Limits To (Machine) Learning

Chen, Kelly, and Malamud

Discussion by Svetlana Bryzgalova

ABFER, Singapore 2026

How much true predictability is there?

What the paper does

Standard out-of-sample R^2 *underestimates* true predictability, because models overfit in-sample.

Key idea: if we know the structure of the estimator, we can find out how much it overfits, Limits to Learning Gain (LLG), $\hat{\mathcal{L}}$, and get a lower bound on true predictability.

Key result: if an estimator achieves R^2_{OOS} and has overfitting loss $\hat{\mathcal{L}}$, then the true maximum R^2 satisfies:

$$R^{2*} \geq \frac{R^2_{\text{OOS}} + \hat{\mathcal{L}}}{1 + \hat{\mathcal{L}}}$$

The worse is estimator OOS *relative to its overfitting capacity*, the higher true predictability.

Why it matters

Predictability literature has been stuck: models perform well in-sample R^2 but fail OOS.

CKM show that poor OOS performance could still imply high true predictability.

The logic: A complex estimator that overfits (high $\hat{\mathcal{L}}$) but still achieves some OOS fit ($R^2_{\text{OOS}} > -\hat{\mathcal{L}}$) means the signal *must* be large enough to reveal at least some learning.

Empirics:

- 14 Goyal-Welch variables using random-feature ridge estimators
- Lower bounds on true R^{*2}: from 0% to 67%.

The LLG Bound: Mechanics

Setup

Observing signals $S = (S_1, \ldots, S_T)$ and outcomes $y = (y_1, \ldots, y_T)$ in-sample, we predict y_{T+1} given S_{T+1} out-of-sample:

$$\hat{f}_{T+1} = \sum_{t=1}^{T} K_t(S_{T+1}, S)\, y_t$$

where the kernel weights K_t depend on all signals but **not on outcomes** y: "linear-in-y" estimator.

The Limits to Learning Gain:

$$\hat{\mathcal{L}} = \frac{1}{T_{\text{OOS}}} \sum_{\tau} \sum_{t=1}^{T} K_t(S_\tau, S)^2$$

Large $\hat{\mathcal{L}}$ means high sensitivity to each training observation.

Intuition for the bound

Decompose an estimator's expected performance:

$$\underbrace{\mathbb{E}[R^2_{\text{OOS}}]}_{\text{what we observe}} = \underbrace{R^{2*}}_{\text{true signal}} - \underbrace{\hat{\mathcal{L}}\,(1 - R^{2*})}_{\text{cost of overfitting noise}}$$

The key insight: even when $R^2_{\text{OOS}} < 0$ (model fails OOS), if $R^2_{\text{OOS}} > -\hat{\mathcal{L}}$, the bound implies $R^{2*} > 0$.

What makes $\hat{\mathcal{L}}$ large?

- Many features relative to observations (P/T)
- Weak regularization (small ridge penalty z)
- High correlation in features

Key property: The bound is valid for *any* estimator linear in y.

Key empirical finding

Headline results

Applying the bound to 14 Welch–Goyal variables:

- **Dividend-price ratio (dp):** $R^{2*} \geq 33\%$ (ReLU).
- **Dividend payout (de):** $R^{2*} \geq 67\%$ (tanh), the highest bound in the paper.
- **Earnings-price ratio (ep):** $R^{2*} \geq 38\%$ (ReLU).
- **Market excess return (retx):** $R^{2*} \geq 19\%$ (ReLU).

These are striking claims: the bound implies predictability far exceeding what any published model achieves out-of-sample.

Why it matters

The bounds are much larger than the best OOS R^2 achieved by any model:

Variable	Bound	Best model	Ratio
`dp`	33%	2%	17×
`retx`	19%	2%	10×
`lty`	27%	1%	27×
`bm`	23%	0%	∞
`de`	67%	42%	1.6×

Either:

- there exist dramatically better estimators waiting to be discovered, or
- something about the methodology may inflate the bound.

Key points of the discussion

My discussion focuses on

- What exactly is the object of prediction (economic value)?
- What affects the bound?
- What does the bound cover, and what modern ML actually does?

What are we predicting?

Data transformation

The LLG bound applies to the **processed** series, not original data. What does that series represent?

Step 1. Rolling standardize: $\tilde{X}_{i,t} = (X_{i,t} - \hat{\mu}_{t-1})/\hat{\sigma}_{t-1}$ — removes slow-moving level & scale

Step 2. Clip at $[-3, 3]$ stdev — removes tail events

Step 3. Take AR(1) residual: $\tilde{y}_t = \tilde{X}_t - \hat{\rho}\,\tilde{X}_{t-1}$ — removes persistent component

Data transformation

The LLG bound applies to the **processed** series, not original data. What does that series represent?

Step 1. Rolling standardize: $\tilde{X}_{i,t} = (X_{i,t} - \hat{\mu}_{t-1})/\hat{\sigma}_{t-1}$ — removes slow-moving level & scale

Step 2. Clip at $[-3, 3]$ stdev — removes tail events

Step 3. Take AR(1) residual: $\tilde{y}_t = \tilde{X}_t - \hat{\rho}\,\tilde{X}_{t-1}$ — removes persistent component

Numerical example: Dividend-Price ratio (dp)

Raw dp: monthly AR(1) $\rho = 0.98$ (GW data, 1930–2024). Fraction of variance in the innovation:

$$\frac{\mathrm{Var}(u)}{\mathrm{Var}(x)} = 1 - \rho^2 = 1 - 0.98^2 \approx \mathbf{0.04} \;\; (\mathbf{4\%})$$

Data processing retains ∼4% of dp's total variance.

Paper reports: $R^{2*} \geq 33\%$ for dp.

The *innovation* of rolling-standardized dp is 33% predictable from the signal cross-section.

Predicting 1/3 of only 4% of variation seems marginal.

Maybe it matters in some applications, but the paper needs to explain why.

Persistence and economic value of predictability

Green = large impact, Red = innovation $\neq$ level, Yellow = mixed, Gray = no difference.

Var	Full Name	ρ	tanh	ReLU	Best	Object predicted after Procedure 1	Economic value
`retx`	Excess market return	0.08	9%	19%	2%	$\approx$ vol-standardized returns	Yes
`dp`	Log dividend–price ratio	0.98	24%	33%	2%	Innovation of dp: only 4% of total dp variance survives	No
`ep`	Log earnings–price ratio	0.99	26%	38%	15%	Innovation of ep	Partial
`de`	Log dividend payout ratio	0.98	67%	41%	42%	Payout ratio innov.	Partial
`bm`	Book-to-market ratio	0.99	23%	17%	0%	Innovation of bm: only 3% of variance	No
`svar`	Stock variance (realized)	0.58	17%	0%	0%	Volatility innov + tanh vs ReLU disagree (17% vs 0%)	Mixed
`ntis`	Net equity expansion	0.98	0%	0%	1%	no predictability	n/a
`tbl`	Treasury bill rate	0.99	10%	0%	10%	$\approx \Delta$(T-bill rate), very tight bound	Yes
`lty`	Long-term gov't yield	1.00	27%	15%	1%	$\approx \Delta$(long yield)	Mixed
`ltr`	Long-term bond return	0.04	7%	0%	1%	$\approx$ vol-adjusted bond returns	Yes
`tms`	Term spread (lty $-$ tbl)	0.96	0%	0%	8%	Bound = 0% but models find 8% — **bound misses signal**	**Missing signal**
`dfy`	Default yield spread	0.97	27%	4%	14%	Δ(credit spread) — **tanh/ReLU disagree**	Mixed
`dfr`	Default return spread	-0.10	0%	2%	3%	$\approx$ raw default return spread (low persistence)	n/a
`infl`	Inflation (CPI)	0.50	0%	0%	0%	Inflation surprises — no predictability	n/a

Lots of really persistent variables, where the bound applies to innovations of 3–5% of total variance.

Why such a massive disagreement between ReLU and tanh?

"$R^{2*} \geq 33\%$ for dp" is not what it sounds like

The decomposition

$x_t = \rho\, x_{t-1} + u_t$ with $\rho \approx 0.98$.
After data transformation, the target $\approx u_t$:

$$\mathrm{Var}(u) = (1 - \rho^2)\,\mathrm{Var}(x)$$

If R^{2*} of innovation $= 50\%$, share of *total* x variance explained:

$$50\% \times (1 - \rho^2) = 50\% \times 0.04 = \mathbf{2\%}$$

Implied share of total variance

AR(1) coefficients from GW monthly data.

Var	ρ	$1-\rho^2$	Bound	Total
dp	0.98	3.5%	33%	1.2%
ep	0.99	2.3%	38%	0.9%
de	0.98	4.9%	67%	3.3%
bm	0.99	2.7%	23%	0.6%
lty	1.00	0.7%	27%	0.2%
dfy	0.97	5.0%	27%	1.4%

"Total" = Bound $\times$ $(1-\rho^2)$: share of total variance.

The missing step

The paper does not provide:

$$R^{2*}(\text{processed}) \longrightarrow R^{2*}(\text{raw})$$

This mapping depends on ρ, the rolling window, and clipping. Without it, the bounds cannot be interpreted as claims about economically relevant raw quantities.

Are outliers harmful or useful?

With a 36-month rolling window, the $\pm 3\sigma$ boundaries are time-varying.

What gets clipped (returns, 1926–2024)

25 observations clipped (2.2% of sample):

- **Great Depression** (1929–33): 6 months, including Sep 1931 (-4.5σ)
- **Oct 1987** (Black Monday): -5.6σ
- **Aug 1998** (LTCM/Russia): -4.8σ
- **2008 crisis**: 4 months clipped (Jan, Jun, Sep, Oct), with Oct 2008 -5.9σ.
- **Mar 2020** (Covid): -3.8σ; **Apr 2020** (recovery): $+3.2\sigma$

These are largely recessions

Why this matters: predictability in the tails

- **Henkel, Martin, Nardari (2011, JFE):** $R^2 \approx 15\%$ during NBER recessions vs. $< 1\%$ during expansions, using the same GW predictors.
- **Rapach, Strauss, Zhou (2010, RFS):** Forecast combination gains are all in recessions
- **Meligkotsidou et al. (2014):** Quantile regressions show predictors are significant in both tails but insignificant at the median.

Implication: For returns, clipping is not about removing 2% of "noise"; it is removing 2% of the observations that carry a disproportionate share of the predictive signal.

What affects the bound?

The LLG bound depends on signal geometry, not on predictability

(Theorem 2): $\hat{\mathcal{L}} = \frac{1}{T_{\text{OOS}}}\text{tr}(\hat{\mathcal{K}}'\hat{\mathcal{K}})$ depends **entirely on the signals** S, not on the outcomes y. You can compute $\hat{\mathcal{L}}$ without ever looking at what you are predicting.

The thought experiment

Take the 13 GW variables and compute 20,000 random features. Now predict three different targets:

a) $y =$ market excess returns

b) $y =$ a random permutation of returns

c) $y =$ i.i.d. Gaussian noise

All three produce **exactly the same** $\hat{\mathcal{L}}$, but will differ in R^2_{OOS}:

a): $R^2_{\text{OOS}} < 0$ (some signal, lots of overfitting), and in (b),(c) $R^2_{\text{OOS}} \approx -\hat{\mathcal{L}}$ (pure overfitting, no signal)

The bound $R^{2*} \geq \frac{R^2_{\text{OOS}}+\hat{\mathcal{L}}}{1+\hat{\mathcal{L}}}$ gives ≈ 0 in (b) and (c), and a positive number in (a). So the bound *does* distinguish signal from noise, **but only through the numerator** $R^2_{\text{OOS}} + \hat{\mathcal{L}}$, a small difference of two large numbers.

Stability of the bound

The arithmetic of the bound

The numerator is $R^2_{\text{OOS}} + \hat{\mathcal{L}}$. In the high dim regime ($P/T \approx 28$), both quantities are large:

Quantity	Typical scale
$\hat{\mathcal{L}}$	100–300%
R^2_{OOS}	-80% to -250%
$R^2_{\text{OOS}} + \hat{\mathcal{L}}$	5–50%

A 5% change to either R^2_{OOS} or $\hat{\mathcal{L}}$ can shift the bound by 50% or more of its value.

Example: If $\hat{\mathcal{L}} = 2.0$ and $R^2_{\text{OOS}} = -1.70$:

$$\text{Bound} = \frac{-1.70 + 2.0}{1 + 2.0} = \frac{0.30}{3.0} = 10\%$$

If R^2_{OOS} shifts to -1.80 (a 6% change):

$$\text{Bound} = \frac{0.20}{3.0} = 6.7\% \quad (\textbf{33\% drop})$$

What affects the measurement

Pre-processing choices shift R^2_{OOS} by small amounts relative to $\hat{\mathcal{L}}$, but these map to large changes in the bound:

- Clipping: clipping at ± 2.5 vs. ± 3 vs. ± 3.5 (crash months carry the most signal).
- Rolling window length.
- Activation function: tanh vs. ReLU produces different $\hat{\mathcal{L}}$ and different R^2_{OOS} for the same data. For example, for stock variance *tanh* gives 17%, *ReLU* gives 0%.
- Random seed: W_k's are drawn randomly, and different draws produce different features, different $\hat{\mathcal{L}}$, and different R^2_{OOS}.

How the bound is constructed

The reported bounds come from **ridge regression on random nonlinear features**:

Step 1: Generate random features

For original GW predictors $X_t \in \mathbb{R}^d$ ($d = 13$), draw $P = 20{,}000$ random weight vectors $W_k \overset{\text{iid}}{\sim} \mathcal{N}(0, I_{d\times d})$. For each k and t:

$$S_{k,t} \;=\; g(W_k' X_t)\,, \quad g \in \{\tanh,\ \text{ReLU}\}$$

where *tanh* is hyperbolic tangent, and ReLu is rectified linear unit, max(0,x).

Each time period, we get an augmented feature vector $S_t \in \mathbb{R}^{20000}$ instead of $X_t \in \mathbb{R}^{13}$.

Step 2: Ridge regression on random features

Run ridge regression of y on S: small shrinkage maximizes the LLG by allowing the estimator to overfit noise aggressively. Compute the LLG.

Why tanh $\neq$ ReLU:

Activation determines the implicit kernel being approximated (Rahimi–Recht 2007):

- $\tanh(x)$ saturates at ± 1: output is bounded, less sensitive to outliers
- $\text{ReLU}(x) = \max(x, 0)$: grows without bound, captures tails, but many features are 0.

Different kernels $\Rightarrow$ different $\hat{\Psi}$ $\Rightarrow$ different $\hat{\mathcal{L}}$ $\Rightarrow$ **different bounds on the same R^2_***. Bounds can disagree dramatically.

Two bounds are estimating different quantities. The true R^2_* is above both, but neither is necessarily close to it.

The bound depends on free parameters: g, P, and z

The tanh/ReLU bounds depend on the activation function g, the number of random features P, and the regularization z. The paper reports the maximum over P for each g.

Activation dependence

Variable	tanh	ReLU	Ratio
svar	17%	0%	∞
dfy	27%	4%	6.8×
tbl	10%	0%	∞
lty	27%	15%	1.8×
dp	24%	33%	0.7×
de	67%	41%	1.6×

For svar and tbl, tanh finds predictability that ReLU misses entirely. For dp, the opposite.

Which bound to use?

What drives the disagreement

1. Tail sensitivity. ReLU preserves tail information ($\max(x, 0)$ is unbounded); tanh compresses it ($\tanh(x) \to \pm 1$).

2. Sparsity of ReLU features. Half of ReLU features are exactly zero (when $W_k' X_t < 0$). The effective dimensionality is lower, producing a different $\hat{\Psi}$ and hence a different LLG.

3. Kernel approximation quality. As $P \to \infty$, random features converge to a specific kernel (arc-cosine for ReLU, a smooth kernel for tanh).

Bound is a function of kernel, not just X.

Where does the bound apply?

What does the bound cover?

$$\hat{f}_T \;=\; K(S_T,\, S)\cdot y \;=\; \sum_{t=1}^{T} K_t(S_T,\, S_1, \ldots, S_T)\, y_t$$

The **kernel weights** K_t depend on signals S only, **not on outcomes** y. Once signals are fixed, the map $y \mapsto \hat{f}_T$ is linear. The feature map inside K can be arbitrarily nonlinear in S.

OLS + Ridge regression

$\hat{f}_T = S_T' \,(\hat{\Psi} + zI)^{-1}\, \frac{1}{T} S'\, y$

Kernel: $K_t = S_T'(\hat{\Psi} + zI)^{-1} S_t / T$.
$\hat{\Psi} = S'S/T$ depends only on signals.

Linear in both S and y.

Kernel ridge regression

$\hat{f}_T = k(S_T, S)\,(zI + k(S,S))^{-1}\, y$

$k(\cdot,\cdot)$ can be polynomial, etc.
Highly nonlinear in signals, still a weighted sum of y_t's.

Random features / kitchen sinks

Generate $\phi(S_t) = \sigma(WS_t + b)$ with random W, b, then ridge on ϕ: $\hat{f}_T = \phi(S_T)'\,(\hat{\Psi}_\phi + zI)^{-1}\, \frac{1}{T}\Phi'\, y$

Features are nonlinear but *fixed* (independent of y).

Approximates kernel ridge for a large number of features.

PCA

Project S onto top-k PCs of $S'S$, then OLS.
Kernel weights are from the signal cov. structure.

Widely used in empirical finance (e.g., Kelly–Pruitt 2015).

Linear in y by construction.

OLS vs virtue of complexity

The bound applies to OLS too: $y_{t+1} = \alpha + \beta\,\mathrm{dp}_t + \varepsilon_{t+1}$

OLS is linear in y:

$$\hat{y}_\tau = X'_\tau \underbrace{(X'X)^{-1}X'}_{K \in \mathbb{R}^{2\times T}} y$$

With $d = 2$ (intercept + dp) and $T \approx 720$:

$$\hat{\mathcal{L}}_{\text{OLS}} \approx \frac{d}{T} \approx \frac{2}{720} \approx \mathbf{0.003}$$

The LLG is negligible because OLS has only 2 parameters, so it cannot overfit enough to generate useful information.

$$R^{2*} \geq \frac{R^2_{\text{OOS}} + 0.003}{1.003} \approx R^2_{\text{OOS}}$$

If $R^2_{\text{OOS}} \approx 0\%$, the bound says $R^{2*} \geq 0\%$, which is **true but uninformative**.

Random features: 20,000 nonlinear transforms of all 13 variables

The paper's estimator: $S_{k,t} = g(W'_k X_t)$, $P = 20{,}000$, ridge with $z_{\text{ref}} = 0.01$.

With $P/T \approx 28$:

$$\hat{\mathcal{L}}_{\text{RF}} \sim \mathbf{1.0\text{–}3.0} \quad (100\text{–}300\%)$$

The estimator overfits massively, producing $R^2_{\text{OOS}} \ll 0$, but the LLG is large enough that the bound is non-trivial:

$$R^{2*} \geq \frac{R^2_{\text{OOS}} + \hat{\mathcal{L}}}{1 + \hat{\mathcal{L}}} \approx \mathbf{19\%} \text{ (for returns)}$$

The virtue of complexity

What it does not cover

Proposition 4: A DNN with parameters θ trained by gradient descent satisfies, as width $\to \infty$:

$$f(S_t;\, \theta_{\text{trained}}) \;\approx\; f(S_t;\, \theta_0) \;+\; \underbrace{\nabla_\theta f(S_t;\, \theta_0)}_{\text{feature map (fixed at init)}}{}' (\theta_{\text{trained}} - \theta_0)$$

The gradient $\nabla_\theta f(S_t; \theta_0)$ depends only on signals and initialization, **not on** y. So the trained network is kernel ridge with the **Neural Tangent Kernel**: $K^{\text{NTK}}(S_i, S_j) = \nabla_\theta f(S_i; \theta_0)' \nabla_\theta f(S_j; \theta_0)$.

1. Finite-width networks with feature learning. When the width is finite, and the network learns representations (features adapt to y). The kernel is y-dependent.

2. Gradient boosted trees / XGBoost. Each tree is fit to residuals from prior trees; learners depend on y through the residual sequence.

3. Random forests. Split decisions in each tree depend on y, making the partition of feature space y-dependent.

4. Any method with y-dependent hyperparameters. Cross-validated ridge (where z is chosen to minimize validation error): the kernel depends on y through the selected z.

5. Neural nets with adaptive early stopping. Stopping rule based on validation loss makes the effective number of gradient steps y-dependent.

6. Transformers. Self-attention weights are y-dependent in a way that makes the output nonlinear in y.

The bound seems tightest where we need it least

The tension

The LLG bound is *exact* for kernel ridge regression. It is *approximate* for neural networks, relying on linearization:

- Width $\to \infty$ (lazy training regime)
- Learning rate scaled as 1/width
- No feature learning

But neural networks are interesting *precisely because* they learn features, i.e., they operate outside the NTK regime.

Finite-width networks in the "rich" regime likely outperform any kernel method (Yang & Hu 2021, Ba et al 2022).

Empirical evidence of the gap

Table 1 provides indirect evidence:

- The **recursive ridge** (a linear-in-y method with data-driven z) is the best-performing model for most variables. The gap between the LLG bound and recursive ridge is moderate (1.5–2$\times$) for `de`, `tbl`, `dfy`.
- How tight is the bound and what does this depend on?
- Would be nice to report results for boosted trees, random forests, etc, although they are *outside* the bound's scope.

"Limits to Kernel Methods" - all the more reason to use advanced ML

What methods are currently used?

Green = covered by LLG, Red = not covered.

Paper	Journal	Best method(s)	Comment	LLG applies?
Gu, Kelly, Xiu (2020)	RFS	NN3 (32-16-8), GBRT	Deep NN with feature learning + boosted trees dominate ridge/PCR	No
Gu, Kelly, Xiu (2021)	JoE	Conditional autoencoder	Deep encoder–decoder learns latent factors; multi-layer, trained end-to-end	No
Kelly, Pruitt, Su (2019)	JFE	IPCA	Instrumented PCA; linear in y by construction	Yes
Kelly, Malamud, Zhou (2024)	JF	Random features + ridge	Same construction as CKM's bound; linear in y	Yes
Kozak, Nagel, Santosh (2020)	JFE	Ridge on PCA factors	Shrinkage estimator; linear in y	Yes
Chen, Pelger, Zhu (2024)	MS	GAN + LSTM	Two adversarial deep networks + LSTM for macro states	No
Bryzgalova, Pelger, Zhu (2025)	JF	Decision trees	Splits selected to maximize SDF spanning; y-dependent partition	No
Feng, Polson, Xu (2020)	—	Deep NN (2–4 layers)	Deep feedforward networks for nonlinear factor models	No
Bianchi, Büchner, Tamoni (2021)	RFS	NN + extreme trees	Bond risk premia; NNs and trees outperform ridge	No
Leippold, Wang, Zhou (2022)	JFE	NN3, gradient boosting	GKX framework applied to China; deep methods dominate	No
Avramov, Cheng, Metzker (2023)	MS	Deep NN	Returns predictability under economic restrictions	No
Freyberger, Neuhierl, Weber (2020)	RFS	Adaptive group LASSO	Nonparametric basis with data-driven selection	No
Cong, Tang, Wang, Zhang (2022)	—	Transformer + RL	Cross-asset attention; self-attention is data-dependent	No

Note: this implies that, at least asymptotically, the paper's bound is conservative.

Lower bound may be too low?

Literature, in a nutshell:

In *every comparative study* that tests both linear and nonlinear methods (GKX 2020, Bianchi et al. 2021, Leippold et al. 2022), **the best performing methods are deep networks and gradient boosted trees**.

They are not covered.

The field moves toward deeper architectures: ridge/LASSO (2015-era) $\rightarrow$ shallow NNs and trees (2018–2020) $\rightarrow$ GANs, autoencoders, and Transformers (2021–present).

To be clear:

1. The bound is valid and informative for the methods it covers. Ridge remains popular.

2. Kelly, Malamud, Zhou (2024) show that random features (inside the bound) achieve strong market-timing performance.

3. Avramov, Cheng, Metzker (2023) show that deep learning's advantage shrinks after accounting for transaction costs and excluding microcaps.

4. The LLG is a *lower bound*. Even if deep networks are outside its scope, the bound still provides a floor on R^2_* that any method, including deep ones, must exceed.

Tightness of the bound is key.

Conclusion

The LLG is a really elegant approach to measuring true predictability.

Main suggestions:

- provide a mapping from predictability gains in transformed data to those in the original variables
- discuss how data choices affect the bound (they often make it conservative)
- guide empirical researchers on the bound interpretation, tightness, and robustness

Next paper (hopefully): How to achieve the bound?