TIERED GOSSIP LEARNING COMMUNICATION-FRUGAL AND SCALABLE COLLABORATIVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern edge deployments require collaborative training schemes that avoid both the single-server bottleneck of federated learning (FL) and the high communication burden of peer-to-peer (P2P) systems. We propose Tiered Gossip Learning (TGL), a two-layer push–gossip–pull protocol that combines the fault tolerance of P2P training with the efficiency of hierarchical aggregation. In each round, device-level leaves push their models to a randomly selected set of relays; relays gossip among themselves; and each leaf then pulls and averages models from another random subset of relays. Unlike other hierarchical schemes, TGL is fully coordinator-free, with communication and aggregation decentralized across nodes. It achieves competitive accuracy while reducing per-round communication (model exchanges) by nearly two-thirds compared to flat P2P baselines on diverse datasets, including CIFAR-10, FEMNIST, and AG-News. We provide convergence guarantees for TGL under standard smoothness, bounded variance and heterogeneity assumptions, and show how its layered structure enables tunable consensus control at each stage. Altogether, TGL brings together the strengths of FL and P2P design, enabling robust, low-cost mixing enabling large scale collaborative learning.

1 INTRODUCTION

Collaborative training is increasingly vital for machine learning applications spanning edge devices, sensor networks, and distributed organizations. These settings demand decentralized solutions due to limited resources, privacy constraints, and heterogeneity in local (non-iid) data. As data ownership becomes increasingly localized, retaining data at the source has become essential. Federated Learning (FL) (McMahan et al., 2017; Yang et al., 2019; Kairouz et al., 2021) emerged as a practical response to these constraints, enabling clients to share a global model but train locally and periodically send model updates to a central server for aggregation. While FL is attractive for its simplicity and ease of client participation, it struggles to scale. Congestion at the server increases with more clients, slowing down training (Lian et al., 2017) and introducing a single point of failure. A central challenge, therefore, is to support *large-scale participation* without overwhelming any single node.

Hierarchical Federated Learning (HFL) (Liu et al., 2020; Abad et al., 2020) extends this idea by introducing intermediate edge servers beneath a single root server to distribute the aggregation load. Yet, HFL inherits, and often amplifies the limitations of FL: each added server introduces an additional point of failure, and the root server remains a bottleneck. Centralized coordination offers simpler orchestration at the cost of fault tolerance and long-term scalability, motivating decentralized alternatives. Although privacy and security concerns in centralized systems are beyond this paper's scope, such considerations have also contributed to the growing interest in fully decentralized paradigms.

Peer-to-Peer Learning (P2PL) (Lian et al., 2017; Koloskova et al., 2020; Kong et al., 2021) eliminates central servers entirely, distributing communication uniformly across all participating nodes. This enhances fault-tolerance and removes single points of failure, but introduces a new trade-off: maintaining strong model mixing requires higher node degrees. The quality of this mixing is governed by the spectral gap of the gossip matrix, which reflects how effectively informa-

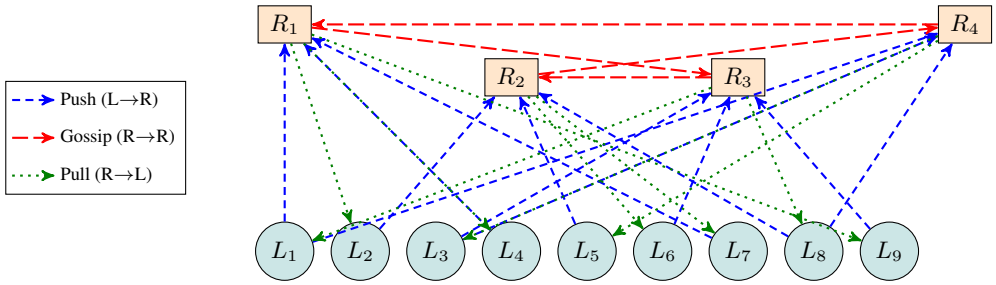

Figure 2: A snapshot of the Tiered Gossip Learning (TGL) network with 9 leaves and 4 relays using 25 directed edges, where connections dynamically change in each round, illustrating the three-stage communication process. In **Stage 1 (Leaf-to-Relay Push)**, relays aggregate models from randomly sampled leaves (shown as blue dashed lines). In **Stage 2 (Relay Gossip)**, each relay exchanges models with other relay (red dashed lines) and averages received models. In **Stage 3 (Relay-to-Leaf Pull)**, each leaf retrieves a model from randomly selected relay (green dotted lines). TGL's two-tier hierarchical design—featuring a decentralized relay layer atop a leaf layer with random dynamic connections enables scalable and fault-tolerant training while significantly reducing communication cost, a key reason behind its superior empirical performance.

tion flows through the network and controls the speed at which models reach consensus. Think of gossip as repeatedly *averaging* with your neighbors. The *spectral gap* measures how quickly the network *forgets where information started*. If the gap is large, every round mixes information well and the network rushes toward consensus; if the gap is small, information leaks slowly through narrow bridges or long chains. A larger spectral gap implies faster convergence under bounded heterogeneity, but preserving it becomes increasingly expensive as the network scales. As shown in Figure 1, when the number of nodes n is small, even a modest node degree k yields a sufficiently large spectral gap. However, *as n increases, flat P2P systems must raise k to preserve mixing quality*, leading to higher per-node and total communication. Prior work on exponential graphs (Ying et al., 2021) partially addresses this by scaling degree as $\mathcal{O}(\log n)$, but the communication burden still grows system-wide, limiting scalability.

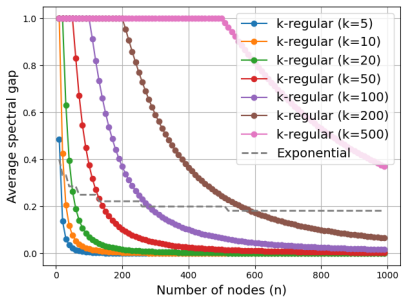

Figure 1: Spectral gap vs. number of nodes n Each curve fixes node degree k. For k-random regular the gap (higher is better) decreases as n grows, so maintaining the same mixing quality typically requires larger k. Exponential graphs, with effective degree $\Theta(\log n)$, retain larger gaps at large n. At higher k, curves plateau—once major bottlenecks are removed, each extra neighbor yields only small additional gains (*diminishing returns*).

Each prior approach concentrates the burden of scalability differently: FL violates the "no root" constraint and overloads a single aggregator as $n \uparrow$, HFL relaxes load at leaves but still concentrates risk/traffic at a root and adds failure points; flat P2P preserves decentralization but must increase k with n to maintain the same contraction ϵ, violating fixed B_ℓ.

We address this challenge with Tiered Gossip Learning (TGL), a hybrid design illustrated in Figure 2 that incorporates the key principles of *hierarchy, asymmetric load sharing, and a decentralized top layer with random, dynamic connections.* Leaves interact only with a small, randomly chosen subset of relays; relays gossip among themselves; and leaves then pull updates from another random relay subset. This distributes aggregation across decentralized relays, avoiding single-server bottlenecks while enabling more leaves to participate under fixed per-leaf degree. Spectrally, the two-layer process multiplies the push, gossip, and pull mixing matrices; although each matrix can be sparse, their product yields a denser effective mixing matrix that improves consensus without necessarily increasing leaf degrees. TGL thus operationalizes these core principles to enable scalable, fault-tolerant, and communication-efficient collaborative learning.

We summarize our key contributions as follows:

1. We introduce *Tiered Gossip Learning* (TGL), a two-layer push–gossip–pull protocol that synthesizes structural decentralization with hierarchical efficiency. TGL leverages a tiered architecture with asymmetric load sharing to achieve strong model mixing at low communication volume, while ensuring that no central coordination is required and all aggregation and communication remain decentralized. This design enhances fault tolerance and enables scalable participation without overloading any single node, addressing key limitations of prior hierarchical and decentralized approaches.
2. We prove convergence of TGL under standard assumptions of smoothness, bounded variance, and bounded heterogeneity. We also derive stage-wise and overall bounds on the expected consensus distance, expressed in terms of network parameters including node count and degree, highlighting how the tiered mixing structure improves global consensus with low node degrees and enables independent control of consensus at each layer.
3. We evaluate TGL on vision tasks: FEMNIST with a CNN and CIFAR-10 with a ResNet—and on a language task, AG News with a TinyTransformer, under non-iid data. We compare performance across varying per-round model exchange budgets. TGL consistently achieves better or comparable accuracy while using up to **3× less communication** than prior decentralized baselines, including EL-Local, Exponential, Base-$(k+1)$, and Erdős–Rényi topologies.

2 Background and Related Work

2.1 Related work

Related Work. The idea of fully decentralized collaborative learning was popularized by Lian et al. (2017), who showed that Peer-to-Peer Learning (P2PL) using Decentralized SGD (DSGD) can surpass Federated Learning (FL) in wall-clock time by eliminating the central server that easily gets congested. This sparked extensive research on decentralized optimization (Assran et al., 2019; Koloskova et al., 2020), with subsequent work focusing on accelerating convergence via algorithmic refinements (Yu et al., 2019; Yuan et al., 2021; Chen et al., 2021).

A key challenge in collaborative learning is the presence of non-iid data across clients. Under bounded heterogeneity, prior work has focused on strengthening consensus: either by reducing sensitivity to the communication graph (Tang et al., 2018; Li et al., 2019; Kong et al., 2021), or by designing sparse yet well-mixing topologies such as expanders and logarithmic-degree graphs (Nedić et al., 2018; Chow et al., 2016; Ying et al., 2021; Takezawa et al., 2023; Wang et al., 2019; Song et al., 2022). More recently, *Epidemic Learning* (EL) (De Vos et al., 2023) showed that dynamic random graphs can improve mixing more efficiently than fixed topologies. EL also highlighted the importance of relaxing the requirement for doubly stochastic mixing: while its Oracle variant relies on a coordinator to construct such matrices, the Local variant achieves full decentralization by operating with only row-stochastic weights. Our work continues in this direction, aiming to improve mixing efficiency under constrained communication without any centralized component.

When heterogeneity is large or unbounded, an alternative line of work has explored clustering strategies, where nodes preferentially interact with peers holding similar data or model updates (Ghosh et al., 2020; Sattler et al., 2020). Such methods mitigate the degradation in local performance that arises from enforcing a single global consensus, but generally require orchestration by a central scheduler, and thus remain structurally closer to FL. Some P2PL approaches (Onoszko et al., 2021; Li et al., 2022) have also incorporated clustering mechanisms for adaptive peer discovery.

Chaoyue: I don't quite understand the following sentence. Do you mean: "A complementary orthogonal direction of reducing communication cost is by limiting ..."? A complementary orthogonal direction reduces communication cost by limiting client participation per round. Methods such as Teleportation (Takezawa & Stich, 2025) and Plexus (de Vos et al., 2023) lower communication cost by activating only a subset of nodes per iteration, but at the cost of discarding many updates. Chaoyue: However, after a careful check of the theory of Teleportation (Takezawa & Stich, 2025), the method of Teleportation, contrary to its claim, actually has a degradation in its convergence rate, which theoretically needs more communication cost due to a larger number of iterations. Other

methods reduce the per-message size through compressed communication (Koloskova et al., 2019) or quantization (Chen et al., 2024). In contrast, our work targets scaling to larger node participation under bounded heterogeneity by improving mixing efficiency at limited communication volume, without relying on sampling or compression, and by relaxing the requirement for doubly stochastic matrices to eliminate central orchestration.

2.2 BACKGROUND

Consider a distributed learning setup with n_l nodes, each holding a private dataset $\mathcal{D}_i$. The nodes collaboratively train a global model by periodically exchanging updates. Each node initializes the same model $\mathbf{x}_0$ and follows a shared training routine. The local objective for node i is: $f^{(i)}(\mathbf{x}) := \mathbb{E}_{\xi\sim\mathcal{D}_i}[f(\mathbf{x},\xi)]$, ξ being a mini-batch sampled from $\mathcal{D}_i$. The global objective F minimizes the average local objectives across all n_l nodes:

$$\min_{\mathbf{x}} F(\mathbf{x}) = \min_{\mathbf{x}} \frac{1}{n_l} \sum_{i=1}^{n_l} f^{(i)}(\mathbf{x}). \tag{1}$$

Let $X_t \in \mathbb{R}^{n_l \times d}$ denote the global model matrix, whose rows are the node models at round t: $X_t = [\mathbf{x}_t^{(1)}, \ldots, \mathbf{x}_t^{(n_l)}]^\top$. This matrix is not available to any individual node but serves as a convenient global representation for analysis. Each node locally updates $\mathbf{x}_t^{(i)} \to \mathbf{x}_{t'}^{(i)}$ for $t < t' < t+1$. The global matrix then updates via mixing, expressed as:

$$X_{t+1} = W_t X_{t'}, \tag{2}$$

where $W_t \in \mathbb{R}^{n_l \times n_l}$ is the mixing matrix. Every row of W_t represents the aggregation weights assigned by a node to all other nodes.

Mixing Matrix. Each row i of W_t specifies the aggregation weights assigned by node i to all others. Equivalently, column i represents the weights assigned to node i, i.e., its contribution to mixing. Some works assume W_t is doubly stochastic, where both rows and columns sum to one. This assumption simplifies analysis by ensuring equal contribution from all nodes, but enforcing it typically requires centralized coordination, as highlighted by the Oracle variant of Epidemic Learning (EL) (De Vos et al., 2023). Denser matrices typically yield larger spectral gaps and faster mixing, but at the cost of high node degrees and communication cost. In our design, detailed in Section 3, we instead compose multiple sparse matrices across rounds to attain both low degree per step and strong effective mixing through their product.

Consensus Distance. In decentralized settings, where only partial averaging occurs, local models may remain different after each round. To quantify this divergence, we use the *consensus distance* (CD), defined in prior works Kong et al. (2021) as the average squared Euclidean distance of each node's model from the global mean:

$$\text{CD}_t = \frac{1}{n_l} \sum_{i=1}^{n_l} \left\| \mathbf{x}_t^{(i)} - \bar{\mathbf{x}}_t \right\|^2, \tag{3}$$

where $\bar{\mathbf{x}}_t = \frac{1}{n_l} \sum_{i=1}^{n_l} \mathbf{x}_t^{(i)}$. To measure mixing efficiency, we define the *consensus distance ratio* (CDR) as:

$$\text{CDR} = \frac{\text{CD}_{t+1}}{\text{CD}_{t'}}, \tag{4}$$

for $t < t' < t+1$, where t' denotes the pre-gossip stage and $t+1$ the post-gossip stage. A lower CDR indicates stronger mixing. When CD approaches zero, nodes approach exact averaging; how close CDR gets to zero depends on the spectral properties of W_t.

Spectral Gap. The spectral gap of a mixing matrix W is defined as $1-\lambda_2$, where λ_2 is the second-largest eigenvalue in magnitude ($\lambda_1 = 1$ for a row stochastic matrix). This gap measures how quickly models converge to their average. A larger spectral gap implies faster information diffusion and better mixing (Ying et al., 2021; Lovász, 1993; Chung, 1997) under bounded heterogeneity.

Number of Edges as a Proxy for Communication Cost. In our setting, model updates are fixed-size messages, so each directed edge corresponds to exactly one exchange. We define one directed edge as one unit of communication cost. A node's degree reflects its individual budget, while the total number of directed edges captures system-wide bandwidth and aggregation effort. Since computation per node also scales with its in-degree, the edge count serves as a unified measure of both communication and aggregation cost. This abstraction provides a simple metric using which we design and evaluate scalable collaborative learning protocols to reduce the communication cost.

3 Design of Tiered Gossip Learning (TGL)

TGL, as illustrated in Figure 2 is composed of two kinds of nodes: *relays*, which act as server-like intermediaries, and *leaves*, which hold private data and perform local training. A small number of relays (n_r) support a larger population of leaves (n_l), mixing and redistributing their models without any direct leaf-to-leaf communication. Leaves may be constrained, with limited connections to relays, while relays are assumed more capable and can sustain larger fan-out to many leaves. This intentional asymmetry offloads heavier communication to the relay layer.

Figure 1 shows that strong mixing can be achieved with lower communication burden when fewer nodes are involved, motivating a modest choice of n_r based on the relay fanout budget. With unbounded relay budget, a single relay suffices and TGL reduces to FL, while tighter budgets can be accommodated by using multiple relays to share the load.

The relays themselves form a fully decentralized gossip layer with no central coordinator. Moreover, all connections in TGL are random and dynamic across rounds. This improves fault tolerance: if a relay fails, the system continues to collaborate through the remaining $n_r - 1$ relays and resampled connections, unlike FL or HFL. In this way, TGL synthesizes the design principles of hierarchy, asymmetric load sharing, decentralized relay layer, and random dynamic connectivity to enable effective mixing, fault-tolerance, and practical scalability without necessarily burdening constrained nodes as the system grows. We parameterize the protocol below.

Between consecutive rounds t and $t+1$, we define three synchronizing steps: $t+\frac{1}{4}, t+\frac{2}{4}, t+\frac{3}{4}$, which correspond to the three mixing stages described below. Extensions to asynchronous operation are also possible and are discussed in Appendix **??**.

Each leaf i performs local training from $\mathbf{x}_t^{(i)}$ to $\mathbf{x}_{t+\frac{1}{4}}^{(i)}$ via T_{loc} steps of SGD on its private dataset, where each step uses one mini-batch. The updated models are then mixed in three fixed hops described below.

Stage 1: Leaf-to-Relay Push. Each relay k independently selects a random subset of b_{lr} leaves (also its in-degree or budget), denoted by $\mathcal{L}_k$, which respond by pushing their current models to relay k, which aggregates them as:

$$\mathbf{x}_{t+\frac{2}{4}}^{(k)} = \frac{1}{b_{lr}} \sum_{i \in \mathcal{L}_k} \mathbf{x}_{t+\frac{1}{4}}^{(i)}, \quad X_{t+\frac{2}{4}} = W_{lr}\, X_{t+\frac{1}{4}},$$

where $W_{lr} \in \mathbb{R}^{n_r \times n_l}$ is the leaf-to-relay mixing matrix formed from the sampled leaf sets and $\mathbf{x}_{t+\frac{2}{4}}^{(k)}$ is the aggregated model at relay k. Sampling is independent across relays and across rounds, ensuring fair selection of leaves over time, with each leaf having an expected out-degree of $n_r b_{lr}/n_l$. Only sampled leaves perform local training in that round.

Algorithm 1 Tiered Gossip Learning (TGL)

Input: n_l (leaves), n_r (relays), T (global rounds), T_{loc} (local SGD steps), η (learning rate), b_{lr}, b_{rr}, b_{rl}: push, gossip, pull budgets
Init: $\mathbf{x}_0^{(i)} \leftarrow \mathbf{x}_0,\ \forall i \in [n_l]$
for $t = 1$ to T **do**
  Step 0: Local Training
  $\mathbf{x}_{t+\frac{1}{4}}^{(i)} \leftarrow \mathrm{SGD}(\mathbf{x}_t^{(i)}, \mathcal{D}_i, T_{\text{loc}}, \eta)$
  Step 1: Push
  Each relay k samples $\mathcal{L}_k, |\mathcal{L}_k| = b_{lr}$
  $\mathbf{x}_{t+\frac{2}{4}}^{(k)} \leftarrow \frac{1}{b_{lr}} \sum_{i \in \mathcal{L}_k} \mathbf{x}_{t+\frac{1}{4}}^{(i)}$
  Step 2: Gossip
  Each relay k sends to b_{rr} relays and receives from a set of $\mathcal{R}_k$ relays.
  $\mathbf{x}_{t+\frac{3}{4}}^{(k)} \leftarrow \frac{1}{|\mathcal{R}_k|+1}(\mathbf{x}_{t+\frac{2}{4}}^{(k)} + \sum_{m \in \mathcal{R}_k} \mathbf{x}_{t+\frac{2}{4}}^{(m)})$
  Step 3: Pull
  Each leaf i samples $\mathcal{S}_i, |\mathcal{S}_i| = b_{rl}$
  $\mathbf{x}_{t+1}^{(i)} \leftarrow \frac{1}{b_{rl}} \sum_{k \in \mathcal{S}_i} \mathbf{x}_{t+\frac{3}{4}}^{(k)}$
end for
Output: $\{\mathbf{x}_T^{(i)}\}_{i=1}^{n_l}$

Stage 2: Relay-to-Relay Gossip. Relays then engage in decentralized gossip, each sending its aggregated model to b_{rr} (its out-degree) randomly selected relays. While the out-degree is fixed, we allow the in-degree to vary across rounds, with each relay receiving models from a set $\mathcal{R}_k$ of peers. Over time, the expected in-degree also equals b_{rr}, but this relaxation avoids the need for central coordination. Each relay then updates its state by averaging its own model with those it receives:

$$\mathbf{x}_{t+\frac{3}{4}}^{(k)} = \frac{1}{|\mathcal{R}_k|+1}\Big(\mathbf{x}_{t+\frac{1}{2}}^{(k)} + \sum_{m\in\mathcal{R}_k} \mathbf{x}_{t+\frac{1}{2}}^{(m)}\Big), \quad X_{t+\frac{3}{4}} = W_{rr}\, X_{t+\frac{1}{2}},$$

$W_{rr} \in \mathbb{R}^{n_r \times n_r}$ is the relay-relay mixing matrix and $\mathbf{x}_{t+\frac{3}{4}}^{(k)}$ is the updated model at the relay k.

Stage 3: Relay-to-Leaf Pull. Each leaf i then pulls updates from a randomly selected subset $\mathcal{S}_i$ of $b_{rl} \geq 1$ relays and averages the received models:

$$\mathbf{x}_{t+1}^{(i)} = \frac{1}{b_{rl}} \sum_{k\in\mathcal{S}_i} \mathbf{x}_{t+\frac{3}{4}}^{(k)}, \quad X_{t+1} = W_{rl}\, X_{t+\frac{3}{4}},$$

where $W_{rl} \in \mathbb{R}^{n_l \times n_r}$ is the relay-to-leaf mixing matrix and $\mathbf{x}_{t+1}^{(i)}$ is the updated model at leaf i for the next round. Since models have already been mixed in the relay layer, even a small pull budget b_{rl} (as low as 1) can suffice, making it easy for leaves to join and for collaboration to scale. While using $b_{rl} = 1$ may leak the exact pre-training model of a leaf, like in FL, a higher value can mitigate this when privacy is a concern.

Combining all three stages yields the end-to-end transformation:

$$X_{t+1} = W_{rl}\, W_{rr}\, W_{lr}\, X_{t+\frac{1}{4}} \;\equiv\; W_{TGL}\, X_{t+\frac{1}{4}},$$

where $W_{TGL} \in \mathbb{R}^{n_l \times n_l}$ is the overall mixing matrix for TGL as formed by the connections in a given round. Although the individual matrices may be sparse due to constrained budgets, W_{TGL}, being the product of three matrices is typically dense, enabling effective mixing across leaves. This dense composition, enabled by the hierarchical structure, is key to TGL's improved performance over all flat P2PL baselines.

Each mixing stage in TGL has its own budget parameter (b_{lr}, b_{rr}, b_{rl}), enabling independent control. For example, if leaves are constrained but relays underutilized, increasing b_{rr} densifies W_{rr}, improving mixing without raising the leaf budget. This decoupled control is a key advantage of TGL over P2PL systems, where unified roles prevent such separation. Together with n_r, these budgets (b_{lr}, b_{rr}, b_{rl}) constitute the core design parameters of TGL. We analyze how these parameters influence stage-wise consensus and the overall convergence rate in the next section.

TGL further benefits from several structural properties. Bounding the in-degrees in Stage 1 (relay receive) and Stage 3 (leaf receive) ensures that no receiver is overwhelmed, either due to too many senders (as in relays receiving from many leaves) or limited capacity (as in leaves pulling from relays). Stage 2, which involves only the small relay set, is less sensitive to such constraints.

The use of dynamic, random sampling offers two key advantages: it gracefully handles node failures by aggregating only over received messages, and it enables seamless scaling, as new leaves or relays can be added by simply augmenting the set of valid nodes to sample from. For example, a new leaf can immediately begin participation by pulling a recent model in Stage 3. Finally, the total number of directed edges per round, $n_r b_{lr} + n_r b_{rr} + n_l b_{rl}$, serves as a direct proxy for communication and computation cost, which we use to normalize comparisons across baselines in our evaluation.

4 Theoretical Analysis

In this section, we analyze the convergence behavior of TGL under standard assumptions used in stochastic first-order methods. Specifically, we assume for local functions f and global function F:

Assumption 4.1 (Smoothness)**.** For each $i \in [n_l]$, the local function $f^{(i)} : \mathbb{R}^d \to \mathbb{R}$ is differentiable, and there exists a constant $L < \infty$ such that for all $x, y \in \mathbb{R}^d$: $\|\nabla f^{(i)}(y) - \nabla f^{(i)}(x)\| \leq L\|y - x\|$.

Assumption 4.2 (Bounded Stochastic Noise)**.** There exists a constant $\sigma < \infty$ such that for all $i \in [n_l]$ and $x \in \mathbb{R}^d$: $\mathbb{E}_{\xi\sim D^{(i)}}\big[\|\nabla f(x,\xi) - \nabla f^{(i)}(x)\|^2\big] \leq \sigma^2$, where σ captures variance introduced by stochastic gradients due to batch sampling.

Assumption 4.3 (Bounded Heterogeneity)**.** There exists a constant $\mathcal{H} < \infty$ such that for all $x \in \mathbb{R}^d$: $\frac{1}{n_l}\sum_{i\in[n_l]} \|\nabla f^{(i)}(x) - \nabla F(x)\|^2 \le \mathcal{H}^2$, where $\mathcal{H}$ quantifies the heterogeneity arising from the non-iid data distribution.

Theorem 4.4. *Consider Algorithm 1 under the above assumptions. Let the initial optimization gap be:* $\Delta_0 := F(x_0) - \min_{x\in\mathbb{R}^d} F(x)$*. Then, for any* $T \ge 1$*, with* $n_l \ge 2$ *leaves with pull budget* $b_{rl} \ge 1$*, and* $n_r \ge 2$ *relays with push and gossip budgets* $b_{lr} \ge 1, b_{rr} \ge 1$*, selecting the step size as:*

$$\gamma \in \Theta\left(\min\left\{\sqrt{\frac{n_l\Delta_0}{TL((1+\beta')\sigma^2+\beta'\mathcal{H}^2)}},\ \sqrt[3]{\frac{\Delta_0}{TL^2\beta_{TGL}(\sigma^2+\mathcal{H}^2)}},\ \frac{1}{L}\right\}\right),$$

we obtain:

$$\frac{1}{n_lT}\sum_{t=0}^{T-1}\sum_{i=1}^{n_l}\mathbb{E}\left[\left\|\nabla F(x_t^{(i)})\right\|^2\right] \in \mathcal{O}\left(\sqrt{\frac{L\Delta_0}{Tn_l}((1+\beta')\sigma^2+\beta'\mathcal{H}^2)}+\sqrt[3]{\frac{L^2\beta_{TGL}\Delta_0^2(\sigma^2+\mathcal{H}^2)}{T^2}}+\frac{L\Delta_0}{T}\right). \tag{5}$$

where

$$\begin{aligned}\beta_{TGL} :=& \beta_{lr}\beta_{rr}\beta_{rl}, \quad \beta' := \tfrac{1}{2}\left[\beta_{TGL} + \tfrac{n_l}{n_r}\beta_{lr}(1+\beta_{rr})\right], \quad \beta_{lr} := \tfrac{1}{b_{lr}}\left(1 - \tfrac{b_{lr}-1}{n_l-1}\right),\\ \beta_{rr} :=& \tfrac{1}{b_{rr}}\left(1 - \left(1 - \tfrac{b_{rr}}{n_r-1}\right)^{n_r}\right) - \tfrac{1}{n_r-1}, \quad \beta_{rl} := \tfrac{1}{b_{rl}}\left(1 - \tfrac{b_{rl}-1}{n_r-1}\right)\end{aligned} \tag{6}$$

These β-terms appear in Lemma **??** as stage-wise contraction factors, bounding the expected consensus distance after each step:

$$\frac{\mathbb{E}[\mathrm{CD}_{t+\frac{2}{4}}]}{\mathbb{E}[\mathrm{CD}_{t+\frac{1}{4}}]} \le \beta_{lr}, \quad \frac{\mathbb{E}[\mathrm{CD}_{t+\frac{3}{4}}]}{\mathbb{E}[\mathrm{CD}_{t+\frac{2}{4}}]} \le \beta_{rr}, \quad \frac{\mathbb{E}[\mathrm{CD}_{t+1}]}{\mathbb{E}[\mathrm{CD}_{t+\frac{3}{4}}]} \le \beta_{rl} \ \Rightarrow\ \frac{\mathbb{E}[\mathrm{CD}_{t+1}]}{\mathbb{E}[\mathrm{CD}_{t+\frac{1}{4}}]} \le \beta_{\mathrm{TGL}}.$$

As shown in Equation 6, each budget parameter b_{lr}, b_{rr}, b_{rl} directly controls its corresponding β-value: increasing the budget reduces the contraction factor, yielding faster consensus. Since β_{TGL} is multiplicative, improving any single stage strengthens the overall contraction rate.

We structure the proof through three key lemmas: average preservation in Lemma **??**, bounding stage-wise consensus contraction in Lemma **??**, and bounding the deviation of the average model in Lemma **??**. These results are combined in Lemmas **??** and **??** to yield the full convergence proof in Appendix **??**, and additional analysis in Appendix **??**, with an overview provided in Appendix **??**.

5 Submission of conference papers to ICLR 2026

ICLR requires electronic submissions, processed by `https://openreview.net/`. See ICLR's website for more instructions.

If your paper is ultimately accepted, the statement `\iclrfinalcopy` should be inserted to adjust the format to the camera ready requirements.

The format for the submissions is a variant of the NeurIPS format. Please read carefully the instructions below, and follow them faithfully.

5.1 Style

Papers to be submitted to ICLR 2026 must be prepared according to the instructions presented here.

Authors are required to use the ICLR LaTeX style files obtainable at the ICLR website. Please make sure you use the current files and not previous versions. Tweaking the style files may be grounds for rejection.

5.2 Retrieval of style files

The style files for ICLR and other conference information are available online at:

http://www.iclr.cc/

The file iclr2026_conference.pdf contains these instructions and illustrates the various formatting requirements your ICLR paper must satisfy. Submissions must be made using LaTeX and the style files iclr2026_conference.sty and iclr2026_conference.bst (to be used with LaTeX2e). The file iclr2026_conference.tex may be used as a "shell" for writing your paper. All you have to do is replace the author, title, abstract, and text of the paper with your own.

The formatting instructions contained in these style files are summarized in sections 6, 7, and 8 below.

6 General formatting instructions

The text must be confined within a rectangle 5.5 inches (33 picas) wide and 9 inches (54 picas) long. The left margin is 1.5 inch (9 picas). Use 10 point type with a vertical spacing of 11 points. Times New Roman is the preferred typeface throughout. Paragraphs are separated by 1/2 line space, with no indentation.

Paper title is 17 point, in small caps and left-aligned. All pages should start at 1 inch (6 picas) from the top of the page.

Authors' names are set in boldface, and each name is placed above its corresponding address. The lead author's name is to be listed first, and the co-authors' names are set to follow. Authors sharing the same address can be on the same line.

Please pay special attention to the instructions in section 8 regarding figures, tables, acknowledgments, and references.

There will be a strict upper limit of **9 pages** for the main text of the initial submission, with unlimited additional pages for citations. This limit will be expanded to **10 pages** for rebuttal/camera ready.

7 Headings: first level

First level headings are in small caps, flush left and in point size 12. One line space before the first level heading and 1/2 line space after the first level heading.

7.1 Headings: second level

Second level headings are in small caps, flush left and in point size 10. One line space before the second level heading and 1/2 line space after the second level heading.

7.1.1 Headings: third level

Third level headings are in small caps, flush left and in point size 10. One line space before the third level heading and 1/2 line space after the third level heading.

8 Citations, figures, tables, references

These instructions apply to everyone, regardless of the formatter being used.

8.1 Citations within the text

Citations within the text should be based on the natbib package and include the authors' last names and year (with the "et al." construct for more than two authors). When the authors or the publication are included in the sentence, the citation should not be in parenthesis using \citet{} (as in "See **?** for more information."). Otherwise, the citation should be in parenthesis using \citep{} (as in "Deep learning shows promise to make progress towards AI (**?**).").

Table 1: Sample table title

PART	DESCRIPTION
Dendrite	Input terminal
Axon	Output terminal
Soma	Cell body (contains cell nucleus)

The corresponding references are to be listed in alphabetical order of authors, in the REFERENCES section. As to the format of the references themselves, any style is acceptable as long as it is used consistently.

8.2 FOOTNOTES

Indicate footnotes with a number[1] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches (12 picas).[2]

8.3 FIGURES

All artwork must be neat, clean, and legible. Lines should be dark enough for purposes of reproduction; art work should not be hand-drawn. The figure number and caption always appear after the figure. Place one line space before the figure caption, and one line space after the figure. The figure caption is lower case (except for first word and proper nouns); figures are numbered consecutively.

Make sure the figure caption does not get separated from the figure. Leave sufficient space to avoid splitting the figure and figure caption.

You may use color figures. However, it is best for the figure captions and the paper body to make sense if the paper is printed either in black/white or in color.

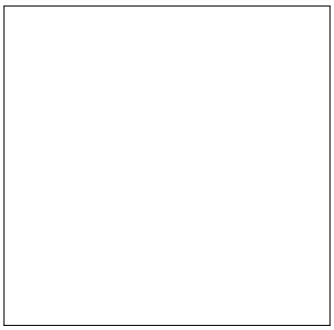

Figure 3: Sample figure caption.

8.4 TABLES

All tables must be centered, neat, clean and legible. Do not use hand-drawn tables. The table number and title always appear before the table. See Table 1.

Place one line space before the table title, one line space after the table title, and one line space after the table. The table title must be lower case (except for first word and proper nouns); tables are numbered consecutively.

[1] Sample of the first footnote

[2] Sample of the second footnote

9 DEFAULT NOTATION

In an attempt to encourage standardized notation, we have included the notation file from the textbook, *Deep Learning* **?** available at `https://github.com/goodfeli/dlbook_notation/`. Use of this style is not required and can be disabled by commenting out `math_commands.tex`.

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathbf{A}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	The parents of x_i in $\mathcal{G}$

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,j,k}$	Element (i, j, k) of a 3-D tensor $\mathbf{A}$
$\mathbf{A}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor containing derivatives of y with respect to $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E} f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\frac{1}{1 + \exp(-x)}$
$\zeta(x)$	Softplus, $\log(1 + \exp(x))$
$\Vert \boldsymbol{x} \Vert_p$	L^p norm of $\boldsymbol{x}$
$\Vert \boldsymbol{x} \Vert$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$
$\mathbf{1}_{\mathrm{condition}}$	is 1 if the condition is true, 0 otherwise

10 FINAL INSTRUCTIONS

Do not change any aspects of the formatting parameters in the style files. In particular, do not modify the width or length of the rectangle the text should fit into, and do not change font sizes (except perhaps in the REFERENCES section; see below). Please note that pages should be numbered.

11 PREPARING POSTSCRIPT OR PDF FILES

Please prepare PostScript or PDF files with paper size "US Letter", and not, for example, "A4". The -t letter option on dvips will produce US Letter files.

Consider directly generating PDF files using `pdflatex` (especially if you are a MiKTeX user). PDF figures must be substituted for EPS figures, however.

Otherwise, please generate your PostScript and PDF files with the following commands:

```
dvips mypaper.dvi -t letter -Ppdf -G0 -o mypaper.ps
ps2pdf mypaper.ps mypaper.pdf
```

11.1 MARGINS IN LATEX

Most of the margin problems come from figures positioned by hand using `\special` or other commands. We suggest using the command `\includegraphics` from the graphicx package. Always specify the figure width as a multiple of the line width as in the example below using .eps graphics

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.eps}
```

or

```
\usepackage[pdftex]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.pdf}
```

for .pdf graphics. See section 4.4 in the graphics bundle documentation (`http://www.ctan.org/tex-archive/macros/latex/required/graphics/grfguide.ps`)

A number of width problems arise when LaTeX cannot properly hyphenate a line. Please give LaTeX hyphenation hints using the `\-` command.

AUTHOR CONTRIBUTIONS

If you'd like to, you may include a section for author contributions as is done in many journals. This is optional and at the discretion of the authors.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

Mehdi Salehi Heydar Abad, Emre Ozfatura, Deniz Gunduz, and Ozgur Ercetin. Hierarchical federated learning across heterogeneous cellular networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8866–8870. IEEE, 2020.

Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning*, pp. 344–353. PMLR, 2019.

Li Chen, Wei Liu, Yunfei Chen, and Weidong Wang. Communication-efficient design for quantized decentralized federated learning. *IEEE Transactions on Signal Processing*, 72:1175–1188, 2024.

Yiming Chen, Kun Yuan, Yingya Zhang, Pan Pan, Yinghui Xu, and Wotao Yin. Accelerating gossip sgd with periodic global averaging. In *International Conference on Machine Learning*, pp. 1791–1802. PMLR, 2021.

Yat-Tin Chow, Wei Shi, Tianyu Wu, and Wotao Yin. Expander graph and communication-efficient decentralized optimization. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pp. 1715–1720. IEEE, 2016.

Fan R. K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, Providence, RI, 1997.

Martijn de Vos, Akash Dhasade, Anne-Marie Kermarrec, Erick Lavoie, Johan Pouwelse, and Rishi Sharma. Decentralized learning made practical with client sampling. *arXiv e-prints*, pp. arXiv–2302, 2023.

Martijn De Vos, Sadegh Farhadkhani, Rachid Guerraoui, Anne-Marie Kermarrec, Rafael Pires, and Rishi Sharma. Epidemic learning: Boosting decentralized learning with randomized communication. *Advances in Neural Information Processing Systems*, 36:36132–36164, 2023.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in neural information processing systems*, 33:19586–19597, 2020.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210, 2021.

Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International conference on machine learning*, pp. 3478–3487. PMLR, 2019.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized sgd with changing topology and local updates. In *International conference on machine learning*, pp. 5381–5393. PMLR, 2020.

Lingjing Kong, Tao Lin, Anastasia Koloskova, Martin Jaggi, and Sebastian Stich. Consensus control for decentralized deep learning. In *International Conference on Machine Learning*, pp. 5686–5696. PMLR, 2021.

Zexi Li, Jiaxun Lu, Shuang Luo, Didi Zhu, Yunfeng Shao, Yinchuan Li, Zhimeng Zhang, Yongheng Wang, and Chao Wu. Towards effective clustered federated learning: A peer-to-peer framework with adaptive neighbor matching. *IEEE Transactions on Big Data*, 10(6):812–826, 2022.

Zhi Li, Wei Shi, and Ming Yan. A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates. *IEEE Transactions on Signal Processing*, 67 (17):4494–4506, 2019.

X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS 2017)*, pp. 5330–5340. Curran Associates, Inc., 2017.

Lumin Liu, Jun Zhang, SH Song, and Khaled B Letaief. Client-edge-cloud hierarchical federated learning. In *ICC 2020-2020 IEEE international conference on communications (ICC)*, pp. 1–6. IEEE, 2020.

László Lovász. Random walks on graphs. *Combinatorics, Paul erdos is eighty*, 2(1-46):4, 1993.

H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS 2017)*, pp. 1273–1282. PMLR, 2017.

Angelia Nedić, Alex Olshevsky, and Michael G Rabbat. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5):953–976, 2018.

Noa Onoszko, Gustav Karlsson, Olof Mogren, and Edvin Listo Zec. Decentralized federated learning of deep neural networks on non-iid data. *arXiv preprint arXiv:2107.08517*, 2021.

Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722, 2020.

Zhuoqing Song, Weijian Li, Kexin Jin, Lei Shi, Ming Yan, Wotao Yin, and Kun Yuan. Communication-efficient topologies for decentralized learning with $o(1)$ consensus rate. *Advances in Neural Information Processing Systems*, 35:1073–1085, 2022.

Yuki Takezawa and Sebastian U Stich. Scalable decentralized learning with teleportation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AvmBgiQxxp.

Yuki Takezawa, Ryoma Sato, Han Bao, Kenta Niwa, and Makoto Yamada. Beyond exponential graph: Communication-efficient topologies for decentralized learning via finite-time convergence. *Advances in Neural Information Processing Systems*, 36:76692–76717, 2023.

Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. D
textsuperscript2:: Decentralized training over decentralized data. In *International Conference on Machine Learning*, pp. 4848–4856. PMLR, 2018.

Jianyu Wang, Anit Kumar Sahu, Zhouyi Yang, Gauri Joshi, and Soummya Kar. Matcha: Speeding up decentralized sgd via matching decomposition sampling. In *2019 Sixth Indian Control Conference (ICC)*, pp. 299–300. IEEE, 2019.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

Bicheng Ying, Kun Yuan, Yiming Chen, Hanbin Hu, Pan Pan, and Wotao Yin. Exponential graph is provably efficient for decentralized deep training. *Advances in Neural Information Processing Systems*, 34:13975–13987, 2021.

Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. In *International Conference on Machine Learning*, pp. 7184–7193. PMLR, 2019.

Kun Yuan, Yiming Chen, Xinmeng Huang, Yingya Zhang, Pan Pan, Yinghui Xu, and Wotao Yin. Decentlam: Decentralized momentum sgd for large-batch deep training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3029–3039, 2021.

A Appendix

You may include other additional sections here.