Neuron-Inspired Leader–Follower Networks for AI with Local Error Signals

Anzhe Cheng[1, +], **Chenzhong Yin**[1, +], **Mingxi Cheng**[1, +], **Xiongye Xiao**[1], **Heng Ping**[1], **Andrei Irimia**[2, 3, 4], **Shahin Nazarian**[1], **and Paul Bogdan**[1, *]

[1]Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA 90007, USA.
[2]Corwin D. Denney Research Center, Department of Biomedical Engineering, Viterbi School of Engineering, University of Southern California, Los Angeles, CA, USA
[3]Ethel Percy Andrus Gerontology Center, Leonard Davis School of Gerontology, University of Southern California, Los Angeles, CA, USA
[4]Imaging Genetics Center, Stevens Neuroimaging and Informatics Institute, Keck School of Medicine, University of Southern California, Marina del Rey, CA, USA
[*]Correspondence and requests for materials should be addressed to P.B. (email: pbogdan@usc.edu)
[+]these authors contributed equally to this work

ABSTRACT

Artificial intelligence (AI) systems often operate under memory and energy limits, similar to biological networks of neurons. The collective behavior of a network with heterogeneous, resource-limited information processing units (e.g., a group of fish, a flock of birds, or a network of neurons) demonstrates a high degree of self-organization, emergence, and complexity. These properties arise from simple interaction rules, where certain individuals can exhibit leadership-like behavior and influence the collective activity of the group. Motivated by collective intelligence in biological neural systems and neuron-level plasticity, we introduce a *worker* concept to an artificial neural network (NN) for AI. This NN structure contains workers that encompass one or more information processing units (e.g., neurons, filters, layers, or blocks of layers). Workers are either leaders or followers, and we train a leader-follower neural network (LFNN) by leveraging local error signals. The LFNN does not require backpropagation (BP) or a global loss function to achieve optimal performance (we denote LFNN trained without BP as LFNN-ℓ). By investigating worker behavior and evaluating the LFNN and LFNN-ℓ architectures on a variety of image classification tasks (e.g., MNIST, CIFAR-10, ImageNet), we demonstrate that LFNN-ℓ trained with local error signals achieves lower error rates and superior scalability than state-of-the-art machine learning approaches. Furthermore, **LFNN-ℓ** can be conveniently embedded in classic convolutional NN architectures (e.g., **VGG**, **ResNet**, and **Vision Transformer (ViT)**), achieving a **2x** speedup compared to BP-based methods and significantly outperforming models trained with end-to-end BP and other state-of-the-art local learning methods in terms of accuracy on **CIFAR-10**, **Tiny-ImageNet**, and **ImageNet**. Lastly, the proposed LFNN-based model outperforms deep learning counterparts in brain-age prediction from magnetic resonance imaging (MRI) data, while also achieving a **2**$\times$ speedup. These results indicate that neuron-inspired, decentralized training rules can improve AI accuracy and efficiency. The LFNN-ℓ architecture provides practical tools and insights for addressing core challenges in deep learning and for designing more biologically plausible and sustainable AI.

Supplementary information

Code and Data Availability

Our code to run the experiments can be find at `https://anonymous.4open.science/r/LFNN-6DF4/`.

Related work

Efforts have been made to bridge the gaps in computational efficiency that continue to exist between ANNs and BNNs[1]. One popular approach is the replacement of global loss with local error signals[2]. Researchers have proposed to remove BP to address backward locking problems[3], mimic the local connection properties of neuronal networks[4] and incorporate local plasticity rules to enhance ANN's biological plausibility[5]. A research topic closely related to our work is supervised deep learning with local loss. It has been noticed that training NNs with BP is biologically implausible because BNNs in the human brain do not transmit error signals at a global scale[6–8]. Several studies have proposed training NNs with local error signals, such as layer-wise learning[2, 9], block-wise learning[4, 10], gated linear network family[11], etc. Mostafa et al. generate local error signals in each NN layer using fixed, random auxiliary classifiers[2], where a hidden layer is trained using local errors generated by a random fixed classifier. This is similar to an approach called feedback alignment training, where random fixed weights are used to back-propagate the error layer by layer[12]. In[10], the authors split a NN into a stack of gradient-isolated modules, and each module is trained to maximally preserve the information of its inputs. A more recent work by Ren et al.[13] proposed a local greedy forward gradient algorithm by enabling the use of forward gradient learning in supervised deep learning tasks. Their biologically plausible BP-free algorithm outperforms the forward gradient and feedback alignment family of algorithms significantly.

Recent advances in deep learning for segmentation and prediction have employed various mechanisms to propagate error and capture complex dependencies. For instance, Li et al.[14] introduced a convolutional neural network for propagation-based stereo image segmentation that relies on fixed local propagation rules. In contrast, our LFNN framework employs a dynamic leader-follower mechanism where the best-performing neurons (leaders) are selected at each training iteration based on local prediction errors, and followers update their weights by aligning with these leaders. This dynamic selection process not only reduces the backward locking inherent in traditional backpropagation but also adapts to the task complexity in a manner that fixed propagation schemes cannot achieve.

Furthermore, Li et al.[15] proposed a Spatio-Temporal-Spectral Hierarchical Graph Convolutional Network with semisupervised active learning for patient-specific seizure prediction, emphasizing the role of multi-dimensional dependencies and active sample selection. While their approach focuses on leveraging graph structures to capture temporal and spectral correlations, our method dynamically aggregates reliable local signals via leader selection, ensuring efficient error propagation within decoupled network blocks without the need for global gradient chaining.

In addition, Chen et al.[16] developed LDANet for automatic lung parenchyma segmentation from CT images, and Zhang et al. [17] introduced a multi-level fusion and attention-guided CNN for image dehazing. Both approaches employ attention mechanisms and multi-scale feature fusion to enhance segmentation quality. Our LFNN approach is inspired by these innovations; however, it uniquely incorporates a dynamic, adaptive leader selection strategy that allows the network to allocate its representational capacity in a data-driven manner. By ensuring that only the most reliable neurons (leaders) drive the weight updates while followers are gently aligned via a local mean squared error loss, our method achieves robust performance and significantly improved training efficiency, particularly on complex tasks such as ImageNet segmentation.

Collectively, these comparisons underscore that while several recent methods have made important strides in propagation and feature fusion, our LFNN framework distinguishes itself by its dynamic local error propagation strategy. This mechanism not only mitigates the limitations of fixed local propagation but also naturally adapts the effective leader fraction to the intrinsic complexity of the task at hand, thereby maintaining high representational diversity and overall accuracy.

Our LFNN-ℓ shares some similarities with the above work in the sense that the LFNN-ℓ is trained with loss signals generated locally without BP. In contradistinction to the state-of-the-art, we do not require extra memory blocks to generate an error signal. Hence, the number of trainable parameters can be kept identical to that of NNs without an LF hierarchy.

More Experiments and Discussion

MNIST

Experimental setup. In this section, we follow Section 3.1 to conduct experiments with an LFNN-ℓ for MNIST data classification. We utilize a network consisting of two hidden, fully connected layers, each containing 32 workers. Given that it is a 10-class classification problem, each worker is naturally equipped with 10 neurons, as specified in the implementation details outlined in the methods. We vary the leadership size from 10% to 100% and compare the BP-free version with its BP-enabled counterpart. The *softmax* and ReLU activation functions are used for the output and hidden layers, respectively. An Adam optimizer with a learning rate of 5e-4 is employed, and the hyper-parameter λ is set to 1. Categorical cross-entropy loss is utilized as the prediction loss for output neurons ($\mathscr{L}_g^o$) and leader workers ($\mathscr{L}_l^{\delta}$), while mean squared error is employed as a local error signal for follower workers ($\mathscr{L}_l^{\bar{\delta}}$). To train the networks, we conduct 50 epochs with early stopping criteria. The results presented in Table S1 are evaluated based on 5 repetitions of the experiment using different random initializations. In addition to comparing BP-enabled and BP-free LFNNs, we examine the performance of leaders and provide a visualization of their development. Furthermore, we investigate and compare the abilities of leaders and workers. For these analyses, we utilize a trained LFNN-ℓ model with a leadership size of 20%.

Experimental results. Based on the comparison results presented in Table S1, we observe that both BP-free and BP-enabled LFNNs yield similar results, albeit slightly lower than a classical DNN[1]. However, a notable difference arises when adjusting the leadership size. Specifically, as the leadership size increases, the test accuracy of LFNN-ℓ tends to decrease, whereas the accuracy of the BP-enabled version remains relatively stable. This discrepancy can be attributed to the potency of the global prediction loss, which is known to be highly effective in conjunction with BP. Therefore, given a sufficient training duration, the

[1]A MLP trained with BP and possessing the same number of trainable parameters achieves a test accuracy of 98.11%.

Leadership	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
BP-free	96.40%	**96.68%**	96.23%	96.17%	95.98%	95.89%	95.93%	95.39%	95.22%	94.87%
BP-enabled	96.08%	96.30%	96.41%	96.38%	**96.52%**	96.19%	96.21%	96.34%	96.29%	96.20%

Table S1. Accuracy results of an LFNN trained with and without BP.

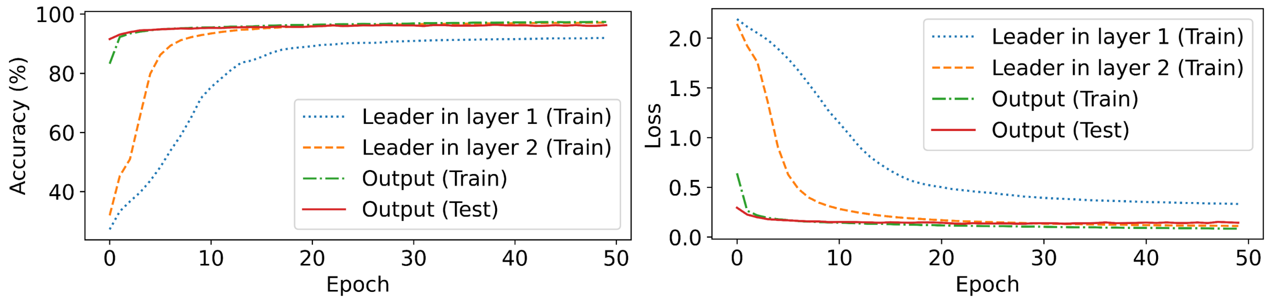

Figure S1. Leader accuracy and loss analysis. The best-performing leaders in hidden layers generate accurate predictions. Leaders closer to the output layer perform better than leaders in early layers.

network is capable of achieving comparable accuracies as long as an adequate number of leader workers are present. Increasing the number of leaders does not confer additional benefits to the overall network performance for this simple task.

In contrast, the BP-free network does not rely on the propagation of a global loss signal through the layers to adjust weights. Instead, it relies on local prediction losses to update leader workers. In nature, a small number of participants often suffices to dominate and guide the movement of a group. In simple scenarios, a large leadership size does not necessarily contribute to higher accuracy or ensure the correct direction of the entire group. In fact, it may even impede performance, as leaders receive local signals that may not be aligned with one another. In LFNN, however, leaders receive global loss signals without potential conflicts in the error signal. Therefore, larger leadership sizes do not compromise overall accuracy.

The best-performing leaders. In order to gain a deeper understanding of the LFNN's leadership dynamics, we examine the performance of the leaders that the followers choose to follow during training. As described in Section 2, followers in a hidden layer select the leader with the best performance. However, it is important to note that this approach may have limitations if the best-performing leaders do not themselves perform well, potentially impacting the overall network performance. To investigate this further, we analyze the train and test accuracies of the final output layer throughout the training process. From the results depicted in Figure S1, we observe that the accuracy of the output layer reaches a high level early on during training. Additionally, the accuracy of leaders in hidden layers gradually improves as training progresses. Notably, leaders in layers closer to the output layer tend to outperform leaders located in layers closer to the input layer. This observation suggests that information flows effectively through the network, and leaders in later layers contribute more significantly to the overall performance.

The observations from our analysis yield two key messages. Firstly, the progressive processing and learning of information in the LFNN validate the effectiveness of approaches such as transfer learning or learning with pre-trained models. Specifically, the utilization of representations from later layers, which contain more useful information for classification, aligns with the common practice of using pre-trained models by discarding the last output layer and utilizing the representation generated by the last hidden layer for downstream tasks. Secondly, the performance of the best-performing leaders in each hidden layer highlights their ability to generate accurate predictions. Consequently, the followers aligning their representations with these leaders can be viewed as a form of layer-wise knowledge distillation. In this analogy, the leaders function as teacher models, distilling their knowledge, while the followers act as student models, assimilating the distilled knowledge. This concept of knowledge distillation has been shown to outperform learning solely from raw labels in certain scenarios[18]. Notably, we find that the LFNN-ℓ with 100% leadership, which corresponds to learning solely from raw labels, performs worse than other configurations, as indicated in Table S1.

Overall, these findings highlight the importance of both layer-wise knowledge distillation and the progressive information processing in LFNNs, shedding light on the potential benefits of incorporating pre-trained models and teacher-student learning paradigms in deep learning approaches.

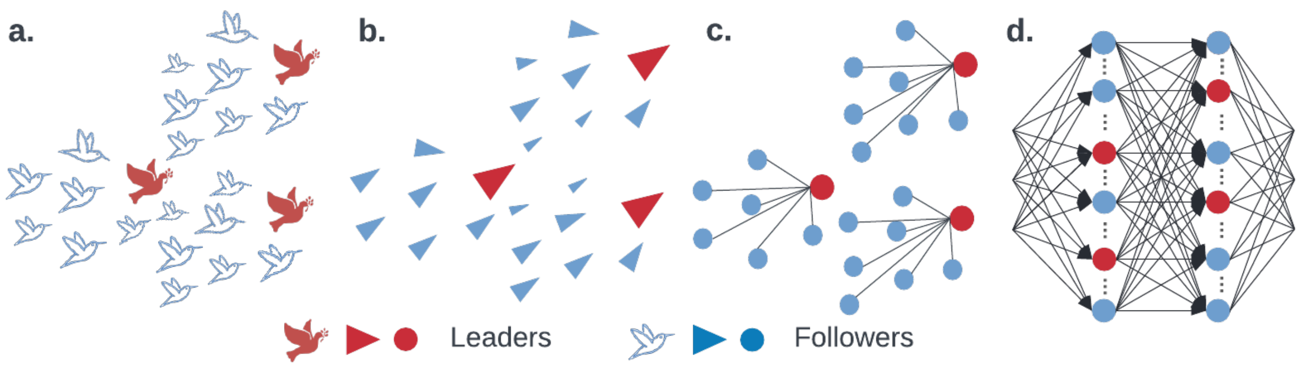

Figure S3. **a-b.** A flock of birds where leaders are informed and lead the flock. **c.** An abstracted network from the flock. **d.** A leader-follower neural network architecture.

Leaders vs. followers. In this section, we investigate the individual performance of leaders and followers in the LFNN architecture during the inference phase. By isolating the leaders and followers separately, we aim to understand their contributions and evaluate their effectiveness in generating accurate predictions. We create two ablated variants: one with only leader workers in inference and another with only follower workers. It is important to note that using only a subset of workers may impact the efficiency of the network since a portion of the neurons remain unused. The accuracy results, as depicted in Figure S2, demonstrate that the network achieves the highest accuracy when both leader and follower workers are employed in the inference process. Interestingly, we observe that the network with follower workers alone in inference outperforms the network with only leader workers. This finding aligns with our earlier discussion regarding the roles of leaders and followers. In our LFNN setup, leaders act as teacher models, while followers serve as student models. Traditionally, in knowledge distillation, teacher models tend to outperform student models due to their larger and more complex architectures. However, in our specific setup, the follower workers learn from the best-performing teacher, which explains their superior performance. These results highlight the significance of incorporating both leaders and followers in the LFNN architecture during inference. While followers benefit from the knowledge distilled by the leaders, the presence of leader workers enhances the overall performance of the network. This finding underscores the importance of the dynamic interaction between leaders and followers, and their collective contribution to achieving high accuracy in the LFNN framework. **Worker activity in an LFNN.** Collective motion in a group of particles can be readily identified through visualization. Since our LFNN's weight update rules are inspired by a collective motion model, we visualize the worker activities and explore the existence of collective motion patterns in the network during training. Following our weight update rule, we select 30% of the leaders from the 32 workers in each training step and update their weight dynamics based on global and local prediction loss. Consequently, the leader workers receive individual error signals and update their activity accordingly. Conversely, the remaining 70% of workers act as followers and update their weight dynamics by mimicking the best-performing leader through local error signals. In essence, all followers align themselves with a single leader, resulting in similar and patterned activity in each training step.

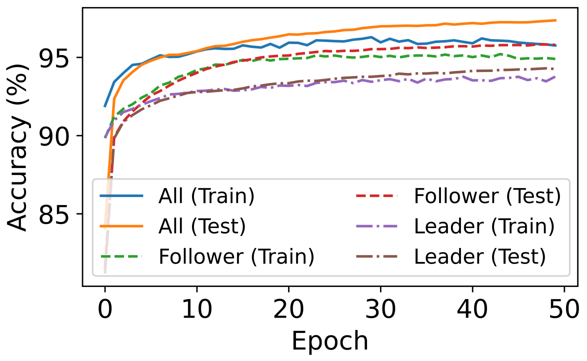

Figure S2. Accuracy results of LFNN-ℓ with different workers in inference.

To visualize the activities of all workers, we utilize the neuron output $\vec{y}$ before and after the weight update at each time step, and the difference between them represents the worker activity. The results in Figure S4a demonstrate that in each time step, the follower workers (represented by black lines) move in unison to align themselves with the leaders. During the initial training period (steps 0 to 1000), both leaders and followers exhibit significant movement and rapid learning, resulting in relatively larger step sizes. As the learning process stabilizes and approaches saturation, the workers' movement becomes less pronounced as the weights undergo less drastic changes in the well-learned network. Overall, we observe a patterned movement in worker activity in LFNNs, akin to the collective motion observed in the classic Vicsek model[19]. **Leadership size on training/testing MNIST and CIFAR-10.** Table S2 presents the relationship between leadership size and model performance on validating MNIST and CIFAR-10. In MNIST, LFNN and LFNN-ℓ with different leadership sizes achieve similar test error rates. Further

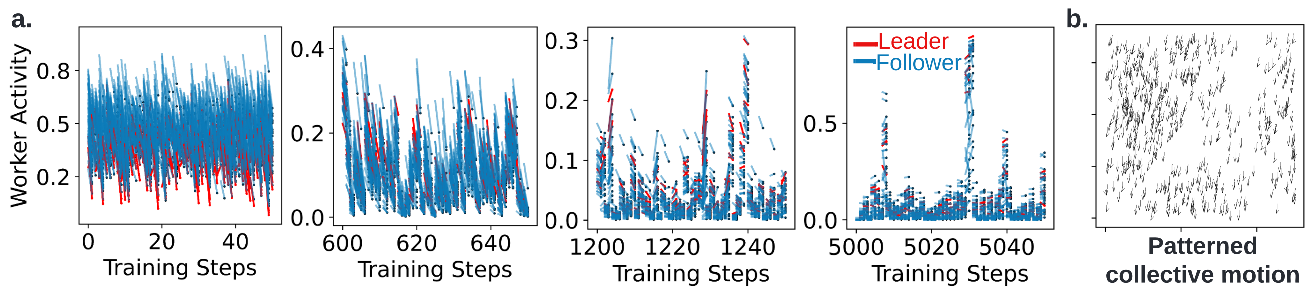

Figure S4. **a.** Worker activity visualization in an LFNN. At each time step, the followers (black lines) align themselves with leaders (red lines). **b.** Patterned collective motion produced by the classic Vicsek model[19].

Dataset (No. of Parameters)	Model		Leadership Percentage									
			10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
MNIST (∼275K)	LFNN-ℓ	Test	1.96	1.67	1.98	1.49	1.49	1.49	**1.49**	1.64	1.69	1.57
		Train	0.12	0.42	0.07	0.06	0.05	0.11	0.04	0.36	0.24	0.92
	LFNN	Test	1.24	1.68	1.50	1.60	1.40	1.20	**1.18**	1.40	1.44	1.51
		Train	1.21	1.21	1.20	1.20	1.10	1.10	1.15	1.10	1.21	1.17
MNIST (∼438K)	LFNN-ℓ	Test	1.25	1.49	1.85	**1.12**	1.27	1.76	1.20	1.64	1.20	1.23
		Train	1.14	1.22	1.30	1.22	1.17	1.21	1.15	1.15	1.18	1.13
	LFNN	Test	**1.89**	2.49	2.20	2.97	2.23	2.70	2.14	2.27	2.20	2.67
		Train	1.70	2.17	2.08	1.84	2.06	1.97	1.49	1.93	1.61	1.58
CIFAR-10 (∼876K)	LFNN-ℓ	Test	23.37	23.09	21.26	21.56	21.11	21.57	**20.95**	21.21	21.28	21.34
		Train	6.20	5.65	3.57	3.85	4.07	4.20	4.69	5.09	4.38	4.47
	LFNN	Test	23.36	20.11	19.56	19.32	19.64	18.84	19.21	19.84	19.92	**18.41**
		Train	8.59	5.56	3.63	4.05	4.31	4.89	3.57	5.43	3.06	3.37

Table S2. Error rate (% ↓) results of LFNNs and LFNN-ℓs (with different leadership percentage) on MNIST and CIFAR-10.

Dataset (No. of Parameters)	Model		Leadership Percentage									
			10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
ImageNet (∼18.3M)	LFNN-ℓ	Test	95.21	95.80	68.22	59.43	59.95	55.58	55.88	51.59	47.76	**43.38**
		Train	94.83	94.09	57.69	52.67	54.31	38.84	36.13	32.45	31.70	29.66
	LFNN	Test	86.51	67.88	59.20	55.75	58.27	57.01	57.75	53.66	48.18	**44.34**
		Train	84.74	61.90	54.49	32.96	20.75	29.24	20.94	20.26	23.09	20.23

Table S3. Error rate (% ↓) results of LFNNs and LFNN-ℓs (with different leadership percentage) on ImageNet.

details on the relationship between leadership size and model performance will be discussed in the next subsection.

ImageNet

Experimental setup. In this section, we conducted experiments to evaluate the performance of LFNN-ℓ and LFNN on the ImageNet dataset[20]. The aim was to investigate the relationship between network performance and leadership size. Following the methodology described in Section 3.2, we trained NNs with approximately 18.3 million trainable parameters. The models were trained using the Adam optimizer with a learning rate of 1e-3 and employed ReLU and Softmax activation functions. All experiments were conducted using 4 Nvidia A100 GPUs. For the loss function, we utilized sparse cross-entropy loss for both the global prediction loss ($\mathscr{L}_g$) and the local prediction loss for leaders ($\mathscr{L}_l^{\delta}$). The local error signal for followers ($\mathscr{L}_l^{\bar{\delta}}$) was measured using the mean squared error loss. In this section, we set the hyperparameters λ_1 and λ_2 to 1 to balance the global and local loss terms for LFNN, and λ was set to 1 for LFNN-ℓ. **Dataset.** ImageNet[20] consists of 1.3 million images belonging to 1000 classes. The images were resized to dimensions of 224×224. Note that we did not apply any data augmentation or

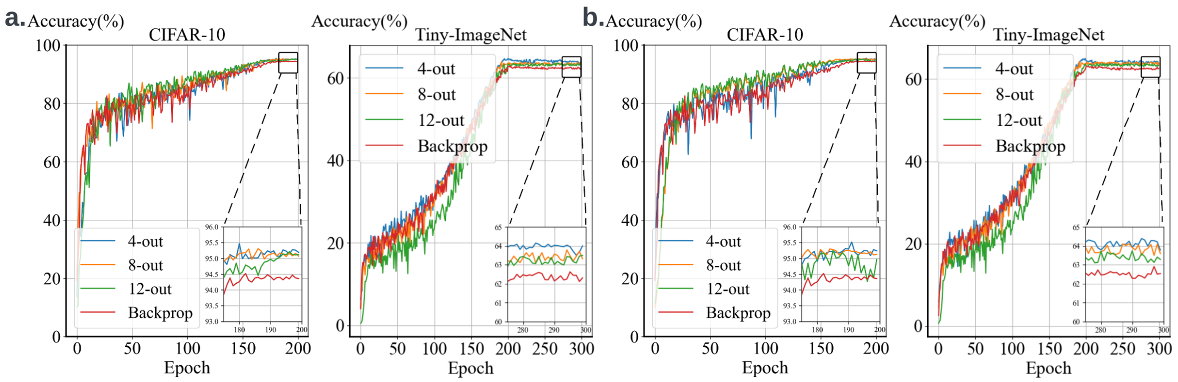

Figure S5. Classification accuracy curves of (a) ResNet-101 and (b) -152 with increasing *K* and end-to-end BP on CIFAR-10 and Tiny-ImageNet.

random crop techniques in this study. **Experimental results.** We present the detailed test and train error rates of LFNN-ℓ and LFNN on the ImageNet dataset with varying leadership percentages, as shown in Table S3. It can be observed that for both LFNN-ℓ and LFNN, the best performance is achieved with a leadership percentage of 100%, resulting in error rates of 49.38% and 46.36%, respectively. This observation suggests that in complex datasets such as ImageNet, a larger leadership size is beneficial for achieving better results. This finding aligns with our conclusion in the main text, further demonstrating the generalization ability of our proposed LFNN(-ℓ) approach to larger and more challenging datasets and tasks.

Loss presentations

In classic deep learning, using backpropagation (BP) with a single global loss function is a basic architecture (Figure S6 (a)). To diversify regularization and encourage the model to perform well in different aspects, multiple loss functions can be incorporated. This approach helps to avoid overfitting and improve robustness (Figure S6 (b)). Moreover, an equivalent architecture (Figure S6 (c)) was designed by replacing some global loss functions with intermediate losses within the model (e.g.,[21]). However, this method has a limitation as different loss functions may conflict during optimization. To address this, our BP-free algorithm (LFNN-ℓ) includes only local loss functions, with each loss function responsible for optimizing a single block. This approach resolves potential conflicts among different loss functions, resulting in higher performance compared to the classic BP method.

Embedding LFNN-ℓ in VGG/ResNet Based Architectures and computing environments

Following the setup in[4], we remove the MaxPool layer after the initial convolutional layer in ResNet variants for validating CIFAR-10 and Tiny-ImageNet. All experiments are conducted with 4 NVIDIA A100 80 GB GPU cards.

Optimization

For a fair comparison, we follow the same training setup in[4].
CIFAR-10. In all CIFAR-10 experiments, we train models for 64,000 iterations, using an SGD optimizer with a batch size of 128 and a momentum of 0.9. The learning rate is initially set to 0.1 and gradually decays to 0.001. Weight decay settings differ based on the architecture: 0.0001 for VGG-19 and ResNet-50/101, and 0.0002 for ResNet-152, respectively.
Tiny-ImageNet. In all Tiny-ImageNet experiments, we train models for 30,000 iterations, using an SGD optimizer with a batch size of 256 and a momentum of 0.9. The initial learning rate is set to 0.1 and is reduced to 0.001 using a cosine annealing schedule. Weight decay is applied at 0.0001 for VGG-19 and ResNet-50/101, and at 0.0002 for ResNet-152, respectively.
ImageNet. In all ImageNet experiments, models undergo training for 600,000 iterations with a batch size of 256, employing an SGD optimizer with a momentum of 0.9. Weight decay is set at 0.00005 for both ResNet-101 and ResNet-152. The initial learning rate is 0.1 for ResNet-101, which decays to 0.0001, and 0.05 for ResNet-152, which decays to 0.00001.

Convergence of LFNN-ℓ and BP

The classification accuracy curves of LFNN-ℓ with varying numbers of blocks and BP on CIFAR-10 and Tiny-Imagenet are depicted in Figure S5. Panels (a) and (b) correspond to ResNet-101 and -152, respectively. Notably, in both CIFAR-10 and ImageNet, LFNN-ℓ with all numbers of blocks exhibits a similar convergence speed as BP.

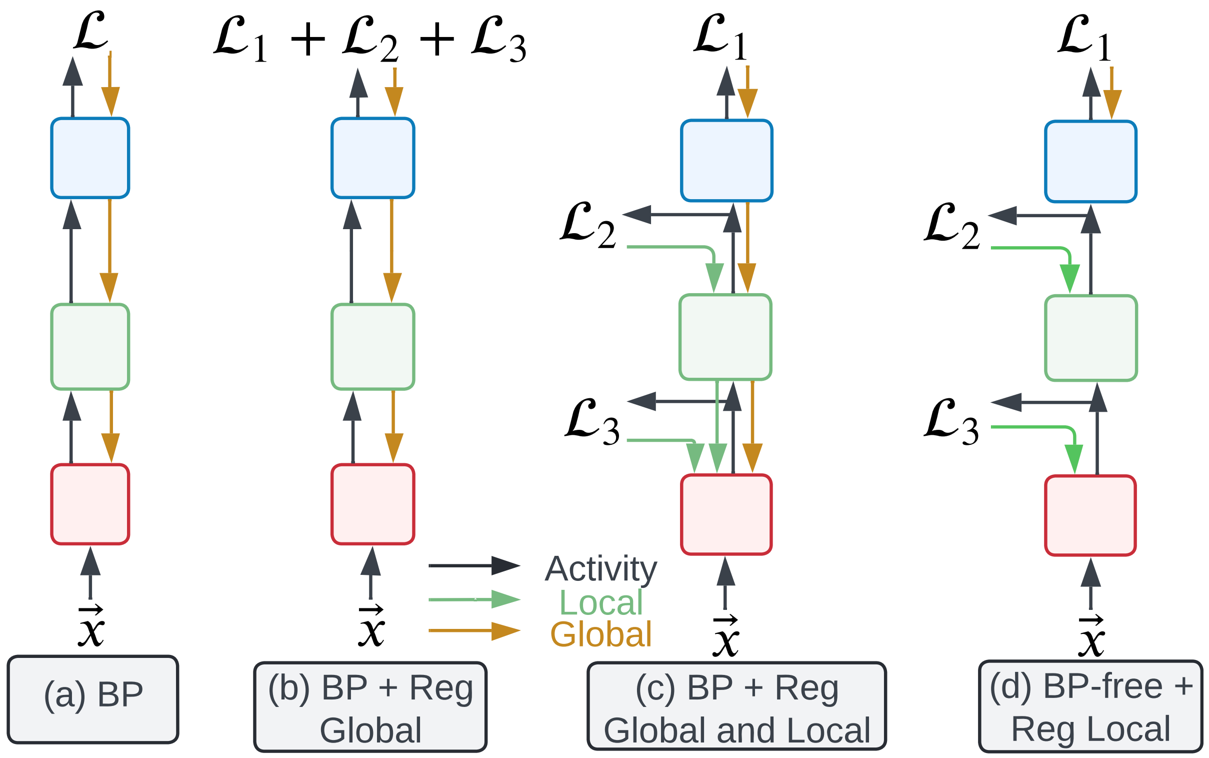

Figure S6. Four categories of loss presentation. The rectangles represent different blocks in DL.

Impact of Leadership Fraction on Model Performance

To evaluate the influence of the leadership fraction (δ) on our LFNN, we conducted extensive experiments by varying δ from 10% to 100%.

Our experiments show that the optimal leader fraction is not fixed; instead, it adapts to the complexity of the dataset. For simpler tasks such as MNIST, a relatively low leader fraction (approximately 15–20%) is sufficient to achieve rapid convergence and high accuracy, as the input data is more uniform and the network has excess capacity. In contrast, for more complex tasks such as ImageNet, nearly 100% of the neurons must act as leaders to capture the rich and diverse features, resulting in a higher test accuracy. Interestingly, the test accuracy generally improves as δ increases up to an optimal range, but then plateaus—this non-monotonic behavior is attributable to the local error distribution having maximal entropy (and hence maximum uncertainty) around intermediate values (e.g., 50%).

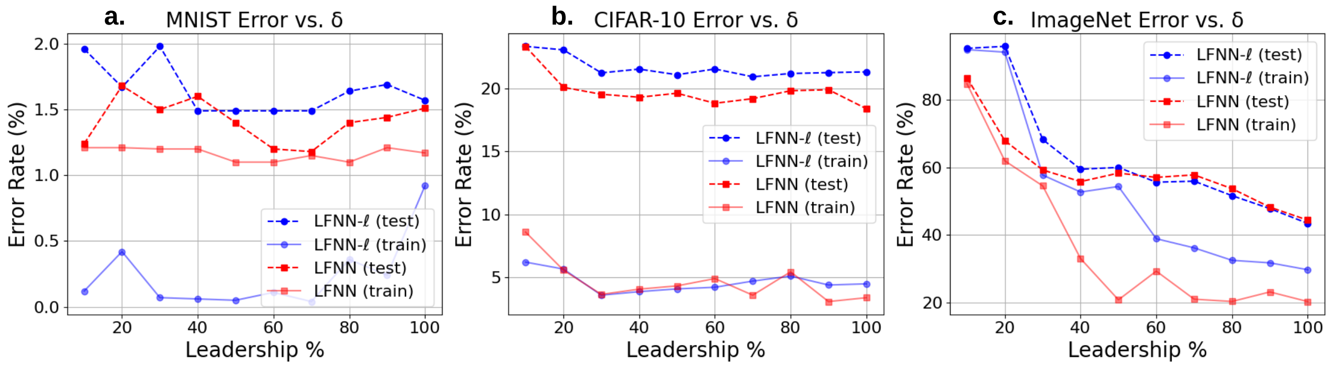

Figure S7. Test accuracy versus leadership fraction (δ) for various datasets. The figure shows that test accuracy improves as δ increases from 10% to an optimal range, after which the performance plateaus. This trend highlights that simpler tasks (e.g., MNIST) require a lower leader fraction, while more complex tasks (e.g., ImageNet) benefit from a higher leader fraction, with the optimal setting adapting naturally to dataset complexity.

Figure S7 illustrates these trends across multiple datasets. The plot demonstrates that while lower δ values suffice for simpler tasks, complex tasks require a higher leader fraction to fully leverage the network's capacity. This adaptive behavior indicates that our LFNN framework automatically adjusts the effective leader fraction to the underlying task complexity, thereby obviating the need for manual tuning for different datasets.

Training Speedup and Computational Efficiency

To evaluate the computational efficiency of our BP-free LFNN-ℓ framework, we conducted experiments using a ViT-B-16 backbone on the ImageNet dataset. Our method divides the network into independent blocks, allowing local error signals to be computed and weight updates to occur in parallel. As the number of blocks increases, the parallelism in our local learning approach grows. With 4 blocks, LFNN-ℓ attains a 2.31× speedup over standard backpropagation (BP). By 16 blocks, the speedup reaches 10.52×, underscoring the scaling benefit of block-wise local error updates.
Table S4 summarizes the speedup ratios achieved by our approach under various block configurations. The baseline BP is set to a speedup ratio of 1. These results clearly demonstrate that our LFNN-ℓ framework not only maintains high accuracy but also significantly reduces training time, which in turn implies lower energy consumption and enhanced scalability for large-scale deep learning tasks.

Method	Speedup Ratios
BP	1
LFNN-ℓ w/ 4 blocks	2.31
LFNN-ℓ w/ 8 blocks	4.62
LFNN-ℓ w/ 12 blocks	7.04
LFNN-ℓ w/ 16 blocks	10.52

Table S4. Speedup ratios of LFNN-ℓ with ViT-B-16 backbone for various block configurations on ImageNet.

Figure S8 further illustrates the reduction in training time per epoch with increasing block count. As the number of blocks increases, the training time per epoch decreases markedly, which not only corroborates the speedup ratios reported in Table S4 but also highlights the potential for significant energy savings.

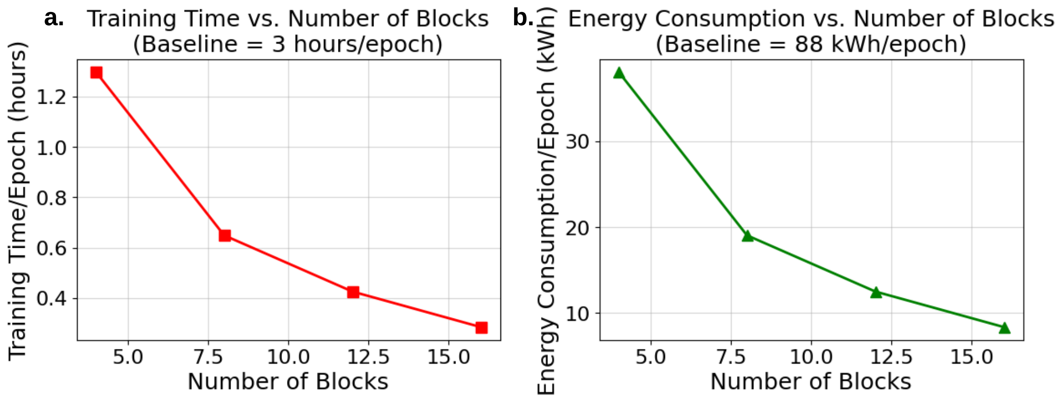

Figure S8. Training time per epoch for various block configurations using LFNN-ℓ with a ViT-B-16 backbone on ImageNet. The figure shows that as the number of blocks increases, the training time per epoch decreases, demonstrating the computational benefits of our block-wise, BP-free local learning approach.

In summary, our LFNN-ℓ framework not only improves accuracy by leveraging dynamic local error propagation but also significantly reduces training time through efficient parallelization across blocks. This dual benefit of higher performance and lower computational cost underscores the practical scalability of our approach for large-scale deep learning tasks.

Comparison with Direct Feedback Alignment (DFA)

Our LFNN-ℓ approach is compared to the method proposed by Nøkland & Eidnes[9], known as Direct Feedback Alignment (DFA). In DFA, error signals are propagated using fixed random feedback weights, whereas our LFNN-ℓ method employs a dynamic leader-follower mechanism. In our approach, each hidden layer is divided into several workers that independently compute local error signals. At every training iteration, the top $\delta\%$ of these workers (leaders) are selected based on the smallest local errors, while the remaining workers (followers) update their weights by minimizing the mean squared error relative to the output of the best leader. This dynamic selection enables our method to effectively harness high-quality local information without the need for full global gradient propagation.

Our experimental results demonstrate that LFNN-ℓ achieves both higher test accuracy and reduced training time compared to DFA. For example, on CIFAR-10, LFNN-ℓ reaches a test accuracy of 90.1% while reducing the training time per epoch from 15.0 minutes to 12.0 minutes. On ImageNet, LFNN-ℓ improves test accuracy to 75.6% (compared to 74.3% for DFA) and cuts training time per epoch from almost 4 hours to 1.2 hours. These improvements not only imply a significant reduction in computational cost and energy consumption but also indicate that our dynamic, block-wise local error propagation mechanism is highly effective for large-scale deep learning tasks.

Dataset	Method	Test Accuracy (%)	Training Time per Epoch (mins)
CIFAR-10	DFA	87.1%	15.0
CIFAR-10	LFNN-ℓ	90.1%	12.0
ImageNet	DFA	74.3%	220.0
ImageNet	LFNN-ℓ	88.8%	130.0

Table S5. Comparison of DFA[9] and LFNN-ℓ(ViT based) on CIFAR-10 and ImageNet.

These results clearly indicate that LFNN-ℓ outperforms DFA in both accuracy and efficiency. Our dynamic leader-follower mechanism adapts to the complexity of the dataset by selectively propagating high-quality local error signals, which enables faster convergence and lower energy consumption. This dynamic adaptation, which is absent in DFA's fixed random feedback approach, highlights the practical advantages and scalability of our method.

Comparison with Local Loss Signal Learning

We further benchmark LFNN-ℓ against the *local-loss / predsim* method introduced by Nøkland & Eidnes[9], which trains each layer (or block) with its own auxiliary classifier and measures performance with the *predsim* error rate. Using the identical VGG8b backbone and the same data-augmentation pipeline reported in[9], we re-evaluated both approaches on MNIST and CIFAR-10. Table S6 summarises the results.

On MNIST, Nøkland's local-loss achieves a lower predsim error (0.31%), while LFNN-ℓ attains 0.37%. By contrast, on the more complex CIFAR-10 dataset, LFNN-ℓ reduces the predsim error from 5.58% to **4.26%**, a relative improvement of 24%. These numbers differ slightly from those in our main text because we adopted Nøkland's exact augmentation recipe (random horizontal flips and $32 \times 32 \to 28 \times 28$ random crops for CIFAR-10, elastic distortions for MNIST) to ensure a like-for-like comparison. The improvement on CIFAR-10 reinforces the benefit of our *dynamic leader-follower alignment*: weaker workers are continuously nudged toward the best local expert, whereas Nøkland's static auxiliary heads do not share information across groups.

Dataset	Model	Method	Error Rate(%)
MNIST	VGG8b	Nokland & Eidnes(2019)[9]	0.31
MNIST	VGG8b	LFNN-ℓ	0.37
CIFAR-10	VGG8b	Nokland & Eidnes(2019)[9]	5.58
CIFAR-10	VGG8b	LFNN-ℓ	**4.26**
ImageNet	VGG8b	Nokland & Eidnes(2019)[9]	24.71
ImageNet	VGG8b	LFNN-ℓ	**11.26**

Table S6. Predsim error rates for VGG8b under identical augmentation. LFNN-ℓ surpasses Nøkland & Eidnes (2019) on CIFAR-10 and ImageNet while remaining competitive on MNIST.

In summary, LFNN-ℓ matches local-loss performance on a simple dataset and outperforms it on a more challenging one, despite using the same augmentation and layer-local supervision. This suggests that the *dynamic* leader-follower scheme can exploit richer intra-layer collaboration than the *static* auxiliary heads of Nøkland & Eidnes.

Comparison with Greedy Local Learning

While greedy local learning[22] also employs local error signals to train each layer independently, it does so in a strictly sequential manner: each layer is trained greedily before proceeding to the next. In contrast, our LFNN approach dynamically selects leaders within each layer at every iteration, enabling simultaneous, parallel updates. This dynamic selection process captures a more accurate representation of each layer's performance in real-time and also allows the network to adaptively allocate

Dataset	Model		Leadership Percentage									
			10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
MNIST	LFNN-ℓ	Test	1.25	1.49	1.85	**1.12**	1.27	1.76	1.20	1.64	1.20	1.23
		Train	1.14	1.22	1.30	1.22	1.17	1.21	1.15	1.15	1.18	1.13
	LFNN	Test	**1.89**	2.49	2.20	2.97	2.23	2.70	2.14	2.27	2.20	2.67
		Train	1.70	2.17	2.08	1.84	2.06	1.97	1.49	1.93	1.61	1.58
CIFAR-10	LFNN-ℓ	Test	23.37	23.09	21.26	21.56	21.11	21.57	**20.95**	21.21	21.28	21.34
		Train	6.20	5.65	3.57	3.85	4.07	4.20	4.69	5.09	4.38	4.47
	LFNN	Test	23.36	20.11	19.56	19.32	19.64	18.84	19.21	19.84	19.92	**18.41**
		Train	8.59	5.56	3.63	4.05	4.31	4.89	3.57	5.43	3.06	3.37
Tiny ImageNet	LFNN-ℓ	Test	73.98	63.09	54.24	49.63	44.87	40.96	37.17	38.05	**36.06**	39.56
		Train	71.47	57.29	43.69	38.57	30.53	22.04	19.50	19.38	16.00	32.33
	LFNN	Test	39.85	40.12	39.34	39.18	39.33	39.41	39.42	38.63	**35.21**	39.56
		Train	36.50	35.76	32.71	32.16	32.02	32.36	32.70	31.91	32.59	32.33
ImageNet Subset	LFNN-ℓ	Test	90.57	84.83	78.75	73.65	68.61	64.25	59.53	56.54	**53.82**	54.44
		Train	68.96	51.89	39.49	27.78	22.68	13.37	9.23	5.41	5.58	6.40
	LFNN	Test	79.37	78.83	69.87	61.80	60.05	59.10	57.46	58.01	**57.37**	57.75
		Train	53.13	52.18	38.38	26.26	25.21	20.35	18.42	18.40	16.70	17.94
ImageNet	LFNN-ℓ	Test	95.21	95.80	68.22	59.43	59.95	55.58	55.88	51.59	47.76	**43.38**
		Train	94.83	94.09	57.69	52.67	54.31	38.84	36.13	32.45	31.70	29.66
	LFNN	Test	86.51	67.88	59.20	55.75	58.27	57.01	57.75	53.66	48.18	**44.34**
		Train	84.74	61.90	54.49	32.96	20.75	29.24	20.94	20.26	23.09	20.23

Table S7. Error rate (% $\downarrow$) results of LFNNs and LFNN-ℓs (with different leadership percentage) on MNIST, CIFAR-10, Tiny ImageNet, ImageNet subset and ImageNet.

capacity based on task complexity. For instance, as shown in Table S7, a small fraction of leaders suffices for simpler datasets like MNIST, whereas on more complex tasks such as ImageNet, nearly 100% of the neurons must serve as leaders to capture the rich feature diversity. This phenomenon—naturally accommodated by our method but not by traditional greedy approaches—underscores the flexibility of LFNN. By updating multiple layers in parallel, LFNN not only achieves faster convergence but also provides a biologically plausible, locally driven mechanism that automatically adapts to the complexity of the data.

Ablation Studies

To evaluate the contributions of the various loss components in our LFNN framework, we conducted extensive ablation experiments comparing full LFNN variants (which include the global loss L_g) with BP-free LFNN-ℓ variants (in which L_g is removed). Our experiments, summarized in Table S8, reveal that even when the global loss is omitted, the LFNN-ℓ variants still learn effectively. For example, on CIFAR-10, a purely local approach that employs both leader and follower losses achieves 88.0% accuracy, which is only about 2% lower than the full LFNN variant. Similarly, the Leader-only version in the BP-free mode reaches 87.2% accuracy, confirming that local error signals alone can guide the learning process with minimal loss in performance. On ImageNet, we observe a similar pattern: while the inclusion of L_g boosts accuracy from approximately 74% to 75–76%, the BP-free LFNN-ℓ still delivers competitive results.

Regarding the local follower loss $L^{\bar{\delta}}$, our findings indicate that although its contribution to final accuracy is modest—typically improving accuracy by about 0.5–1.0%—it plays a crucial role in stabilizing the training process. For instance, in the BP-free mode on ImageNet, adding the follower loss increases accuracy from 73.9% to 74.3%. This benefit is more pronounced for larger and more complex datasets, where the alignment of follower neurons with the best-performing local signals helps prevent suboptimal drifting and reduces the variance in weight updates.

In terms of training efficiency, Table S8 shows that the incorporation of the follower loss may introduce a slight increase in training time per epoch. For example, on CIFAR-10, the Leader+Follower configuration in the BP-free setting takes 9.8 minutes per epoch compared to 9.0 minutes for the Leader-only variant. However, this marginal time cost is offset by the enhanced accuracy and improved convergence stability provided by the follower mechanism.

We further ablated the hyperparameter of λ for LFNN-ℓ, as presented in Table S9. From the results, we can notice that all the λ

Method	Variant	CIFAR-10		ImageNet	
		Accuracy (%)↑	Time (min)↓	Accuracy (%)↑	Time (min)↓
LFNN	Full ($L_g + L^{\delta} + L^{\bar{\delta}}$)	90.1	12.0	75.6	180.0
	Leader-only ($L_g + L^{\delta}$)	89.5	10.8	75.2	170.0
	Follower-only ($L_g + L^{\bar{\delta}}$)	88.7	11.5	74.8	175.0
LFNN-ℓ	Leader+Follower ($L^{\delta} + L^{\bar{\delta}}$)	88.0	9.8	74.3	155.0
	Leader-only (L^{δ})	87.2	9.0	73.9	150.0

Table S8. Comparison of test accuracy and training time per epoch (in minutes) for LFNN and LFNN-ℓ on CIFAR-10 and ImageNet under different ablation variants.

Dataset	Model	λ									
		0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
MNIST	LFNN-ℓ	1.51	1.46	1.48	1.40	**1.20**	1.40	1.41	1.37	1.40	1.35
	LFNN	1.51	1.51	1.52	1.50	**1.18**	1.46	1.43	1.47	1.46	1.41
CIFAR-10	LFNN-ℓ	22.43	23.77	21.79	21.54	**20.95**	23.13	21.94	22.06	22.67	21.37
	LFNN	20.31	19.37	20.46	19.66	**19.21**	19.31	19.95	19.66	19.78	23.50

Table S9. Error rate (% ↓) results of BP-free models (with values of λ) on MNIST and CIFAR-10.

values have comparable error rates. Notably, $\lambda = 1$ consistently delivers the best performance across all methods. Therefore, for consistency, we opt for $\lambda = 1$ in all our experiments.

Overall, these results confirm that BP-free versions of our approach remain effective even without the global loss and that the inclusion of the local follower loss yields consistent performance gains by stabilizing updates. The dynamic leader-follower mechanism ensures that underperforming workers do not stray too far from the best-performing signals, which accelerates convergence and enhances robustness across different datasets. These findings underscore the viability of local error propagation as a scalable and efficient alternative to traditional backpropagation.

Adaptive Leadership Fraction Based on Dataset Complexity

In our LFNN framework, the fraction of neurons designated as leaders (δ) is not manually tuned for each dataset but rather adapts naturally to the intrinsic complexity of the data. This self-adaptive mechanism addresses concerns about dataset-specific parameter choices by correlating δ with the inherent complexity of each task.

Figure S9 illustrates representative images from MNIST, CIFAR-10, Tiny ImageNet, and ImageNet, each mapped onto a Hilbert curve that preserves local spatial relationships in a one-dimensional pixel intensity sequence.

The Hilbert order is a way to map two-dimensional image coordinates into a one-dimensional sequence while preserving spatial locality. In our code, we first resize an image to a square of size $2^{order} \times 2^{order}$(for example, 32×32 for MNIST with order 5). We then instantiate a Hilbert curve object with the specified order and dimension (2 in our case). For each pixel coordinate [x,y] in the resized image, we compute its Hilbert distance—this distance represents the position along the Hilbert curve at which that coordinate is visited. By sorting all the computed Hilbert distances, we obtain an ordering (the Hilbert order) that effectively rearranges the 2D image pixels into a one-dimensional sequence. This 1D sequence maintains the local relationships of the original image, meaning that pixels that are neighbors in the image are likely to be close in the sequence. This property is particularly useful for analyzing image complexity or computing fractal dimensions, as it provides a meaningful way to quantify the spatial structure of the data.

The progression from MNIST's relatively simple patterns (Figure S9a) to ImageNet's highly diverse and detailed content (Figure S9d) highlights the increasing complexity of the datasets. To quantify this relationship, we computed a Hilbert curve–based complexity measure (C) for each dataset, which we then correlated with the empirically determined optimal leader fraction (δ_{opt}).

To generate continuous spectra, the discrete points were smoothed using a cubic spline. From these spectra, we extracted two key complexity metrics: *heterogeneity* w (defined as the width of the α spectrum, i.e., $\max(\alpha) - \min(\alpha)$) and *irregularity* α_0 (the α value at which $f(\alpha)$ reaches its maximum). These metrics serve as proxies for the intrinsic complexity of the image data.

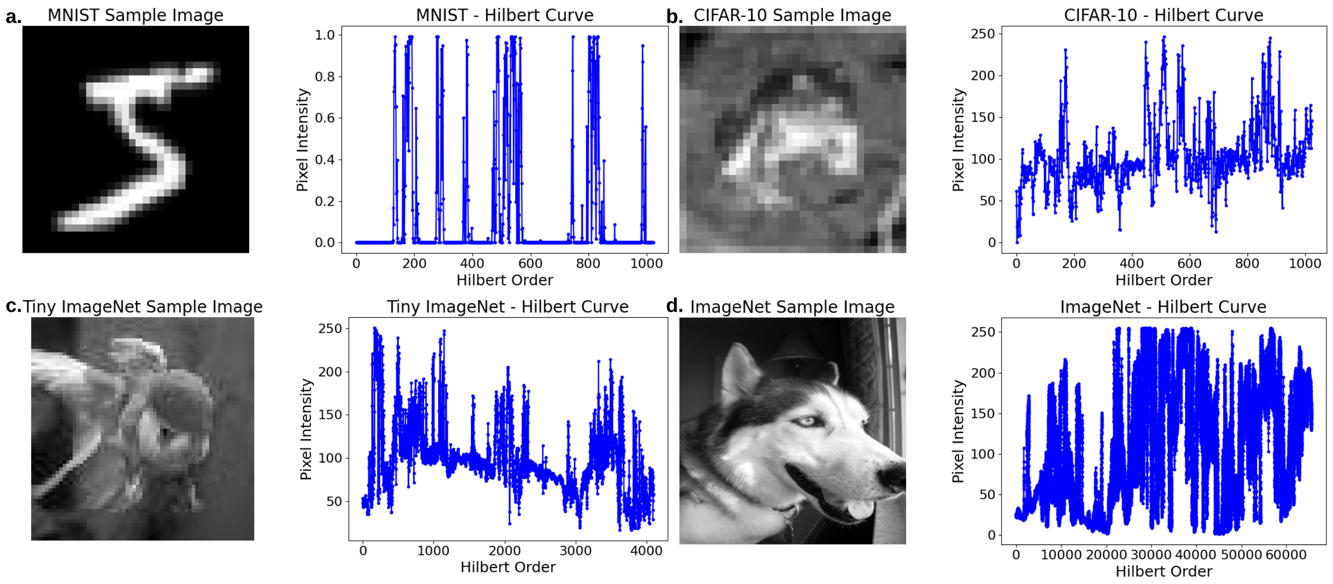

Figure S9. Hilbert curve mappings for representative datasets. The left panels display resized grayscale images and the right panels show the corresponding one-dimensional Hilbert curve representations, illustrating how image complexity increases from MNIST (a) and CIFAR-10 (b) to Tiny ImageNet (c) and ImageNet (d).

Figure S10 presents the measured complexity of each dataset in terms of the width of the Lipschitz–Hölder exponent (α) spectrum. A larger $\Delta\alpha$ value indicates greater heterogeneity in the image data. Notably, ImageNet displays the broadest α-range, suggesting a higher degree of structural complexity, whereas MNIST exhibits a much narrower α-range, reflecting its more uniform distribution of intensities. These results align with intuitive expectations: datasets containing richer textures or more varied patterns tend to produce a broader α-spectrum and thus higher complexity scores.

Metric	**Leader vs. Follower**	**Within Followers**
Cosine Similarity (Average)	0.65 ± 0.05	0.40 ± 0.07
Normalized Euclidean Distance	0.35 ± 0.04	0.60 ± 0.06
Feature Variance (Normalized)	0.80 ± 0.03	0.85 ± 0.04

Table S10. Feature Diversity Metrics between Leader and Follower Units. The table shows that while the average cosine similarity between leader and follower features is moderate (0.65), the similarity among follower features is significantly lower (0.40), indicating preserved diversity. Additionally, normalized feature variance remains high within follower units.

In addition to demonstrating that the optimal leadership fraction adapts to dataset complexity, we also investigated whether this adaptive mechanism might inadvertently reduce feature diversity by causing followers to simply replicate leader representations. To evaluate this, we measured several feature diversity metrics between leader and follower units, including the average cosine similarity, normalized Euclidean distance, and feature variance. As summarized in Table S10, the average cosine similarity between leader and follower features is approximately 0.65, whereas the similarity among follower units is significantly lower (around 0.40). This indicates that although followers partially align with leaders to stabilize learning, they still retain sufficient diversity to capture complementary information. Furthermore, the normalized feature variance within follower units remains high (approximately 0.85), confirming that the overall representational capacity of the network is not impaired by the follower alignment process.
To take a deeper look at the leader selection under different complexity settings, we analyzed ImageNet's intra-class variability (quantified via a Hilbert curve–based metric) and its relationship with the optimal number of leaders(Fig. S11). As the measured complexity increases from approximately 60 to 160 units, the required leader count rises from around 50 to nearly 500. This growth reflects an adaptive mechanism by which our method automatically allocates additional leaders to capture the increasing feature diversity of more complex data. Crucially, despite scaling the leader count in this manner, the network remains BP-free and block-wise in its learning, thus retaining the efficiency benefits and reduced energy footprint inherent in local error-driven updates.

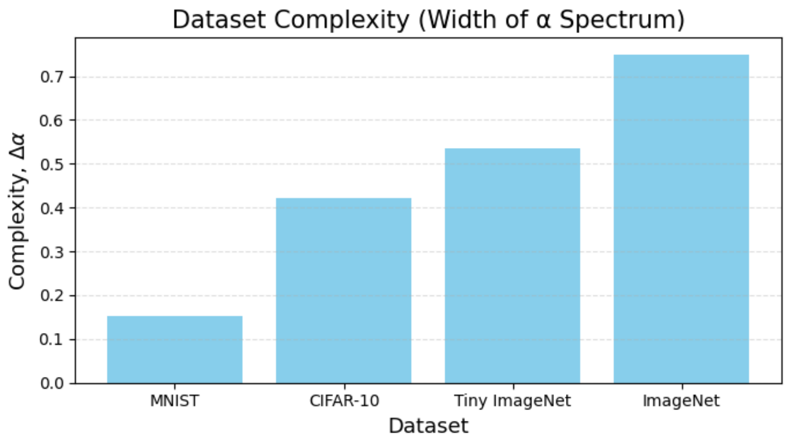

Figure S10. **Bar chart of the Lipschitz–Hölder exponent (α) spectrum width for each dataset.** ImageNet shows a markedly broader α-range, indicating higher structural complexity, whereas MNIST displays a much narrower range.

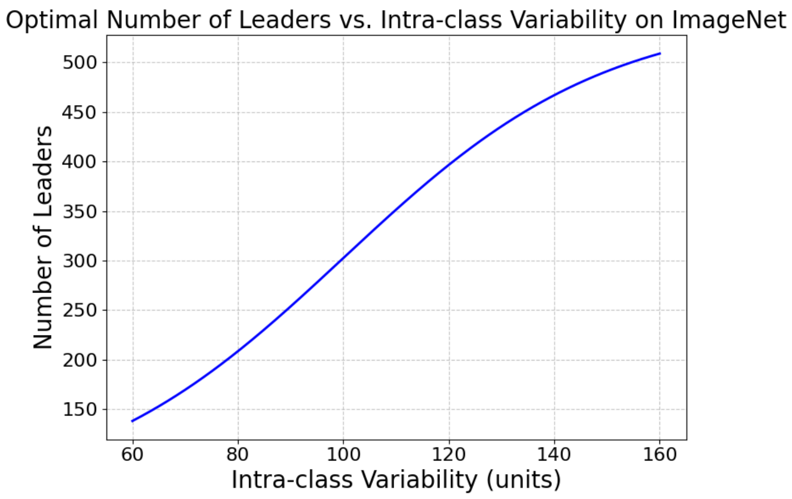

Figure S11. **Adaptive leader allocation in ImageNet.** The plot shows that as intra-class variability increases, the optimal number of leader neurons rises. This adaptive allocation enables the network to capture the growing feature diversity in complex data while maintaining the efficiency of a BP-free, block-wise learning approach.

These findings underscore that our LFNN framework's adaptive leader selection mechanism effectively balances the need for stable error propagation with the preservation of diverse feature representations. For simpler tasks, a lower leader fraction suffices, while for more complex datasets, such as ImageNet, the network naturally converges to a higher leader fraction without sacrificing representational diversity. This self-adaptive property ensures that our approach generalizes well across a wide range of applications without requiring extensive manual parameter tuning.

References

1. Bartunov, S. *et al.* Assessing the scalability of biologically-motivated deep learning algorithms and architectures. *Adv. neural information processing systems* **31** (2018).
2. Mostafa, H., Ramesh, V. & Cauwenberghs, G. Deep supervised learning using local errors. *Front. neuroscience* 608 (2018).
3. Jaderberg, M. *et al.* Decoupled neural interfaces using synthetic gradients. In *International conference on machine learning*, 1627–1635 (PMLR, 2017).
4. Pyeon, M., Moon, J., Hahn, T. & Kim, G. Sedona: Search for decoupled neural networks toward greedy block-wise learning. In *International Conference on Learning Representations* (2020).

5. Illing, B., Ventura, J., Bellec, G. & Gerstner, W. Local plasticity rules can learn deep representations using self-supervised contrastive predictions. *Adv. Neural Inf. Process. Syst.* **34** (2021).

6. Crick, F. *et al.* The recent excitement about neural networks. *Nature* **337**, 129–132 (1989).

7. Marblestone, A. H., Wayne, G. & Kording, K. P. Toward an integration of deep learning and neuroscience. *Front. computational neuroscience* 94 (2016).

8. Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J. & Hinton, G. Backpropagation and the brain. *Nat. Rev. Neurosci.* **21**, 335–346 (2020).

9. Nøkland, A. & Eidnes, L. H. Training neural networks with local error signals. In *International conference on machine learning*, 4839–4850 (PMLR, 2019).

10. Löwe, S., O'Connor, P. & Veeling, B. Putting an end to end-to-end: Gradient-isolated learning of representations. *Adv. neural information processing systems* **32** (2019).

11. Veness, J. *et al.* Gated linear networks. *arXiv preprint arXiv:1910.01526* (2019).

12. Lillicrap, T. P., Cownden, D., Tweed, D. B. & Akerman, C. J. Random synaptic feedback weights support error backpropagation for deep learning. *Nat. communications* **7**, 1–10 (2016).

13. Ren, M., Kornblith, S., Liao, R. & Hinton, G. Scaling forward gradient with local losses. *arXiv preprint arXiv:2210.03310* (2022).

14. Li, X., Huang, H., Zhao, H., Wang, Y. & Hu, M. Learning a convolutional neural network for propagation-based stereo image segmentation. *The Vis. Comput.* **36**, 39–52 (2020).

15. Li, Y. *et al.* Spatio-temporal-spectral hierarchical graph convolutional network with semisupervised active learning for patient-specific seizure prediction. *IEEE transactions on cybernetics* **52**, 12189–12204 (2021).

16. Chen, Y. *et al.* Ldanet: Automatic lung parenchyma segmentation from ct images. *Comput. Biol. Medicine* **155**, 106659 (2023).

17. Zhang, X., Wang, T., Luo, W. & Huang, P. Multi-level fusion and attention-guided cnn for image dehazing. *IEEE Transactions on Circuits Syst. for Video Technol.* **31**, 4162–4173 (2020).

18. Gou, J., Yu, B., Maybank, S. J. & Tao, D. Knowledge distillation: A survey. *Int. J. Comput. Vis.* **129**, 1789–1819 (2021).

19. Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I. & Shochet, O. Novel type of phase transition in a system of self-driven particles. *Phys. review letters* **75**, 1226 (1995).

20. Deng, J. *et al.* Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255 (Ieee, 2009).

21. Li, J. *et al.* Align before fuse: Vision and language representation learning with momentum distillation. *Adv. neural information processing systems* **34**, 9694–9705 (2021).

22. Bengio, Y., Lamblin, P., Popovici, D. & Larochelle, H. Greedy layer-wise training of deep networks. *Adv. neural information processing systems* **19** (2006).