Is Hyper-Parameter Optimization Different for SE?

Yes. And here's the math to prove it.

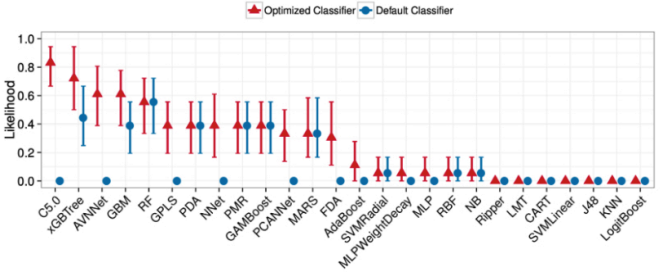
Rahul Yedida (Lexis Nexis) Tim Menzies (NCSU)

NC State University, USA
timm@ieee.org
http://timm.fyi

TSE, 51(6), 2025, https://ieeexplore.ieee.org/document/10919477
SLIDES: http://timm.fyi/26smooth.pdf

Apr 17, 2026

Motivation: Off-the-Shelf AI for SE Data



[Tantithamthavorn et al., ICSE'16]

Most SE research uses AI defaults without **audit**.

- HPO = hyperparameter optimization. Black art of selecting tuning parameters.
- HPO is applied to defect prediction and issue triage.
- Current tools (NSGA-II, MOEA/D, SMAC3, BOHB, DEHB, etc) target **general AI data**.
- Question: Is SE data different? Does it need different kinds of learning?

Message

- All you LLM users– you are missing what **matters**.
- SE-AI $\neq$ standard AI.
- And you can **leverage** that.

So... many... studies....

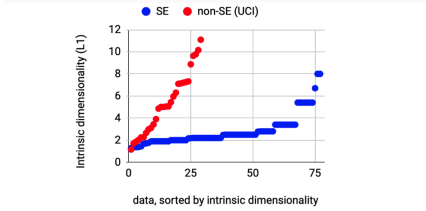


Fig. 1: Differences in the intrinsic dimensionality of SE and non-SE data. From [30].

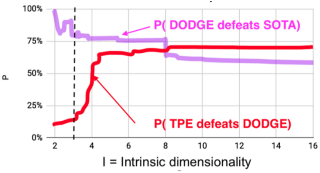


Fig. 2: According to Agrawal et al. [30] different algorithms work best at different intrinsic dimensionalities. Vertical dashed lines shows the median of the SE data from Figure 1.

- 2021: Intrinsic dimensionality
  doi.org/10.1109/TSE.2021.3073242
- DRR (data reduction ratio):
  - SE data "flattens" more than other data.
- Count neighbors at R_1 vs. $n \times R_1$.
  - Selects **very simple**optimization problems.
  - Most SE data sets are simple this way.
- **Issue**: not enough data to generalize

So... many... (more) studies....

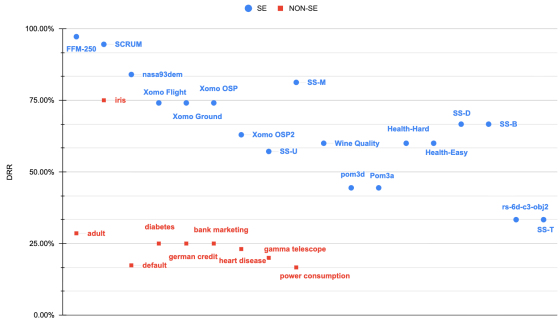
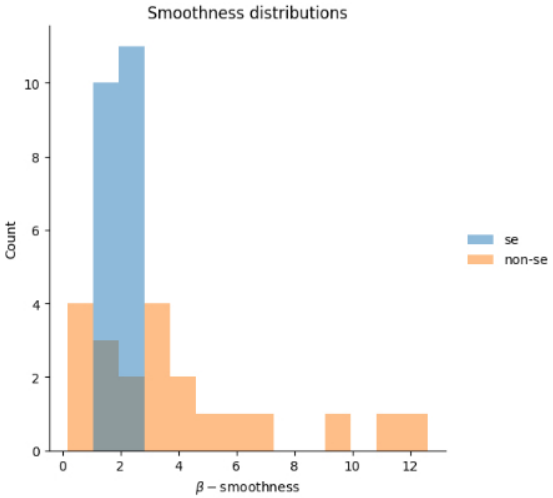


Fig. 4: Shown here are the Table 1 data sets, scored in the y-axis by Equation 1 (and the data is spaced out across the x-axis for readability). For the blue points, in all cases, very simple optimizers (that use only 30 samples) perform as well as more complex optimizers (that require 3000 samples).

- 2024: DRR (data reduction ratio) **arxiv.org/abs/2503.21086**
- SE data “flattens” more than other data.
- Above the “Andre threshold”, astonishingly simple methods work **well**:
  - Configuration
  - Project management
  - Hyperparameter optimization for **predicting** OSS project health
  - Banking, health, power;
  - etc.
- **Issue**: could not explain ”why”

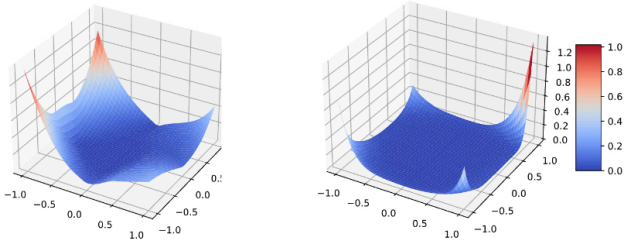
So ... many... (yet more).... studies...



- 2025: this study
  - SE data has “smoother” **boundaries**between classes.
  - Vs. AI data, magnitude of 2nd derivative of loss is much **smaller**.
  - Better HPO via **AI designed for SE**.
- 2026: FSE’26: BINGO effect
  arxiv.org/pdf/2512.13524
  - An even greater simplification.
- 2027: **Rashomon** sets (work in progress)
  - Potentially: an end to hyperparameter optimization

The Observation: SE Loss Landscapes are Smoother

Claim: SE model loss landscapes are **exploitably** smooth.



What "smooth" means (formally)

f is β-smooth if: $\frac{\|\nabla f(y) - \nabla f(x)\|}{\|y - x\|} \leq \beta$.

- SE data: mean β-smoothness ≈ 2.
- Standard AI (UCI) data: mean β-smoothness ≈ 5.

Why SE Data is Smooth: The Naturalness Argument

- Human code is surprisingly **repetitive** [Hindle et al. ICSE'12]
- Maintainability forces repeated use of idioms and patterns.
- Repetition ⇒ minimal variation between clusters.
- Minimal variation ⇒ flat loss **landscapes**.

The core insight

Maintainable code produces data with flat loss landscapes.

Effects of HPO

Smoothing the landscape.

Experiments with Hyper-parameters (not an exhaustive list)

Pre-processing (Basic)

- Normalization: $x \rightarrow \frac{x}{\|x\|}$
- Standardization: $x \rightarrow \frac{x-\mu_x}{\sigma_x}$
- Min-max scaling: $x \rightarrow \frac{x-\min x}{\max x-\min x}$
- Max-abs scaling: $x \rightarrow \frac{x}{\max |x|}$
- Robust scaling: $x \rightarrow \frac{x-P_{50}(x)}{P_{75}(x)-P_{25}(x)}$

Pre-processing (Advanced)

- SMOTE: Synthetic minority samples via interpolation.
- Fuzzy sampling: Allocates samples across layers ($\frac{1}{2^i}$) to rebalance.
- Semi-supervised labeling: Label deletion and k-NN rebuild.

Neural Networks

- Feedforward: 3–20 layers, 2–6 units/layer.
- Deep Learners: More control parameters, e.g., topology, activation functions.

Traditional Classifiers

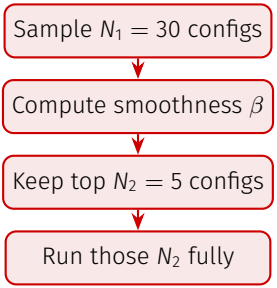
- Neighbor classifiers: (a) Distance measure, (b) neighbor count (k), (c) combination method, (d) pre-clustering.
- Random forests: Number of trees, features per tree, polling method (majority or weighted).
- Logistic regression: (a) Regularization type (L1, L2, elastic net), (b) strength ($C \in \{0.1, 1, 10\}$).

The Archive: 23 Datasets across 30 Years

Dataset	#Trn	#Tst	Feat
Defect prediction			
camel	1819	442	20
ivy	352	352	20
log4j	244	205	20
synapse	379	256	20
velocity	410	229	20
xalan	2411	909	20
Static code warnings			
ant	28	7	329
cassandra	13	4	265
commons	17	5	127
derby	366	92	292
jmeter	14	4	287
lucene-solr	24	6	278
tomcat	147	37	284

Dataset	#Trn	#Tst	Feat
Issue Lifetime Prediction			
Eclipse	44545	25459	12
Firefox	44800	25201	12
Chromium	44801	25200	12
Bayesmark (from NuerIPS'20)			
breast	455	114	30
digits	1438	359	64
iris	120	30	4
wine	142	36	13
diabetes	354	88	10

The Algorithm: SMOOTHIE



Finding 1 (RQ1): SMOOTHIE Wins on SE Data

- **Static warnings:** SMOOTHIE **wins** 4, ties 3.
- **Issue lifetime:** Ties best in all 3 datasets.
- **Defect prediction:** All methods tie (low **complexity**).
- **Data from AI:** On NuerIPS challenge data, no win.

RQ1 Answer

SMOOTHIE outperforms or matches SOTA on SE-specific data.

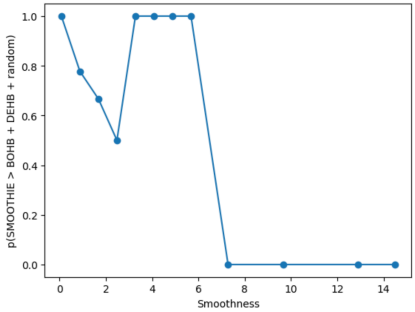


Fig. 7: Probability of SMOOTHIE outperforming (or match

Finding 2 (RQ2): SMOOTHIE is Much Faster

- Computing β (Get-Smoothness) is **negligible**.
- End-to-end, SMOOTHIE is 4.7× faster than BOHB.
- End-to-end, SMOOTHIE is 8.3× **faster** than DEHB.

RQ2 Answer

SMOOTHIE achieves SOTA accuracy 4–8× faster.

The End

- All you LLM users– you are missing what **matters**.
- SE-AI $\neq$ standard AI.
- And you can **leverage** that.
- Questions? Comments?