DISCO: Diversifying Sample Condensation for Efficient Model Evaluation

Alexander Rubinstein

Benjamin Raible

Martin Gubri

Seong Joon Oh

STAI

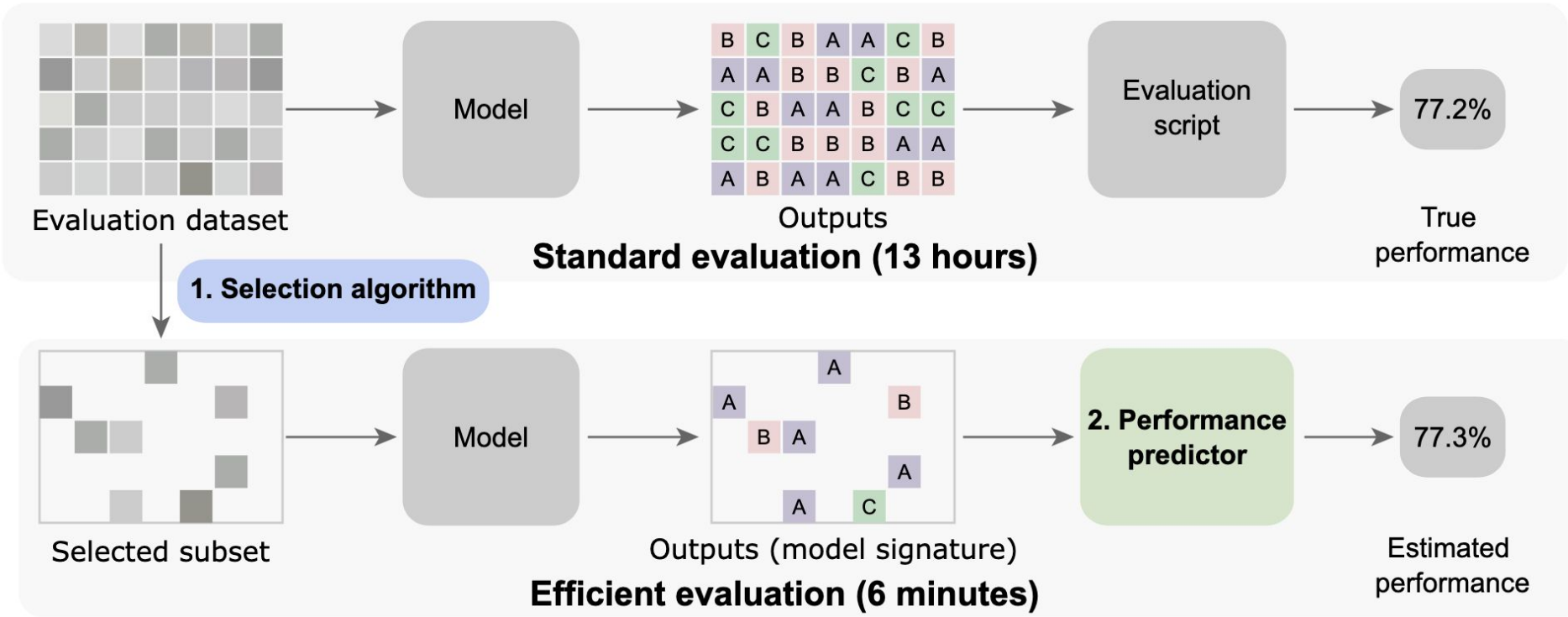
imprs-is

Tübingen AI Center

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Problem statement

Efficient evaluation = selection + prediction



Status quo

PERFORMANCE PREDICTION METHOD

1. DEFINE

A: Datapoints
Specific instance variable

Z: Models
Model-specific variable

Likelihood Model:
$P(S|A,Z)$
- Probability of model Z predicting correct answer on datapoint A

Correctness Score (s):
Binary, whether model Z is correct on data point A

2. FIT (EM)

ITERATIVE EM
Estimates of A and Z with EM of $P(S|A,Z)$

(EM

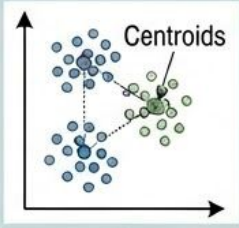
E-Step:
Estimate latent variables A, Z

M-Step:
Maximize likelihood $P(S|A,Z)$

PARAMETERS:
A, Z

3. SELECT

CLUSTER A



SELECT ANCHORS
(A_{anc})

4. FIT & PREDICT

FIT

Scores on Anchors
Input: S_{anc}, Z

Input: S_{full} Scores (S_{full})

TRAIN
Model g
$g(S_{anc}, Z) = S_{full}$
Maps correctness on anchor points to full set performance

TEST TIME

NEW
Z_{test}

Observe S_{anchor} on Anchors

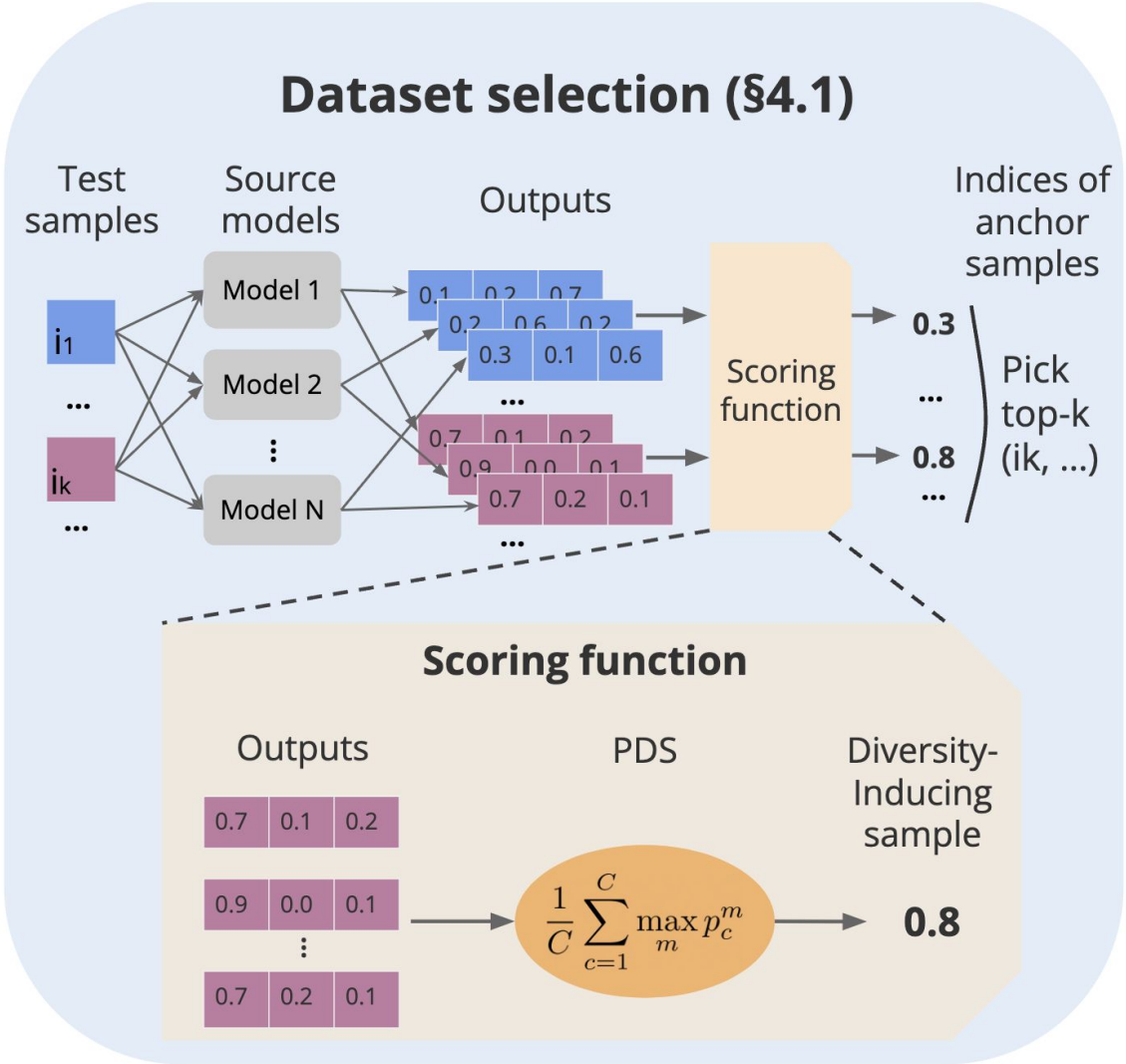
OBSERVE S_{anc}
MLE for Z_{test} (S_{anc})

PREDICT:
$S_{pred} = g(S_{anc}, Z_{test})$

Our solution

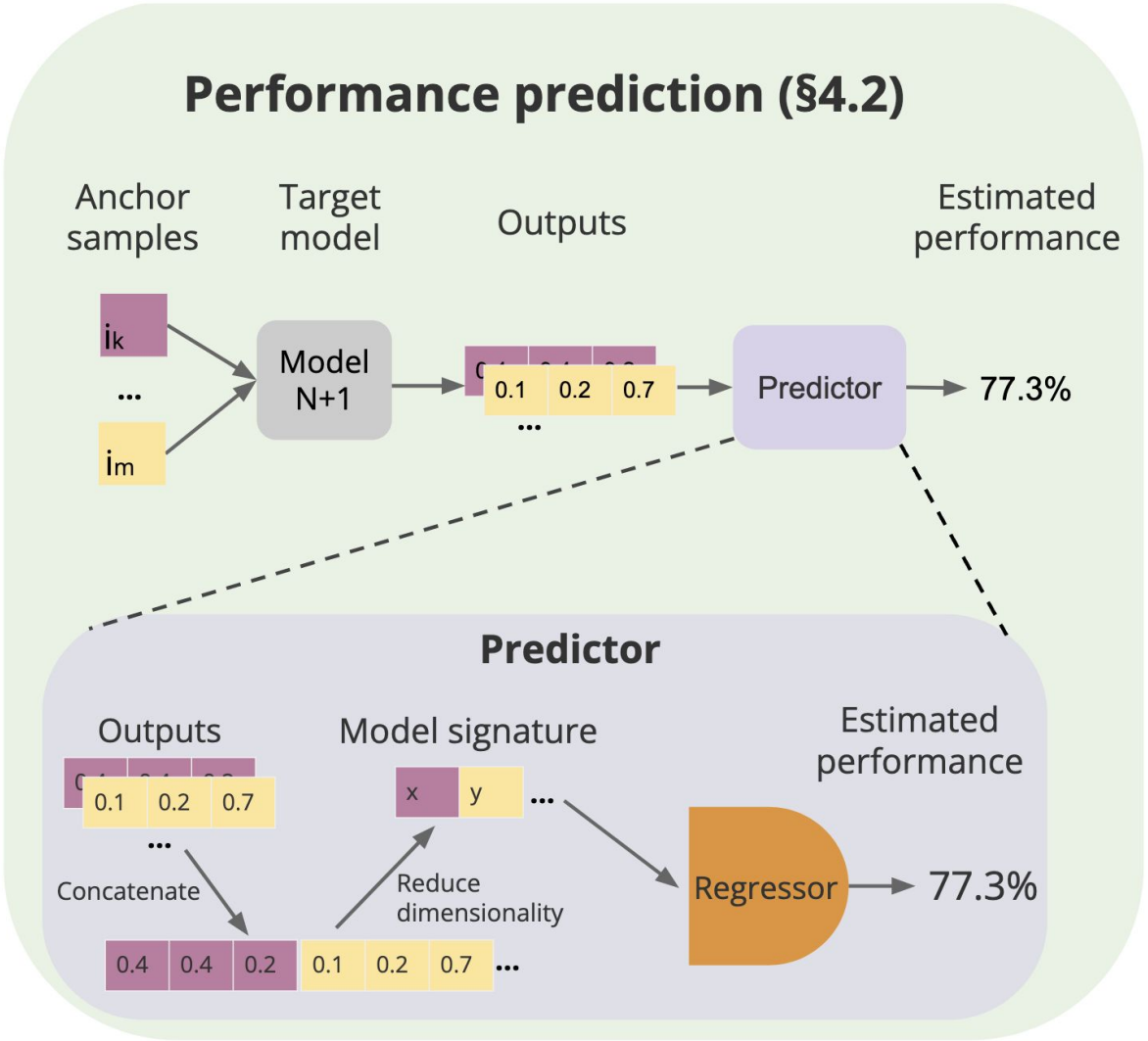
DISCO: select anchor data points

- No latent data point variables - only use models disagreement
- Select not based on datapoint inner qualities, but on responses diversification



DISCO: predict performance

- No latent model variables - only use model signatures



Approach	Selection	Prediction	MMLU (14k)		HS (10k)		WG (1.3k)		ARC (1.2k)	
	§4.1	§4.2	MAE↓	Rank↑	MAE↓	Rank↑	MAE↓	Rank↑	MAE↓	Rank↑
Baseline	Random	Direct eval.	3.45	0.916	2.85	0.839	3.60	0.827	2.61	0.898
tinyBenchmarks	Random	gp-IRT	2.79	0.922	1.96	0.819	1.64	0.928	2.22	0.921
	Anchor-IRT	gp-IRT	3.25	0.922	2.19	0.830	2.24	0.850	4.55	0.708
	Anchor-corr	gp-IRT	2.08	0.927	1.27	0.937	1.95	0.918	2.18	0.948
Metabench	Best for val.	ability-IRT	2.08†	0.904†	**0.80**†	0.974†	1.23†	0.947†	**1.14**†	**0.971**†
Model signature	Random	Sig. + kNN	1.82	0.912	1.49	0.899	1.58	0.920	2.30	0.905
		Sig. + RF	1.81	0.933	1.36	0.938	1.29	0.926	1.72	0.938
DISCO (ours)	High PDS	Sig. + kNN	1.31	0.972	1.32	0.956	1.19	0.951	1.96	0.937
		Sig. + RF	**1.07**	**0.987**	1.01	**0.984**	**1.00**	0.967	1.47	**0.971**
	High JSD	Sig. + kNN	1.14	0.975	1.50	0.944	1.26	0.955	2.11	0.939
		Sig. + RF	1.30	**0.987**	0.86	0.972	1.09	**0.973**	1.75	0.938

Approach	Selection §4.1	Prediction §4.2	IN val (50k) MAE↓	Rank↑
Baseline	Random	Direct eval.	3.03	0.652
Lifelong Bench.	Uniform correctness	Weighted sum	2.06	0.838
SSEPY	Uniform confidence	Weighted sum	3.05	0.762
Model signature	Random	Sig. + kNN	1.72	0.808
		Sig. + RF	0.86	0.944
DISCO (ours)	High PDS	Sig. + kNN	1.68	0.819
		Sig. + RF	**0.63**	**0.969**

Conclusions

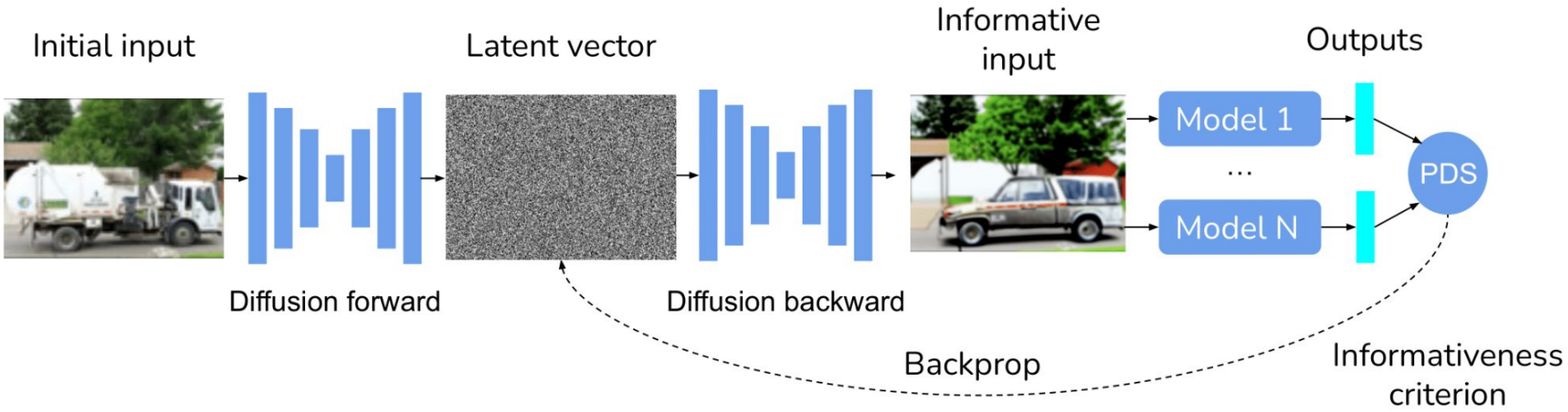
If you need to evaluate models more efficiently:

If you want to save up to 99.3% of compute at the cost of 1-2% error:

1. Select data points that diversify models' responses
2. Fit a lightweight prediction model from outputs on these data points to full test performance

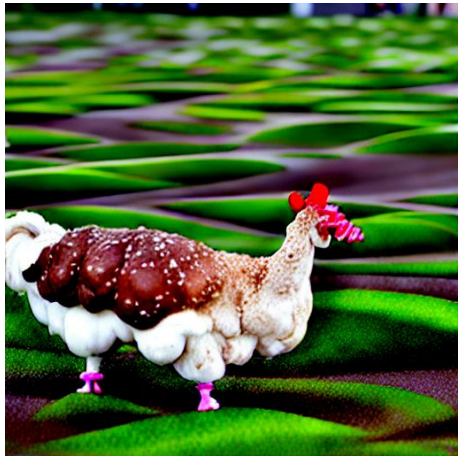
Appendix

Let’s look which samples diversify model outputs:



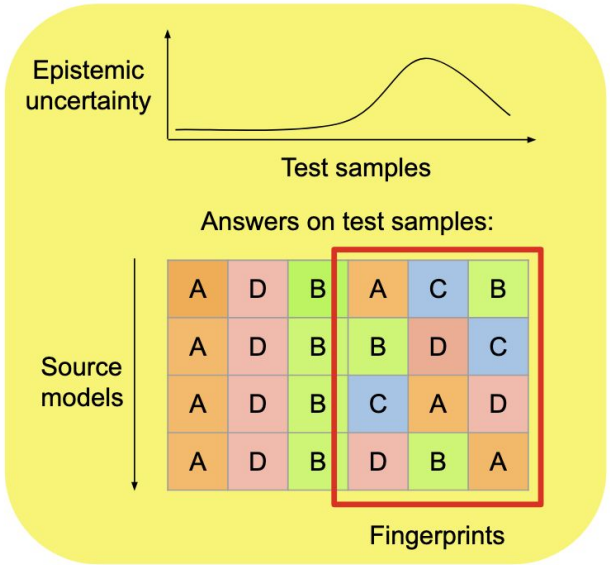
Inspired by: Dig-in: Diffusion guidance for investigating networks-uncovering classifier differences neuron visualisations and visual counterfactual explanations (CVPR 2023)

Some funny qualitative examples

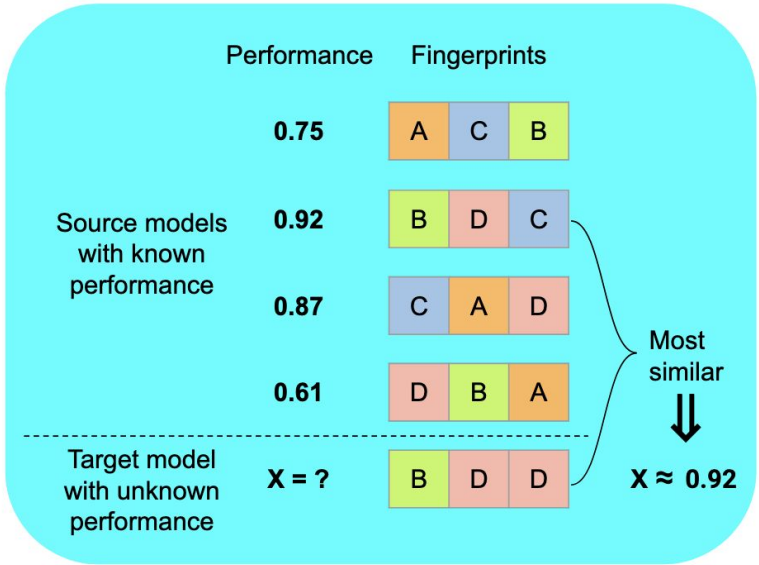


Very nice, but can they help us with anything?

Estimate performance based on models' similarity



Compute models' fingerprints



Predict performance based on fingerprints similarity