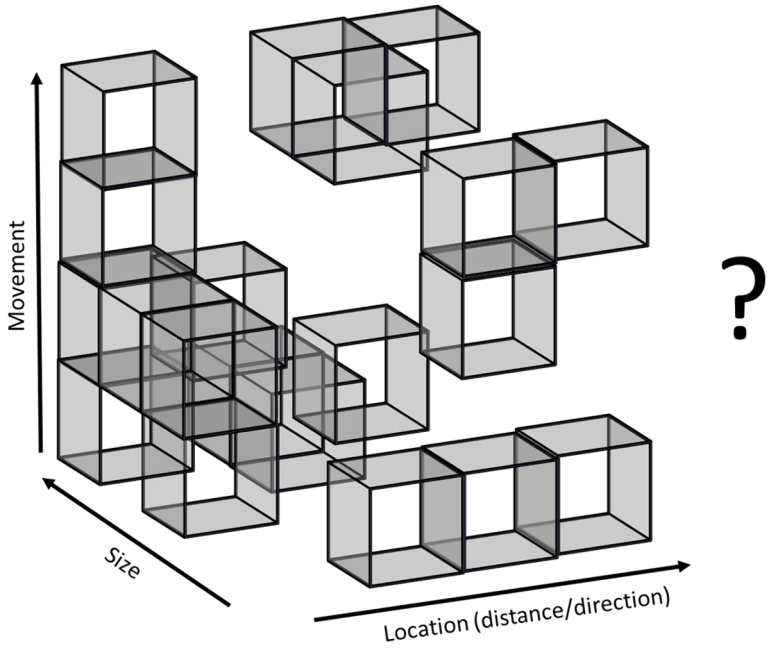
A : use assessor models
(Using all variables or only the relevant ones?)



assessors = let's use all the power of ML to characterise the system's performance!!

We need data at the granular level!





Need for granular data!

- Instance-level data:
  - For building good predictive models of AI validity, we need evaluation results at the instance level.

Is sharing code open source (github) enough?
Re-running the experiments is not feasible/sustainable anymore.

ARTIFICIAL INTELLIGENCE

Rethink reporting of evaluation results in AI

Aggregate metrics and lack of access to results limit understanding

By Ryan Burnell[1], Wout Schellaert[2], John Burden[1,3], Tomer D. Ullman[4], Fernando Martinez-Plumed[2], Joshua B. Tenenbaum[5], Danaja Rutar[1], Lucy G. Cheke[1,6], Jascha Sohl-Dickstein[7], Melanie Mitchell[8], Douwe Kiela[9], Murray Shanahan[10,11], Ellen M. Voorhees[12], Anthony G. Cohn[13,14,15,16], Joel Z. Leibo[10], Jose Hernandez-Orallo[1,2,3]

Artificial intelligence (AI) systems have begun to be deployed in high-stakes contexts, including autonomous driving and medical diagnosis. In contexts such as these, the consequences of system failures can be devastating. It is therefore vital that researchers and policymakers have a full understanding of the capabilities and weaknesses of AI systems so that they can make informed decisions about where these systems are safe to use and how they might be improved. Unfortunately, current approaches to AI evaluation make it exceedingly difficult to build such an understanding, for two key reasons. First, aggregate metrics make it hard to predict how a system will perform in a particular situation. Second, the instance-by-instance evaluation results that could be used to unpack these aggregate metrics are rarely made available (*1*). Here, we propose a path forward in which results are presented in more nuanced ways and instance-by-instance evaluation results are made publicly available.

Across most areas of AI, system evaluations follow a similar structure. A system is first built or trained to perform a particular set of functions. Then, the performance of the system is tested on a set of tasks relevant to the desired functionality of the system. In many areas of AI, evaluations use standardized sets of tasks known as "benchmarks." For each task, the system will be tested on a number of example "instances" of the task. The system would then be given a score for each instance based on its performance, e.g., 1 if it classified an image correctly, or 0 if it was incorrect. For other systems, the score for each instance might be based on how quickly the system completed its task, the quality of its outputs, or the total reward it obtained. Finally, performance across the various instances and tasks is usually aggregated to a small number of metrics that summarize how well the system performed, such as percentage accuracy.

But aggregate metrics limit our insight into performance in particular situations, making it harder to find system failure points and robustly evaluate system safety. This problem is also worsening as the increasingly broad capabilities of state-of-the-art systems necessitate ever more diverse benchmarks to cover the range of their capabilities. This problem is further exacerbated by a lack of access to the instance-by-instance results underlying the aggregate metrics, making it difficult for researchers and policy-makers to further scrutinize system behavior.

AGGREGATE METRICS

Use of aggregate metrics is understandable. They provide information about system performance "at a glance" and allow for simple comparisons across systems. But aggregate performance metrics obfuscate key information about where systems tend to succeed or fail (*2*). Take, for example, a system that was trained to classify faces as male or female that achieved classification accuracy of 90% (*3*). Based on this metric, the system appears highly competent. However, a subsequent breakdown of performance revealed that the system misclassified females with darker skin types a staggering 34.5% of the time, while erring only 0.8% of the time for males with lighter skin types. This example demonstrates how aggregation can make it difficult for policymakers to determine the fairness and safety of AI systems.

Compounding this problem, many benchmarks include disparate tasks that are ultimately aggregated together. For example, the Beyond the Imitation Game Benchmark (BIG-bench) for language models includes over 200 tasks that evaluate everything from language understanding to causal reasoning (*4*). Aggregating across these disparate tasks—as the BIG-bench leaderboard does—reduces the rich information in the benchmark to an overall score that is hard to interpret.

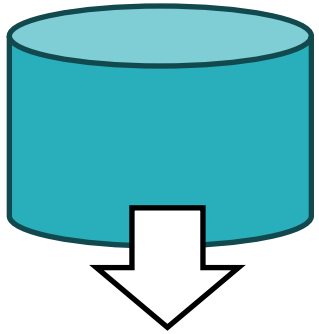
It is also easy for aggregation to introduce unwarranted assumptions into the evaluation process. For example, a simple average across tasks implicitly treats every task as equally important—in the case of BIG-bench, a sports understanding task has as much bearing on the overall score as a causal reasoning task. These aggregation decisions have huge implications for the conclusions that are drawn about system capabilities, yet are seldom considered carefully or explained.

Aggregate metrics depend not only on the capability of the system but also on the characteristics of the instances used for evaluation. If the gender classification system above were reevaluated by using entirely light-skinned faces, accuracy would skyrocket, even though the system's ability to classify faces has not changed. Aggregate metrics can easily give false impressions about capabilities when a benchmark is not well constructed.

Problems and trade-offs that arise when considering aggregate versus granular data and metrics are not specific to AI, but they are exacerbated by the challenges inherent in AI research and the research practices of the field. For example, machine learning evaluations usually involve randomly splitting data into training, validation, and test sets. An enormous amount of data is required to train state-of-the-art systems, so these datasets are often poorly curated and lack the detailed annotation necessary to conduct granular analyses. In addition, the research culture in AI is centered around outdoing the current state-of-the-art performance, as evidenced by the many lea-

[1]Leverhulme Centre for the Future of Intelligence, University of Cambridge, Cambridge, UK. [2]Valencian Research Institute for Artificial Intelligence, Universitat Politècnica de Valencia, Valèncida, Spain. [3]Centre for the Study of Existential Risk, University of Cambridge, Cambridge, UK. [4]Department of Psychology, Harvard University, Cambridge, MA, USA. [5]Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA. [6]Department of Psychology, University of Cambridge, Cambridge, UK. [7]Brain team, Google, Mountainview, CA, USA. [8]Santa Fe Institute, Santa Fe, NM, USA. [9]Stanford University, Stanford, CA, USA. [10]DeepMind, London, UK. [11]Department of Computing, Imperial College London, London, UK. [12]National Institute of Standards and Technology (Retired), Gaithersburg, MD, USA. [13]School of Computing, University of Leeds, Leeds, UK. [14]Alan Turing Institute, London, UK. [15]Tongji University, Shanghai, China. [16]Shandong University, Jinan, China. Email: rb967@cam.ac.uk

From characteristic grids to capabilities profiles



Past test data

Success

Convolution Capability	8.3
Visual Capability	2.5
Traffic Capability	7.2
...	

Cognitive Profile

Fail

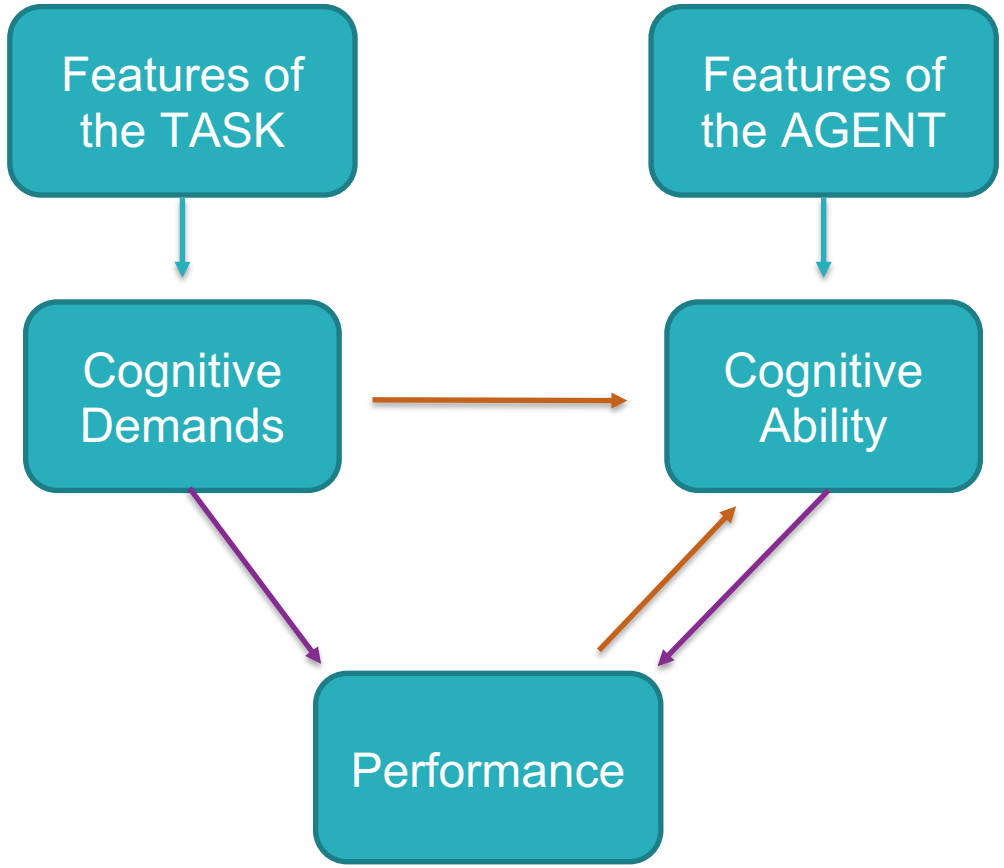
Modelling capabilities

We build on:

- The (cognitive) demands for each task instance
- Performance dictated from agent abilities meeting demands
- The assumed relationship between demands and abilities

…to infer the capability profile of a subject from the performance data of that subject only across a number of diverse instances.

The Measurement Layouts are Hierarchical Bayesian networks, which allow this forwards and backwards inference.

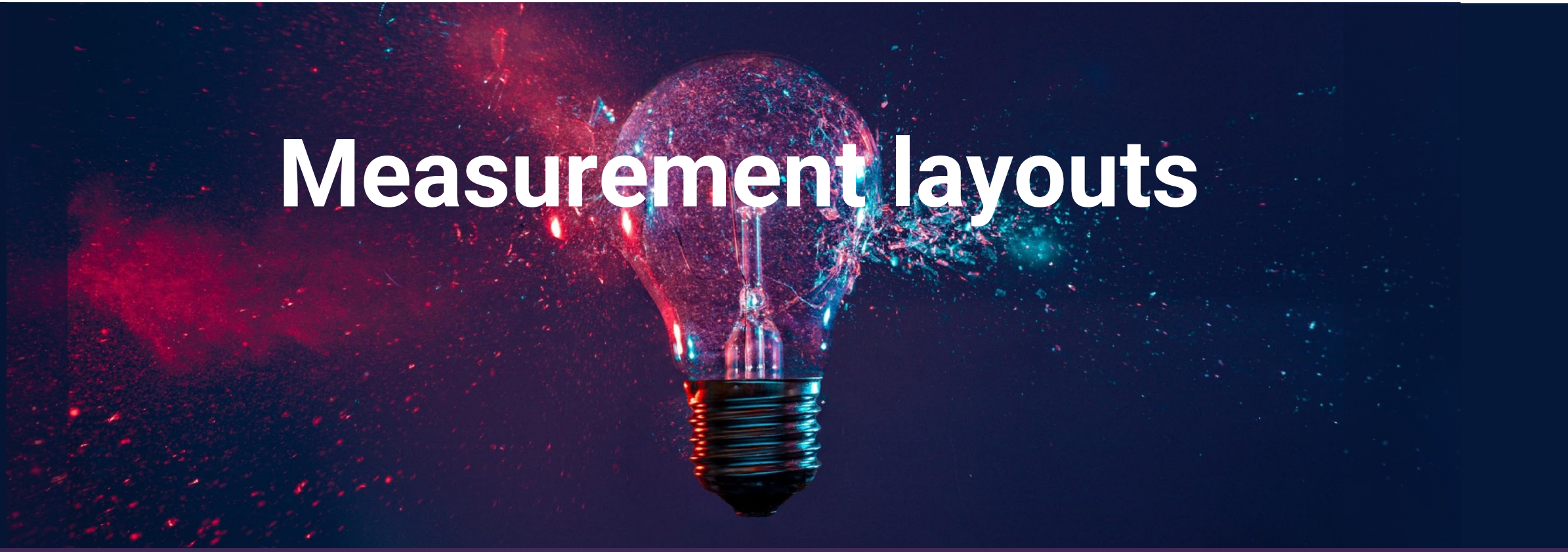


Pointers and Resources

- Burden, J., Voudouris, K., Burnell, R., Rutar, D., Cheke, L. & Hernandez-Orallo, J. (2023) Inferring Capabilities from Task Performance with Bayesian Triangulation. Arxiv preprint arXiv:2309.11975
- Burnell, R., Burden, J., Rutar, D., Voudouris, K., Cheke, L., & Hernandez-Orallo, J. (2022) Not a Number: Identifying Instance Features for Capability-Oriented Evaluation, *International Joint Conference on Artificial Intelligence* (IJCAI).
- Burnell, R., Schellaert, W., Burden, J., Ullman, T.D., Martinez- Plumed, F., Tenenbaum, J.B., Rutar, D., Cheke, L.C., Sohl-Dickstein, J. Mitchell, M., Kiela, D., Shanahan, M. Voorhees, E.M., Cohn A. G., Leibo, J.Z. & Hernandez-Orallo, J. (2023) “Rethink reporting of evaluation results in AI: Aggregate metrics and lack of access to results limit understanding”, *Science*, Vol 380, Issue 6641, pp. 136-138.

https://github.com/Kinds-of-Intelligence-CFI/measurement-layout-tutorial/

UNIVERSITY OF CAMBRIDGE



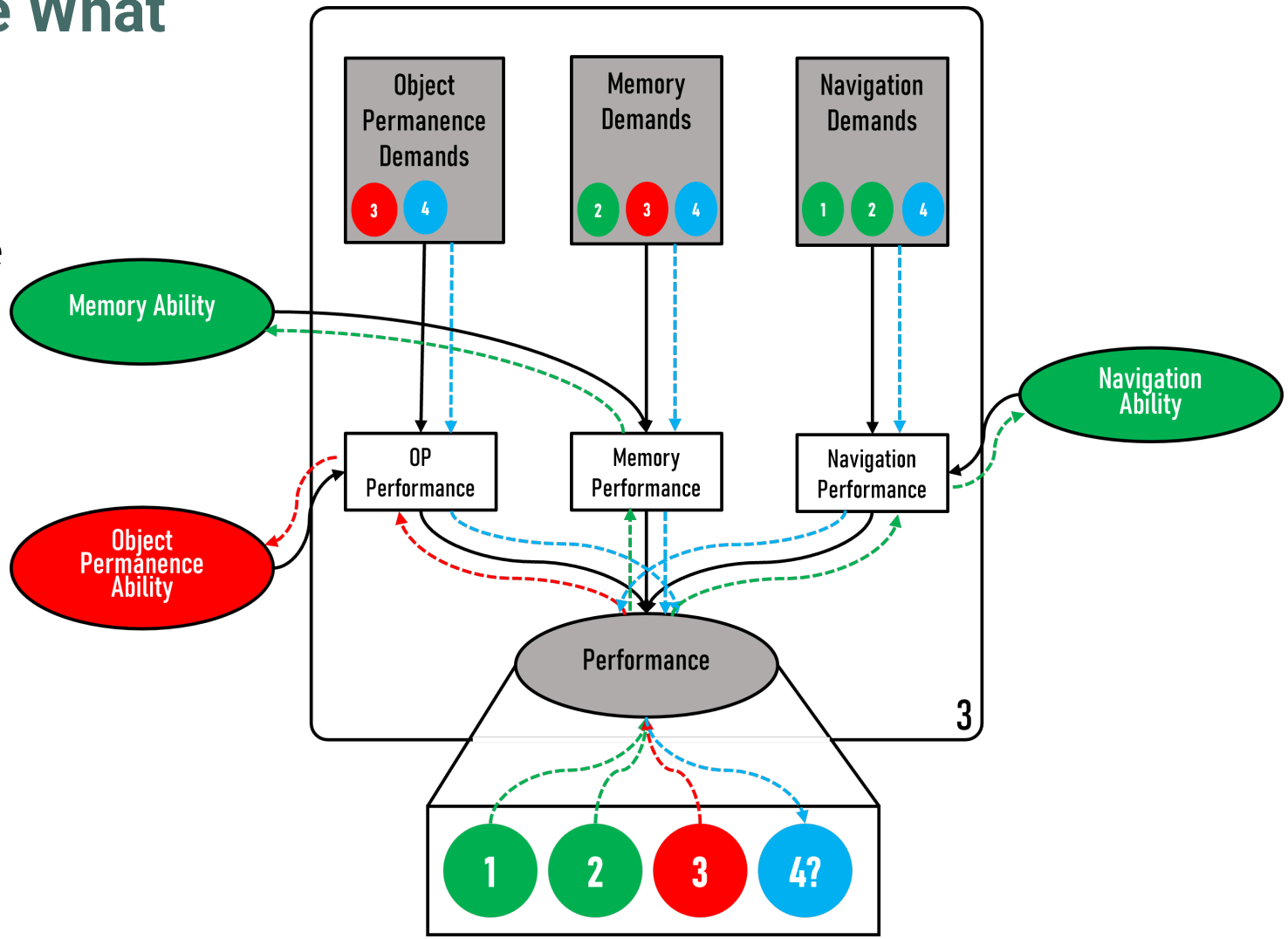
Measurement layouts

Intro to measurement layouts

- What are measurement layouts?
- How are they used?
- Why do we think they can provide capability-oriented evaluation?

Measurement Layouts – The What

- If we know the demands of a set of tasks or task instances and how they are related, we can build a Bayesian model that connects system capabilities and task demands.
- From a collection of instances, we can infer capabilities.
- Then we can predict future performance for new instances.

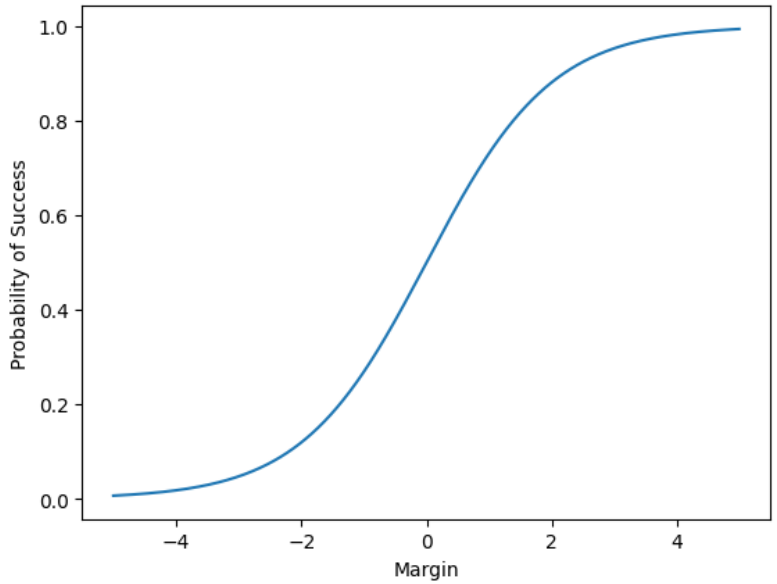


Measurement Layouts – The What

- Increased capabilities make success more likely, while increased demands decrease likelihood of success.
- For a single demand and capability:

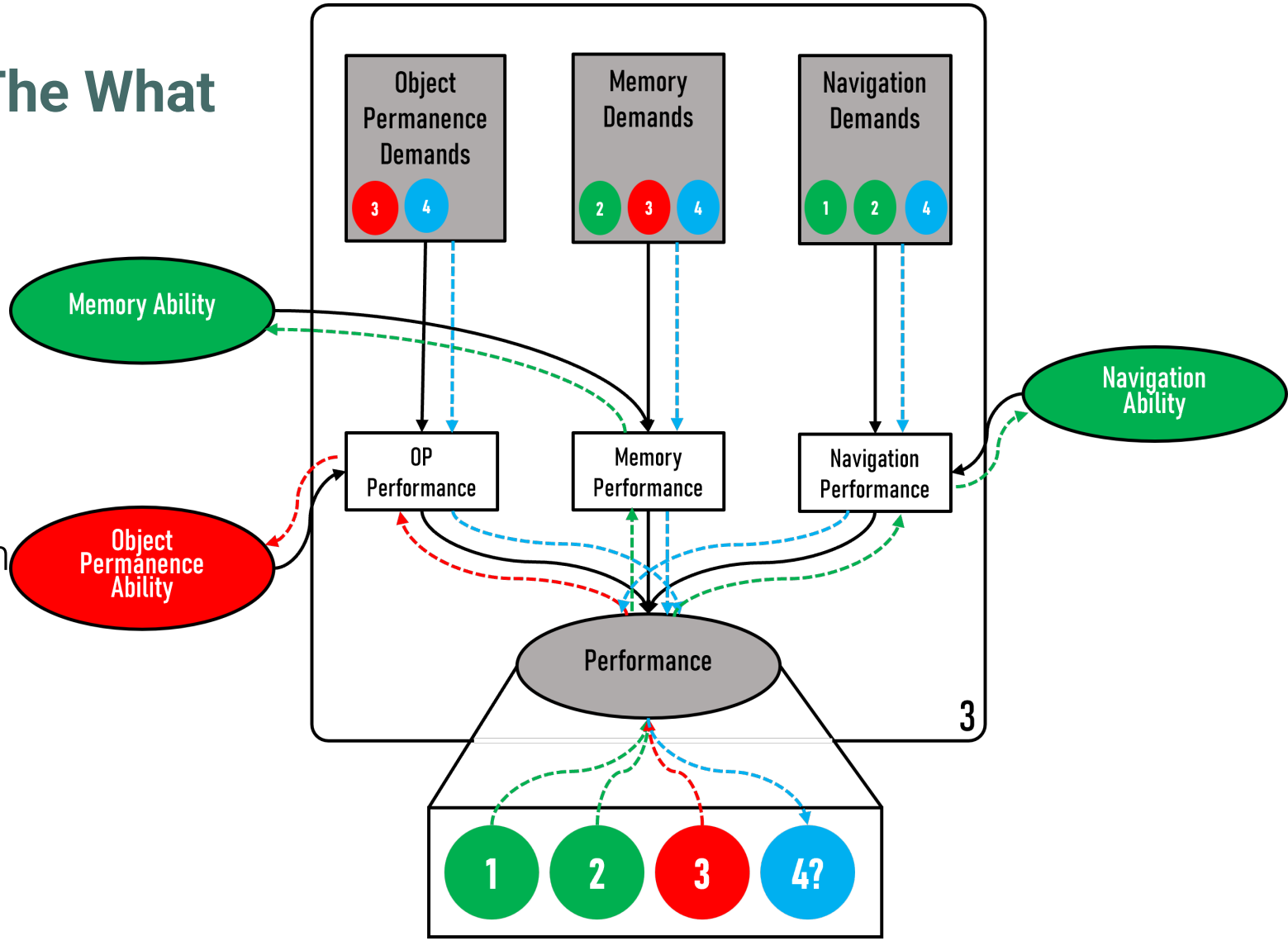
$$P(\mathrm{s}|\mathrm{capability}, \mathrm{demand}) = \sigma(\mathrm{capability} - \mathrm{demand})$$

- The difference between capability and demand is referred to as the *margin*.



Measurement Layouts – The What

- More than a single ability and demand.
- Introduce *partial performance* nodes.
- Final performance will depend on some combination of these.



Measurement Layouts – The What

- Partial performances are combined to give final performance. Two broad ways of thinking about that:
- Some tasks will be non-compensatory. High capabilities in one area will not compensate for lacking in others.
- Some tasks compensatory. One high capability can compensate for low levels in others.

Measurement Layouts – The How

- So, we have the structure of a measurement layout in place. What do we do with it?
- Need a dataset with performance outcomes labelled along with demands for each instance.
- Bayesian model needs priors.

  Either encode system knowledge or uninformative.
- We can then leverage a probabilistic programming implementation of the network using PyMC.

Measurement Layouts – The Why

- Why do we expect Measurement Layouts to help provide capability-oriented evaluation?

  Capabilities are independent of the distribution of tasks instances.

  Allows us to predict future performance at the instance-level. The Measurement Layout models reflect domain expertise.

  - Higher internal validity.
  - Enables predictions to be valid even out of distribution.

UNIVERSITY OF CAMBRIDGE

Evaluating capabilities of LLMs

Key challenges in building measurement layouts for LLMs: Performance data

- We need instance-level performance data
  - For each prompt need to have how the language model performed
    - Success/failure or if multiple choise which are the choises, which one did LLM pick and if correct or not
    - Addition example -- prompt: '`Sum of 112359321 and 23456543 is`'; success: `yes`

- Some good examples include: HELM and BIG-bench

Key challenges in building measurement layouts for LLMs: Meta-features/demands

- Many benchmarks don't include and/or don't have meta-features that describe each instance in the benchmark
  - How was the benchmark built?
  - What was considered when building the benchmark beyond just this a benchmark that tests X?
  - Are there any features that were explicitly manipulated?
- Addition: **number of digits in each number**; create new features: **average number of digits**, **number of carrying operations**, etc.

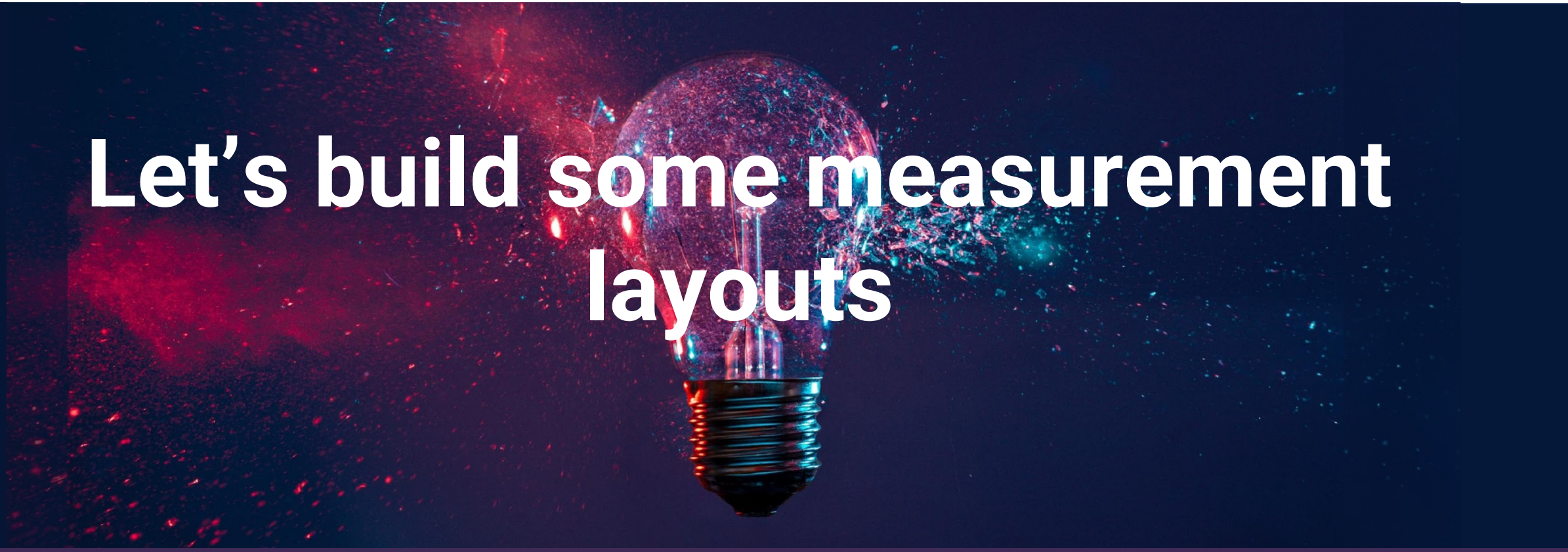
Key challenges in building measurement layouts for LLMs: Capabilities

- What capabilities are tested by those benchmarks?
  - Addition: what are the abilities that are being tested/informed or that need to match demands such as **average number of digits**, **number of carry operations** etc.
- Are those capabilities characterizable and have some backing in cognitive science (or some other sciences)?
- How do these capabilities interact? Does having one capability compensate for not having some other capability test on the benchmark?

Summary

- Need **instance-level** performance data
- Need **meta-features/demands** that describe each instance
- Need **capabilities** that are tested/informed by those demands

UNIVERSITY OF CAMBRIDGE



Let’s build some measurement layouts

CFI LEVERHULME CENTRE FOR THE FUTURE OF INTELLIGENCE

Exploring Measurement Layouts in PyMC

All notebooks, data and presentation are at:

- https://github.com/markotesic/Robust-Evaluation-of-Generative-AI/

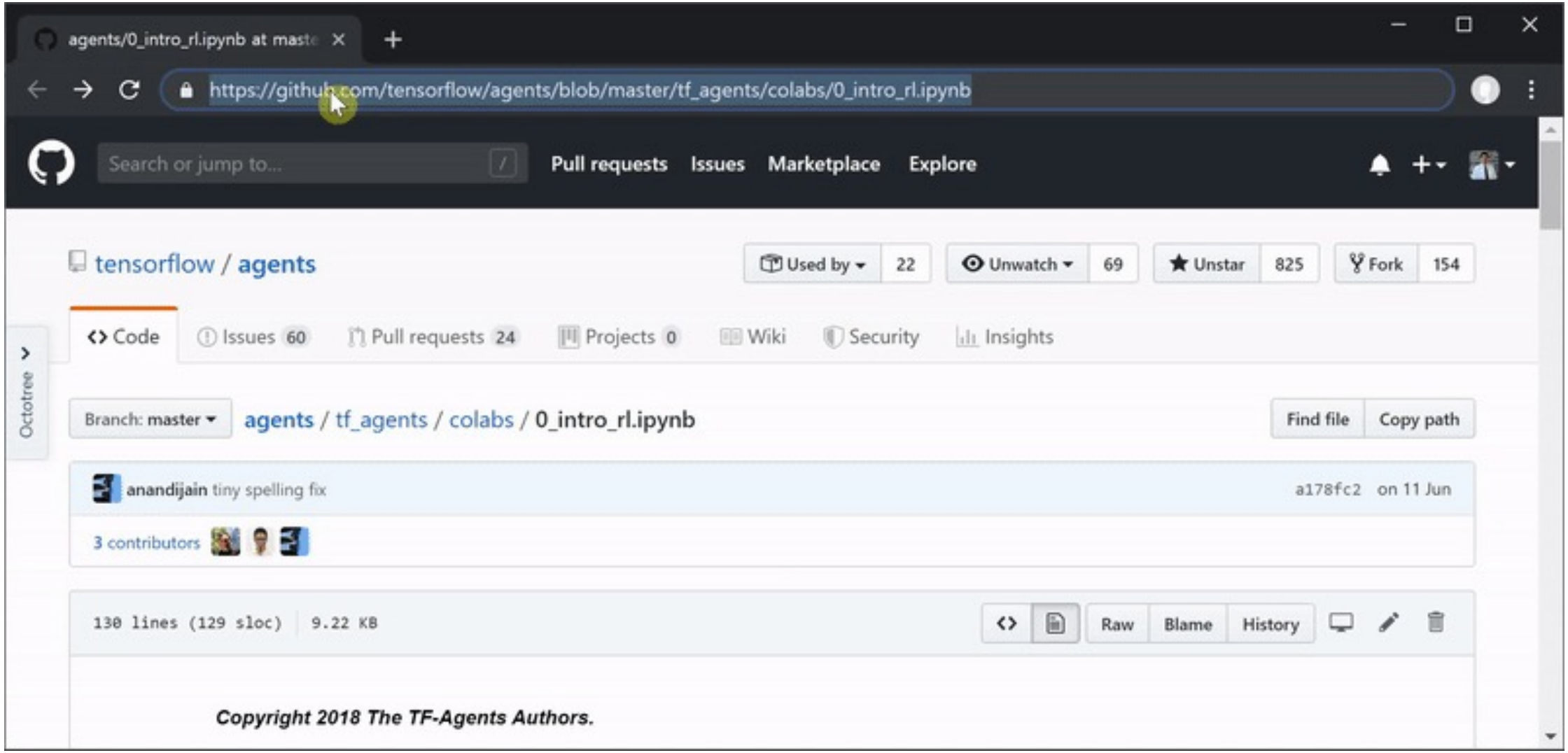
UNIVERSITY OF CAMBRIDGE

Addition data set

Addition dataset

- A simple benchmark testing LLMs abilities to add two numbers
  - E.g. prompt: `The sum of 656050998910983832047 and 4871137 is`
- 10 LLMs tested: GPT-3 Ada, GPT-3 Babbage, GPT-3 Curie, GPT-3 Davinci, text-davinci-001, text-davinci-002, text-davinci-003, GPT-3.5-turbo, GPT-4-0314, GPT-4-0613
- We will see how to create some of the **meta-features** for this benchmark and how to relate them to **capabilities** learnt by the measurement layout
- We will also train some common predictive models and baselines to compare to the performance of measurement layouts

Addition dataset



Addition dataset: overview

- `instance_id`: ID of an instance.
- `template_id`: ID of a prompt template. Each addition prompt was phrased in 8 different ways.
- `llm`: The large language model tested.
- `prompt`: The actual prompt.
- `response`: The LLM's response.
- `summand1`: The first number to be added.
- `summand2`: The second number to be added.
- `target`: The correct sum of the two summands.
- `digits1`: The number of digits in the first summand.
- `digits2`: The number of digits in the second summand.
- `min_digits`: $min(digits_1, digits_2)$, i.e., the number of digits in the smaller summand.
- `harm_mean`: $2/(1/digits_1 + 1/digits_2)$, i.e., the harmonic mean of the number of digits in the two summands.
- `art_mean`: $(digits_1 + digits_2)/2$, i.e., the arithmetic mean of the number of digits in the two summands.
- `max_digits`: $max(digits_1, digits_2)$, i.e., the number of digits in the larger summand.
- `carry`: The number of carrying operations required to add the two numbers.
- `success`: Indicates whether the response is correct (1) or incorrect (0).

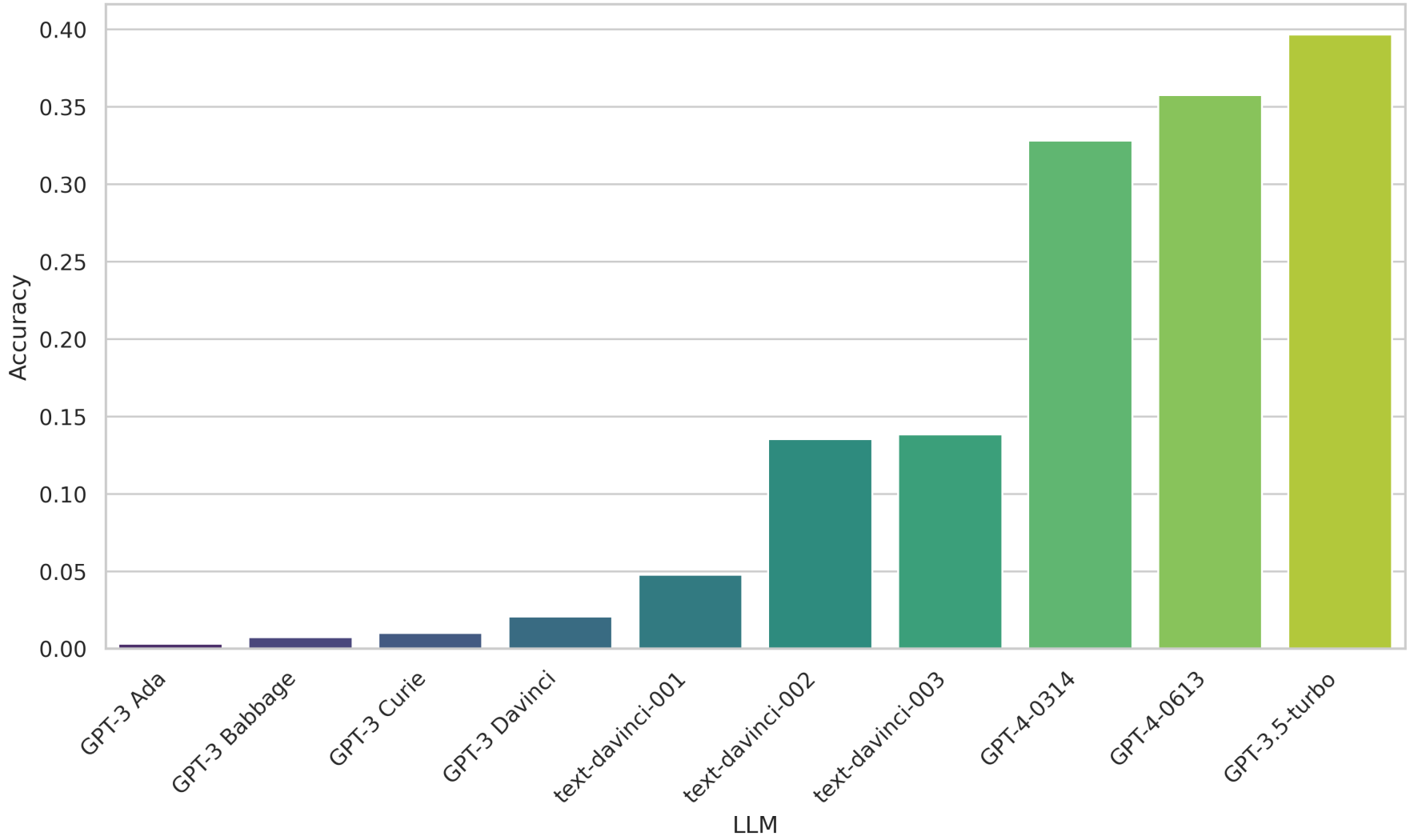
```
Template 1: The sum of 656050998910983832047 and 4871137 is
Template 2: Add: 656050998910983832047 + 4871137 =
Template 3: Make the addition of 656050998910983832047 and 4871137.
Template 4: By adding 656050998910983832047 and 4871137, the result is
Template 5: If you add 656050998910983832047 and 4871137, you get
Template 6: If you have 656050998910983832047 and 4871137, and you add them up, you get
Template 7: Find the value of x + y when x=656050998910983832047 and y=4871137.
Template 8: Imagine you have two numbers, 656050998910983832047 and 4871137, and you added them together. What number would you get?
```

Between 1 and 30 digits

Addition dataset

success	0	1
llm		
GPT-3 Ada	3286	10
GPT-3 Babbage	3272	24
GPT-3 Curie	3263	33
GPT-3 Davinci	3228	68
GPT-3.5-turbo	1989	1307
GPT-4-0314	2214	1082
GPT-4-0613	2117	1179
text-davinci-001	3139	157
text-davinci-002	2850	446
text-davinci-003	2840	456

Addition dataset: aggregate accuracy



Meta-features/demands

- `digits1`: The number of digits in the first summand.
- `digits2`: The number of digits in the second summand.
- `min_digits`: $min(digits_1, digits_2)$, i.e., the number of digits in the smaller summand.
- `harm_mean`: $2/(1/digits_1 + 1/digits_2)$, i.e., the harmonic mean of the number of digits in the two summands.
- `art_mean`: $(digits_1 + digits_2)/2$, i.e., the arithmetic mean of the number of digits in the two summands.
- `max_digits`: $max(digits_1, digits_2)$, i.e., the number of digits in the larger summand.
- `carry`: The number of carrying operations required to add the two numbers.

- Do these meta-features capture everything that could we could potentially think of when it comes to addition of two number? What are some of the things that make the addition of two number ‘difficult’?
- Size of the two numbers
- Number of carrying operations
- Can we have lots of carrying operations but the additions is still ‘easy’?

Meta-features/demands

- Standard deviation of digits in the two numbers (digit variety):

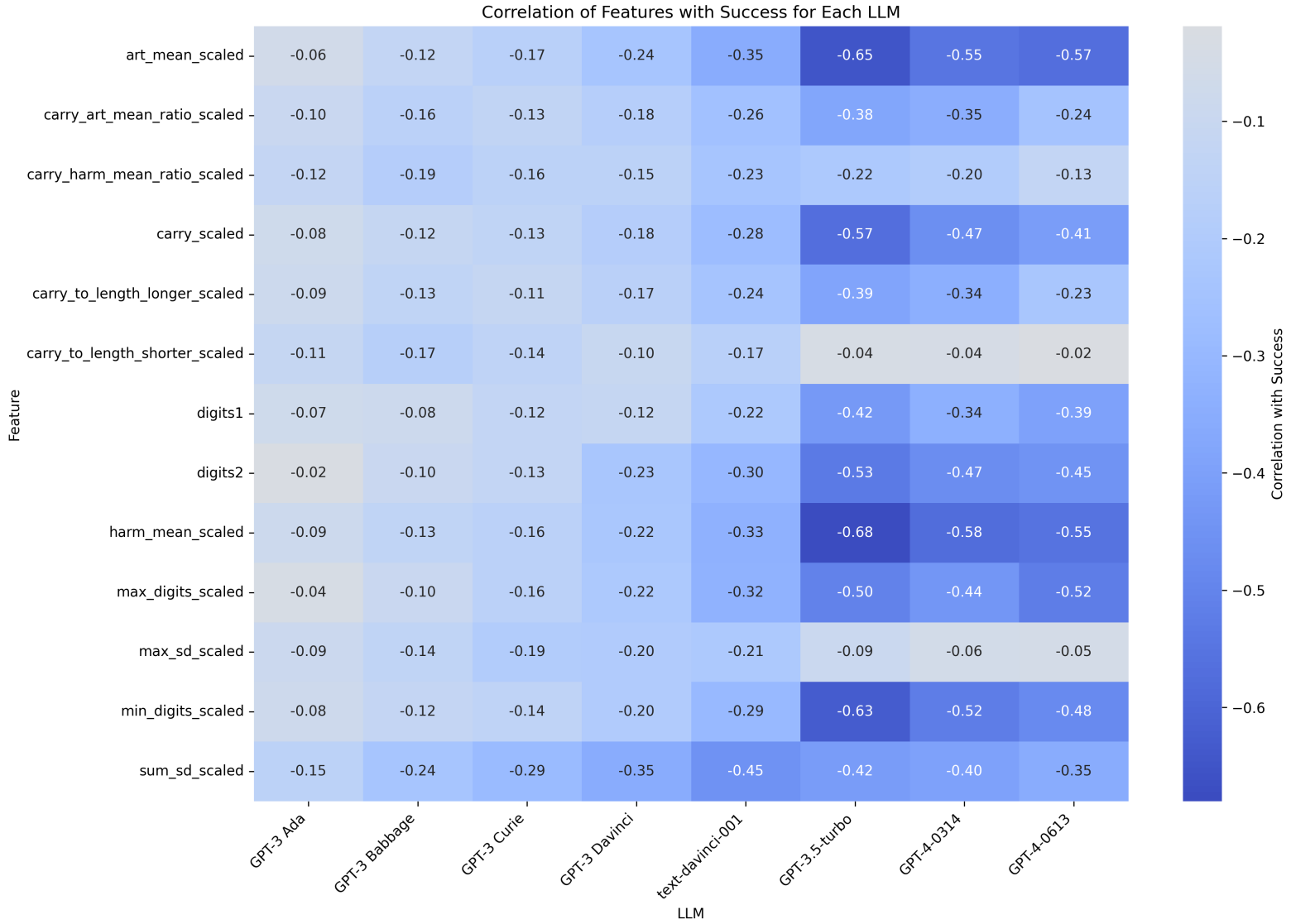
```
# Calculate max and sum of digit standard deviations
addition_data['max_sd'] = addition_data.apply(lambda x: max(num_sd(x['summand1']), num_sd(x['summand2'])), axis=1)
addition_data['sum_sd'] = addition_data.apply(lambda x: num_sd(x['summand1']) + num_sd(x['summand2']), axis=1)
```

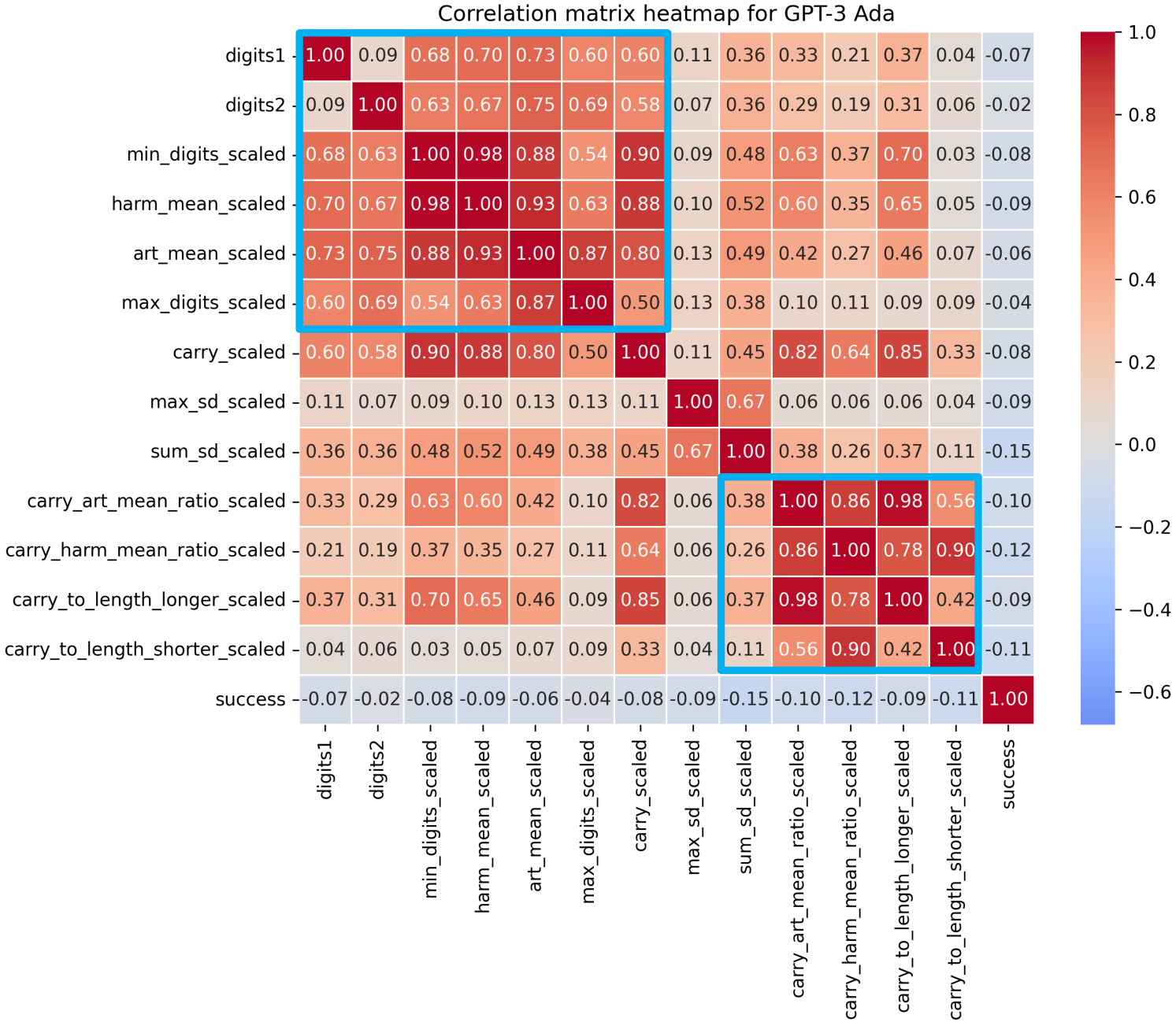
- Calculate the ratio of carry to size:

```
# Calculate ratio of carry and harmonic mean columns
addition_data['carry_harm_mean_ratio'] = addition_data['carry'] / addition_data['harm_mean']
addition_data['carry_art_mean_ratio'] = addition_data['carry'] / addition_data['art_mean']
addition_data['carry_to_length_longer'] = addition_data['carry'] / addition_data['max_digits']
addition_data['carry_to_length_shorter'] = addition_data['carry'] / addition_data['min_digits']
```

- Scale data: min-max normalization to range between 0 and 10

Meta-features/demands: correlations with success




Correlation matrix heatmap for GPT-3 Ada
digits1 1.00 0.09 0.68 0.70 0.73 0.60 0.60 0.11 0.36 0.33 0.21 0.37 0.04 -0.07
digits2 0.09 1.00 0.63 0.67 0.75 0.69 0.58 0.07 0.36 0.29 0.19 0.31 0.06 -0.02
min_digits_scaled 0.68 0.63 1.00 0.98 0.88 0.54 0.90 0.09 0.48 0.63 0.37 0.70 0.03 -0.08
harm_mean_scaled 0.70 0.67 0.98 1.00 0.93 0.63 0.88 0.10 0.52 0.60 0.35 0.65 0.05 -0.09
art_mean_scaled 0.73 0.75 0.88 0.93 1.00 0.87 0.80 0.13 0.49 0.42 0.27 0.46 0.07 -0.06
max_digits_scaled 0.60 0.69 0.54 0.63 0.87 1.00 0.50 0.13 0.38 0.10 0.11 0.09 0.09 -0.04
carry_scaled 0.60 0.58 0.90 0.88 0.80 0.50 1.00 0.11 0.45 0.82 0.64 0.85 0.33 -0.08
max_sd_scaled 0.11 0.07 0.09 0.10 0.13 0.13 0.11 1.00 0.67 0.06 0.06 0.06 0.04 -0.09
sum_sd_scaled 0.36 0.36 0.48 0.52 0.49 0.38 0.45 0.67 1.00 0.38 0.26 0.37 0.11 -0.15
carry_art_mean_ratio_scaled 0.33 0.29 0.63 0.60 0.42 0.10 0.82 0.06 0.38 1.00 0.86 0.98 0.56 -0.10
carry_harm_mean_ratio_scaled 0.21 0.19 0.37 0.35 0.27 0.11 0.64 0.06 0.26 0.86 1.00 0.78 0.90 -0.12
carry_to_length_longer_scaled 0.37 0.31 0.70 0.65 0.46 0.09 0.85 0.06 0.37 0.98 0.78 1.00 0.42 -0.09
carry_to_length_shorter_scaled 0.04 0.06 0.03 0.05 0.07 0.09 0.33 0.04 0.11 0.56 0.90 0.42 1.00 -0.11
success -0.07 -0.02 -0.08 -0.09 -0.06 -0.04 -0.08 -0.09 -0.15 -0.10 -0.12 -0.09 -0.11 1.00
digits1
digits2
min_digits_scaled
harm_mean_scaled
art_mean_scaled
max_digits_scaled
carry_scaled
max_sd_scaled
sum_sd_scaled
carry_art_mean_ratio_scaled
carry_harm_mean_ratio_scaled
carry_to_length_longer_scaled
carry_to_length_shorter_scaled
success
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

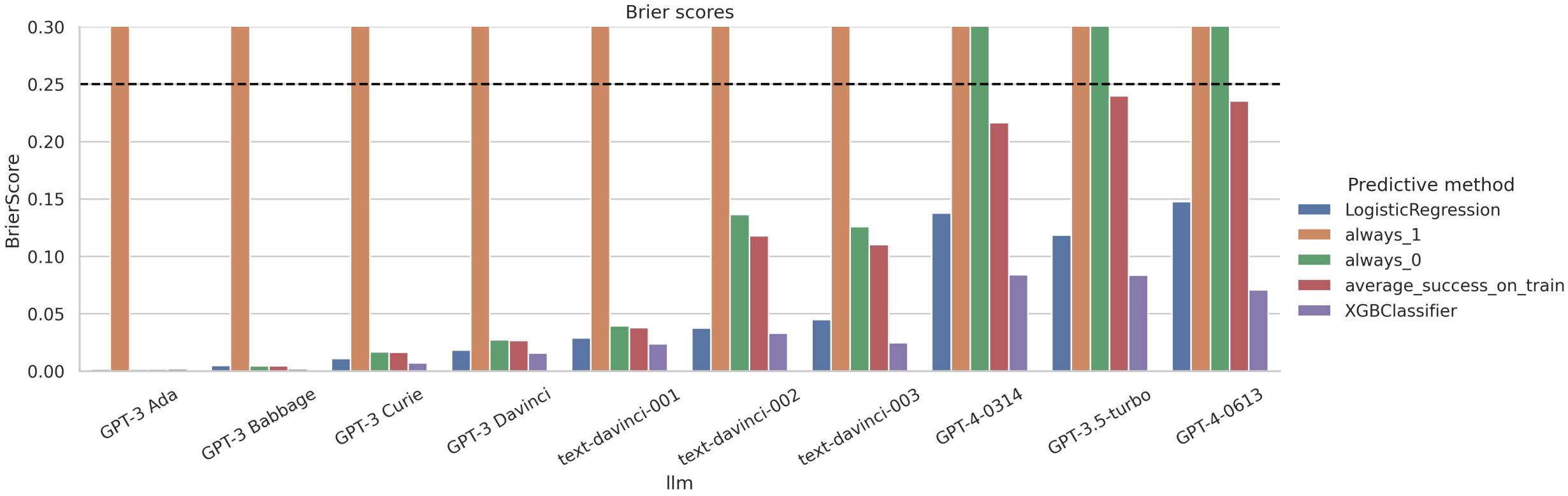

harm_mean_scaled
sum_sd_scaled
carry_art_mean_ratio_scaled
GPT-3 Ada
GPT-3 Babbage
GPT-3 Curie
GPT-3 Davinci
text-davinci-001


text-davinci-002
text-davinci-003
GPT-4-0314
GPT-3.5-turbo
GPT-4-0613

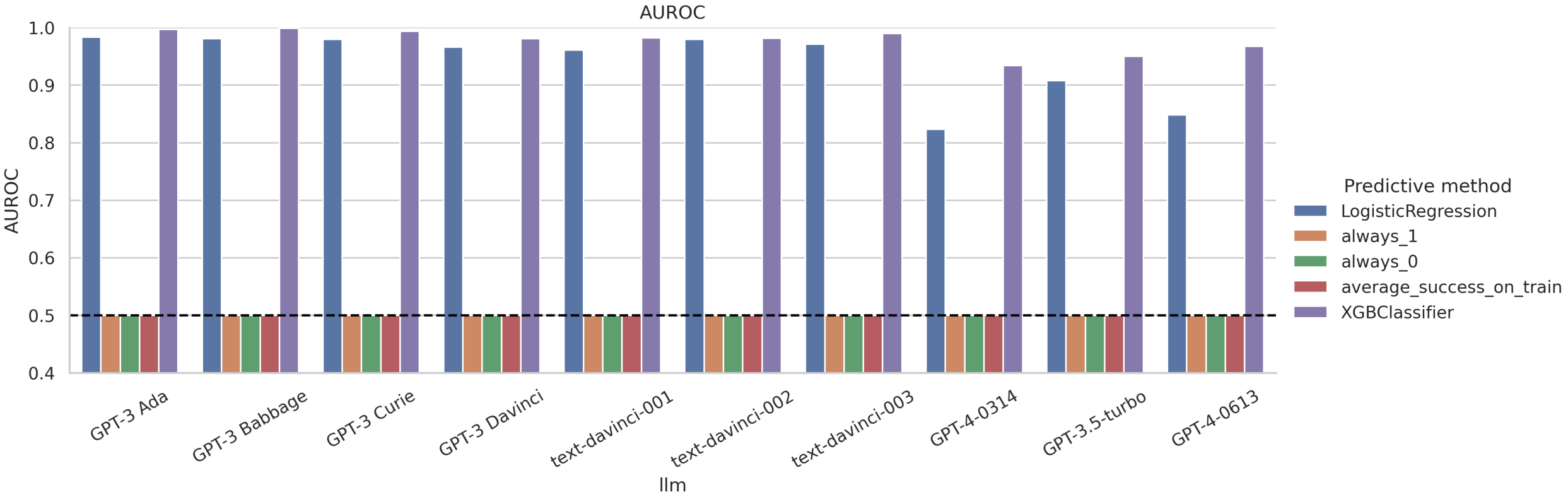
Common predictive models

- Build Logistic regression and XGBoost classifier for each LLM to predict success using the 3 meta-features (harm_mean, carry_art_mean_ratio, sum_sd)
- Add baselines:
  - Predict success with probability 1 for each instance
  - Predict success with probability 0 for each instance
  - Predict success with probability equal to aggregate probability of success on the training set (i.e. the proportion of successes on the training data per LLM)
- Evaluate the performance of the predictive models using AUROC and Brier score

Common predictive models



Common predictive models



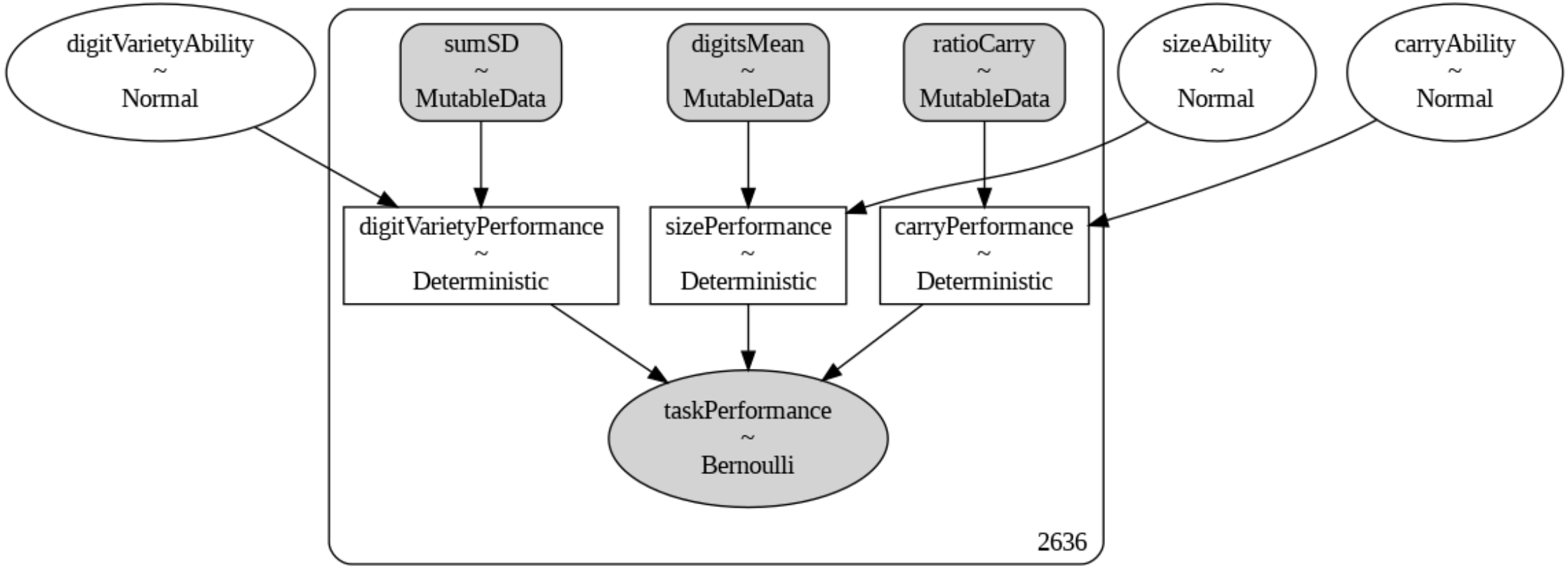
Capabilities

- 3 meta-features/demands: harm_mean, carry_art_mean_ratio, sum_sd
- Capabilities that are informed by/measured/match those demands:
  - harm_carry ↔ size ability
  - carry_art_mean_ratio ↔ carry ability (ability to perform addition that require carry to various extents)
  - sum_sd ↔ digit variety ability
- Simple case of one demand, one ability (in general doesn't have to be as we've seen in the previous session)
- Abilities non-compensatory (to an extent)

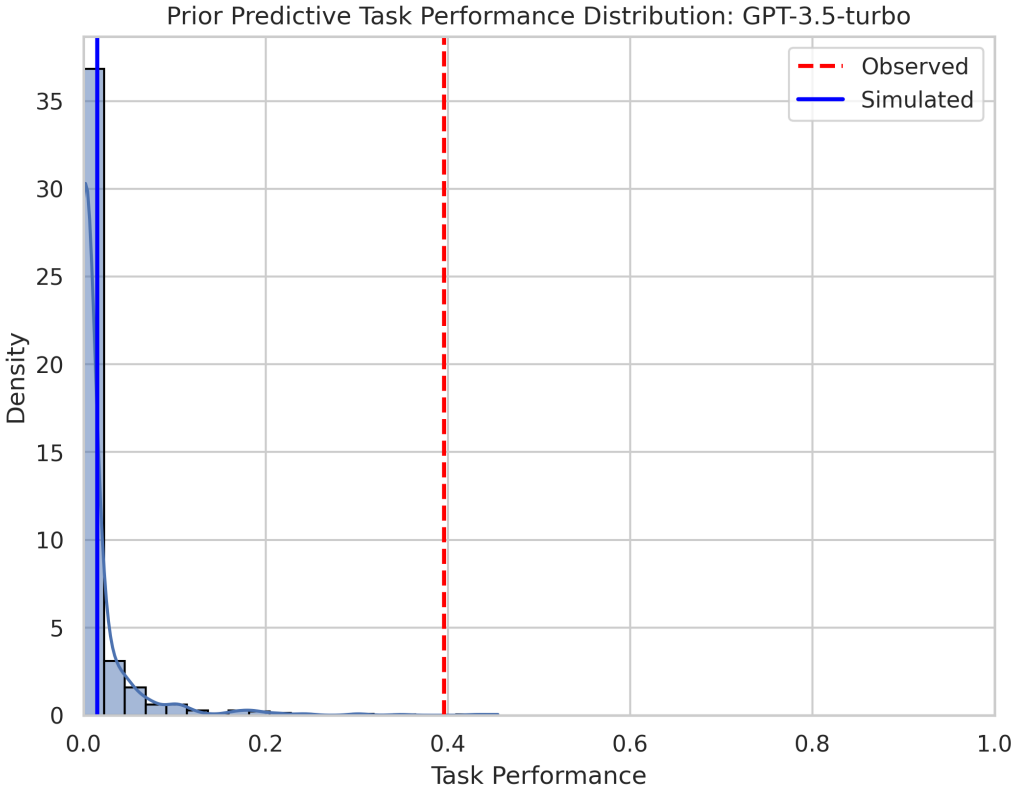
Measurement layouts

```
def setupSumModel(data):

    m = pm.Model()
    with m:

      # Define abilities and their priors
      sizeAbility = pm.Normal("sizeAbility", 0,6)
      carryAbility = pm.Normal("carryAbility", 0,6)
      digitVarietyAbility = pm.Normal("digitVarietyAbility", 0,6)

      # Define environment variables as MutableData
      digitsMean = pm.MutableData("digitsMean", data['harm_mean'].values)
      ratioCarry = pm.MutableData("ratioCarry", data['carry_art_mean_ratio'].values)
      sumSD = pm.MutableData("sumSD", data['sum_sd'].values)

      # Margins / partial performance
      sizePerformance = pm.Deterministic("sizePerformance", logistic_general(margin(sizeAbility,digitsMean),min=0,max=12,c=0,p=0.99))
      carryPerformance = pm.Deterministic("carryPerformance", logistic_general(margin(carryAbility,ratioCarry),min=0,max=12,c=0,p=0.99))
      digitVarietyPerformance = pm.Deterministic("digitVarietyPerformance", logistic_general(margin(digitVarietyAbility,sumSD),min=0,max=12,c=0,p=0.99))

      #Non-compensatory interaction
      finalPerformance =  sizePerformance * carryPerformance * digitVarietyPerformance

      taskPerformance = pm.Bernoulli("taskPerformance", finalPerformance, observed = data['success'])

    return m
```

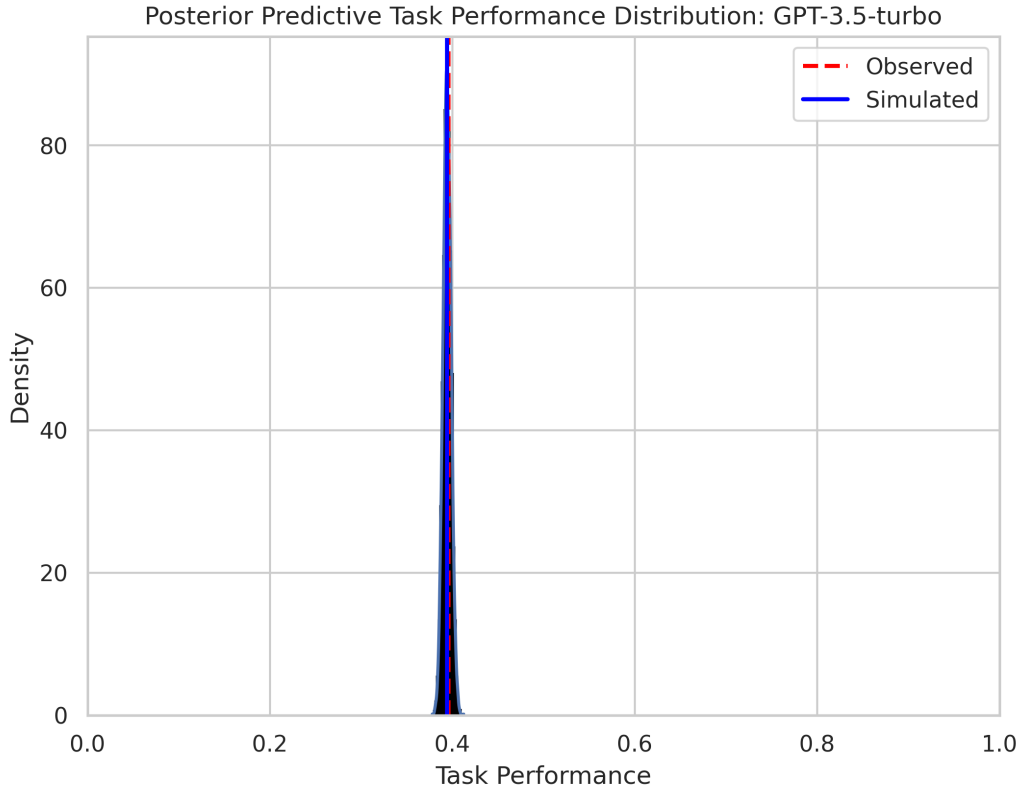
Measurement layouts



Measurement layouts: prior and posterior predictive checks

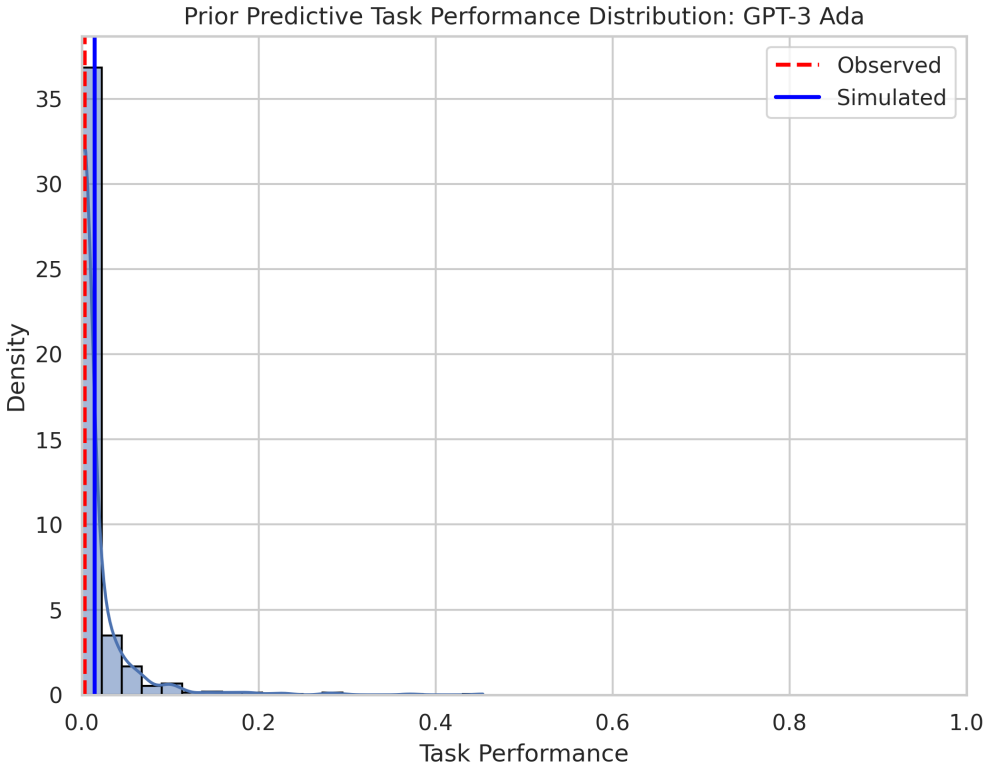


Sample from the posterior

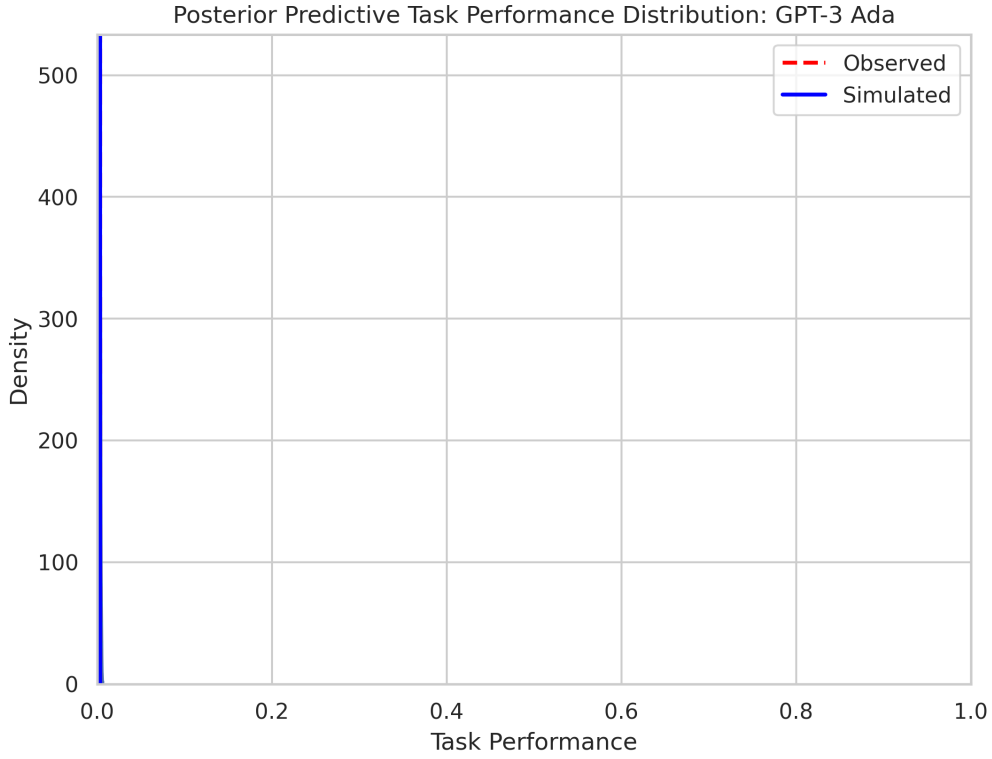


$$\theta^{sim} \sim p(\theta) \qquad y^{sim} \sim p(y|\theta^{sim})$$

Measurement layouts: prior and posterior predictive checks

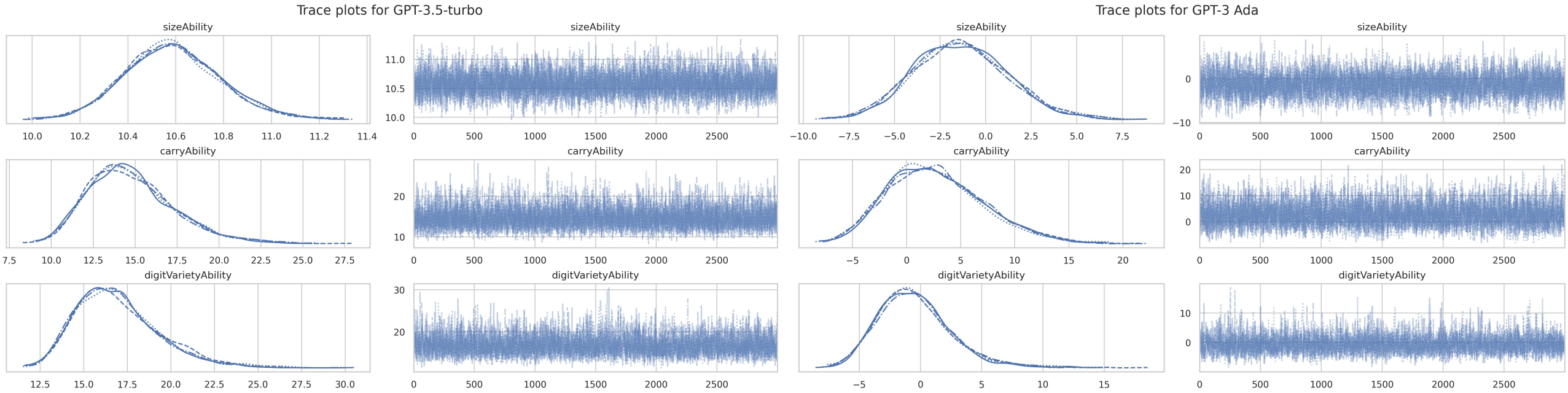


Sample from the posterior

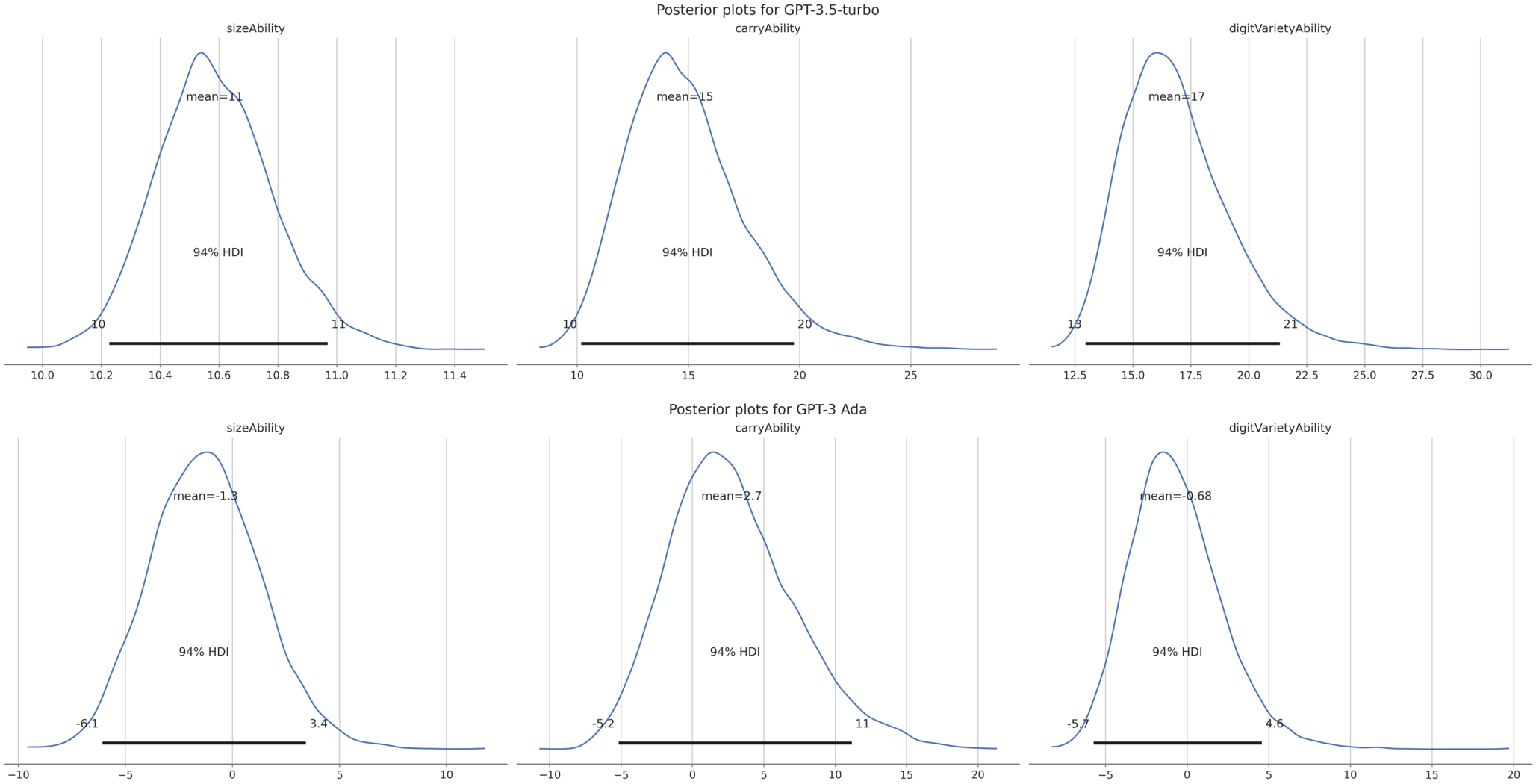


$$\theta^{sim} \sim p(\theta) \qquad y^{sim} \sim p(y|\theta^{sim})$$

Measurement layouts: Markov chains

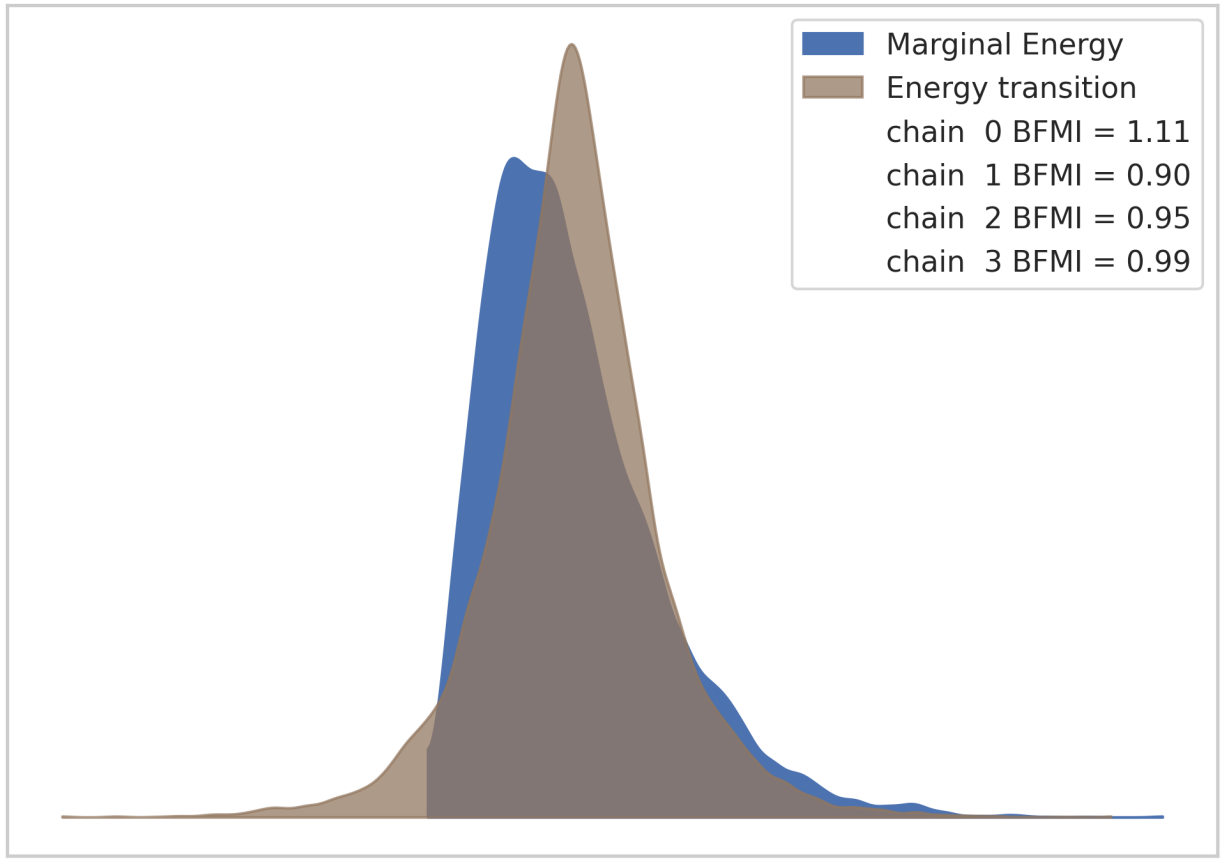


Measurement layouts: posterior plots

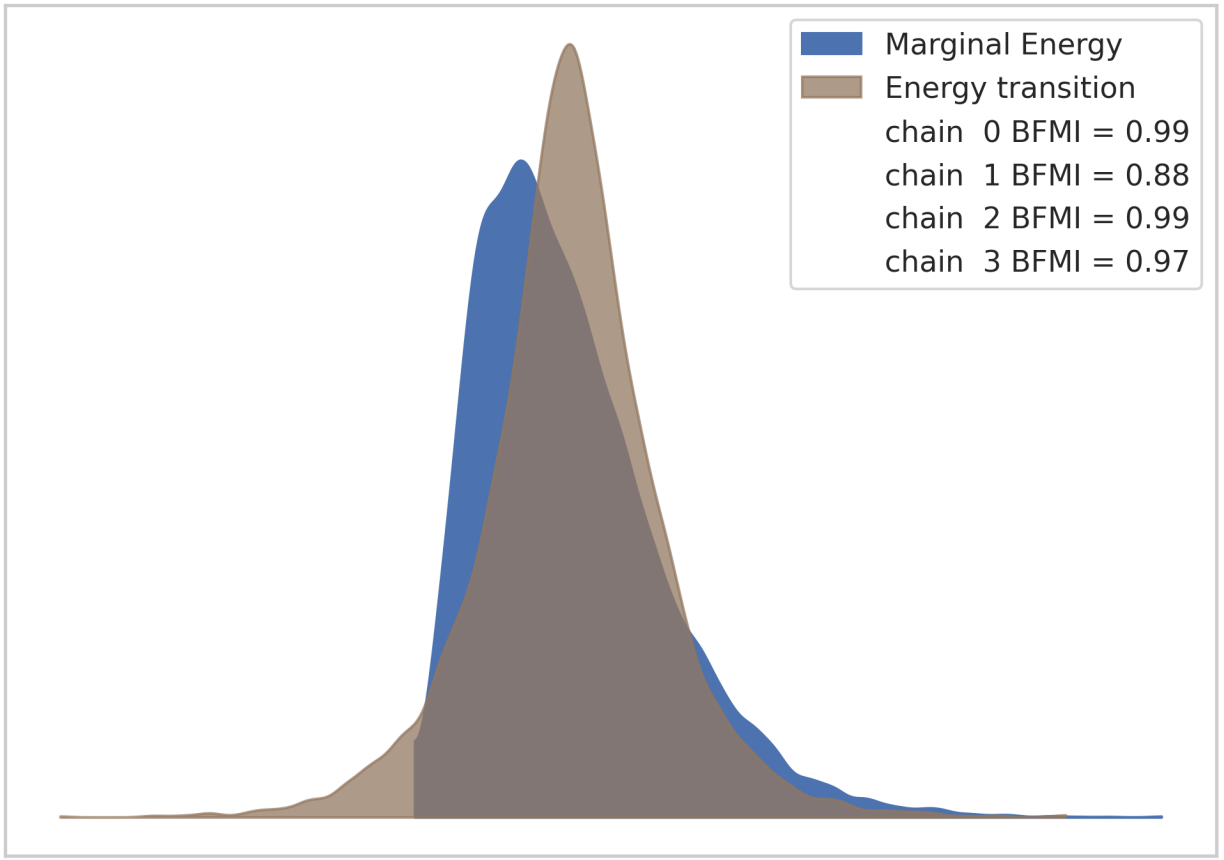


Measurement layouts: convergence metrics

Energy plot for GPT-3.5-turbo



Energy plot for GPT-3 Ada



Measurement layouts: convergence metrics

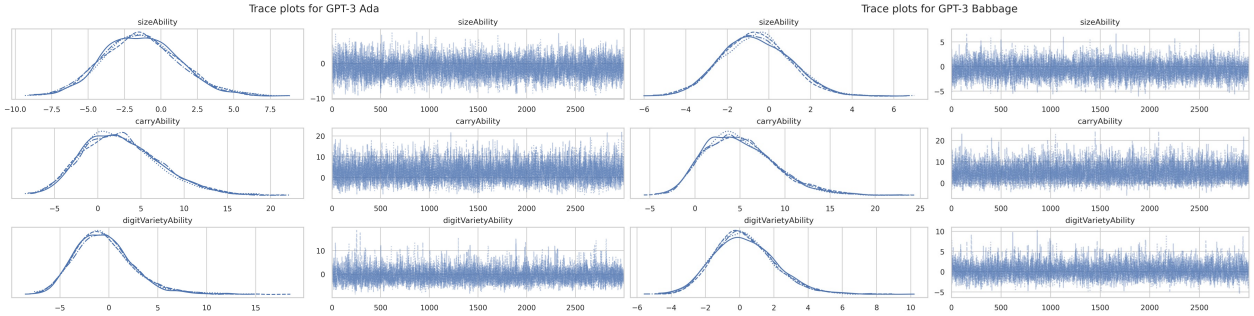
Summary statistics: GPT-3.5-turbo

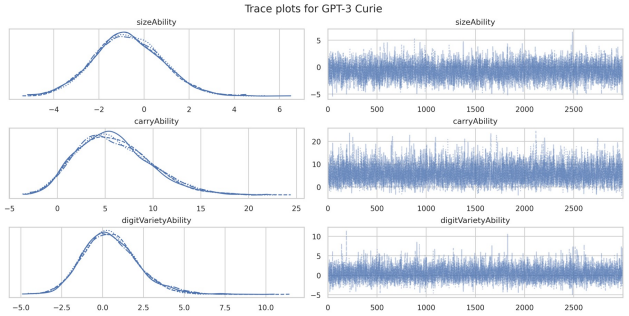
	mean	sd	hdi_3%	hdi_97%	ess_bulk	ess_tail	r_hat
sizeAbility	10.585	0.198	10.227	10.968	5746.943	5346.377	1.001
carryAbility	14.839	2.652	10.175	19.743	6126.873	4941.861	1.001
digitVarietyAbility	16.887	2.336	12.949	21.330	4821.880	4500.382	1.001

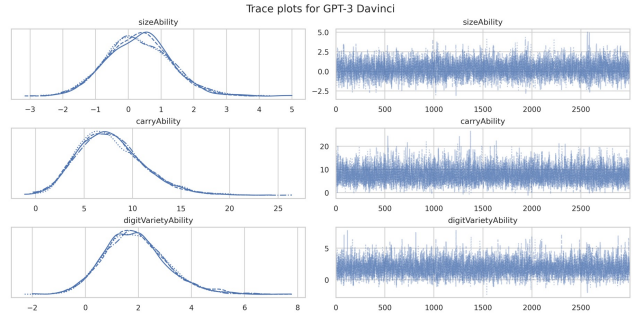
Summary statistics: GPT-3 Ada

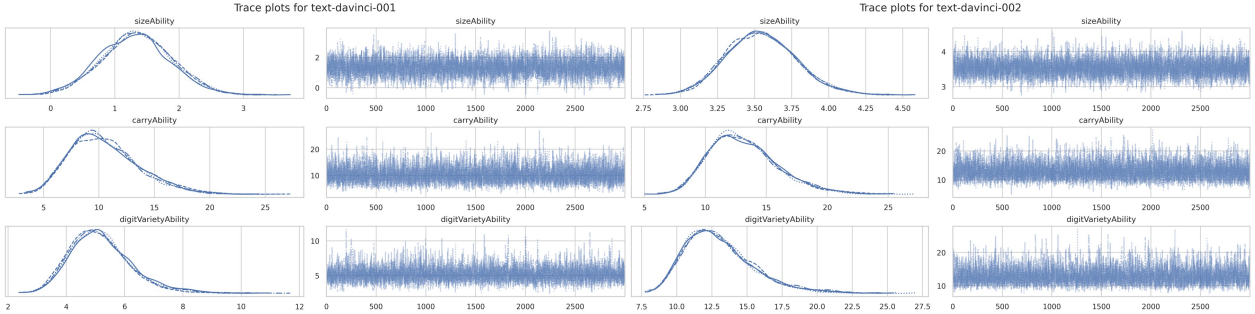
	mean	sd	hdi_3%	hdi_97%	ess_bulk	ess_tail	r_hat
sizeAbility	-1.252	2.569	-6.075	3.425	4289.046	4371.565	1.001
carryAbility	2.739	4.415	-5.183	11.158	4434.910	4728.066	1.001
digitVarietyAbility	-0.685	2.822	-5.737	4.564	4432.195	4482.631	1.000

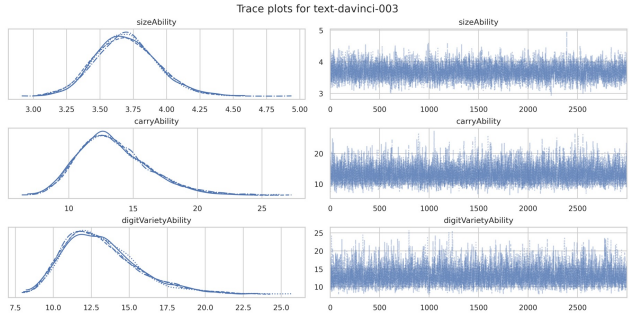
Build measurement layouts for each LLM

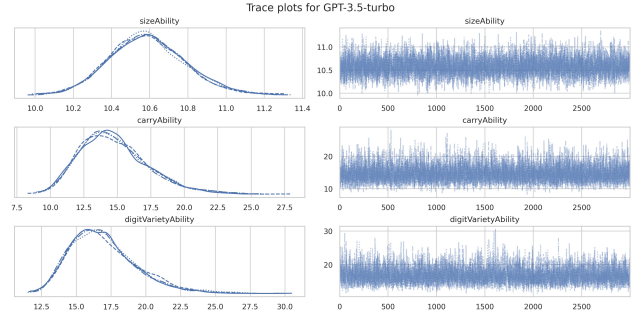


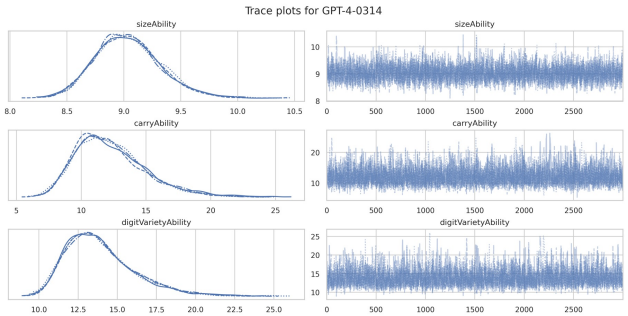


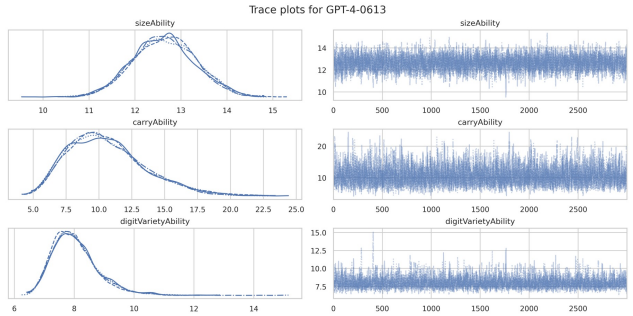






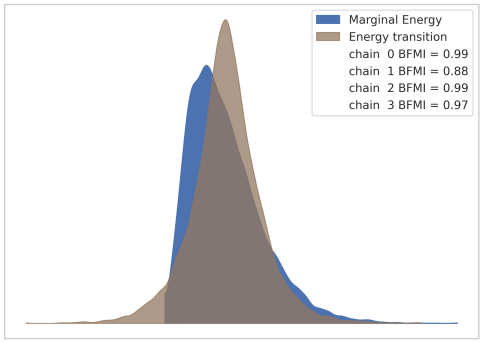




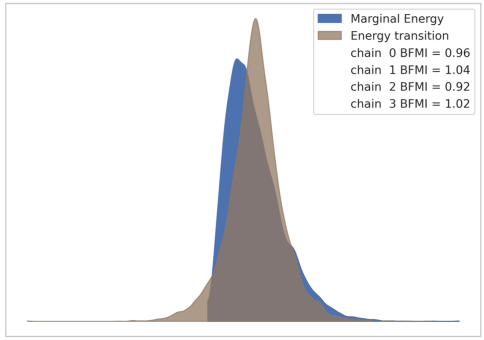


Build measurement layouts for each LLM

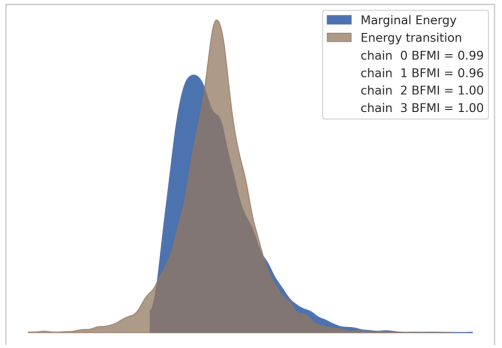
Energy plot for GPT-3 Ada



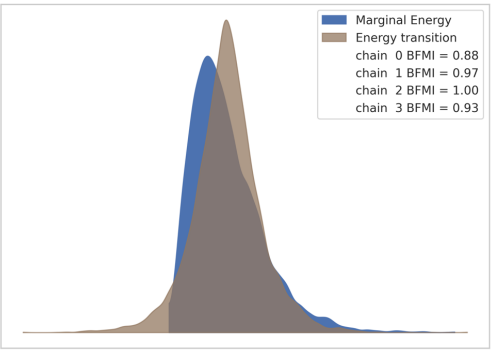
Energy plot for GPT-3 Babbage



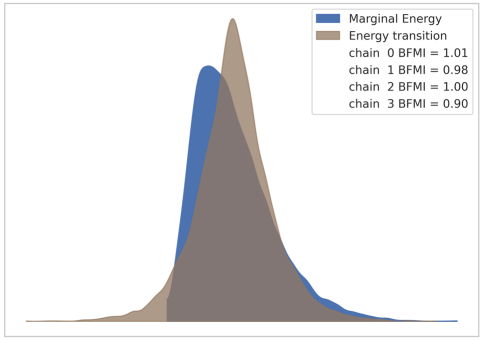
Energy plot for GPT-3 Curie



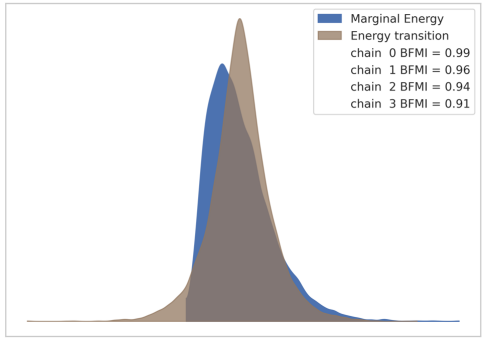
Energy plot for GPT-3 Davinci



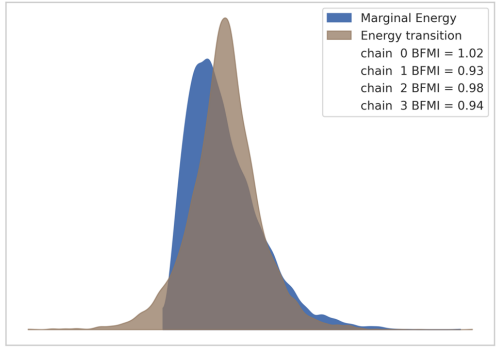
Energy plot for text-davinci-001



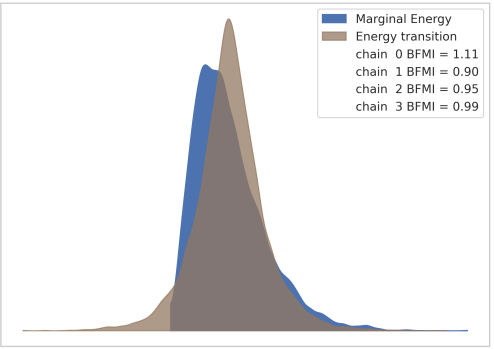
Energy plot for text-davinci-002



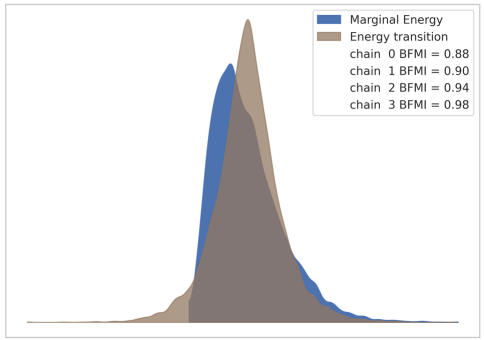
Energy plot for text-davinci-003



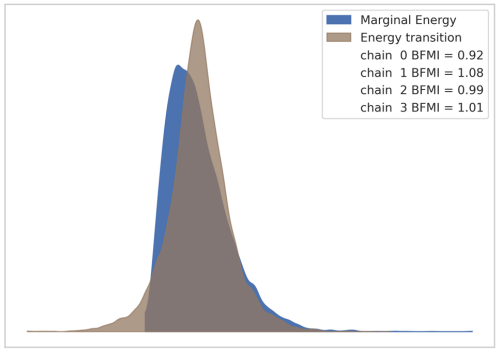
Energy plot for GPT-3.5-turbo



Energy plot for GPT-4-0314

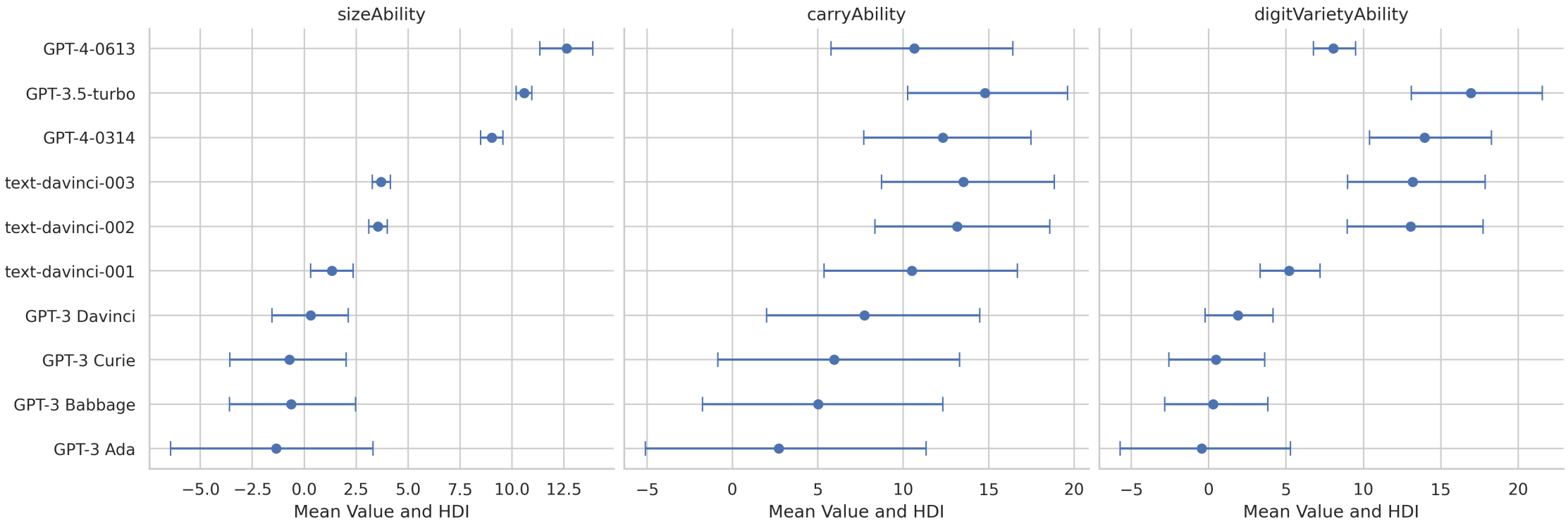


Energy plot for GPT-4-0613

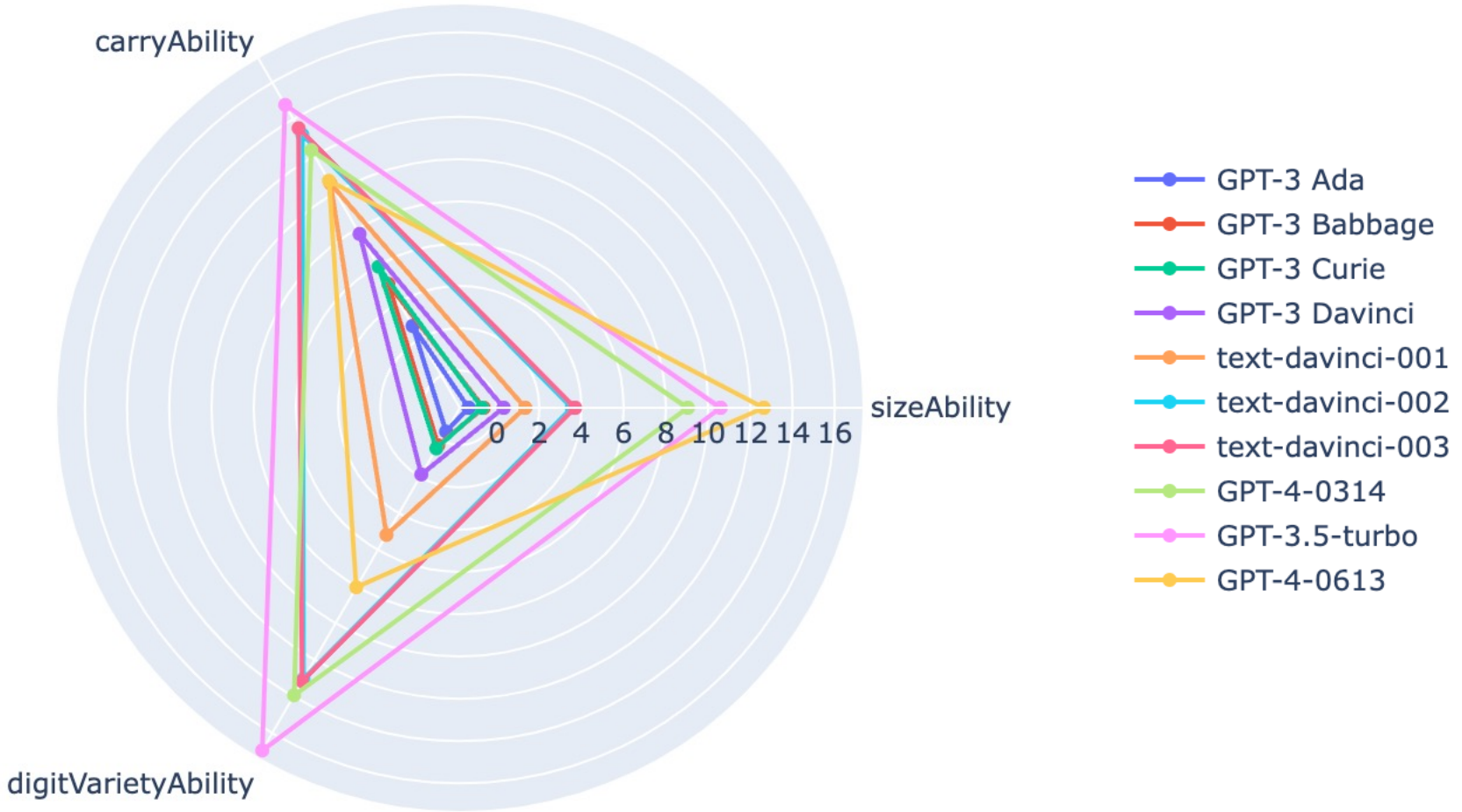


		ess_bulk	ess_tail	r_hat
LLM	Ability			
GPT-3 Ada	sizeAbility	4498.985	4433.078	1.001
	carryAbility	5223.856	5411.293	1.000
	digitVarietyAbility	4185.243	3874.632	1.000
GPT-3 Babbage	sizeAbility	4267.570	4854.430	1.002
	carryAbility	4414.343	4259.494	1.001
	digitVarietyAbility	4199.861	4260.734	1.001
GPT-3 Curie	sizeAbility	4326.282	4850.581	1.000
	carryAbility	4959.339	4330.477	1.001
	digitVarietyAbility	4645.460	4959.820	1.000
GPT-3 Davinci	sizeAbility	4422.249	4616.055	1.001
	carryAbility	5512.185	4857.278	1.000
	digitVarietyAbility	4336.236	4424.901	1.000
text-davinci-001	sizeAbility	3978.109	4799.510	1.002
	carryAbility	4826.458	4416.341	1.001
	digitVarietyAbility	4042.979	3612.911	1.002
text-davinci-002	sizeAbility	5163.584	5466.039	1.000
	carryAbility	5874.716	4886.958	1.002
	digitVarietyAbility	4165.493	3805.924	1.001
text-davinci-003	sizeAbility	4737.856	4706.722	1.001
	carryAbility	5922.350	4829.169	1.002
	digitVarietyAbility	3918.935	4393.516	1.000
GPT-4-0314	sizeAbility	3589.105	4003.518	1.001
	carryAbility	4773.185	3827.450	1.001
	digitVarietyAbility	3006.980	2697.055	1.001
GPT-3.5-turbo	sizeAbility	5088.514	5161.921	1.001
	carryAbility	5734.960	5162.670	1.000
	digitVarietyAbility	4352.323	4721.511	1.001
GPT-4-0613	sizeAbility	3951.369	4506.992	1.001
	carryAbility	3837.764	4128.004	1.001
	digitVarietyAbility	3581.884	3887.807	1.001

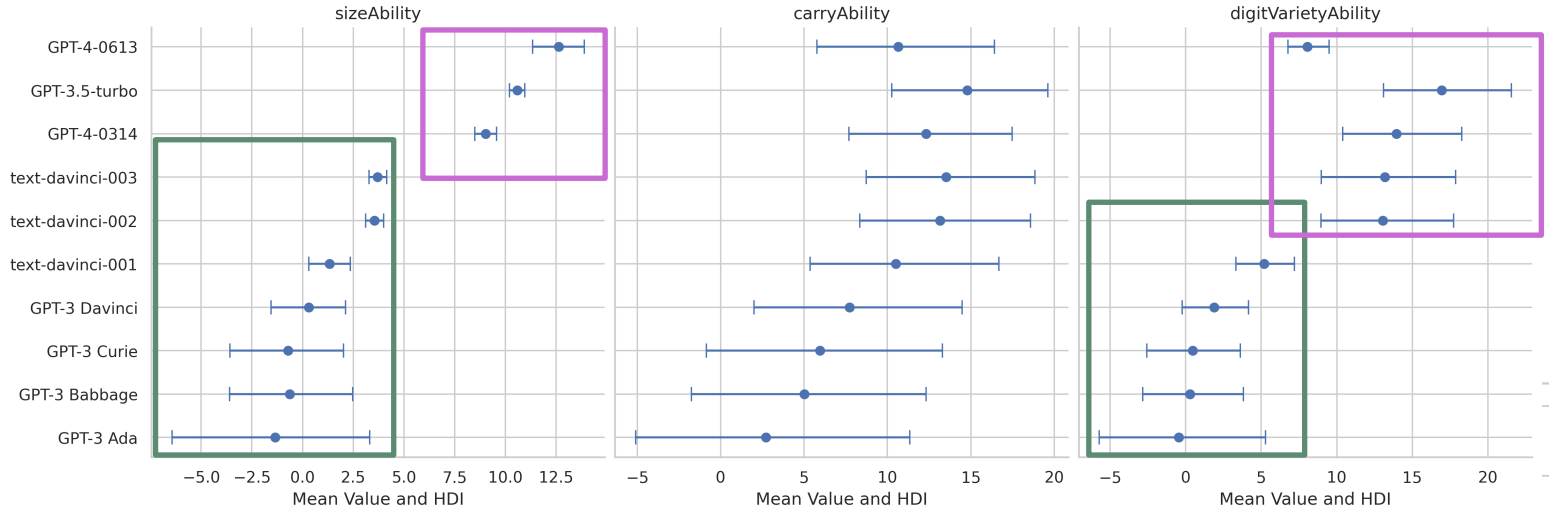
Build measurement layouts for each LLM

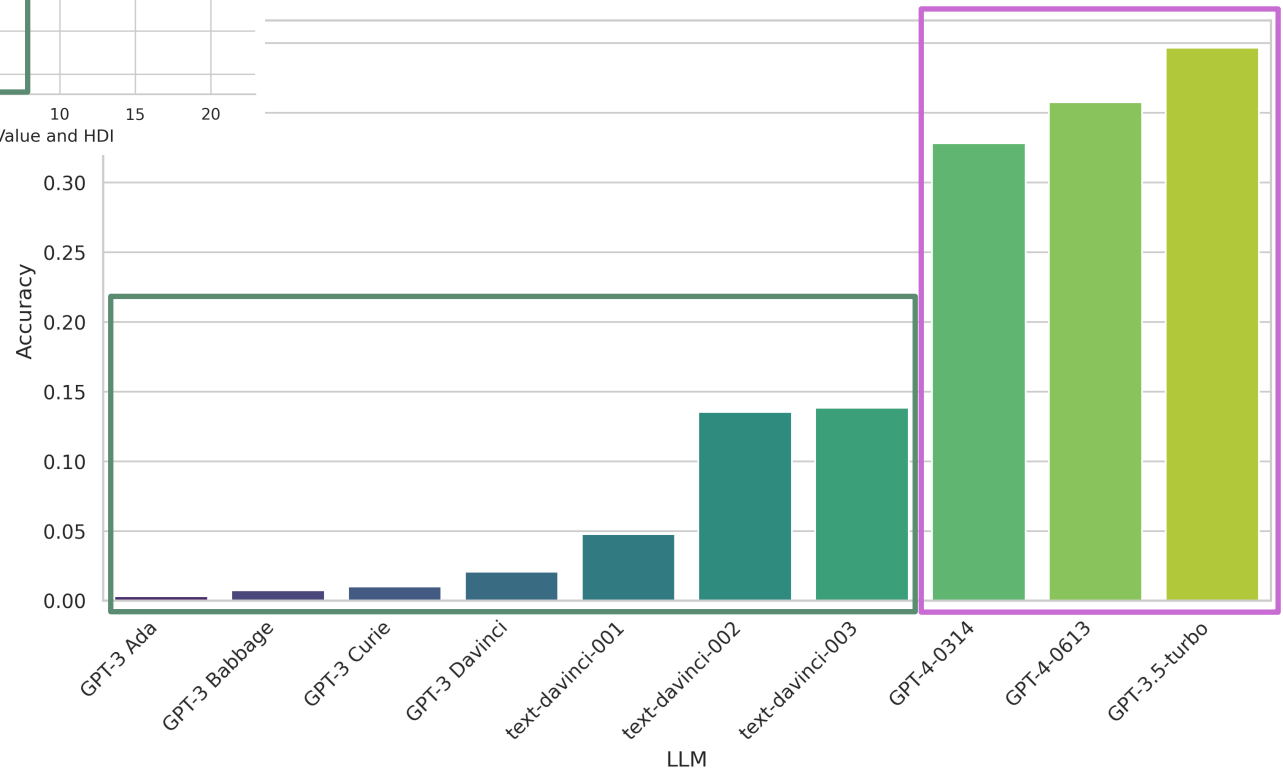


Build measurement layouts for each LLM

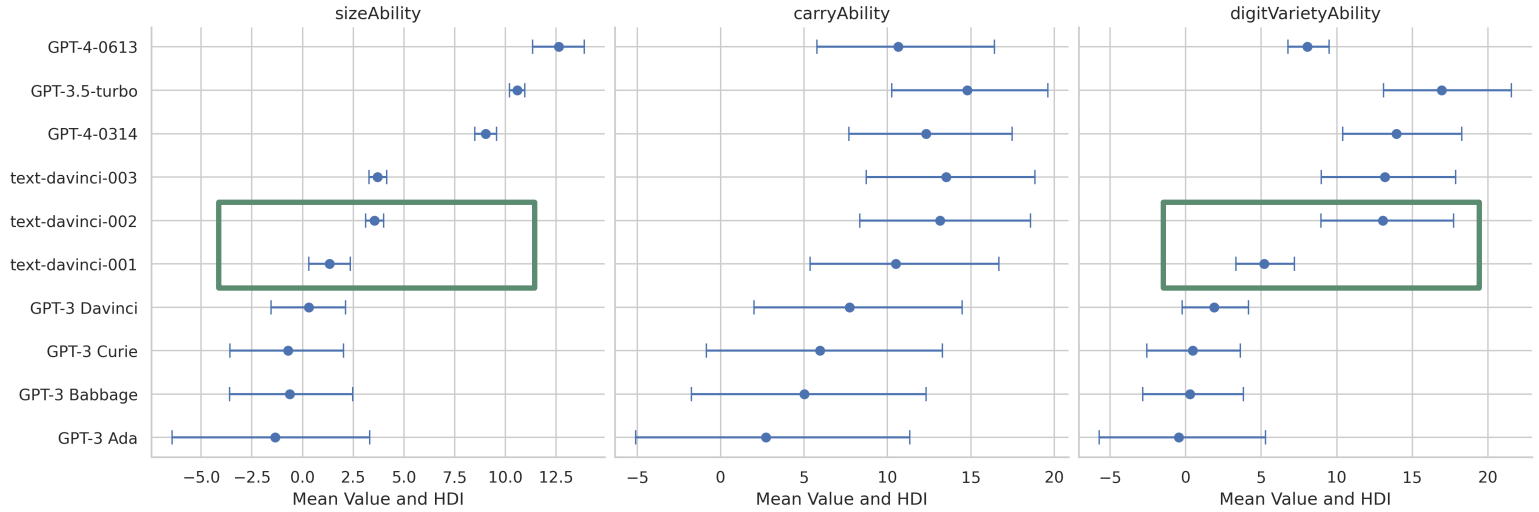


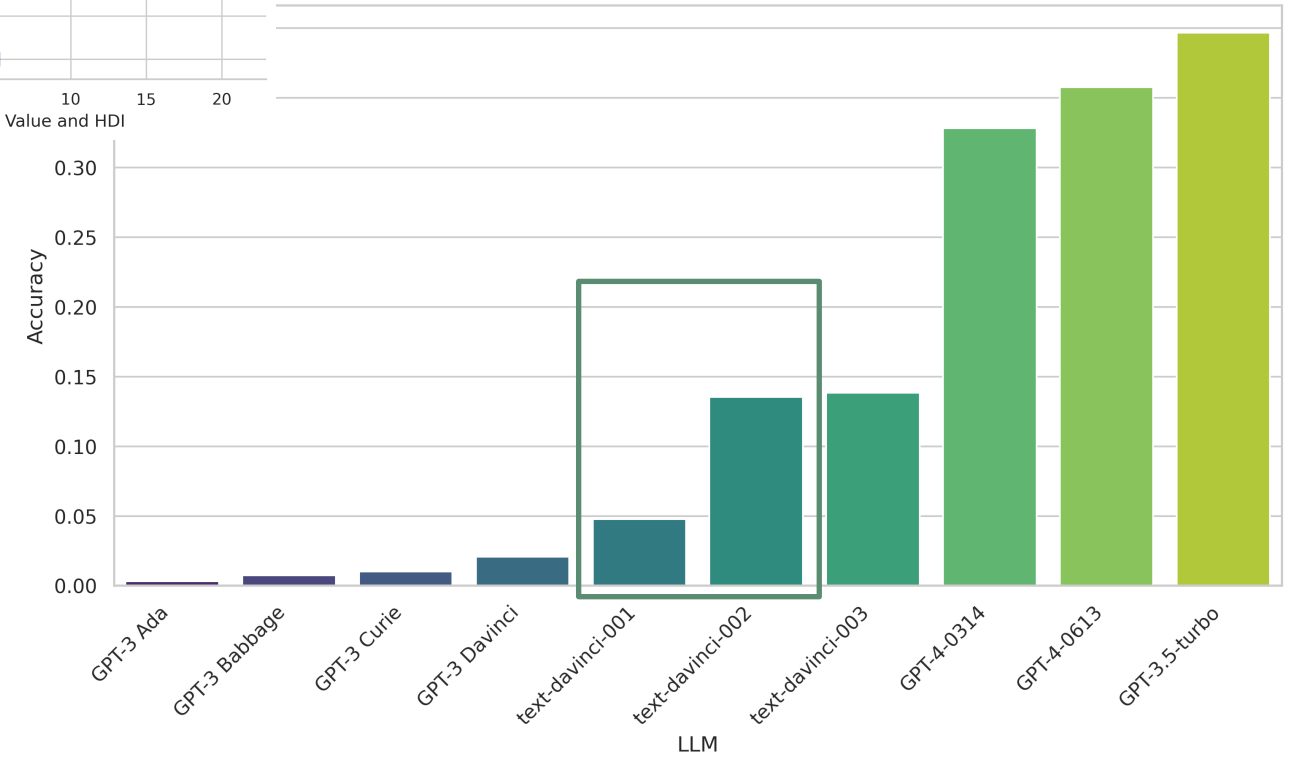
Build measurement layouts for each LLM



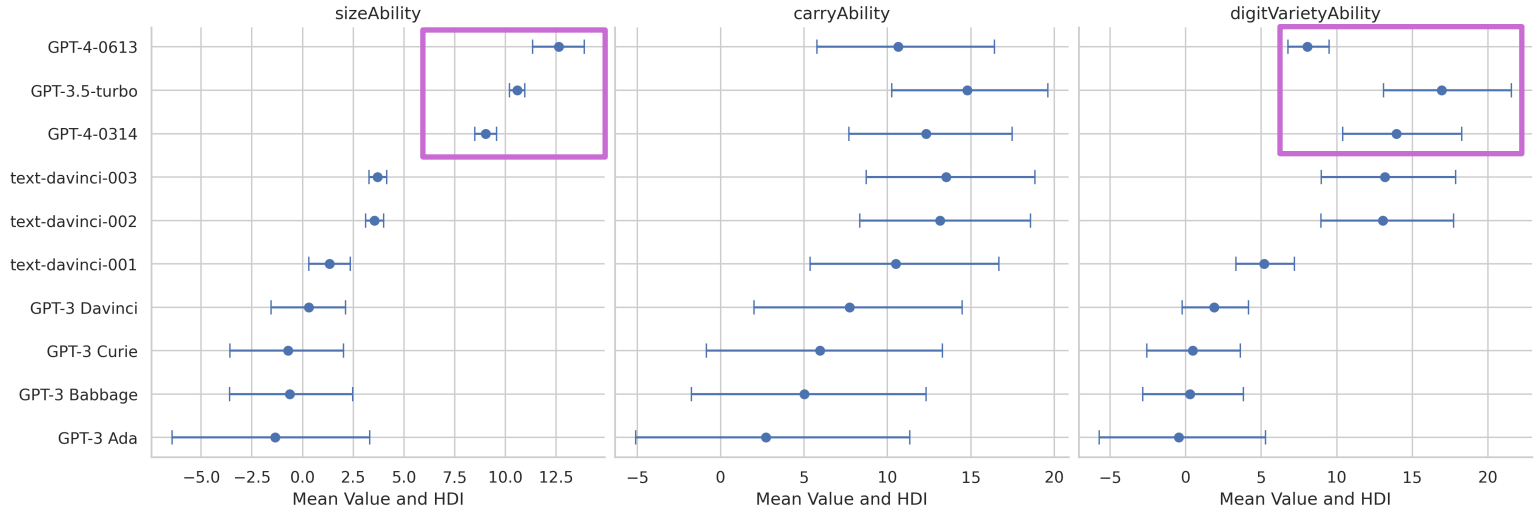


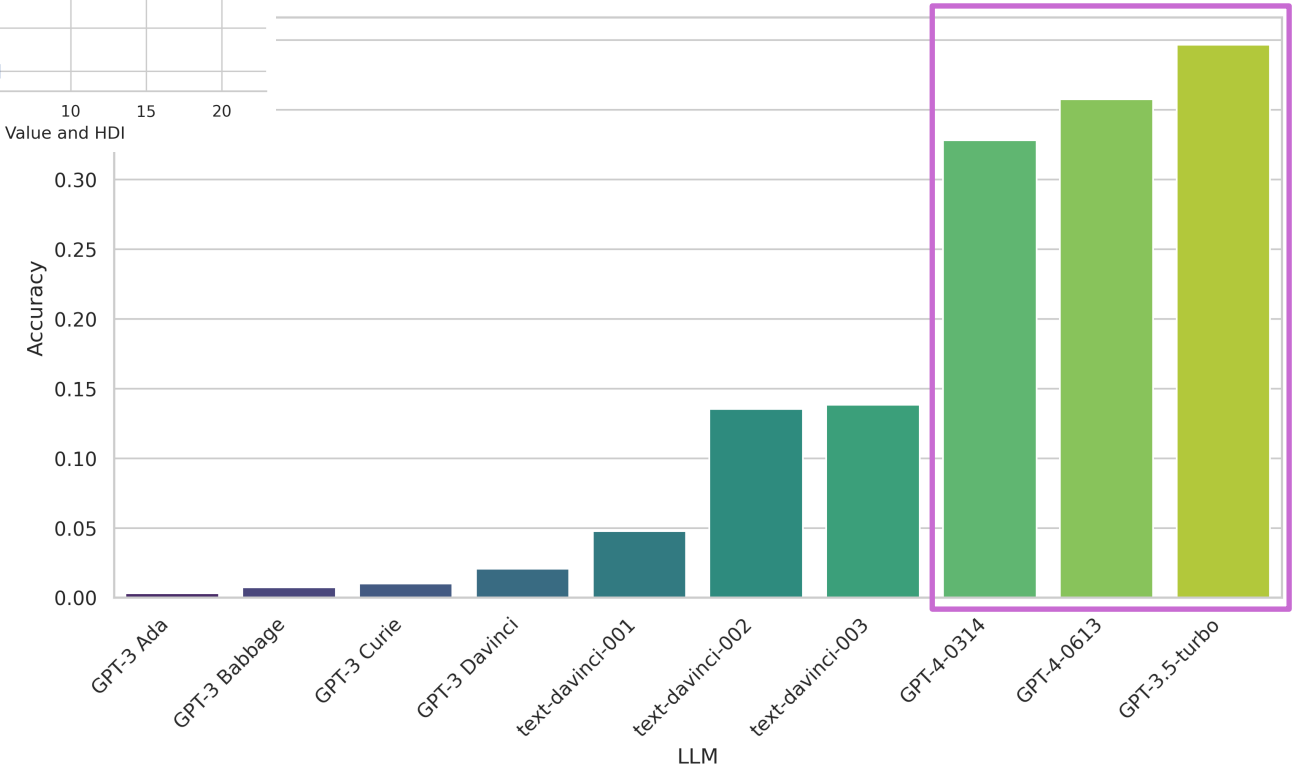
Build measurement layouts for each LLM



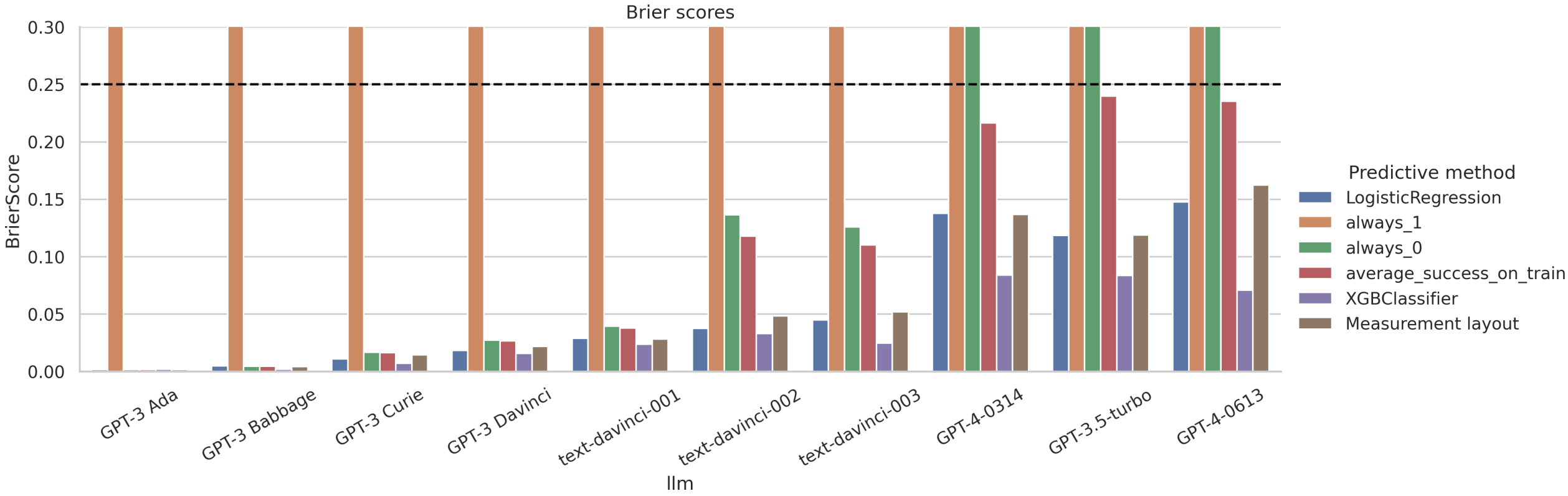


Build measurement layouts for each LLM

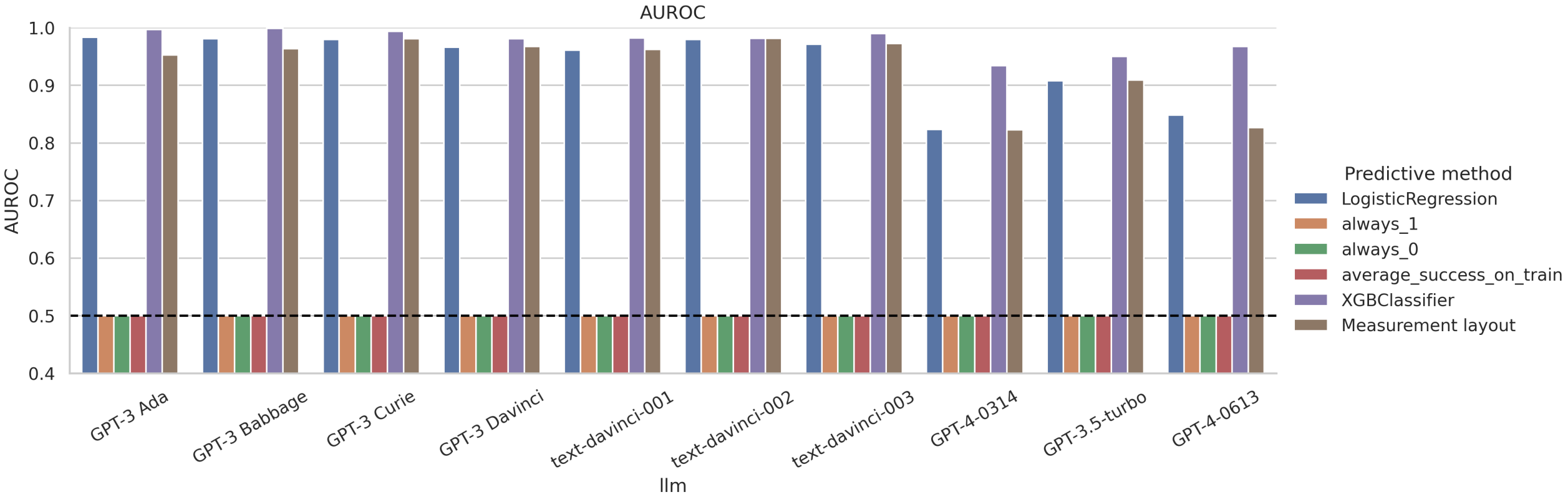




Prediction accuracy



Prediction accuracy



Take-aways

- Capability is not average performance
  - Capability allows to predict performance, at the instance level and even OOD
- System Capabilities and Task Demands are related through a “margin”
- Measurement layouts capture domain knowledge and intuitions
  - Need instance level data
  - Meta-features demands
  - Capabilities that are meant to meet the demands
- Backward inference: estimate capabilities for one single system
- Forward inference: infer performance for new task instances.

CFI LEVERHULME CENTRE FOR THE FUTURE OF INTELLIGENCE

With Thanks to…

Kozzy Voudouris

José Hernández-Orallo

John Burden

Lorenzo Pacchiardi

Lucy Cheke

UNIVERSITY OF CAMBRIDGE

DARPA

E·S·R·C ECONOMIC & SOCIAL RESEARCH COUNCIL

LEVERHULME TRUST

OECD

GoodAI

aws

WBAI Whole Brain Architecture Initiative

unity

CFI LEVERHULME CENTRE FOR THE FUTURE OF INTELLIGENCE