NEURIPS 2026 · POSTER

Beyond Flat Gossip: Tiered Gossip Learning for Scalable Collaborative AI

Atul Sharma, Kavindu Herath, Saurabh Bagchi, Chaoyue Liu, Somali Chaterji

Elmore Family School of Electrical and Computer Engineering, Purdue University

TIERED GOSSIP LEARNING

At scale, every topology pays somewhere

Federated learning

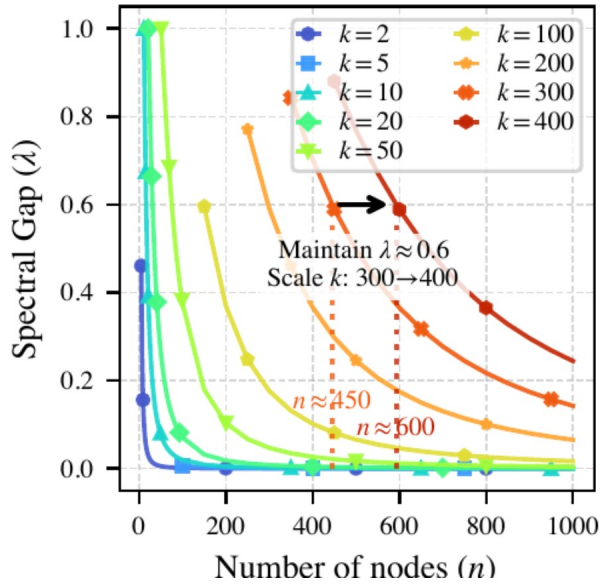
One server aggregates every update: congestion grows with clients, and it is a single point of failure.

Hierarchical FL

Edge servers spread the load, but each one can cut off a whole sub-tree, and the root bottleneck stays.

Peer-to-peer gossip

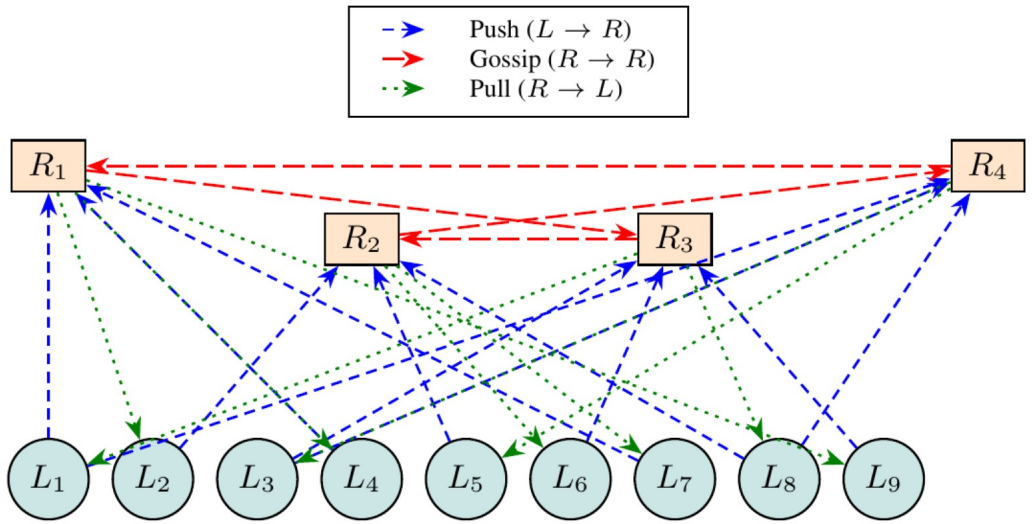
No server, but to keep mixing quality fixed as the network grows, every node must raise its degree.



Holding spectral gap λ≈0.6 as n grows 450→600 needs degree k 300→400.

THE IDEA

TGL: push, gossip, pull — every round



1 · PUSH b_{lr} relay in-degree

Relays average models from random leaves.

2 · GOSSIP b_{rr} relay out-degree

Relays mix with random relay peers.

3 · PULL b_{rl} leaf in-degree

Leaves average models from random relays.

DESIGN PRINCIPLES

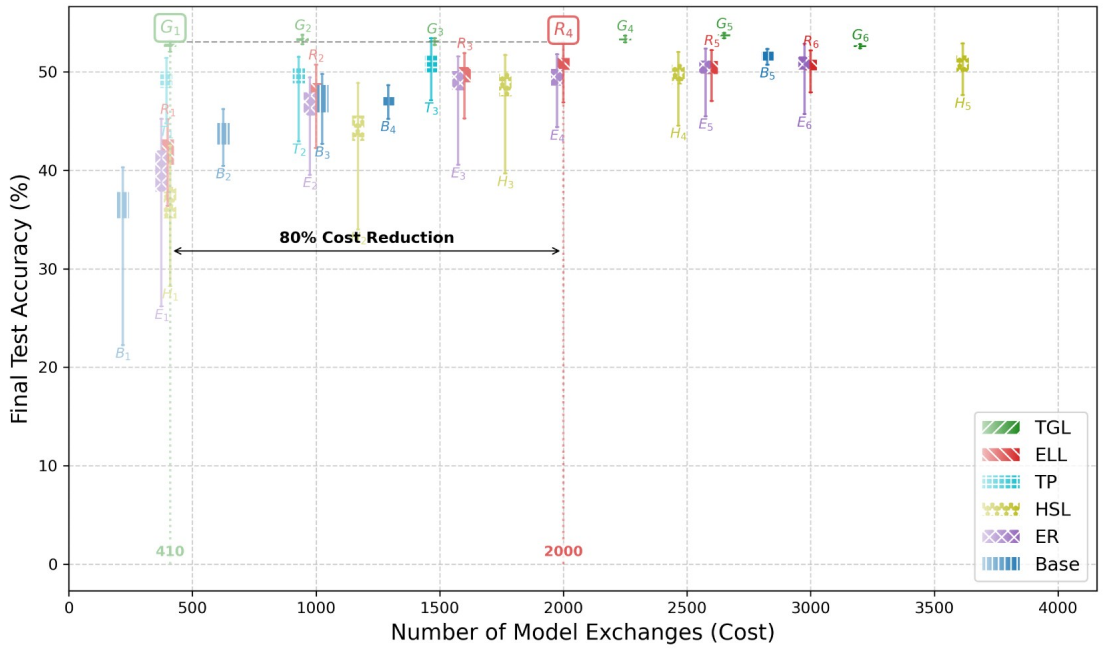
- **Hierarchy**
- **Asymmetry**
- **Decentralized top layer**
- **Random dynamic links**

$$\text{cost/round} = n_r\, b_{lr} + n_r\, b_{rr} + n_l\, b_{rl}$$

Decoupled budgets: leaf cost **can** stay fixed while relay budgets tune mixing.

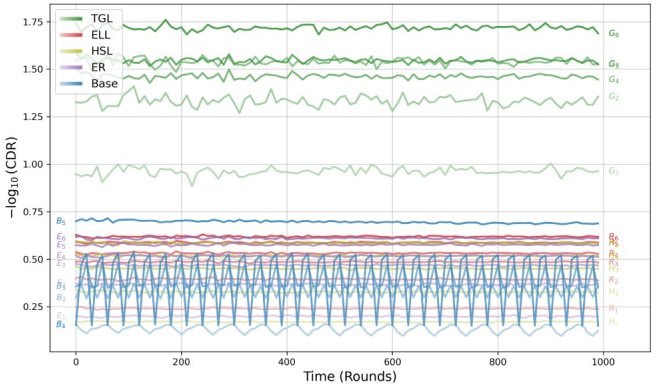
RESULTS

Sparse TGL configs compete with denser baselines



Accuracy vs. cost · CIFAR-10 · 200 leaves · non-iid (α = 0.5)

Config	Setup	Edges
TGL G1	15 relays · (10, 4, 1)	410
TGL G2	15 relays · (20, 3, 3)	945
ELL R4	degree k = 10	2000



Mixing efficiency (−log CDR, higher is better)

THEORY & FAULT TOLERANCE

Budgets set the mixing rate and absorb failures

Convergence

Under smoothness, bounded noise and heterogeneity, the rate is governed by

$$\beta_{TGL} = \beta_{lr}\,\beta_{rr}\,\beta_{rl}$$

$$\beta_{lr} = \frac{1}{b_{lr}}\left(1 - \frac{b_{lr}-1}{n_l - 1}\right)$$

$$\beta_{rr} = \frac{1}{b_{rr}}\left(1 - \left(1 - \frac{b_{rr}}{n_r - 1}\right)^{n_r}\right) - \frac{1}{n_r - 1}$$

$$\beta_{rl} = \frac{1}{b_{rl}}\left(1 - \frac{b_{rl}-1}{n_r - 1}\right)$$

Each bounds one stage's contraction of consensus distance, roughly 1 / budget.

Fault-tolerant variant (TGL-FT)

Vanilla TGL: pick any budgets to meet a cost. TGL-FT: given relay crash probability *p*, derive

Relays needed

$$n_r^{\min} = \left\lceil \frac{n_l\, b_{rl}}{C_{\max}\,(1-p)} \right\rceil$$

Relay capacity cap

$$c_{cap} = \left\lceil \frac{n_l\, b_{rl}}{n_r(1-p)}(1+\delta) \right\rceil$$

Leaf retries (95th pct.)

$$b_{rl}^{\max} = \left\lceil \frac{b_{rl}}{1-p} + 1.645\sqrt{\frac{b_{rl}\,p}{(1-p)^2}} \right\rceil$$

C_max: per-relay bandwidth ceiling. δ = 0.2: headroom for round-to-round load variance. A crash is just a timeout.

TAKEAWAYS

Mixes well, scales well, tolerates faults

Mixes well

Push, gossip, pull mixes more per edge than flat P2P: sparse TGL configs compete with dense baselines.

Scales well

Decoupled budgets: leaf cost can stay fixed as the network grows, while relays carry global mixing.

Fault tolerant

No central server, links resampled every round, and TGL-FT provisions for relay crashes.

Thank you — find us at the poster.