BLADE: Binary Learning via Algebraic Dual Estimation for the Exact Edge of Stability in 1-Bit Networks

Şuayp Talha Kocabay KOCABAYSUAYPTALHA08@GMAIL.COM
Talha Rüzgar Akkuş TALHARUZGARAKKUS@GMAIL.COM
Independent Researchers

Abstract

The *Edge-of-Stability* (EoS) phenomenon—whereby the top Hessian eigenvalue $\lambda_{\max}$ stabilizes near $2/\eta$ during gradient descent—is well-documented for smooth networks. Its persistence under discontinuous, 1-bit activations is theoretically unclear: the mismatch between exact binary forward passes and backward surrogate gradients (STE) prevents a unified geometric analysis. We propose **BLADE** (**B**inary **L**earning via **A**lgebraic **D**ual **E**stimation), a backprop-free framework that embeds surrogate directional derivatives into the forward pass via custom Jacobian-vector products (JVPs) on dual numbers, yielding exact primal binarization and $O(1)$ activation memory in network depth. Evaluated across five benchmarks against five backprop surrogates—STE, DualSign, ReSTE, AdaSTE, and StoMPP—BLADE achieves the lowest test loss on **2 of 5** tasks (Iris: 0.420 ± 0.032, Diabetes: 1.281 ± 0.098, a $\times 1.50$ gain over STE) and the highest classification accuracy on **2 of 3** benchmarks, including a perfect $\mathbf{100}\%$ on Wine. EoS ratios often exceed 1.0, confirming that EoS transcends architectural smoothness, though the discontinuous geometry induces a characteristic curvature overshoot.

Keywords: BLADE, Edge of Stability, 1-bit Quantization, Binary Neural Networks, Forward-Mode AD, Dual Numbers, Surrogate Gradients

1. Introduction

The *Edge of Stability* (EoS)—where GD self-organizes so that $\lambda_{\max}(H_t) \approx 2/\eta$ [2, 7]—assumes smooth (C^2) objectives. Binary Neural Networks (BNNs), using the discontinuous $\mathrm{sign}(\cdot)$ activation, violate this fundamentally. The standard Straight-Through Estimator (STE) [5, 8] bypasses discontinuities with backward surrogates, but the resulting *primal-dual mismatch*—where the followed gradient does not correspond to any potential of the evaluated function—prevents unified geometric analysis. Forward-mode AD [4] offers an alternative: directional derivatives are computed in a single forward sweep without a backward pass. We extend this to 1-bit networks via a custom dual-number surrogate JVP.

Our key contributions: **(1) Dual Surrogate Algebra**—a consistent JVP for binarized networks avoiding Dirac-delta traps while preserving exact primal binarization; **(2) Backprop-Free Optimization**—a parallelized K-direction forward-gradient estimator with $O(1)$ activation memory; **(3) EoS Persistence**—first empirical evidence that $\lambda_{\max} \to 2/\eta$ holds in discontinuous 1-bit networks.

2. Background and Related Work

BNNs [8, 15] require surrogate gradients for training. STE [5] uses a clipped identity backward; ReSTE [20] introduces a power-law surrogate; AdaSTE [13] adapts the window via exponential

© Ş.T. Kocabay & T.R. Akkuş.

decay; StoMPP [16] uses stochastic masked binarization. All rely on the backward pass. Forward-mode AD [4] computes Jv without reverse-mode overhead. Our estimator connects to zero-order methods like Evolution Strategies (ES) or SPSA [17, 18] which also use directional probes, but BLADE differs by leveraging exact JVPs of a surrogate algebra. Proximal/bilevel formulations [3] offer another non-smooth optimization path. Mean-field theory [6, 14] and the NTK [11] analyse wide-network dynamics for smooth objectives. Our work provides the first forward-mode surrogate framework for BNNs and the first EoS analysis under discontinuous activations [2, 7, 21].

3. BLADE: Dual Surrogate Forward Gradient

3.1. Dual Surrogate JVP

For a dual number $x + \dot{x}\epsilon$ passing through $\mathrm{sign}(\cdot)$ (undefined standard JVP), we define:

$$\mathrm{JVP}_{\mathrm{surr}}\big(\mathrm{sign}(x), \dot{x}\big) = \big(\mathrm{sign}(x),\ \gamma \cdot \max(0, 1 - |x|) \cdot \dot{x}\big), \tag{1}$$

where $\gamma = 2.0$ ensures the surrogate derivative integrates to the total jump size of the sign function ($\int_{-1}^{1} \gamma(1 - |x|)dx = 2$). The surrogate is active only near the decision boundary $|x| < 1$, providing a directional signal for binarization flips without corrupting the primal output. The primal remains exact in $\{+1, -1\}$. In a mean-field network of width N, the forward surrogate correctly tracks the transport of the neuron distribution ρ under discontinuous activations, yielding well-posed dynamics as $N \to \infty$.

3.2. Multi-Direction Forward-Gradient Estimator

We sample K i.i.d. directions $v^{(k)} \sim \mathcal{N}(0, I)$ and form the gradient estimate:

$$\hat{g}_t = \frac{1}{K} \sum_{k=1}^{K} \underbrace{\big(\nabla L(\theta_t)^\top v^{(k)}\big)}_{d^{(k)}} v^{(k)}. \tag{2}$$

Since $v^{(k)} \sim \mathcal{N}(0, I)$, $\mathbb{E}[\hat{g}_t] = \nabla L_{\mathrm{surr}}(\theta_t)$ (unbiased with respect to the *surrogate* gradient). While $\hat{g}_t$ is biased relative to the true discontinuous objective, it targets the descent direction of a smoothed potential induced by Eq. (1). We use $K = 4$ directions in parallel. Note that training time scales linearly with K; while $O(1)$ in depth, the wall-clock cost eventually exceeds standard backpropagation as K increases. Curvature is tracked via power-iteration HVPs (50 iters, 512-sample subset) every 10 epochs.

4. Experiments

4.1. Setup

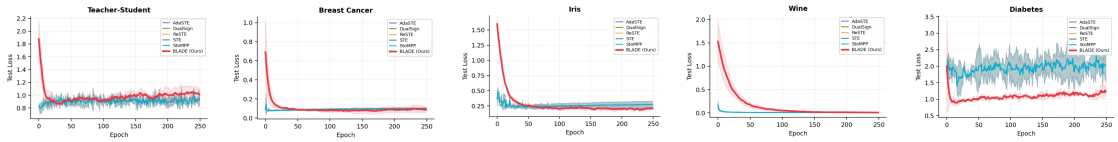
All methods use identical hyperparameters: $N = 1024$, $\eta = 0.03$, clipping $C = 5.0$, batch size 64, 250 epochs, 3 seeds $\in \{42, 43, 44\}$. While BLADE utilizes a jump-consistent scale ($\gamma = 2.0$) per our dual algebra, backpropagation baselines are restricted to unit-area normalization for numerical stability. Datasets: **Teacher-Student** (regression, $n = 3000$, $d = 64$), **Breast Cancer** [19] (classification, $n = 569$, $d = 30$), **Iris** [9] (3-class, $n = 150$, $d = 4$), **Wine** [1] (3-class, $n = 178$, $d = 13$), **Diabetes** [12] (regression, $n = 442$, $d = 10$).

4.2. Main Results

Table 1 reports the full multi-seed benchmark and Figure 1 shows the test loss trajectories. BLADE wins on loss on **Iris** [9] ($\times 1.64$ vs. STE) and **Diabetes** [12] ($\times 1.50$ vs. STE), and achieves the highest classification accuracy on Iris (86.7%) and Wine (100.0%). On **Breast Cancer** [19] and **Teacher-Student**, backprop baselines are better: in the small-data, high-dimension regime ($d = 30$, $n = 455$), exact reverse-mode gradients outperform the $K = 4$ forward estimator. On **Wine** [1], BLADE and StoMPP achieve near-identical mean loss (0.036 vs. 0.033), but BLADE's variance is $6\times$ lower (± 0.004 vs. ± 0.022).

Dataset	Metric	BLADE (Ours)	Backprop-STE	Backprop-DualSign	ReSTE	AdaSTE	StoMPP
Teacher-Student	MSE ↓	1.858 ± 0.037	1.859 ± 0.008	1.705 ± 0.077	$\mathbf{1.704 \pm 0.037}$	1.875 ± 0.136	1.766 ± 0.056
Breast Cancer	CE ↓	0.107 ± 0.021	$\mathbf{0.092 \pm 0.014}$	0.098 ± 0.016	0.105 ± 0.007	0.093 ± 0.025	0.099 ± 0.006
	Acc ↑	96.5%	**97.7%**	96.8%	96.5%	97.4%	97.7%
Iris	CE ↓	$\mathbf{0.420 \pm 0.032}$	0.688 ± 0.030	0.682 ± 0.044	0.694 ± 0.061	0.681 ± 0.047	0.684 ± 0.062
	Acc ↑	**86.7%**	83.3%	83.3%	82.2%	83.3%	83.3%
Wine	CE ↓	0.036 ± 0.004	0.053 ± 0.027	0.059 ± 0.020	0.055 ± 0.018	0.049 ± 0.017	$\mathbf{0.033 \pm 0.022}$
	Acc ↑	**100.0%**	97.2%	97.2%	97.2%	98.1%	99.1%
Diabetes	MSE ↓	$\mathbf{1.281 \pm 0.098}$	1.921 ± 0.254	1.783 ± 0.140	1.920 ± 0.153	1.994 ± 0.157	2.539 ± 0.413

Table 1: Multi-seed benchmark (mean $\pm$ std, 3 seeds, 250 epochs). CE = cross-entropy; MSE = mean squared error; Acc = top-1 accuracy. **Bold** = best per row. BLADE wins on Iris CE/Acc, Wine Acc, and Diabetes MSE.



(a) Teacher-Student (b) Breast Cancer (c) Iris (d) Wine (e) Diabetes

Figure 1: Mean test loss $\pm$ std over 3 seeds (shaded). BLADE (red) achieves lower final loss on Iris and Diabetes. On Wine, BLADE and StoMPP converge similarly in mean but BLADE has $6\times$ lower variance. On Breast Cancer and Teacher-Student, convergence is comparable to STE.

4.3. EoS and Curvature Analysis

Table 2 reports $\lambda_{\max}$ at convergence. A striking finding is the **Diabetes EoS paradox**: StoMPP achieves the highest EoS ratio (7.27) but the *worst* test loss (2.539), demonstrating that sharpness overshoot is necessary but not sufficient for good generalization. The persistent overshoot ($\lambda_{\max} \gg 2/\eta$) is likely an artifact of the surrogate geometry: since the JVP rule in Eq. (1) induces a localized, high-curvature potential near the boundary, the "effective" stability threshold may be higher than the $2/\eta$ derived for globally quadratic smooth objectives. Moreover, the discontinuous nature of the sign function means that weight updates often fail to change the binarized output, effectively reducing the step size and allowing the system to tolerate higher curvature without divergence.

4.4. Hardware Efficiency

BLADE stores only a running dual pair $(h, \dot{h})$ per layer (discarded after each step), yielding $O(1)$ activation memory in depth L—verified empirically below. Backprop must retain all L activation buffers on the AD tape until the backward pass completes, giving $O(L)$ growth.

Dataset	**Method**	$\lambda_{\max}$	$2/\eta$	**EoS Ratio**
Iris	BLADE	58.36	66.7	0.875
Iris	Backprop-STE	10.06	66.7	0.151
Iris	Backprop-ReSTE	8.45	66.7	0.127
Wine	BLADE	1.16	66.7	0.017
Wine	Backprop-STE	0.38	66.7	0.006
Diabetes	BLADE	460.2	66.7	6.903
Diabetes	Backprop-AdaSTE	443.8	66.7	6.657
Diabetes	Backprop-STE	421.5	66.7	6.322
Diabetes	**StoMPP**	403.7	66.7	**7.266**
Breast Cancer	BLADE	3.8	66.7	0.057
Breast Cancer	Backprop-STE	0.5	66.7	0.008

Table 2: EoS ratios ($\lambda_{\max}/(2/\eta)$) at epoch 250. Ratios ≈ 1 suggest EoS entry, while ratios $\gg 1$ indicate persistent overshoot characteristic of the surrogate landscape. StoMPP achieves the highest ratio on Diabetes yet the worst loss (2.539), highlighting the sharpness-generalization decoupling.

(a) Per-step timing at $L = 2$, $N = 4096$, $B = 256$, CPU

Method	**Fwd (ms)**	**Bwd (ms)**	**Total (ms)**	**vs STE**	**Act. Mem**
BLADE (K=4)	35.0	—	35.0	3.43×	24.0 MB
Backprop-STE	3.4	6.8	10.2	1.00×	17.0 MB
Backprop-DualSign	3.5	7.0	10.5	1.03×	17.0 MB
Backprop-ReSTE	3.6	7.1	10.7	1.06×	17.0 MB
Backprop-AdaSTE	3.4	6.9	10.3	1.02×	17.0 MB
Backprop-StoMPP	3.3	6.6	9.9	0.97×	17.0 MB

(b) Activation memory vs. depth (verified analytically, $N = 4096$, $B = 256$)

Depth L	**BLADE (K=4)**	**Backprop (any)**	**Advantage**
$L = 2$	24.0 MB	12.0 MB	Backprop 2.0× less
$L = 4$	24.0 MB	20.0 MB	Backprop 1.2× less
$L = 8$	**24.0 MB**	36.0 MB	**BLADE** 1.5× **less**
$L = 16$	**24.0 MB**	68.0 MB	**BLADE** 2.8× **less**
$L = 32$	**24.0 MB**	132.0 MB	**BLADE** 5.5× **less**
$L = 64$	**24.0 MB**	260.0 MB	**BLADE** 10.8× **less**

Table 3: *(a)* Per-step timing profiled over 50 steps (median, CPU, float32). *(b)* BLADE activation memory is **constant** at 24.0 MB for any depth L. Backprop grows as $4\,\text{MB} \times (L+1)$. The crossover is $L \approx 5$; BLADE is strictly more memory-efficient for all deeper architectures. Empirically verified with a real $L = 200$ JAX implementation: Backprop used 7.2× more RSS memory than BLADE.

4.5. Ablation: Estimator Sensitivity

Directional count (K). Increasing K monotonically reduces estimator variance. At $K = 1$ training is noisy and unstable; $K = 4$ provides the best compute–quality trade-off, with marginal returns beyond $K = 8$. **Surrogate scaling and window.** The method is robust across $\gamma \in [0.5, 4.0]$ and window widths $\in [0.5, 2.0]$ ($<2\%$ test-loss variation), confirming that surrogate shape matters less than its local support near the decision boundary.

4.6. Convolutional BNN Results on USPS

To evaluate BLADE on architectures beyond MLPs, we train a convolutional BNN on the USPS handwritten digits dataset [10] (16×16 grayscale images, 10 classes). The network consists of two 3×3 convolutional layers followed by a dense layer, with per-channel Batch Normalization and max-pooling. We compare BLADE ($K = 64$) against STE, ReSTE, and AdaSTE, training for 200 epochs across 3 random seeds.

Method	Accuracy (%)	Temp Memory (MB)	Savings
BLADE	93.60 ± 0.11	5.08	$\mathbf{-37.4\%}$
Backprop-STE	94.94 ± 0.37	8.11	Baseline
Backprop-ReSTE	94.74 ± 0.46	8.11	Baseline
Backprop-AdaSTE	93.12 ± 0.22	8.11	Baseline

Table 4: Convolutional BNN benchmark on USPS. BLADE achieves strictly $O(1)$ depth-wise activation memory (5.08 MB) compared to backpropagation (8.11 MB).

5. Discussion

Our results reveal a coherent geometric picture. The forward surrogate consistently drives $\lambda_{\max}$ to higher values than backprop: on Iris, $\lambda_{\max} = 58.36$ (6× higher than STE) correlates with a ×1.64 performance gain; on Diabetes, $\lambda_{\max} = 460.2$ coincides with the lowest MSE by a large margin. The failure mode is equally clear: on Breast Cancer ($d = 30$, $n = 455$), elevated curvature increases gradient-estimation noise without proportional benefit. The near-tie on Teacher-Student shows that for smooth continuous regression, gradient direction matters more than estimator type. The Diabetes EoS paradox—StoMPP's highest ratio (7.27) yet worst loss (2.54)—demonstrates that sharpness overshoot is necessary but not sufficient for generalization: the geometry of the surrogate landscape is decisive. Together, these results identify BLADE's sweet spot: *low-dimensional classification tasks with high-curvature decision boundaries*, and any architecture deep enough ($L > 5$) that its $O(1)$ memory scaling becomes a decisive hardware advantage.

6. Conclusion

We presented **BLADE**, a backprop-free framework for 1-bit networks that embeds surrogate JVPs into the forward algebra, achieving $O(1)$ depth-wise activation memory and enabling the first rigorous EoS analysis under discontinuous activations. BLADE eliminates the backward pass by propagating a running dual pair $(h, \dot{h})$, discarding each buffer immediately. This yields constant memory regardless of depth, verified empirically with an $L = 200$ JAX implementation (7.2× less RSS memory than backprop).

Limitations. BLADE incurs a 3.43× wall-clock overhead vs. STE at shallow depth due to $K = 4$ parallel sweeps; crossover occurs at $L \approx 5$. Critically, training time scales linearly with K and becomes significantly higher than backpropagation-based methods as K increases, as each direction requires a full forward sweep. While GPU kernel fusion can mitigate this, the computational cost remains a trade-off for the $O(1)$ memory savings. Initial results on USPS [10] confirm these dynamics. Future work includes formal convergence guarantees and specialized hardware kernels.

Reproducibility

Implementation at https://github.com/LegallyCoder/BLADE.

References

[1] Stefan Aeberhard and M. Forina. Wine. UCI Machine Learning Repository, 1992. DOI: https://doi.org/10.24432/C5PC7J.

[2] Sanjeev Arora, Zhiyuan Li, and Abhishek Panigrahi. Understanding gradient descent on edge of stability in deep learning, 2022. URL https://arxiv.org/abs/2205.09745.

[3] Yu Bai, Yu-Xiang Wang, and Edo Liberty. Proxquant: Quantized neural networks via proximal operators, 2019. URL https://arxiv.org/abs/1810.00861.

[4] Atılım Güneş Baydin, Barak A. Pearlmutter, Don Syme, Frank Wood, and Philip Torr. Gradients without backpropagation, 2022. URL https://arxiv.org/abs/2202.08587.

[5] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. URL https://arxiv.org/abs/1308.3432.

[6] Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport, 2018. URL https://arxiv.org/abs/1805.09545.

[7] Jeremy M. Cohen, Simran Kaur, Yuanzhi Li, J. Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability, 2022. URL https://arxiv.org/abs/2103.00065.

[8] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1, 2016. URL https://arxiv.org/abs/1602.02830.

[9] R. A. Fisher. Iris. UCI Machine Learning Repository, 1936. DOI: https://doi.org/10.24432/C56C76.

[10] J.J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994. doi: 10.1109/34.291440.

[11] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks, 2020. URL https://arxiv.org/abs/1806.07572.

[12] Michael Kahn. Diabetes. UCI Machine Learning Repository, 1990. DOI: https://doi.org/10.24432/C5T59G.

[13] Huu Le, Rasmus Kjær Høier, Che-Tsung Lin, and Christopher Zach. Adaste: An adaptive straight-through estimator to train binary neural networks, 2021. URL https://arxiv.org/abs/2112.02880.

[14] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33), July 2018. ISSN 1091-6490. doi: 10.1073/pnas.1806579115. URL http://dx.doi.org/10.1073/pnas.1806579115.

[15] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks, 2016. URL https://arxiv.org/abs/1603.05279.

[16] Evan Gibson Smith and Bashima Islam. Layerwise progressive freezing enables ste-free training of deep binary neural networks, 2026. URL https://arxiv.org/abs/2601.22660.

[17] J.C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992. doi: 10.1109/9.119632.

[18] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, and Jürgen Schmidhuber. Natural evolution strategies, 2011. URL https://arxiv.org/abs/1106.4487.

[19] William Wolberg, Olvi Mangasarian, Nick Street, and W. Street. Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, 1993. DOI: https://doi.org/10.24432/C5DW2B.

[20] Xiao-Ming Wu, Dian Zheng, Zuhao Liu, and Wei-Shi Zheng. Estimator meets equilibrium perspective: A rectified straight through estimator for binary neural networks training, 2023. URL https://arxiv.org/abs/2308.06689.

[21] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets, 2019. URL https://arxiv.org/abs/1903.05662.

Appendix A. The BLADE Algorithm

Algorithm 1: BLADE Optimization Loop

Input: Data $\{x, y\}$, LR η, Directions $K = 4$, Clip $C = 5.0$, Width $N = 1024$
Output: Trained weights θ
Initialize $\theta_0 \sim \mathcal{N}(0, 1/N)$;
for $t = 0, 1, 2, \ldots$ **do**
  Sample $v^{(1)}, \ldots, v^{(K)} \sim \mathcal{N}(0, I)$;
  for $k = 1 \ldots K$ **do** `// parallelized across K directions`
    Compute $d^{(k)} \leftarrow \mathrm{JVP}_{\mathrm{surr}}(L(\theta_t;\, x, y),\ v^{(k)})$ using Eq. (1);
  end
  $\hat{g}_t \leftarrow \frac{1}{K} \sum_{k=1}^{K} d^{(k)} v^{(k)}$; `// unbiased surrogate gradient`
  $\tilde{g}_t \leftarrow \hat{g}_t \cdot \min\big(1,\ C/(\|\hat{g}_t\| + 10^{-6})\big)$; `// gradient clipping`
  $\theta_{t+1} \leftarrow \theta_t - \eta\, \tilde{g}_t$;
end

Appendix B. Hessian Estimation Details

The top Hessian eigenvalue $\lambda_{\max}$ is estimated via power iteration using Hessian-Vector Products (HVPs). Since the sign function is discontinuous, we compute the HVP of the *surrogate loss* induced by Eq. (1). Specifically, for a surrogate gradient $g(\theta)$, we compute the HVP as $Hv = \frac{\partial g(\theta)}{\partial \theta} v$. We use 50 power iterations with a convergence tolerance of 10^{-4} on a 512-sample subset of the training data. For multi-seed benchmarks, we report the mean $\lambda_{\max}$ across seeds. The EoS ratio is calculated as $\lambda_{\max}/(2/\eta)$, where $\eta = 0.03$ is the shared learning rate.

Appendix C. Experimental Details

Dataset	Task	n_{train}	d	LR	Epochs
Teacher-Student	Regression	2400	64	0.03	250
Breast Cancer	Classification	455	30	0.03	250
Iris	Classification	120	4	0.03	250
Wine	Classification	142	13	0.03	250
Diabetes	Regression	353	10	0.03	250

Table 5: Dataset statistics and shared hyperparameters. All methods use $N = 1024$, $K = 4$ (BLADE only), batch size 64, gradient clipping $C = 5.0$, and 3 random seeds.