GENERALIZED EXAMPLE DIFFICULTY METRICS AND THEIR APPLICATIONS FOR TABULAR CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantifying example difficulty in ML is crucial for addressing prevalent data quality issues. Our goal is to design the most effective example difficulty metric for each data-centric application. To this end, we investigate questions about the design space itself: Which design axes matter most? Which values are the "best" for each axis, given all else equal? Existing surveys, which are catalogs of published metrics, cannot answer these questions rigorously. Furthermore, existing research under-represents the pervasive domain of tabular data, and none rigorously contrasts the architecture families of deep learning and gradient-boosted decision trees (GBDTs). We remedy these limitations by defining a broad taxonomy that generalizes the design knobs of existing metrics, yielding 96 metrics in a five-dimensional design space, spanning DL and GBDTs. We then study their behavior across 33 tabular classification tasks and their performance across five data-centric applications. We show the following: 1) Most published metrics correlate together despite a vast design space. 2) Model probing method matters more than other axes, such as whether to compute influence or direct train- or test-time measurements. 3) Final-model and snapshot-mean measurements are redundant, and train-time and test-time measurements are nearly redundant. 4) Training and measuring GBDTs is more effective for application tasks on tabular data than the DL equivalent; loss remains strong for many tasks. We conclude by identifying the best metric, from our experiments, for five data-centric application tasks.

1 INTRODUCTION

Data quality is a defining problem of ML applications in practice. The ideal is clear: using high-quality data improves ML performance (Li et al., 2021). Yet data quality issues are omnipresent, and practitioners spend an outsized share of their time on data preparation and cleaning (Côté et al., 2024). This gap has motivated the recent rise in data-centric AI research (Côté et al., 2024). A line of work on example difficulty quantifies how challenging a data point is for a model to learn. There are now dozens of proposed metrics. This direction is promising because the most difficult data points are often mislabeled or outliers, causing model performance degradation (Paul et al., 2021; Carlini et al., 2019; Kwok et al., 2024; Seedat et al., 2024; George et al., 2024). Identifying the most difficult data points is a prerequisite for many remediation strategies. These applications span the full ML pipeline: label cleaning, outlier detection and removal, data pruning, curriculum learning, dataset enrichment, and drift detection, among others (Forouzesh & Thiran, 2024; Carlini et al., 2019; Qin et al., 2024; Soviany et al., 2022; Chang et al., 2024; Baldock et al., 2021).

Given numerous proposed metrics and their applications, the key question is: What kind of example difficulty metric should we use for each data-centric application? We argue that existing surveys leave two gaps that hinder our ability to answer this question effectively for practitioners. First, existing surveys have scant coverage of tabular data and, consequently, gradient-boosted decision trees (GBDTs). Hence, these surveys fail to answer whether conclusions drawn from deep learning on other modalities generalize. Second, existing surveys are collections of published metrics, each making specific choices across several possible axes to constitute a metric. Merely cataloging metrics precludes research questions of our interest: Does the model probing method matter more than the aggregation method? Do train-time measurements fundamentally differ from test-time measurements across many metrics? Should we train GBDTs or tabular DL models to measure example difficulty? These are questions about the general properties of the design space of example difficulty

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

metrics and key recurring design axes. Answering them rigorously requires an unbiased representation of the design knobs. We elaborate on both contributions below.

As tabular data is the most prevalent domain for enterprises, it is the setting where a large proportion of practitioners' data quality woes lie (Borisov et al., 2022; Sambasivan et al., 2021; Polyzotis et al., 2018). Indeed, data quality issues in tabular data are well documented for both real-world tables and popular benchmark tables (Li et al., 2021; Tschalzev et al., 2025). Despite clear practical motivation, most research has focused on vision and NLP (Paul et al., 2021; Baldock et al., 2021; Carlini et al., 2019; Kwok et al., 2024; Seedat et al., 2024). This imbalance challenges engineers who want to utilize example difficulty metrics on tabular data. The data distributions are different, so not all conclusions transfer (McElfresh et al., 2023). Crucially, since both tabular DL and GBDTs are popular, cleanly contrasting the two architectural families is critical. The two families have fundamental differences: adapting activations to GBDTs requires tree kernels; early trees are fixed, whereas DL trains all weights; decision boundaries are less smooth (Davies & Ghahramani, 2014; Chen & Guestrin, 2016; Grinsztajn et al., 2022). It is not known whether these data and model architectural characteristics cause example difficulty to behave differently. To remedy the limitations of George et al. (2024), our design directly compares the same set of methods across the two architectural families across a wider range of applications.

In order to design the most effective example difficulty metric for specific tasks, we would like to understand the properties of the design space: What design axes matter more than others? What values are most effective for each axis? Are there combinations of existing design knobs that prove effective for certain tasks? Existing surveys are effective at cataloging and characterizing published metrics, but are not able to rigorously answer fundamental questions about the design space (Carlini et al., 2019; Kwok et al., 2024; Seedat et al., 2024; George et al., 2024). To that end, we design a taxonomic framework that includes a wide variety of example difficulty metrics, including published metrics and others implied by the design space. We then fix the design axes' domains and exhaustively enumerate all combinations to create a battery of metrics that is balanced across all design knobs. With this taxonomy-first approach, we can design statistical tests and experiments to answer our research questions, and mine metrics much like Alshammari et al. (2025).

This paper's organization and findings are as follows. We define a taxonomic framework for generalized example difficulty metrics, with five design knobs (Section 3.1), and a concrete instantiation that yields 96 metrics (Section 3.2). For the first half of our experiments, we collect data over dozens of tabular classification tasks to answer how the metrics correlate to published proposals, and which design axes are the most important (Section 4.1, Section 5.1). This analysis reveals the limited behavior of published metrics, and the importance of the probing method over all other axes. For the second half, we measure and analyze the metrics' performance over data cleaning, dirtiness detection, and data pruning applications (Section 4.2, Section 5.2). We isolate the effect of each axis on performance, and identify standout metrics.

2 Related Work

Example difficulty. Many example difficulty metrics were devised to solve a particular target application. Simple cross-entropy loss, forgetting events, EL2N, and GraNd were used for data-efficient learning (Qin et al., 2024; Toneva et al., 2019; Paul et al., 2021). Prediction depth and variance of gradient were motivated by the detection of problematic data in vision tasks (Baldock et al., 2021; Agarwal et al., 2022). Consistency score, area under the margin, learning risk, and static centroid distance were proposed for noisy label detection (Jiang et al., 2021; Pleiss et al., 2020; Deng et al., 2024; Forouzesh & Thiran, 2024). Data-IQ was proposed for data acquisition (Seedat et al., 2022). Two prior works study example difficulty for tabular learning. George et al. (2024) is a collection of published metrics with no controlled cross-architecture comparison and no controlled assessments of the design axes. Seedat et al. (2024) includes only two tabular tasks.

Influence. Influence metrics quantify the impact of removing a training point on the loss of other data points. Although the line of work on "canonical" example difficulty and influence don't completely overlap, we include influence in our study as they also score data points, and are used for the same applications. The original definition, leave-one-out influence over the entire held-out data, remains computationally expensive (Koh & Liang, 2017; Guo et al., 2021; Pruthi et al., 2020). As such, we include two cheaper alternatives: self-influence, and simplified local influence. Self-influence,

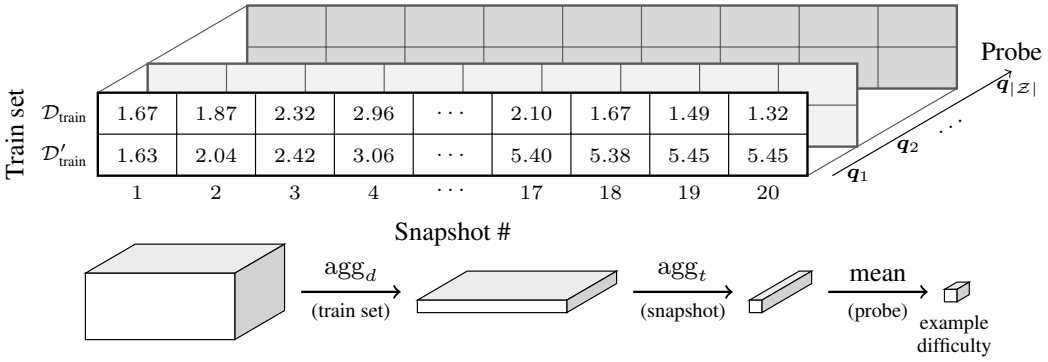

Figure 1: **Visualization of the raw measurements and their aggregations into an example difficulty metric. (Top)** The probe functions are measured across two train sets $\boldsymbol{r} \in \mathcal{D}_{\text{train}}$, $\boldsymbol{r} \notin \mathcal{D}'_{\text{train}}$, $T = 20$ snapshots, and $|\mathcal{Z}|$ probed data points, forming a cuboid. **(Bottom)** This is reduced to a scalar via agg_d, agg_t and mean-aggregation over $\mathcal{Z}$. We also implicitly take expectations over initializations and training data splits, which is 300 four-fold splits in our experiments.

used for outlier detection, quantifies the impact of excluding a data point in training for the performance on that data point itself (Pruthi et al., 2020). Local influence is the data point's influence over its k-nearest held-out data in activation space (Guo et al., 2021). When combined with Feldman & Zhang (2020)'s formulation that samples subsets of fixed sizes, simplified local influence becomes rather sample-efficient. We omit approximations of influence from training trajectories, as we are interested in measuring the target of their approximations (Pruthi et al., 2020; Bae et al., 2024; Wang et al., 2025).

Instance hardness. Instance hardness (IH) computes the difficulty of data points or entire tasks from the data alone without training a machine learning model (Lorena et al., 2019; Torquette et al., 2022; 2024). While IH is distinct from example difficulty, they are included as baselines.

3 Taxonomic Framework

The design space of example difficulty metrics is itself an object of study in our paper. We begin by considering the major design knobs in example difficulty metrics. Measuring example difficulty involves training, taking model snapshots, probing them on a subset of data, and aggregating the measurements. We identify five axes for design choices: which model architecture to probe, which data points to probe, how to probe the model, and how to aggregate across train/test splits and training snapshots within a training run. Changing these design knobs lets us recover many existing metrics or generate new, implied metrics.

3.1 Framework

Assume a standard ML setting. Our goal is to quantify how challenging some *reference point* $\boldsymbol{r}$ is to some model $\boldsymbol{\theta}$. To handle influence metrics, let $\mathcal{D}_{\text{train}}, \mathcal{D}'_{\text{train}} \subset \mathcal{D}$ be two training sets such that $\boldsymbol{r} \in \mathcal{D}_{\text{train}}$ and $\boldsymbol{r} \notin \mathcal{D}'_{\text{train}}$. The complement are the test sets $\mathcal{D}_{\text{test}}, \mathcal{D}'_{\text{test}}$. Training some initialization $\boldsymbol{\theta}_0$ for T iterations—epochs or boosting rounds—yields model snapshots $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_T$ and $\boldsymbol{\theta}'_1, \ldots, \boldsymbol{\theta}'_T$, respectively. The general form of an example difficulty metric is:

$$f_{(\phi, \mathcal{Z}, \text{agg}_d, \text{agg}_t)}(\boldsymbol{r}, \boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\boldsymbol{\theta}_0, \boldsymbol{q} \in \mathcal{Z}} \left[\text{agg}_t \Big(\text{agg}_d\big(\phi(\boldsymbol{\theta}_1, \boldsymbol{q}), \phi(\boldsymbol{\theta}'_1, \boldsymbol{q})\big), \ldots, \text{agg}_d\big(\phi(\boldsymbol{\theta}_T, \boldsymbol{q}), \phi(\boldsymbol{\theta}'_T, \boldsymbol{q})\big) \Big) \right]. \quad (1)$$

Here $\mathcal{Z}$ is the set of data points to probe, $\phi : \Theta \times \mathbb{R}^d \to \mathbb{R}$ is the probing method, $\text{agg}_d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is the method of aggregating across two different training runs (using $\mathcal{D}_{\text{train}}$ and $\mathcal{D}'_{\text{train}}$), and $\text{agg}_t : \mathbb{R}^T \to \mathbb{R}$ is the method of aggregating across model snapshots at different iterations.

Equation 1 and its illustration in Figure 1 can be interpreted as follows. The raw probe measurements over two train sets, T model snapshots each, and probe points in $\mathcal{Z}$ form a cuboid of data, which

we then aggregate to a scalar by successively applying three aggregation functions along each axis. Although a cuboid can be reduced to a scalar in multiple orders, we fix one natural ordering.

What model architecture to probe. There are two reasonable interpretations of what example difficulty means: difficulty of an example for a particular architecture, or some architecture-agnostic equivalent. We assume that the model $\boldsymbol{\theta}$ belongs to a specified architecture, and that the architecture is one of the design axes. This is because some architectures may be more useful for certain data-centric applications than others due to different inductive biases.

Which data points to probe. All example difficulty metrics probe model behavior on some data points, which might be a single data point or a set. We denote this set of probed data points as $\mathcal{Z}$. A simple baseline is to probe $\boldsymbol{r}$ itself, i.e. $\mathcal{Z} = \boldsymbol{r}$. This corresponds to train-time or test-time behavior. This also includes self-influence, which measures the difference in model behavior when including or excluding $\boldsymbol{r}$ in training by probing $\boldsymbol{r}$. In our study, there is one nontrivial $\mathcal{Z}$: local test influence, corresponding to $\mathcal{Z} = \mathcal{N}_k^{\boldsymbol{r}}(\mathcal{D}_{\text{test}} \cap \mathcal{D}'_{\text{test}})$. Here, $\mathcal{N}_k^{\boldsymbol{r}}(S)$ for a set of data points S denotes $\boldsymbol{r}$'s k closest neighbors in the activation space.

How to probe the model. The probing method $\phi : \Theta \times \mathbb{R}^d \to \mathbb{R}$ maps a model snapshot and $\boldsymbol{q} \in \mathcal{Z}$ to a measurement, typically by reading logits, activations, or gradients. Examples include loss, accuracy, ground-truth label confidence, max confidence, prediction gap, entropy of logits, prototypicality, gradient norm, feature predictability, and more, many cataloged in Table 1.

How to aggregate two training runs using $\mathcal{D}_{\text{train}}$ and $\mathcal{D}'_{\text{train}}$. The first aggregation function $\text{agg}_d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ takes two probe method measurements $\phi(\boldsymbol{\theta}_t, \boldsymbol{q}), \phi(\boldsymbol{\theta}'_t, \boldsymbol{q})$ and returns a scalar. The simplest agg_d is a simple projection of one of the two values for capturing train-time or test-time behavior: $\text{agg}_d(\cdot,\cdot) = \phi(\boldsymbol{\theta}_t, \boldsymbol{q})$ or $\text{agg}_d(\cdot,\cdot) = \phi(\boldsymbol{\theta}'_t, \boldsymbol{q})$. Influence metrics compute the two inputs' difference: $\text{agg}_d(\cdot,\cdot) = \phi(\boldsymbol{\theta}_t, \boldsymbol{q}) - \phi(\boldsymbol{\theta}'_t, \boldsymbol{q})$. Rarer agg_d functions also exist, such as a ratio: $\text{agg}_d(\cdot,\cdot) = \phi(\boldsymbol{\theta}_t, \boldsymbol{q})/\phi(\boldsymbol{\theta}'_t, \boldsymbol{q})$ (Wang, 2019).

How to aggregate T training snapshots. The $\text{agg}_t : \mathbb{R}^T \to \mathbb{R}$ aggregation function aggregates rowwise across model snapshots in the cuboid. Three standard methods include: mean behavior $\text{agg}_t(x_1, \ldots, x_T) = 1/T \sum_{t=1}^{T} x_t$, final-model behavior $\text{agg}_t(x_1, \ldots, x_T) = x_T$, early-epoch model behavior, e.g. $\text{agg}_t(x_1, \ldots, x_T) = x_{\text{early}}$. Other published agg_t functions include variance aggregation, forgetting events $\text{agg}_t(x_1, \ldots, x_T) = |i : i \in [T-1] \wedge x_i = 1 \wedge x_{i+1} = 0|$, and learned epoch $\text{agg}_t(x_1, \ldots, x_T) = \min t : x_{t:T} = 1$.

Numerous published example difficulty metrics map into our framework (Table 1). This immediately highlights underexplored design combinations. While most fit cleanly into the framework, gradient variance requires a non-atomic ϕ that returns vectorized input gradients, and gradient agreement requires max aggregation over the probe set. Both are natural generalizations that we omit to avoid overcomplicating Equation 1 (Agarwal et al., 2022; Dhaliwal & Shintre, 2018).

3.2 Concrete Instantiation

In our experiments, instead of utilizing published metrics, we choose a set of legal domain values for the five design knobs—architecture, ϕ, $\mathcal{Z}$, agg_d, and agg_t (Table 2). We then exhaustively enumerate all legal combinations, as aforementioned, to remove selection bias within the scope of our chosen domains. Given this combinatorial enumeration, we necessarily restrict the domain of the design axes. We also want to avoid candidates that are already known to be nearly identical, to avoid spurious correlations (Kline, 2014).

We choose the following domains for each axis. We abstract the large number of tabular learning architectures into just two families—tabular DL and GBDT—since they have big inter-family differences and smaller intra-family differences in inductive bias. The former averages over MLP, ResNet, and FT-Transformer, and the latter over LightGBM, CatBoost, and XGBoost (Gorishniy et al., 2021; Ke et al., 2017; Prokhorenkova et al., 2018; Chen & Guestrin, 2016). We consider four probe methods: cross-entropy loss, confidence, gradient disagreement, and feature predictability. We use max-logit confidence to decouple from loss, omitting softmax. Gradient disagreement is the Euclidean distance between $\boldsymbol{q}$'s last-layer gradients and the class-wise centroid gradient (Dhaliwal & Shintre, 2018). Feature predictability is the distance of feature encoding from a ridge regression model fit from all other data points, a model-dependent adaptation of Zhou et al. (2024). For GBDTs, we use leaf kernel-based pseudo-embeddings and leaf gradients (Davies & Ghahramani,

Table 1: **Mapping of published example difficulty metrics to our framework.**

Metric	Probe method $\phi(\boldsymbol{\theta}, \boldsymbol{q})$	Probe set	Dataset agg. agg_d	Snapshot agg. $\mathrm{agg}_t(x_1, \ldots, x_T)$
Train/test loss (Jiang et al., 2021)	$\mathcal{L}_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{q})$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$ or $\phi(\boldsymbol{\theta}', \boldsymbol{q})$	$\frac{1}{T}\sum_{t=1}^{T} x_t$
AUM (Pleiss et al., 2020)	$z_y - \max_{k \neq y} z_k$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	$\frac{1}{T}\sum_{t=1}^{T} x_t$
Forgetting events (Toneva et al., 2019)	$\mathbb{I}[\hat{y} = y]$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	$\sum_{t=1}^{T-1} \mathbb{I}[x_t > x_{t+1}]$
Learned epoch (Baldock et al., 2021)	$\mathbb{I}[\hat{y} = y]$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	$\min\{t : x_{t:T} = \mathbf{1}\}$
GraNd EP20 (Paul et al., 2021)	$\lVert \boldsymbol{g}(\boldsymbol{q}) \rVert_2$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_{early}
EL2N EP20 (Paul et al., 2021)	$\lVert \boldsymbol{\pi}(\boldsymbol{q}) - \boldsymbol{e}_y \rVert_2$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_{early}
VoG$^{\approx}$ (Agarwal et al., 2022)	$\nabla_{\boldsymbol{q}}\, z_y \in \mathbb{R}^d$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	$\frac{1}{d}\sum_{i=1}^{d} \mathrm{Var}_t(x_{t,i})$
Prototypicality (Sorscher et al., 2022)	$\lVert \boldsymbol{h}(\boldsymbol{q}) - \boldsymbol{\mu}_y \rVert_2$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_T
Model confidence (Carlini et al., 2019; Jiang et al., 2021)	$\max_k [\boldsymbol{\pi}(\boldsymbol{q})]_k$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	$\frac{1}{T}\sum_{t=1}^{T} x_t$
Prediction depth (Baldock et al., 2021)	$\min\{l : \mathrm{kNN}_{l'}(\boldsymbol{q}) = \hat{y} \;\; \forall\, l' \geq l\}$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$ or $\phi(\boldsymbol{\theta}', \boldsymbol{q})$	x_T
Self-influence (Koh & Liang, 2017; Feldman & Zhang, 2020)	$\mathcal{L}_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{q})$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q}) - \phi(\boldsymbol{\theta}', \boldsymbol{q})$	x_T
Local intrinsic dim. (Levina & Bickel, 2004; Ma et al., 2018)	$\left[\frac{1}{k-1}\sum_{j=1}^{k-1} \log \frac{r_k(\boldsymbol{q})}{r_j(\boldsymbol{q})}\right]^{-1}$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_T
Learning risk (Deng et al., 2024)	$\big(\boldsymbol{h}(\boldsymbol{r})^\top \boldsymbol{h}(\boldsymbol{q}) + 1\big) \cdot \big(\boldsymbol{\pi}(\boldsymbol{q}) - \boldsymbol{e}_{y_{\boldsymbol{q}}}\big)^\top \big(\boldsymbol{e}_{y_{\boldsymbol{r}}} - \boldsymbol{\pi}(\boldsymbol{r})\big)$	$C \subseteq \mathcal{D}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_T
Static centroid distance (Forouzesh & Thiran, 2024)	$\lVert \boldsymbol{h}(\boldsymbol{q}) - \boldsymbol{\mu}_y \rVert_2$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_{t^*}
Gradient agreement (Dhaliwal & Shintre, 2018)	$\cos\big(\boldsymbol{g}(\boldsymbol{r}), \boldsymbol{g}(\boldsymbol{q})\big)$	$\mathcal{D}_{\text{train}} \setminus \{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	x_T
Local influence (Guo et al., 2021)	$\mathcal{L}_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{q})$	$\mathcal{N}_k^{\boldsymbol{r}}(\mathcal{D}_{\text{test}})$	$\phi(\boldsymbol{\theta}, \boldsymbol{q}) - \phi(\boldsymbol{\theta}', \boldsymbol{q})$	x_T
Veracity score (Zhou et al., 2024)	$\dfrac{\lvert y - \hat{f}(\boldsymbol{q}) \rvert}{\hat{u}(\boldsymbol{q}) + \hat{\sigma}(\boldsymbol{q})}$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}', \boldsymbol{q})$	x_T
Data-IQ (Seedat et al., 2022)	$[\pi(\boldsymbol{q})]_y$	$\{\boldsymbol{r}\}$	$\phi(\boldsymbol{\theta}, \boldsymbol{q})$	$\frac{1}{T}\sum_t x_t$ (confidence), $\frac{1}{T}\sum_t (1 - x_t)$ (aleatoric)

2014). We combine $\mathcal{Z}$ and agg_d into a single human-readable axis with four legal values: train, test, self-influence, and local test-influence with $k = 0.05 \times |\mathcal{D}_{\text{test}}|$. We use three existing agg_t: final snapshot, mean, and variance, where the latter stands in for other notions of instability. There are 96 valid metrics in total over the full enumeration.

Axis	Value	Definition
Architecture	deep learning	MLP, ResNet, and FT-Transformer
	GBDTs	CatBoost, LightGBM, and XGBoost
Probe method (ϕ)	loss	Cross-entropy loss
	confidence	Maximum logit value
	gradient	L2 distance to the class gradient centroid
	disagreement	
	feature	distance of feature encoding from a ridge regression
	predictability	model fit from other data points
Probe set & dataset aggregation ($\mathcal{Z}$ & agg_d)	train	Reference point in the train split
	test	Reference point in the test split
	self-influence	Change in the atom on itself when removed
	local	Change in the point's atom of the point's neighborhood
	test-influence	of test points
Snapshot aggregation (agg_t)	final	Final snapshot
	mean	Snapshot mean
	variance	Snapshot variance

Table 2: **Concrete instantiation of the taxonomic framework.** Concrete metrics are any combinations of the four design axes, 96 total.

4 METHODS

4.1 DESCRIPTIVE MODELING

The first half of our experiments involves collecting and modeling example difficulty metrics across a large dataset. Our data spans 96 metrics, 33 tasks, 6 architectures, 237,600 models, and 20 snapshots per model. We perform targeted analyses for the following research questions. **RQ1.** How do the metrics in our taxonomy relate to other published example difficulty metrics and IH metrics? **RQ2.** Which axes of the framework are the most influential design knobs for ranking data? **RQ3.** Which values within the axes are effectively redundant, if any?

Tasks, model training & measurement. We selected 33 tasks from the OpenML-based AutoML benchmark AMLB, from 71 total, filtering out tasks that are too large, deterministic, or where trained models predict at chance level (Appendix A.1) (Gijsbers et al., 2024; McElfresh et al., 2023). Three tasks were randomly designated as held-out tasks for RQ6, with all other analyses using 30 held-in tasks. We then artificially corrupted 10% of data points with feature-space jittering and 10% with label corruption, independently, for later experiments. The label corruption was class-conditioned and preserved relative class counts. We hyperparameter-tuned all architectures on each task for 150 iterations (Akiba et al., 2019). Accurate measurement of certain generalized example difficulty metrics for all points is infeasible for large datasets. We therefore sampled up to 500 class-balanced "observed points" as measurement targets. We included other points in the train set but not as $\boldsymbol{r}$ or $\boldsymbol{q}$. We used 300 independent four-fold train/test splits of the observed points, with 20 evenly spaced snapshots. We measured the spurious correlations from the shared variance of k-fold splitting and found it negligible. See Appendix A.2 for full details.

Analyses. RQ1. We removed ten unreliable metrics with split-half self-correlation below 0.75, following Cicchetti (1994). We then computed the correlation between the 86 reliable metrics in our taxonomy and four IH metrics and 13 published example difficulty metrics. We pooled all data and used weighted Spearman's rank correlations over the observed points between metrics, up weighting small datasets ($\times 500/|\text{observed}|$) to ensure equal contribution, and attenuating via self-reliability (Figure 7, 8 in Appendix A.3). We RQ2. We conducted a G-study, treating each observed point as a facet alongside the design axes (Cronbach et al., 1972). We then computed the relative proportion of variance explained by each axis' interaction term with the observed point facet. RQ3. We implemented a monotrait-heteromethod analysis (Campbell & Fiske, 1959). For each axis and each pair of domain values, we pairs of metrics identical in all other axes, then computed the

weighted Spearman correlation over the pooled observations, and disattenuated the results using split-half reliabilities (Spearman, 1904).

4.2 Data-Centric Applications

The second half directly tests the metrics on five data-centric applications: noisy label, noisy feature, conflicting duplicate, missingness, and data pruning. **RQ4.** Which axis is the most influential design knob for each application? **RQ5.** Which domain value within each axis is the best for each application? **RQ6.** What individual metrics are the best-performing metrics for each application? Note that RQ2-5 require our grid-enumeration approach, which balances the cohort across axes.

Application setup. Noisy label and feature detection used the corrupted data points. Conflicting duplicates and missingness detection used natural dirtiness properties, denoting rows with identical features but conflicting labels, or missing cells, dropping artificially corrupted data points. These are natural analogues and construct validation tests for the syntehtic corruptions. THey are present in 6 and 4 tasks respectively, too small for their own hold-out. All detection tasks used F1@k where k is the number of target points, and we ranked metrics by task-wise median. For data pruning, we trained LightGBM and MLP models on the observed points, then retrained them after removing 5, 10, or 20% of the most/least difficult points, with random pruning and InfoBatch with MLP as baselines, and reported balanced held-out accuracy delta (Qin et al., 2024). The small training pool for this task is due to its inherent tension with stronger statistics elsewhere.

Analyses. RQ4. We reused the design of RQ2, but used application performance as the response signal rather than rankings, and included dataset as a facet. We then report the share of performance variance for each axis. RQ5. For each axis, we report the performance of each value within an axis, averaged over all combinations of other axes. RQ6. We ranked all metrics by their median performance across tasks, then report the leading metrics against baselines.

5 Results

Figures 2 to 6 in the main text and Appendix A.3 show the results. For brevity, we use several consistent shorthands: gradient for gradient disagreement, feature for feature predictability, self-inf for self-influence, test-inf for local test influence, dataset agg for the probe set + dataset aggregation combined axis, and variance or var for snapshot-wise variance. Conclusions drawn are primarily applicable to the specific instantiation of our taxonomy.

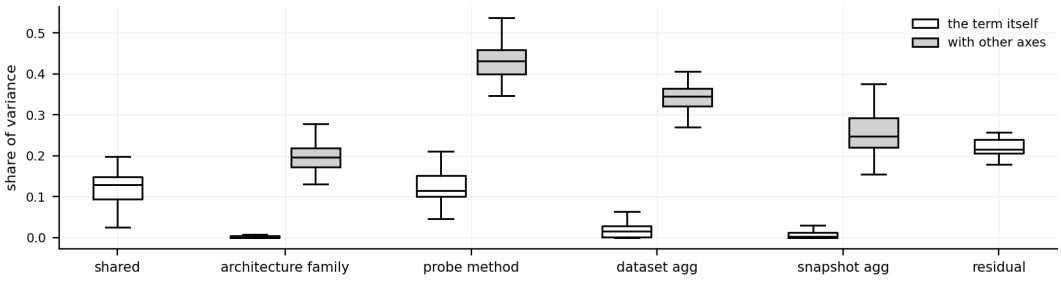

Figure 2: **(RQ2) Share of variance in rankings explained by each design axis.** Box plot shows the task-wise share of total variance explained by each facet, across 30 held-in tasks. Shared facet conditions only on the data point with no information about the metric design axes. Higher-order combination terms with any other axes, summed up, is also shown. Residual is unexplained variance.

5.1 Descriptive Modeling

RQ1. How do the metrics in our taxonomy relate to other example difficulty and IH metrics? See Figure 8 in Appendix A.3.2. Most published example difficulty and IH metrics show a similar correlation pattern to the example difficulty metrics in our taxonomy. They tend to correlate strongly with train/test/self-influence of loss, and moderately with train/test/self-influence of confidence and

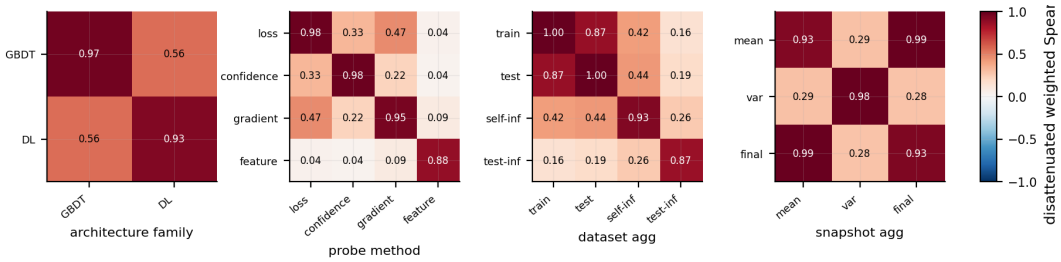

Figure 3: **(RQ3) Rank correlations between pairs of values within each design axis keeping others fixed.** Values show weighted Spearman's rank correlations between two values within an axis when fixing all other axes, disattenuated via split-half reliabilities. Diagonals show split-half reliabilities. Note that correlation $\gtrsim 0.9$ is a standard threshold for redundant values (Brown, 2015).



Figure 4: **(RQ4) Share of variance in application performance explained by each design axis and dataset.** Performance is measured as F1@k and balanced accuracy delta compared to standard training for data pruning. Values are averaged across all included tasks per application task. Higher-order terms collapsed into "with other axes."

gradient disagreement. Test-influence and feature-predictability variants are weakly correlated with the external metrics, indicating that they are orthogonal constructs.

> **Takeaway 1.** *The majority of published example difficulty and IH metrics correlate together and with train/test/self-influence loss. Be aware of this redundancy.*

RQ2. Which axes of the framework are the most influential design knobs for ranking data? See Figure 2. The share of variance measures the impact of changing a facet value from one to another on the ranking of data points by example difficulty. In order of descending importance, we get: 1) unexplained residuals, 2) a shared facet, reflecting a metric-agnostic notion of example difficulty, and 3) the probe method facet. All other axes trail behind.

> **Takeaway 2.** *Changing the probe method is the most influential, followed by the* agg_d *axis. Pay close attention to their design.*

RQ3. Which values within the axes are effectively redundant, if any? See Figure 3. Using the standard threshold of $\gtrsim 0.9$ correlation, final snapshot and snapshot mean (0.99) are redundant, and train- and test-time agg_d are nearly redundant (Brown, 2015). No other pairs are redundant.

> **Takeaway 3.** *Final snapshot measurements and snapshot-wise mean are redundant. Train-time and test-time measurements are nearly redundant.*

5.2 Data-Centric Applications

RQ4. Which axis is the most important design knob for each application task? See Figure 4. For this result, we also measure the effect of the dataset, which is large. The strong effect of the probe method and the follow-up agg_d remains. Notably, the relative importance of the dataset and design axes varies significantly across applications.

> **Takeaway 4.** *The importance of carefully designing the probing method, and secondarily* agg_d*, does not change for data-centric applications.*

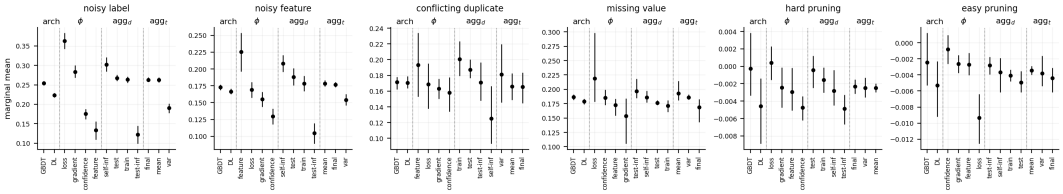

Figure 5: **(RQ5) Marginal mean of each value within each axis, holding all others fixed and partialling out the dataset.** Averaged across datasets with 95% bootstrap intervals ($n = 1{,}200$).

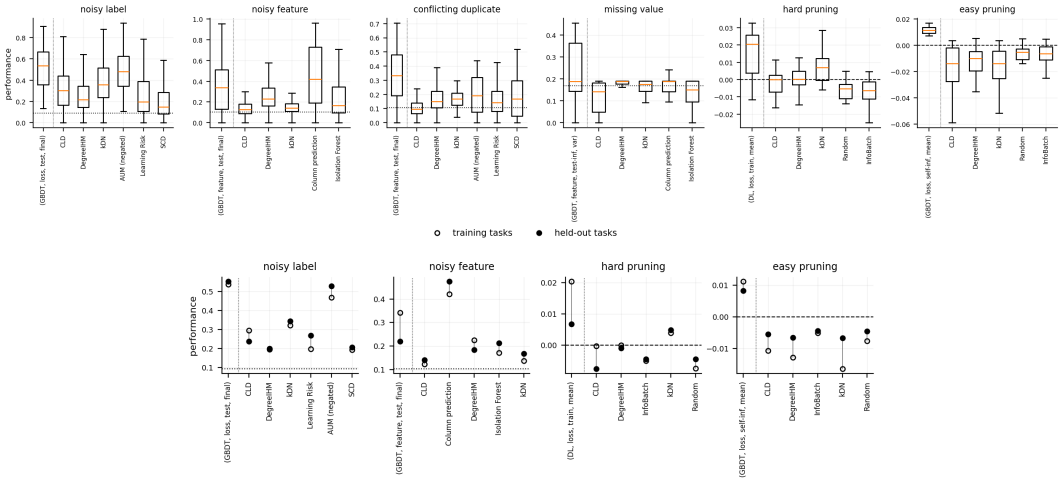

Figure 6: **(RQ6) Best-performing single metric in our taxonomy compared to IH and domain-specific baselines.** Performance is measured as F1@k and balanced accuracy delta compared to standard training for data pruning, ranked via dataset-wise median. Box plot for detection tasks shows distribution over tasks. 5-20% pruning for data pruning tasks. Dotted lines are chance-prediction and standard training baselines. Hold-out repetition at the bottom.

RQ5. Which domain value within each axis is the best for each application task? See Figure 5. The confidence intervals separate large tiers but not always adjacent neighbors. GBDT outperforms DL on average, though with overlapping intervals. Loss performs best for noisy label detection; feature predictability performs best for noisy feature detection. The effect size for pruning is small.

> **Takeaway 5.** *Train and measure GBDTs over DL in general. Measure loss for label-oriented tasks. Beating standard training via pruning remains difficult.*

RQ6. Which individual metric performs the best for each application task? See Figure 6. The best metric in our battery outperforms all baselines, except for noisy feature detection, where Zhou et al. (2024)'s column prediction method—which motivated feature predictability—is the best.

> **Takeaway 6.** Different metrics top the charts for different tasks. GBDT + loss/feature predictability based metrics are the most effective. Column prediction is the winner for noisy features. Top metrics for noisy labels and data pruning largely replicate in the held-out data.

6 Conclusion

We propose a framework to design and taxonomize generalized example difficulty metrics systematically, and study their behavior and applications for tabular classification. Our taxonomy-first approach eliminates inclusion bias within our chosen domains. This lets us draw principled conclusions about the metrics' design space, such as the relative importance and redundancies of design axes. We also use our taxonomy to identify metrics strong at data-centric AI applications.

AI USE DISCLOSURE

We used generative AI tools for literature search, coding the experiments, and for plotting. We did not use AI to formulate the research topic, high-level experimental design, theoretical framework, title, keywords, paper structure, or hypothesis. We did not use AI to interpret experimental results. We used generative AI to draft certain sections' outlines. All AI outlines underwent extensive human revision and rewriting. No writing at submission is AI-generated. Mathematical proofs, synthetic data, translations, qualitative data analysis, or surveys are not applicable to this work. We used generative AI to write code, and AI suggested parameters for certain experiments. We reviewed AI-driven literature search results and AI-written code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All data and code used for this paper will be made public by publication. This includes the code for HPO, model training, metric measurement, analyses, and plotting, and HPO data, model artifacts, and processed data. The entire experiment, analysis, and data visualization pipelines are fully deterministically reproducible.

REFERENCES

Chirag Agarwal, Daniel D'souza, and Sara Hooker. Estimating example difficulty using variance of gradients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10368–10378, 2022.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Shaden Alshammari, John Hershey, Axel Feldmann, William Freeman, and Mark Hamilton. A unifying framework for representation learning. In *International Conference on Learning Representations*, volume 2025, pp. 59257–59287, 2025.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger B Grosse. Training data attribution via approximate unrolling. *Advances in Neural Information Processing Systems*, 37:66647–66686, 2024.

Robert Baldock, Hartmut Maennel, and Behnam Neyshabur. Deep learning through the lens of example difficulty. *Advances in Neural Information Processing Systems*, 34:10876–10889, 2021.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE transactions on neural networks and learning systems*, 35(6):7499–7519, 2022.

Timothy A Brown. *Confirmatory factor analysis for applied research*. Guilford publications, 2015.

Donald T. Campbell and Donald W. Fiske. Convergent and discriminant validation by the multitrait-multimethod matrix. *Psychological Bulletin*, 56(2):81–105, 1959. doi: 10.1037/h0046016.

Nicholas Carlini, Ulfar Erlingsson, and Nicolas Papernot. Distribution density, tails, and outliers in machine learning: Metrics and applications. *arXiv preprint arXiv:1910.13427*, 2019.

Jiwon Chang, Christina Dionysio, Fatemeh Nargesian, and Matthias Boehm. Plutus: Understanding data distribution tailoring for machine learning. In *Companion of the 2024 International Conference on Management of Data*, pp. 528–531, 2024.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Domenic V. Cicchetti. Guidelines, criteria, and rules of thumb for evaluating normed and standardized assessment instruments in psychology. *Psychological Assessment*, 6(4):284–290, 1994. doi: 10.1037/1040-3590.6.4.284.

Pierre-Olivier Côté, Amin Nikanjam, Nafisa Ahmed, Dmytro Humeniuk, and Foutse Khomh. Data cleaning and machine learning: a systematic literature review. *Automated Software Engineering*, 31(2):54, 2024.

Lee J. Cronbach, Goldine C. Gleser, Harinder Nanda, and Nageswari Rajaratnam. *The Dependability of Behavioral Measurements: Theory of Generalizability for Scores and Profiles*. Wiley, New York, 1972.

Alex Davies and Zoubin Ghahramani. The random forest kernel and other kernels for big data from random partitions. *arXiv preprint arXiv:1402.4293*, 2014.

Lihui Deng, Bo Yang, Zhongfeng Kang, Jiajin Wu, Shaosong Li, and Yanping Xiang. Separating hard clean samples from noisy samples with samples' learning risk for dnn when learning with noisy labels. *Complex & Intelligent Systems*, 10(3):4033–4054, 2024.

Jasjeet Dhaliwal and Saurabh Shintre. Gradient similarity: An explainable approach to detect adversarial attacks against deep learning. *arXiv preprint arXiv:1806.10707*, 2018.

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in neural information processing systems*, 33:2881–2891, 2020.

Mahsa Forouzesh and Patrick Thiran. Differences between hard and noisy-labeled samples: An empirical study. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*, pp. 91–99. SIAM, 2024.

Thomas George, Pierre Nodet, Alexis Bondu, and Vincent Lemaire. Mislabeled examples detection viewed as probing machine learning models: concepts, survey and extensive benchmark. *arXiv preprint arXiv:2410.15772*, 2024.

Pieter Gijsbers, Marcos LP Bueno, Stefan Coors, Erin LeDell, Sébastien Poirier, Janek Thomas, Bernd Bischl, and Joaquin Vanschoren. Amlb: an automl benchmark. *Journal of Machine Learning Research*, 25(101):1–65, 2024.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in neural information processing systems*, 34:18932–18943, 2021.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? *Advances in neural information processing systems*, 35: 507–520, 2022.

Cheng Guo and Felix Berkhahn. Entity embeddings of categorical variables. *arXiv preprint arXiv:1604.06737*, 2016.

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. FastIF: Scalable influence functions for efficient model interpretation and debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10333–10350, 2021. doi: 10.18653/v1/2021.emnlp-main.808.

Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. Characterizing structural regularities of labeled data in overparameterized models. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5034–5044. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/jiang21k.html`.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Paul Kline. *An easy guide to factor analysis*. Routledge, 2014.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Devin Kwok, Nikhil Anand, Jonathan Frankle, Gintare Karolina Dziugaite, and David Rolnick. Dataset difficulty and the role of inductive bias. *arXiv preprint arXiv:2401.01867*, 2024.

Elizaveta Levina and Peter J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper/2004/hash/74934548253bcab8490ebd74afed7031-Abstract.html.

Peng Li, Xi Rao, Jennifer Blase, Yue Zhang, Xu Chu, and Ce Zhang. Cleanml: A study for evaluating the impact of data cleaning on ml classification tasks. In *2021 IEEE 37th international conference on data engineering (ICDE)*, pp. 13–24. IEEE, 2021.

Ana C Lorena, Luís PF Garcia, Jens Lehmann, Marcilio CP Souto, and Tin Kam Ho. How complex is your classification problem? a survey on measuring classification complexity. *ACM Computing Surveys (CSUR)*, 52(5):1–34, 2019.

Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Vishak Prasad C, Ganesh Ramakrishnan, Micah Goldblum, and Colin White. When do neural nets outperform boosted trees on tabular data? *Advances in Neural Information Processing Systems*, 36:76336–76369, 2023.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607, 2021.

Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, 33:17044–17056, 2020.

Neoklis Polyzotis, Sudip Roy, Steven Euijong Whang, and Martin Zinkevich. Data lifecycle challenges in production machine learning: a survey. *ACM Sigmod Record*, 47(2):17–28, 2018.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33: 19920–19930, 2020.

Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Zhaopan Xu, Zhou Daquan, Lei Shang, Baigui Sun, Xuansong Xie, et al. Infobatch: Lossless training speed up by unbiased dynamic data pruning. In *International Conference on Learning Representations*, volume 2024, pp. 39723–39744, 2024.

Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M Aroyo. "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2021.

Nabeel Seedat, Jonathan Crabbé, Ioana Bica, and Mihaela van der Schaar. Data-iq: Characterizing subgroups with heterogeneous outcomes in tabular data. *Advances in Neural Information Processing Systems*, 35:23660–23674, 2022.

Nabeel Seedat, Fergus Imrie, and Mihaela van der Schaar. Dissecting sample hardness: A fine-grained analysis of hardness characterization methods for data-centric ai. In *International Conference on Learning Representations*, volume 2024, pp. 9495–9564, 2024.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565, 2022.

Charles Spearman. "general intelligence," objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292, 1904. doi: 10.2307/1412107.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJlxm30cKm.

Gustavo Torquette, Victor Nunes, Pedro Paiva, Lourenço Neto, and Ana Lorena. Characterizing instance hardness in classification and regression problems. In *Anais do X Symposium on Knowledge Discovery, Mining and Learning*, pp. 178–185, Porto Alegre, RS, Brasil, 2022. SBC. doi: 10.5753/kdmile.2022.227758. URL https://sol.sbc.org.br/index.php/kdmile/article/view/24984.

Gustavo P Torquette, Victor S Nunes, Pedro YA Paiva, and Ana C Lorena. Instance hardness measures for classification and regression problems. *Journal of Information and Data Management*, 15(1):152–174, 2024.

Andrej Tschalzev, Lennart Purucker, Stefan Lüdtke, Frank Hutter, Christian Bartelt, and Heiner Stuckenschmidt. Unreflected use of tabular data repositories can undermine research quality. *arXiv preprint arXiv:2503.09159*, 2025.

Jiachen T Wang, Prateek Mittal, and Ruoxi Jia. Efficient data shapley for weighted nearest neighbor algorithms. In *International Conference on Artificial Intelligence and Statistics*, pp. 2557–2565. PMLR, 2024.

Jiachen T. Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HD6bWcj87Y.

Yu-Xiang Wang. Per-instance differential privacy. *Journal of Privacy and Confidentiality*, 9(1), 2019.

Hang Zhou, Jonas Mueller, Mayank Kumar, Jane-Ling Wang, and Jing Lei. Detecting errors in a numerical response via any regression model. *Journal of Data-centric Machine Learning Research*, 2024. URL https://openreview.net/forum?id=CIQ5iemeTw.

A Appendix

A.1 Full Task Suite

Table 3: **AMLB classification tasks used.** We use AMLB classification tasks. We exclude those that are too large ($n \geq 200{,}000$), too wide for DL or GBDTs to be competitive with linear regression (#feat > 100), or where the target classes are too many for measurements to be stable (#class ≥ 100). We also exclude deterministic tasks and tasks where a trained model's accuracy is at chance. The class ratio (CR) is the ratio of the least common label's frequency to that of the most common. 38 AMLB tasks survive the n_rows, #feat, and #class filters. Four tasks are deterministic (`car`, `connect-4`, `jungle_chess_2pcs_raw_endgame_complete`, `kr-vs-kp`), and one task is predicted at chance (`numerai28.6`), yielding the final list of 33 tasks.

Task ID	Name	n	#feat	#class	CR
190411	ada	4147	49	2	0.330
359983	adult	48842	15	2	0.314
359979	Amazon_employee_access	32769	10	2	0.061
146818	Australian	690	15	2	0.801
359982	bank-marketing	45211	17	2	0.132
359955	blood-transfusion-service-center	748	5	2	0.312
359968	churn	5000	21	2	0.164
359992	Click_prediction_small	39948	12	2	0.202
359959	cmc	1473	10	3	0.529
168757	credit-g	1000	21	2	0.428
211986	Diabetes130US	101766	50	3	0.207
359954	eucalyptus	736	20	5	0.490
359969	first-order-theorem-proving	6118	52	6	0.190
359970	GesturePhaseSegmentationProcessed	9873	33	5	0.338
211979	jannis	83733	55	4	0.043
359962	kc1	2109	22	2	0.182
359991	kick	72983	33	2	0.140
359990	MiniBooNE	130064	51	2	0.390
359993	okcupid-stem	50789	20	3	0.134
190137	ozone-level-8hr	2534	73	2	0.067
359958	pc4	1458	38	2	0.139
359971	PhishingWebsites	11055	31	2	0.795
168350	phoneme	5404	6	2	0.415
359956	qsar-biodeg	1055	42	2	0.509
359975	Satellite	5100	37	2	0.014
359963	segment	2310	20	7	1.0
359987	shuttle	58000	10	7	0.0002
168784	steel-plates-fault	1941	28	7	0.081
359972	sylvine	5124	21	2	1.0
190146	vehicle	846	19	4	0.912
146820	wilt	4839	6	2	0.057
359974	wine-quality-white	4898	12	7	0.002
2073	yeast	1484	9	10	0.010

A.2 Experimental Details

Datasets and tasks. We drew datasets and tasks from the OpenML-based AutoML benchmark AMLB, which catalogs 71 tabular classification tasks. We filtered tasks with n_rows $\geq$ 200,000, #feat $>$ 100, or #class $>$ 100. We also excluded tasks generated by deterministic processes, such as games, and one task in which tuned models performed no better than chance. This left 33 AMBL tasks.

Data cleaning. We handled missing values as a separate label in all cases except for missing numbers in DL, which we median-imputed. We dropped rows with no observed features and removed duplicate rows introduced to OpenML via clerical error. `eucalyptus`, `kick`, and `okcupid-stem` are the three tasks where we median-imputed numeric missing values and changed cell values. `ozone-level-8hr`'s missing values were mean-imputed by the OpenML dataset authors. `MiniBooNE` was the only task with points missing all values, totaling 468 points. `segment` and `yeast` contained confirmed duplicate points from clerical error and were deduplicated.

Data corruption. Noisy label and feature detection are key applications of example difficulty, and George et al. (2024) assess the utility of various metrics. Accordingly, we randomly corrupted a subset of rows in each dataset before model training. Specifically, we feature-corrupted 10% of rows with numeric jitters and category perturbations; another 10% were class-conditioned label-corrupted. We performed data corruption after all data-cleaning tasks, on all points, including both observed and non-observed data points. We selected points for label-corruption or feature-corruption independently, yielding approximately 1% of points corrupted in both ways.

The probability of flipping label i to some other label j was the row-normalized off-diagonal of a GaussianNB confusion matrix. Up to $(0.1 \times \text{n_rows})$ points were label-corrupted, capped at the largest feasible total that satisfies the constraints: at least one corruption per gt class, a maximum 10% corruption rate per gt class, and preservation of each class's population size.

For feature corruption, we corrupted up to 10% of points per class, with the constraint of at least one corruption per gt class. We corrupted 25% of columns using both numeric jitters and category perturbations, displacing each data point to its distance to the 5th nearest neighbor in feature space.

Observed points. We randomly selected observed points in a class-balanced manner and constrained them to at most 500 points. Five imbalanced datasets yielded less than 500 observed points: `wine-quality-white`, `yeast`, `shuttle`, `Satellite`, and `blood-transfusion-service-center`. We selected observed points after data corruption and independently of the previous steps.

Hyperparameter optimization. We ran hyperparameter optimization once per (task, architecture) pair for 150 trials. The search used the Gaussian process sampler from Optuna (Akiba et al., 2019) and a three-fold cross-validation balanced accuracy target on already-corrupted data. We constrained training to at least 20 epochs (or boosting rounds) to ensure snapshot aggregations are well-defined and stable. We ensured that no boosting round for GBDTs was trivial, where no leaf is split.

Folds. We drew 300 independent stratified four-fold splits over the observed points, including all non-observed points in the train set.

GBDT architectures. We used CatBoost 1.2.10, LightGBM 4.6.0, and XGBoost 3.2.0 with default values for all parameters not included in HPO.

DL architectures. All architectures used FP32 and ReLU, numeric and categorical embeddings, and skipped dropout on the FC layer. All models were trained on unweighted loss, including on imbalanced tasks. The tabular DL architectures use categorical column embeddings Guo & Berkhahn (2016). We excluded tabular foundation models (TFMs) despite their strong performance because their mechanism differs fundamentally from the architectures studied here. For TFMs, training epochs and sample-wise gradients are not well defined.

A.3 Additional Figures

A.3.1 Self-reliability

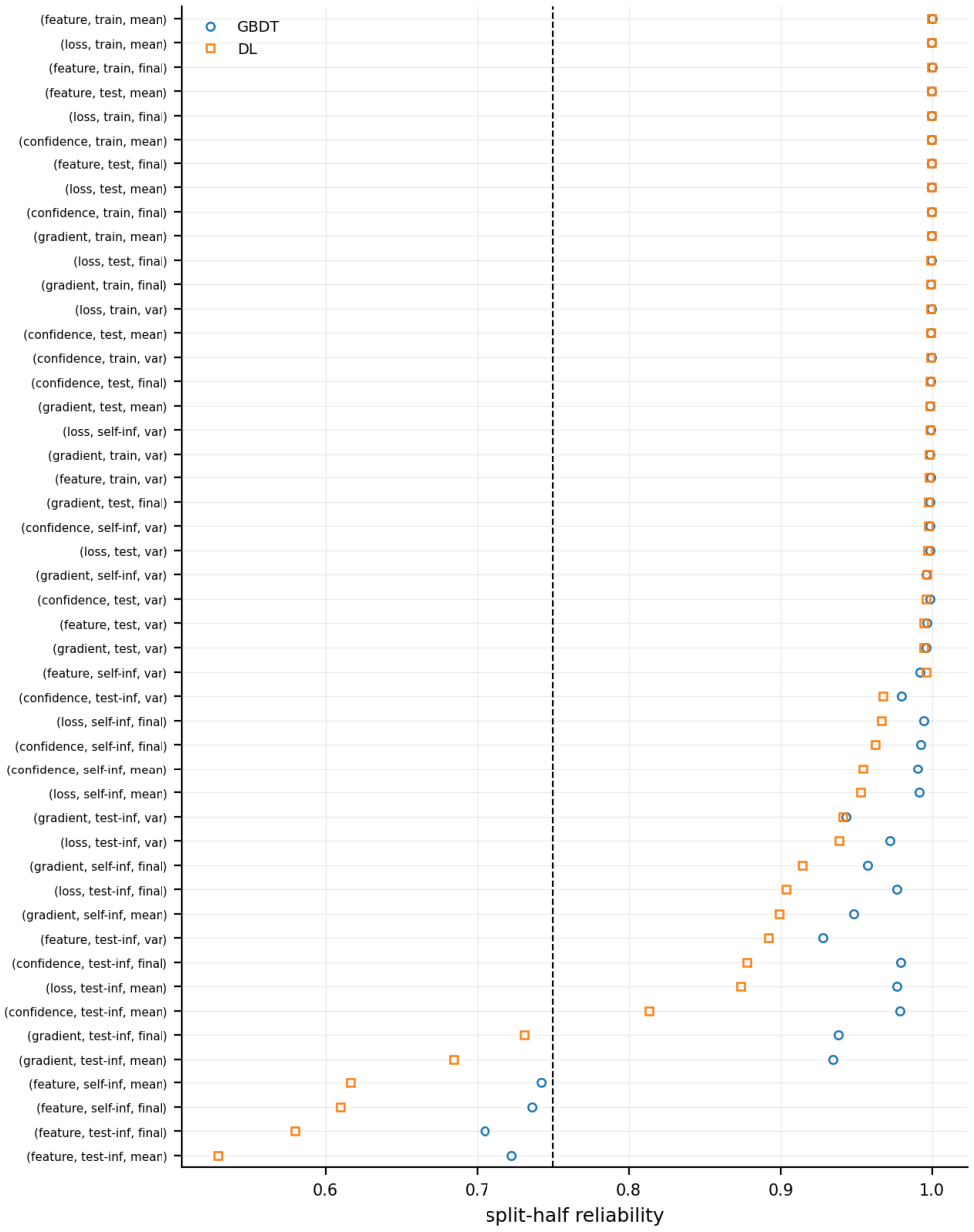

Figure 7: **Split-half self-agreement in rankings of the observed data points for each metric in the taxonomy.** Self-agreement is computed for each metric-task pair via 1200 random half-splits of data, then computing the halves' Spearman rank correlation over observed data points, median-aggregated across tasks. Ten metrics fall below 0.75 self-agreement (dotted vertical line) and are removed from Figure 8.

A.3.2 RQ1: INTER-METRIC RANK CORRELATIONS

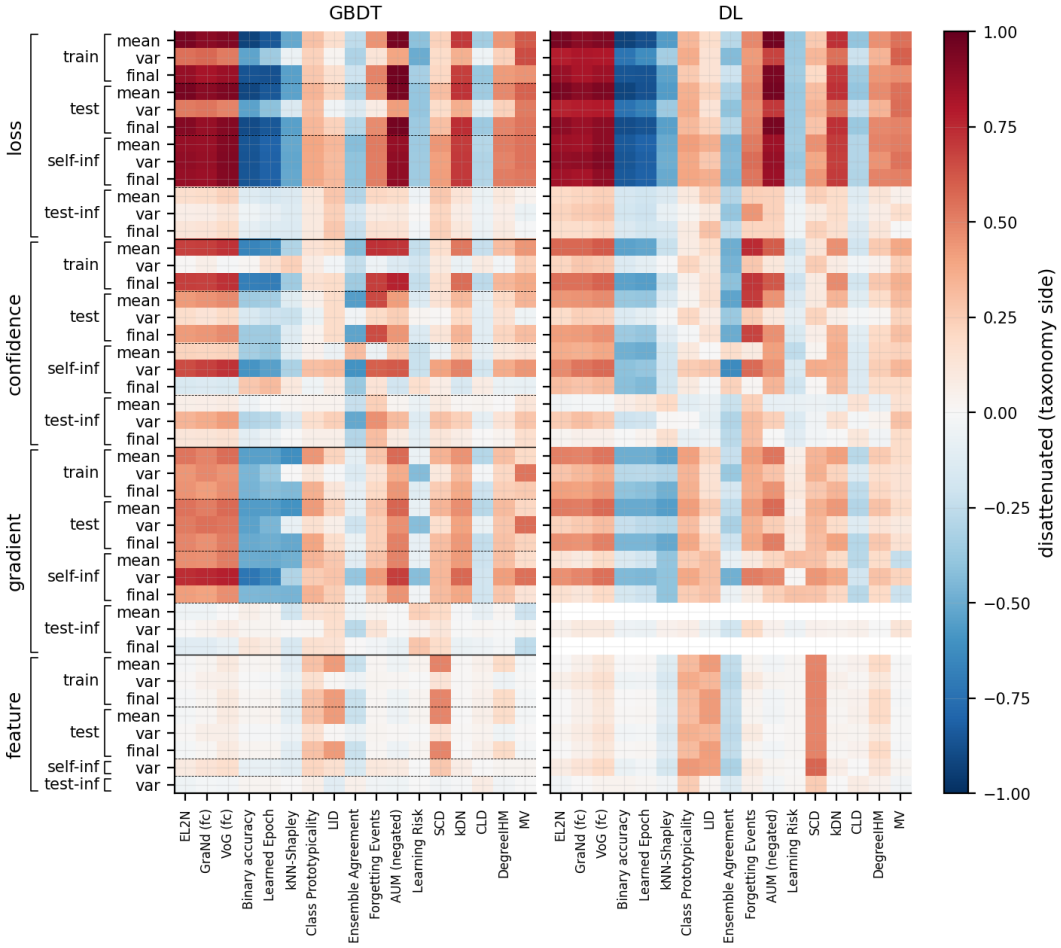

Figure 8: **(RQ1) Pooled rank correlation between the example difficulty metrics in our taxonomy and external metrics: 13 published example difficulty metrics and 4 IH metrics.** The data used here is pooled across all 30 held-in tasks. Weighted Spearman correlation is used to upscale small datasets with fewer than 500 observed points for equal contribution between tasks, and all correlations are disattenuated. **Example difficulty metrics:** error vector L2 normed (EL2N), gradient normed (GraNd; fc layer approximation), area under the margin (AUM; negated), variance of gradient (VoG; fc layer approximation), forgetting events, consistently learned epoch, binary accuracy, ensemble agreement (mean within-task Jensen-Shannon divergence of logits), kNN-Shapley ($k = 10$), Levina-Bickel maximum-likelihood estimator (LID), learning risk, static centroid distance (SCD), class prototypicality (Paul et al., 2021; Pleiss et al., 2020; Agarwal et al., 2022; Toneva et al., 2019; Carlini et al., 2019; Wang et al., 2024; Levina & Bickel, 2004; Deng et al., 2024; Forouzesh & Thiran, 2024; Sorscher et al., 2022). **IH metrics:** k-disagreeing neighbors (kDN), class likelihood difference (CLD), degree-based instance hardness measure (DegreeIHM), and minority value (MV) (Torquette et al., 2024; 2022).