KNOW Center

TU Graz

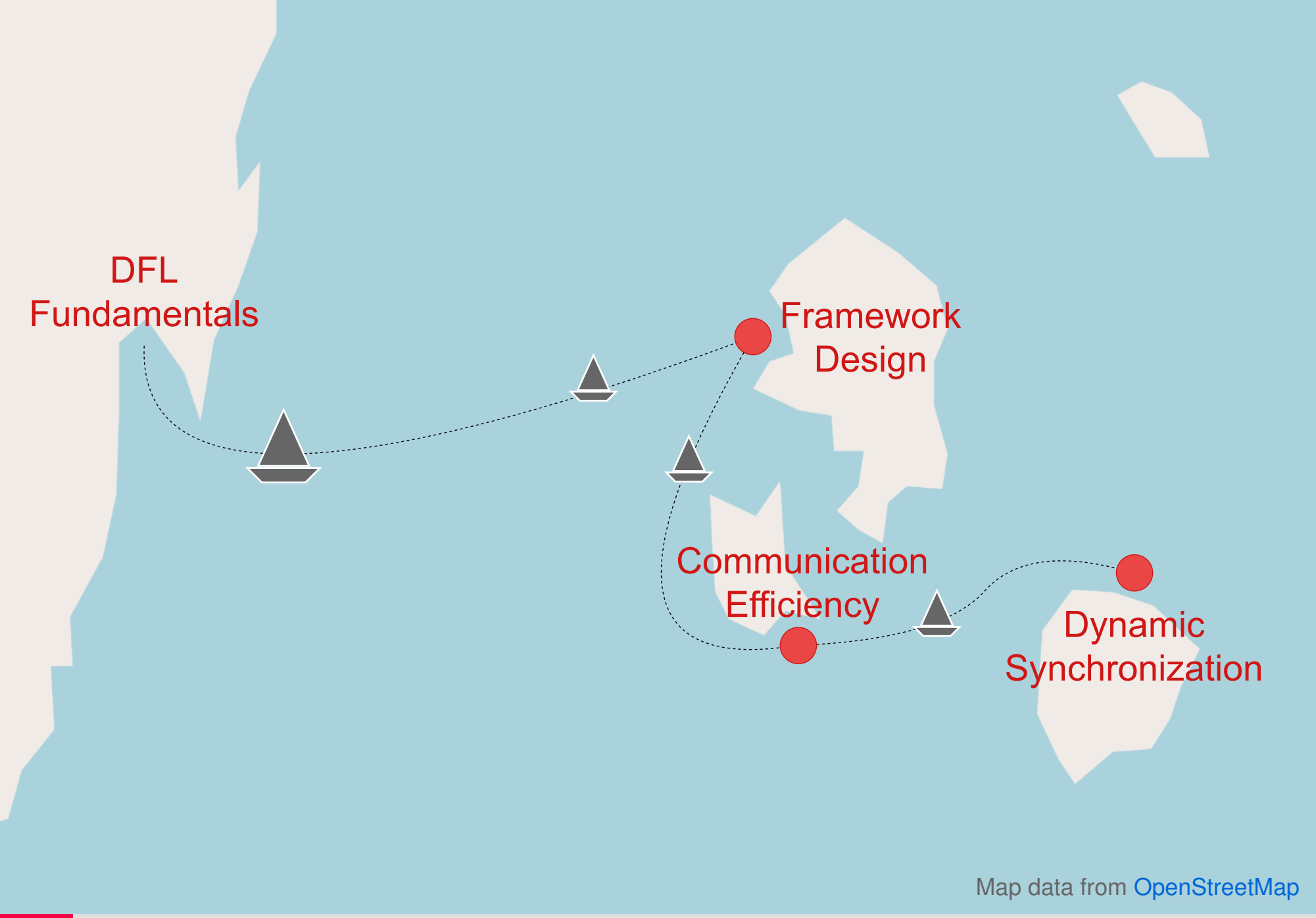
SCIENCE
PASSION
TECHNOLOGY

Decentralized Federated Learning

Framework Design, Communication Efficiency, and Dynamic Synchronization

David Weissteiner

29 January 2026

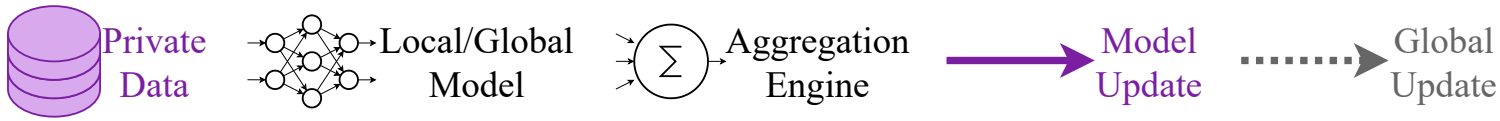

DFL
Fundamentals
Framework
Design
Communication
Efficiency
Dynamic
Synchronization
Map data from OpenStreetMap

DFL Fundamentals

David Weissteiner, Know Center Research GmbH

29 January 2026

Federated Learning (FL)



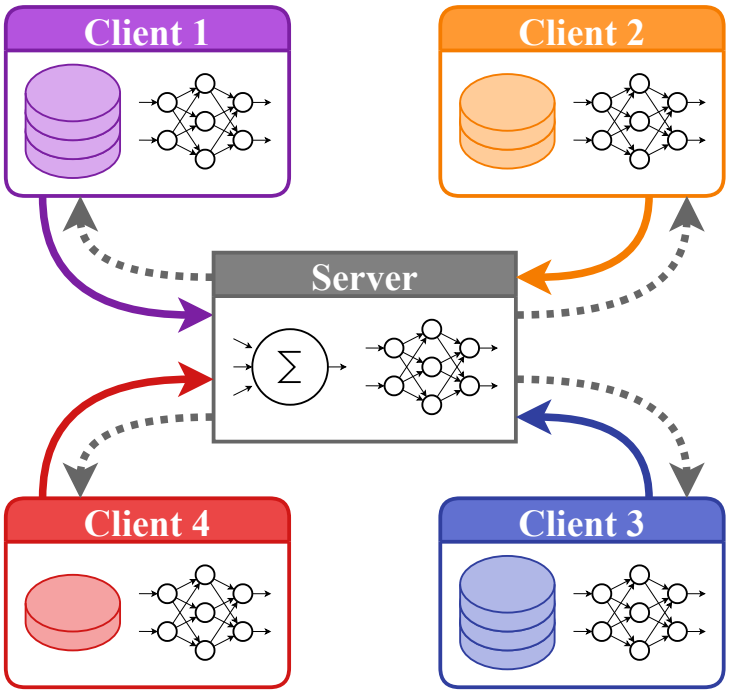
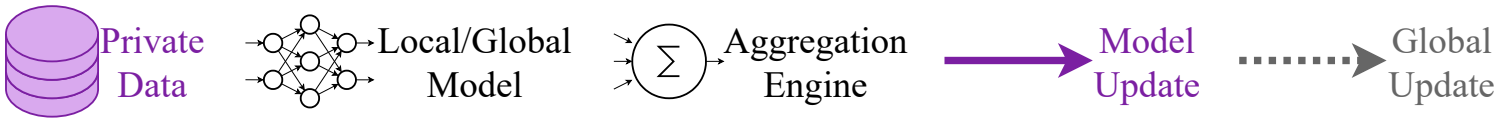


Figure: Centralized FL

McMahan, H.B. et al. (2016). Communication-Efficient Learning of Deep Networks from Decentralized Data. [1]

- Distributed ML paradigm
- Devices w/ data (“Clients”)
- Server aggregates model updates

Decentralized FL (DFL)



- ~~Central server~~
- Arbitrary P2P communication
- "Clients" → "Actors"

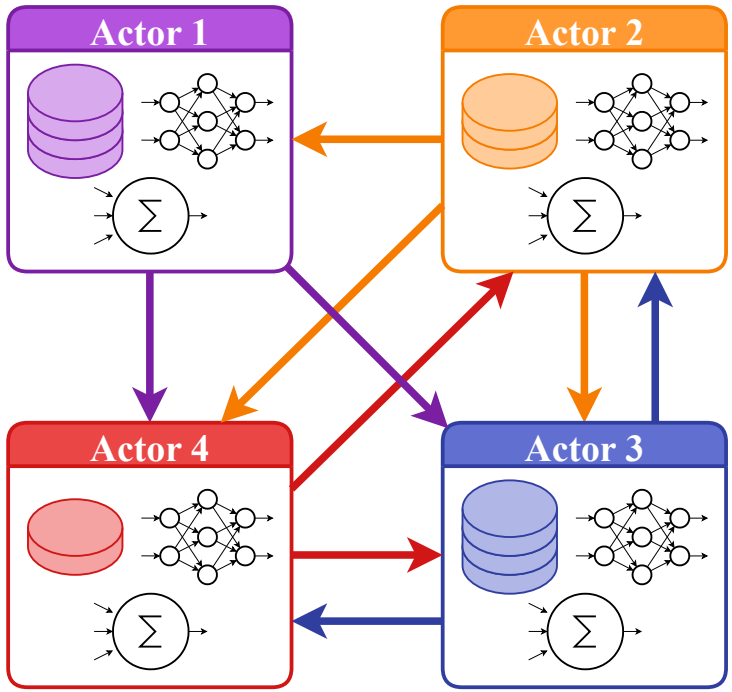


Figure: Decentralized FL

DFL Challenges

- Communication Efficiency
- Security and Privacy
- Data Heterogeneity
- System Heterogeneity

Martínez Beltrán, E.T. et al. (2022). Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges. IEEE Communications Surveys Tutorials, 25, 2983-3013. [2]

DFL Challenges

- Communication Efficiency
- Security and Privacy
- Data Heterogeneity
- System Heterogeneity

Martínez Beltrán, E.T. et al. (2022). Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges. IEEE Communications Surveys Tutorials, 25, 2983-3013. [2]

DFL Challenges

- Communication Efficiency
- Security and Privacy
- Data Heterogeneity
- System Heterogeneity

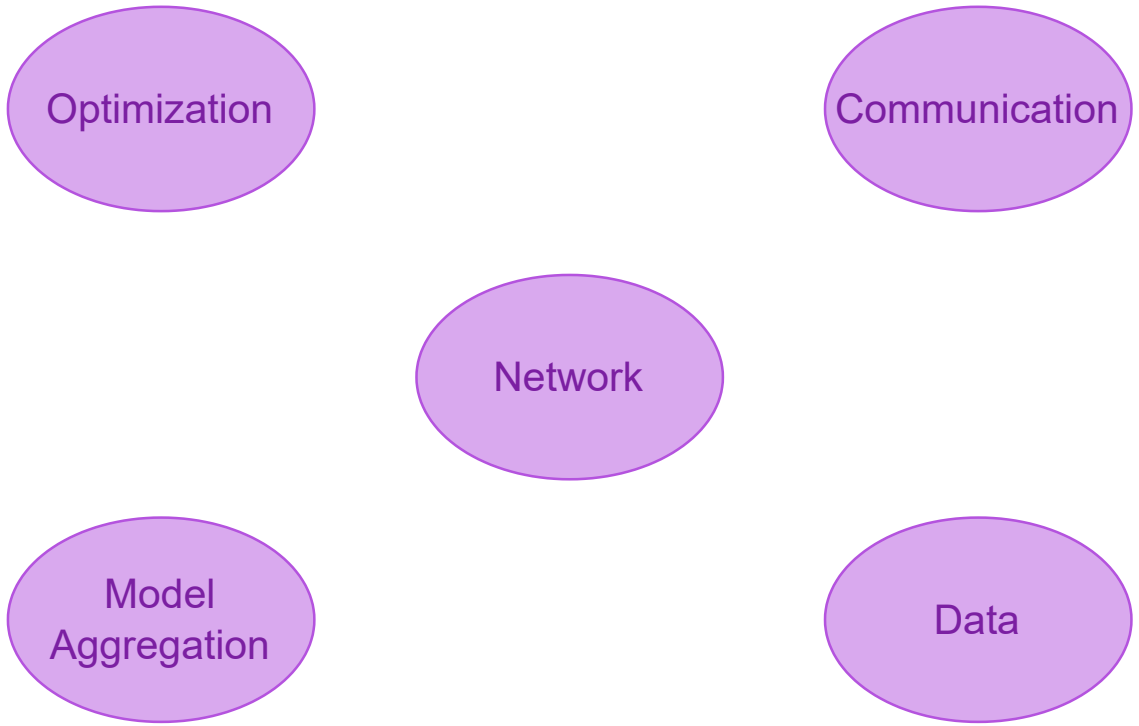
Martínez Beltrán, E.T. et al. (2022). Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges. IEEE Communications Surveys Tutorials, 25, 2983-3013. [2]

DFL Challenges

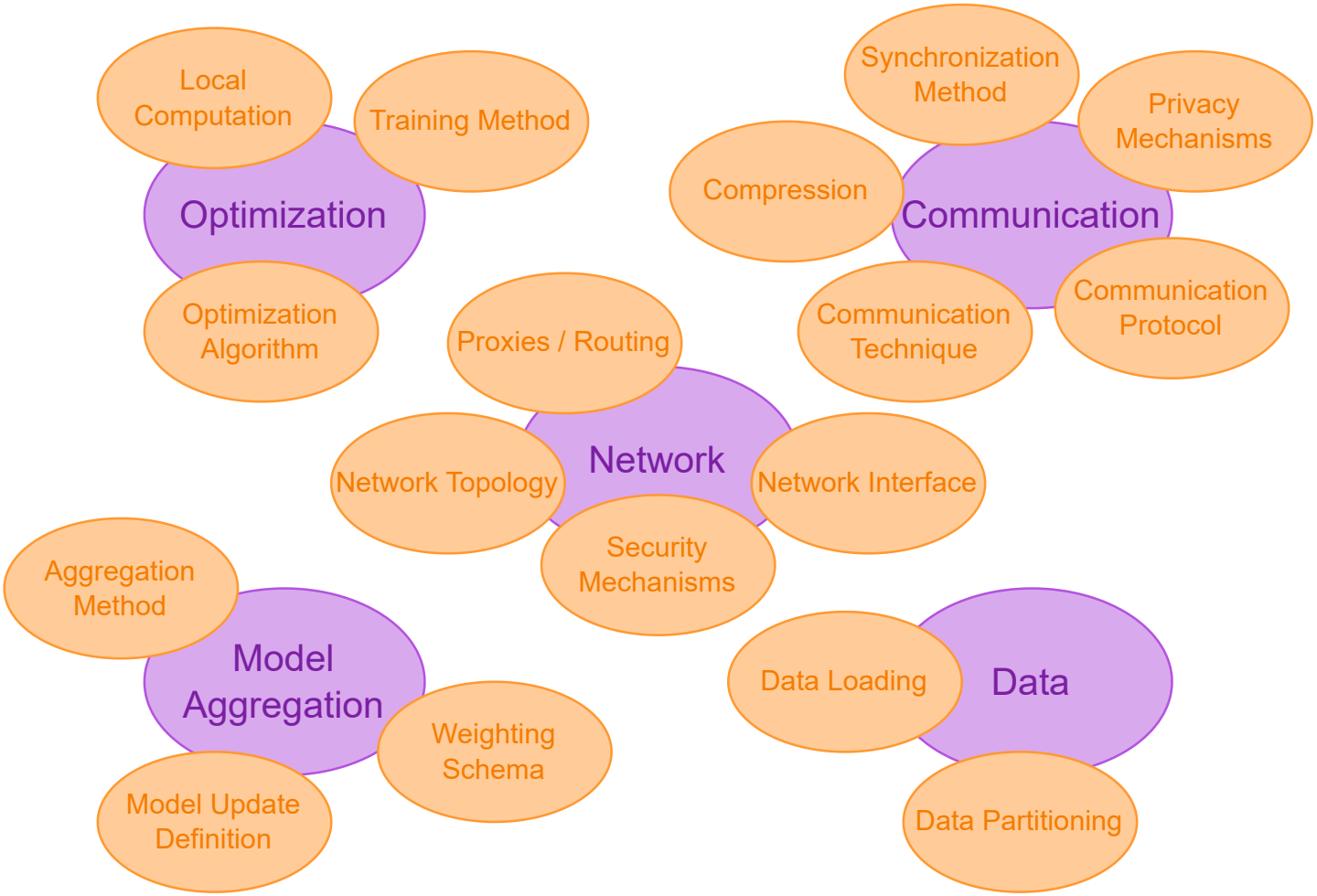
- Communication Efficiency
- Security and Privacy
- Data Heterogeneity
- System Heterogeneity

Martínez Beltrán, E.T. et al. (2022). Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges. IEEE Communications Surveys Tutorials, 25, 2983-3013. [2]

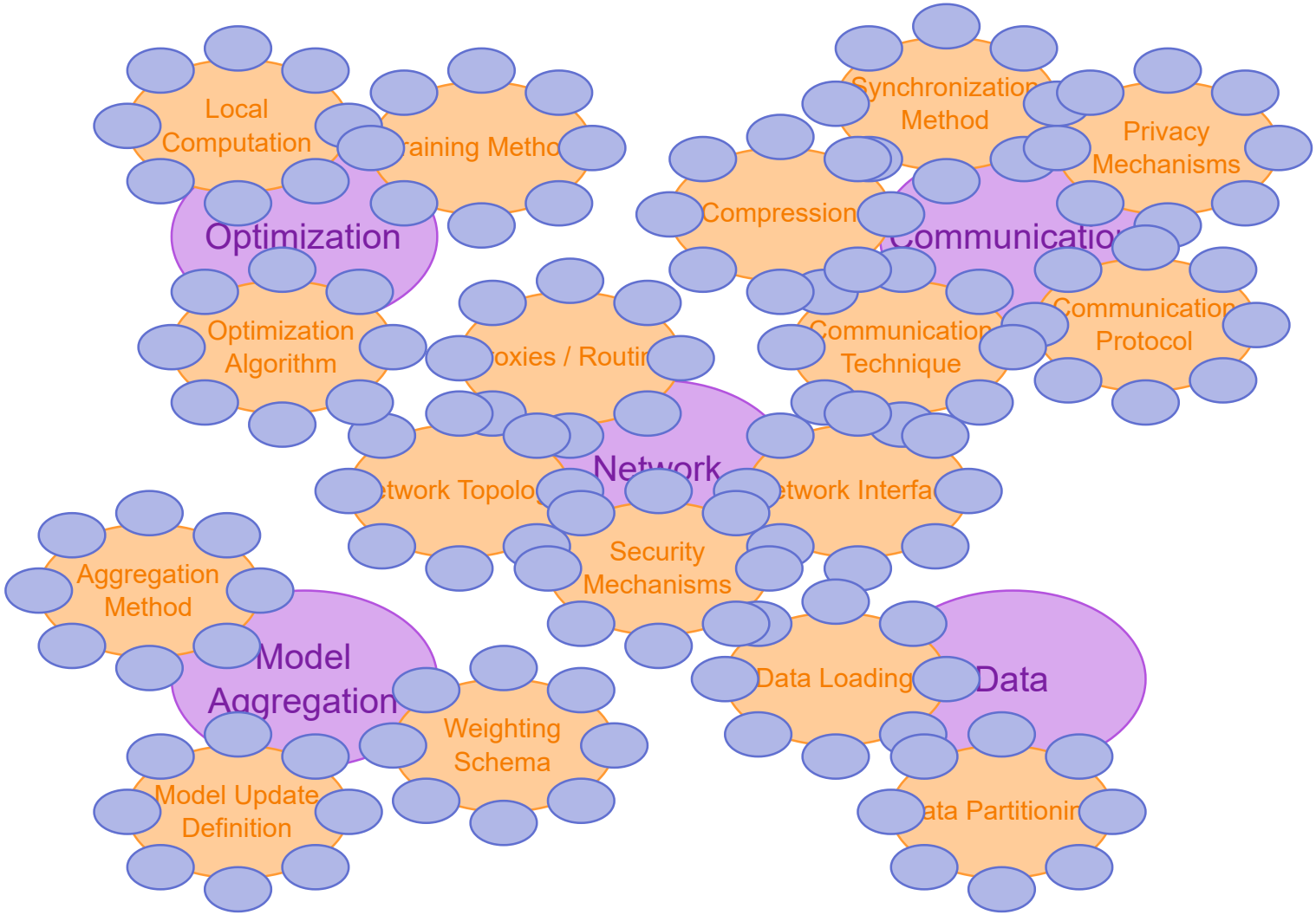
DFL System Components



DFL System Components



DFL System Components

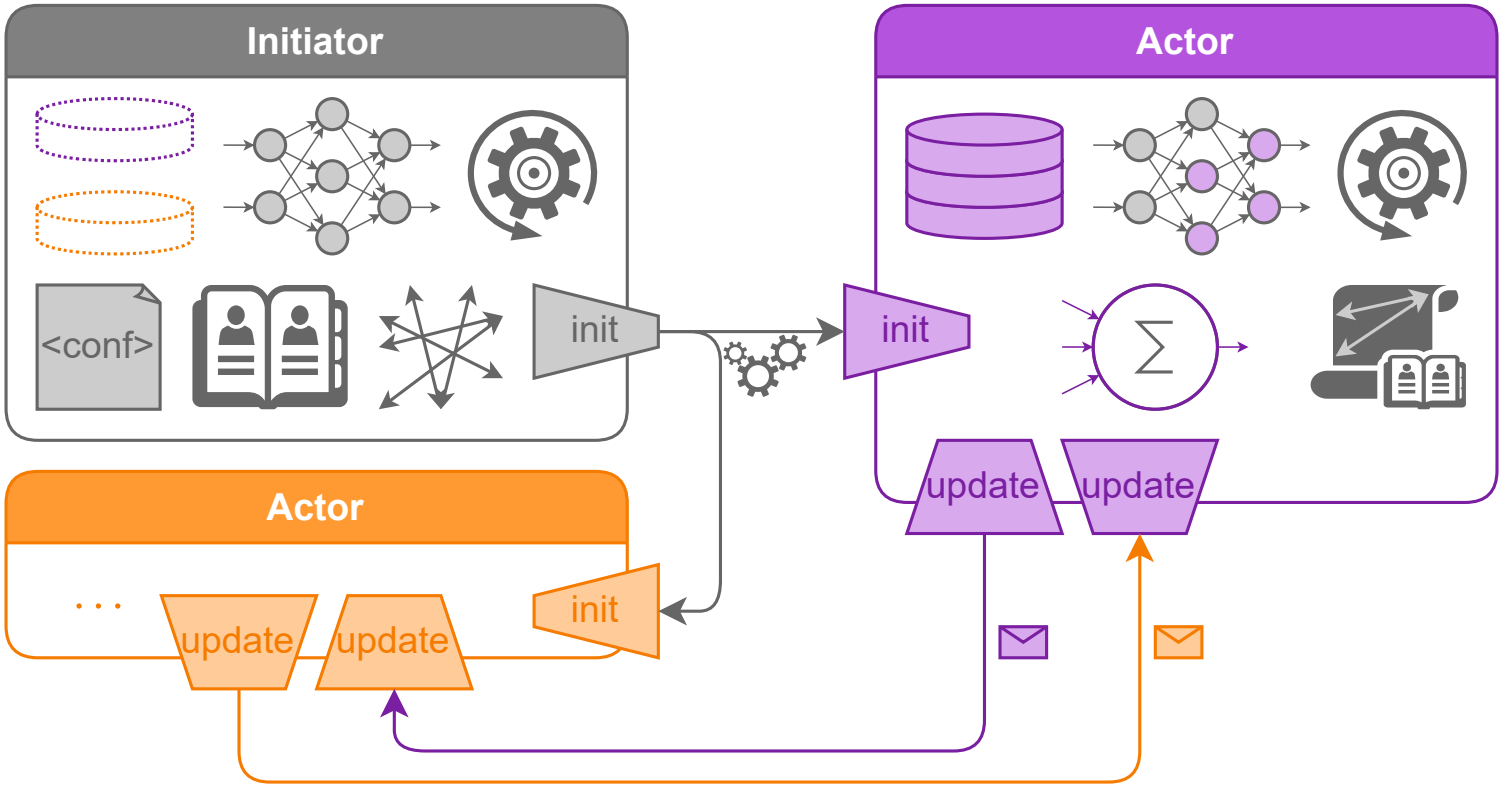


Framework Design

Introduction

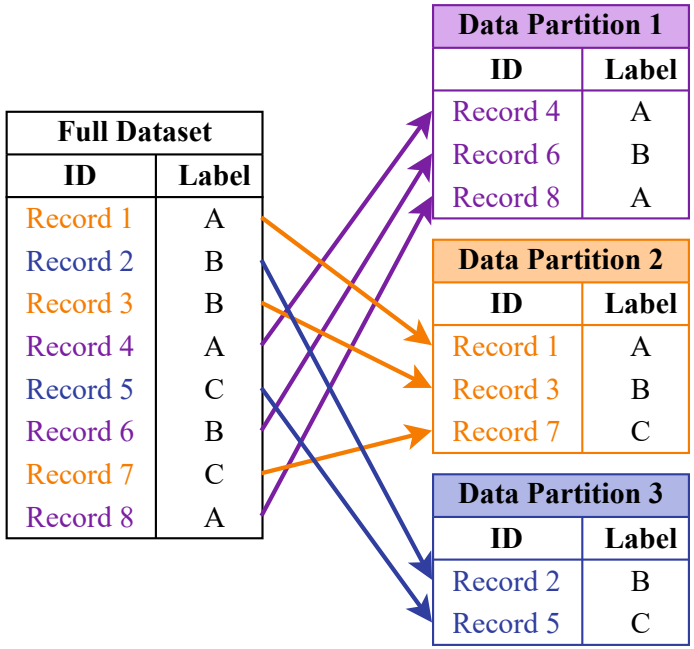
- **Mo**dular **De**centralized **F**ederated **L**earning framework – MoDeFL
- Goal
  - Facilitate experimental comparison of DFL methods
  - Simple integration of novel techniques

MoDeFL Structure

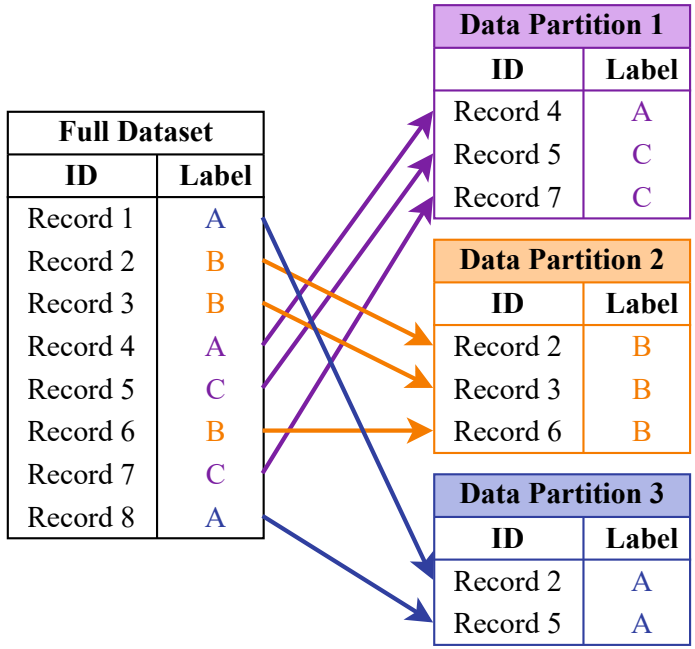


MoDeFL Module Example

- Data partitioning:
  Range, Random, Round-Robin, Dirichlet



(a) Random



(b) Dirichlet

Configuration

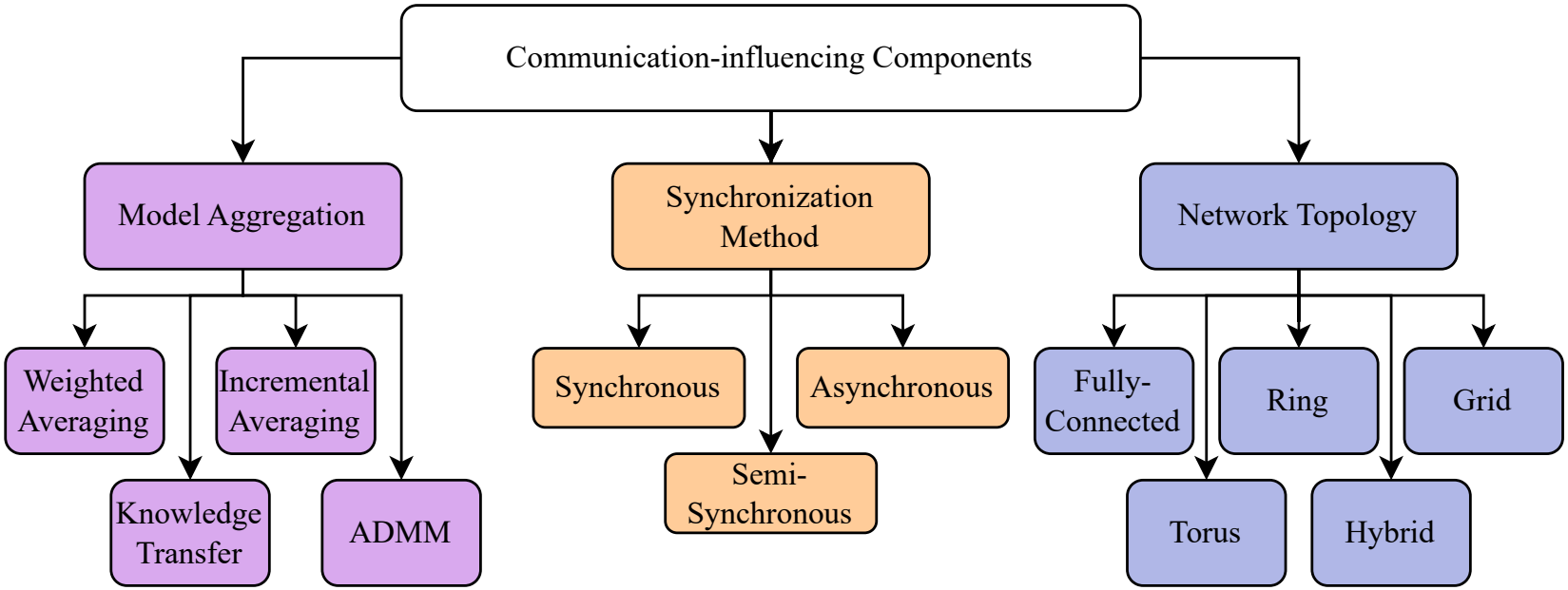
- Configuration file or CLI options
- Options (Common, Actor, Initiator)
  - System — `seed`, `num_threads_server`
  - Logging — `log_level`, `log_*`
  - Network — `addr_file`, `adj_file`
  - Learning — `dataset_id`, `num_*_epochs`, `lr*`
  - Strategy — `learning_strategy`, `sync_strat*`, `compression_*`, `pdp_*`

Configuration

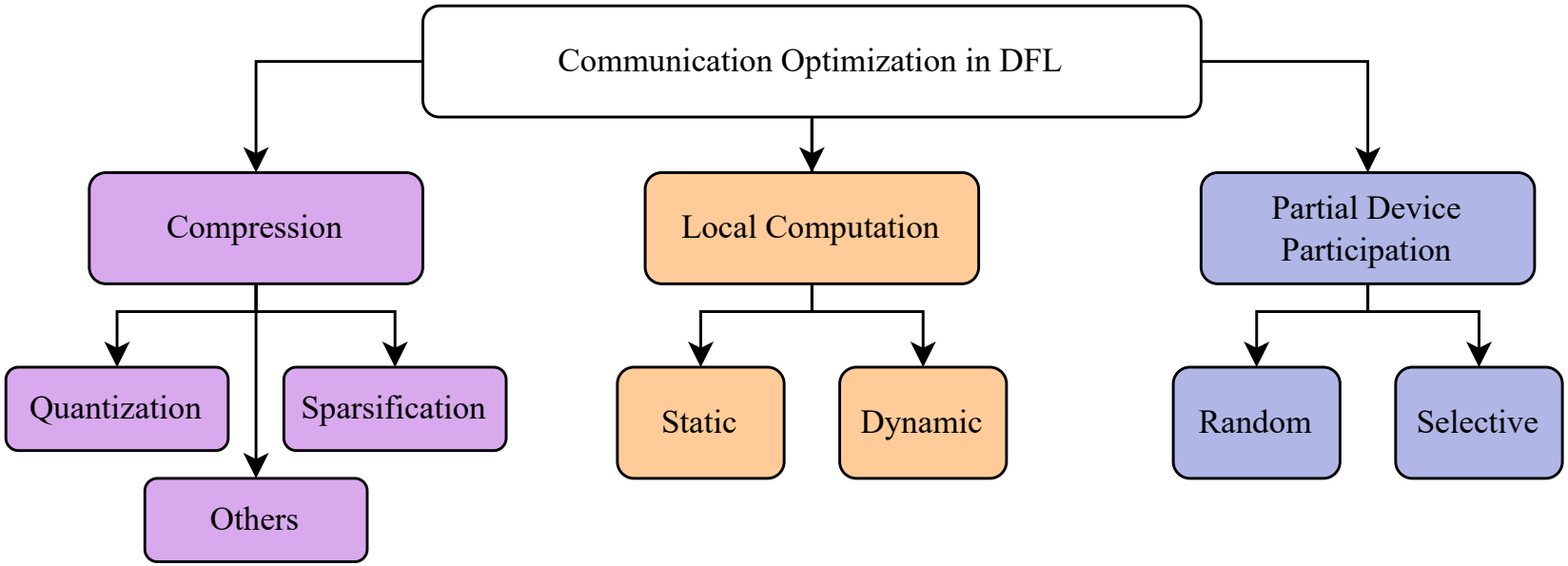
- Configuration file or CLI options
- Options (Common, Actor, Initiator)
  - System — `seed`, `num_threads_server`
  - Logging — `log_level`, `log_*`
  - Network — `addr_file`, `adj_file`
  - Learning — `dataset_id`, `num_*_epochs`, `lr*`
  - Strategy — `learning_strategy`, `sync_strat*`, `compression_*`, `pdp_*`

Communication Efficiency

Taxonomy I


Communication-influencing Components
Model Aggregation
Synchronization Method
Network Topology
Weighted Averaging
Incremental Averaging
Knowledge Transfer
ADMM
Synchronous
Asynchronous
Semi-Synchronous
Fully-Connected
Ring
Grid
Torus
Hybrid

Taxonomy II


Communication Optimization in DFL
Compression
Local Computation
Partial Device Participation
Quantization
Sparsification
Others
Static
Dynamic
Random
Selective

Discussion

- Trade-offs
  - Security and privacy × communication efficiency
  - Data heterogeneity × communication efficiency
  - System heterogeneity × communication efficiency
- Underexplored research directions
  - e.g., dynamic synchronization

Dynamic Synchronization

Challenges

1. Communication cost minimization
   - Omit synchronization steps
2. Divergence detection
   - Synchronization is needed when actors diverge
3. Limited communication
   - Actor is limited to the information exchanged during synchronization

Challenges

1. Communication cost minimization
   - Omit synchronization steps
2. Divergence detection
   - Synchronization is needed when actors diverge
3. Limited communication
   - Actor is limited to the information exchanged during synchronization

Challenges

1. Communication cost minimization
   - Omit synchronization steps
2. Divergence detection
   - Synchronization is needed when actors diverge
3. Limited communication
   - Actor is limited to the information exchanged during synchronization

Idea

- Speculate on the next synchronized gradient

Example

Step . . . step . . . next step?

- Actors develop similarly $\Longrightarrow$ no synchronization
- “Gradient Thresholding”

Idea

- Speculate on the next synchronized gradient

Example

Step . . . step . . . next step?

- Actors develop similarly $\Longrightarrow$ no synchronization
- “Gradient Thresholding”

Idea

- Speculate on the next synchronized gradient

Example

Step . . . step . . . next step?

- Actors develop similarly $\Longrightarrow$ no synchronization
- “Gradient Thresholding”

Idea

- Speculate on the next synchronized gradient

Example

Step . . . step . . . next step?

- Actors develop similarly $\Longrightarrow$ no synchronization
- “Gradient Thresholding”

Illustration – Epoch 1

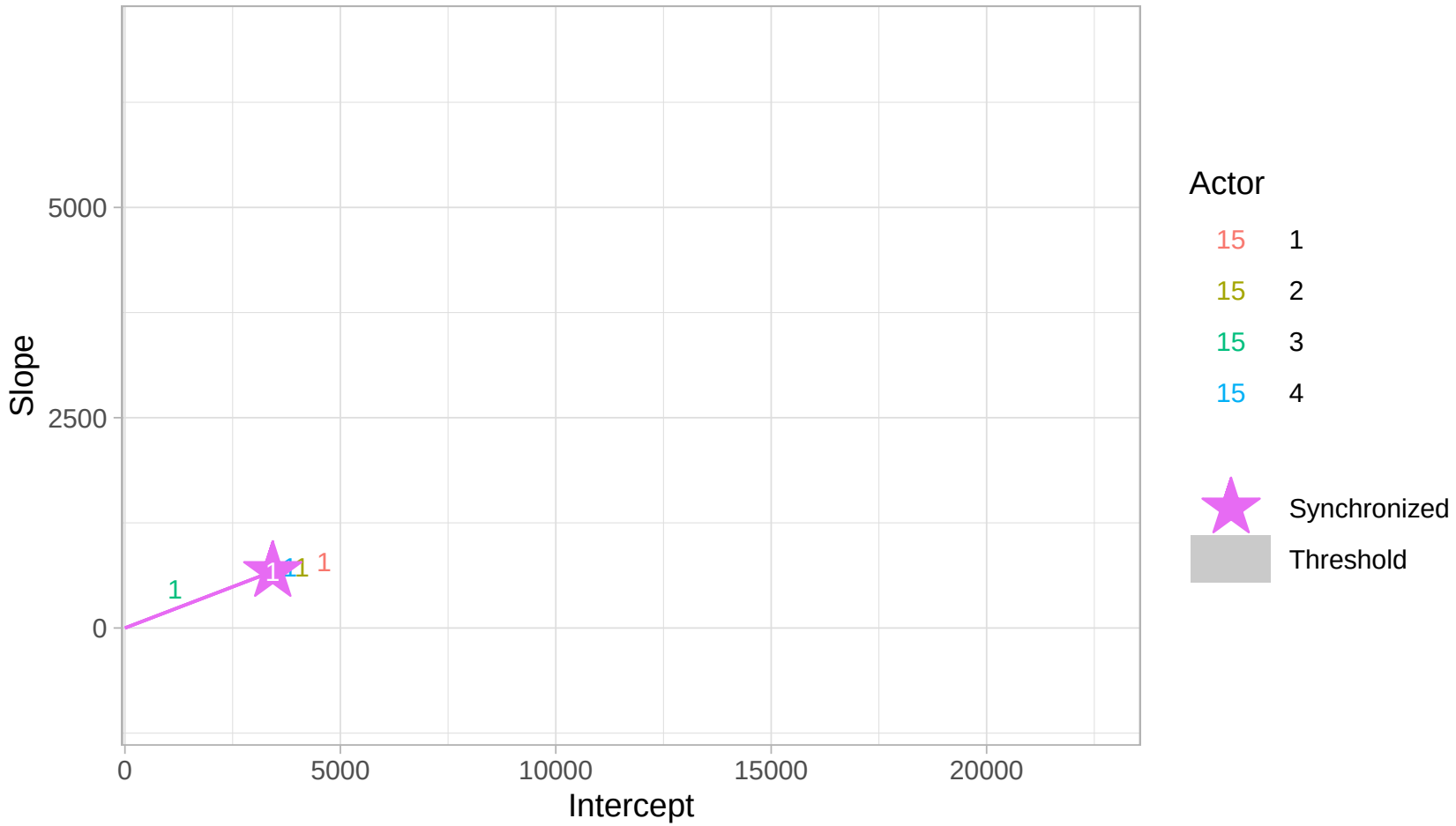


Illustration – Epoch 2

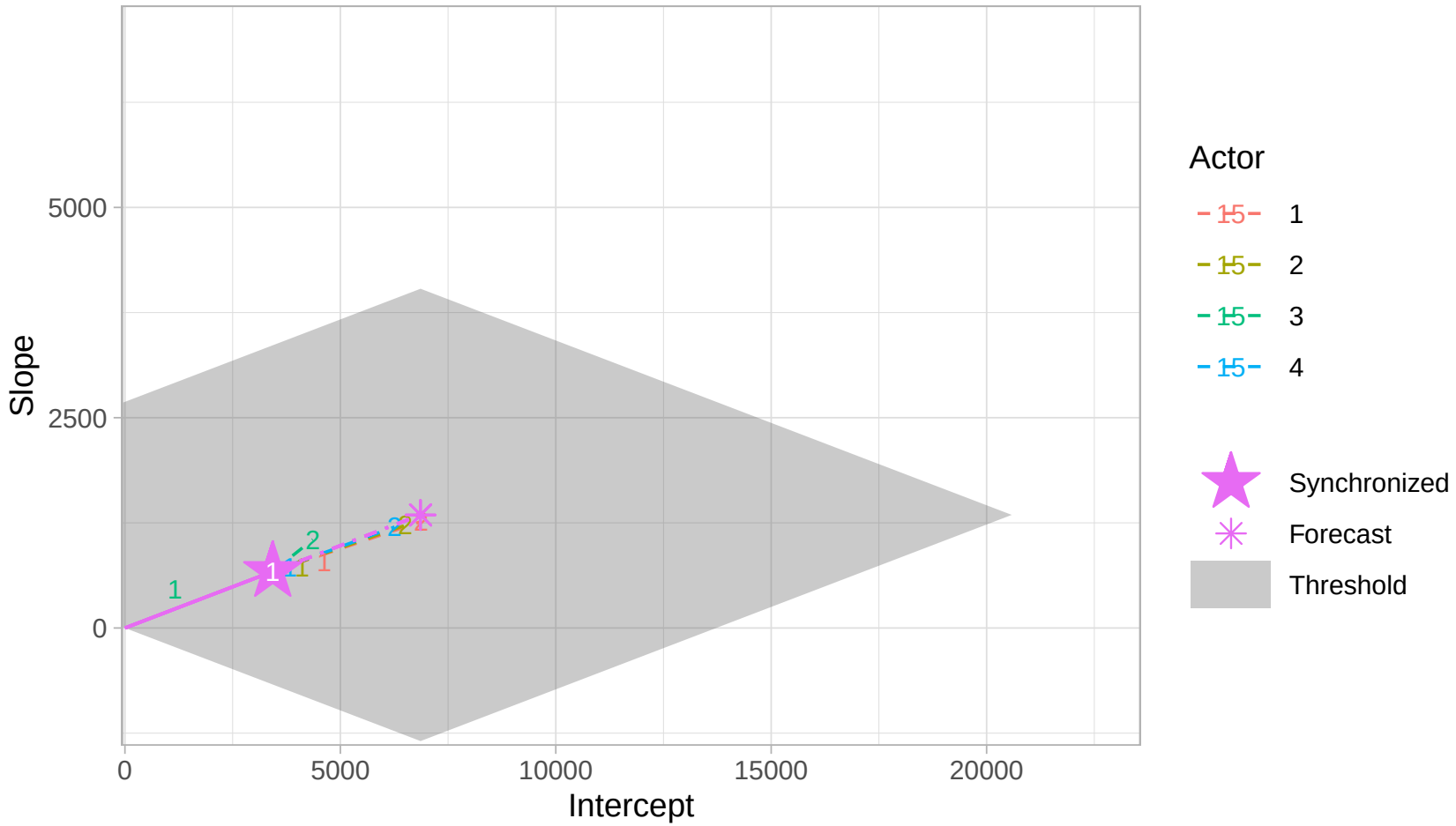


Illustration – Epoch 3

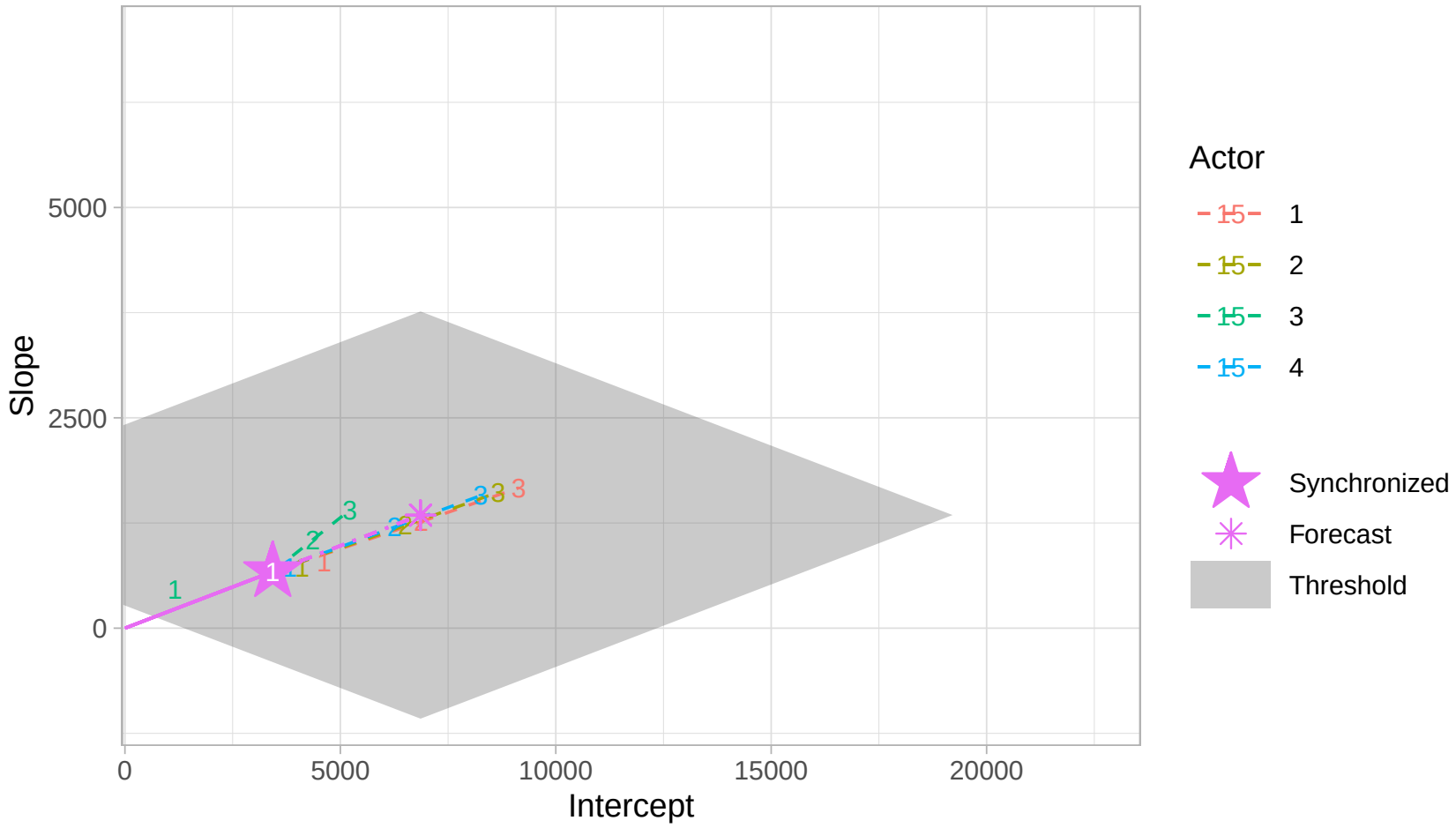


Illustration – Epoch 4

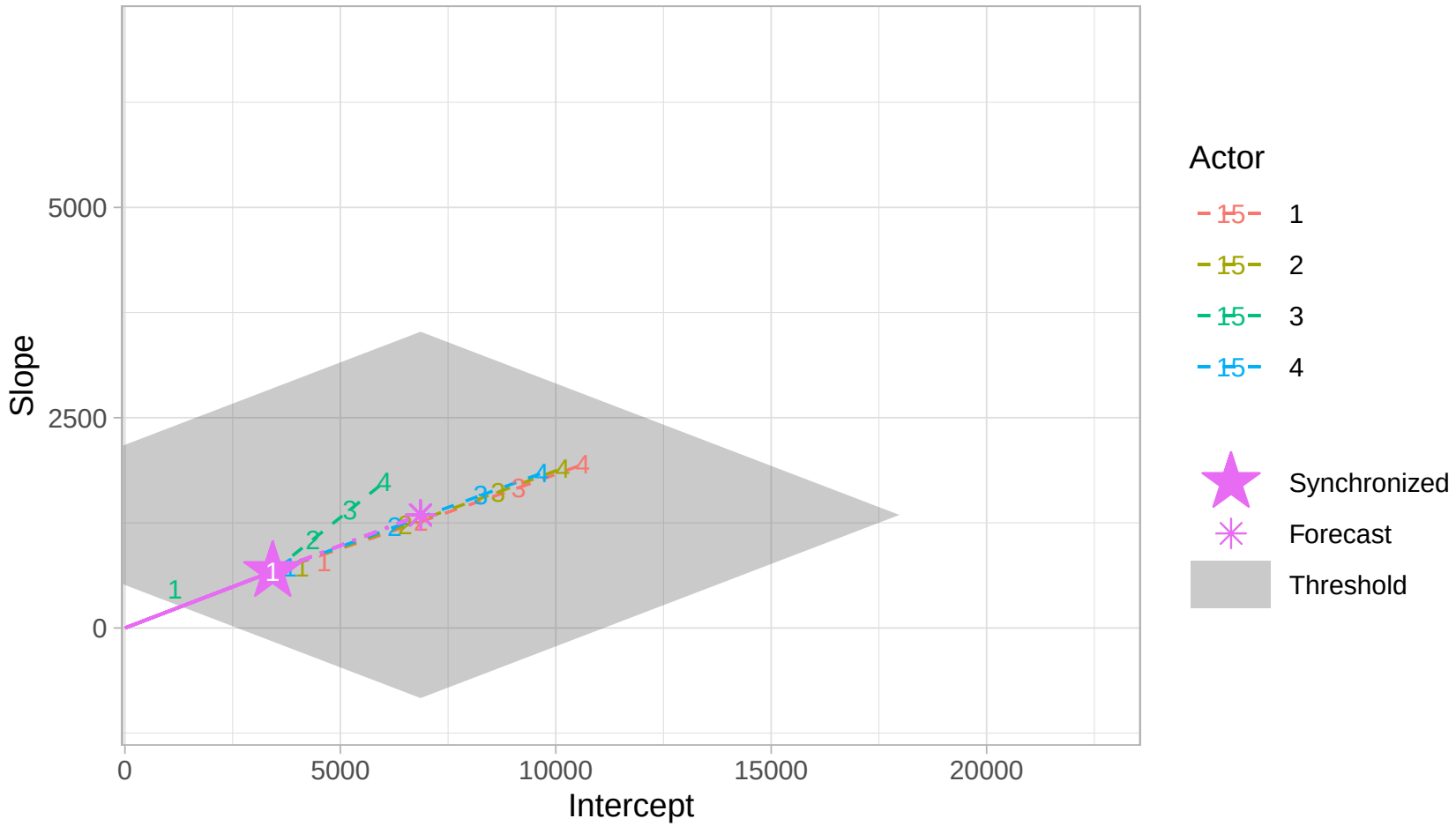
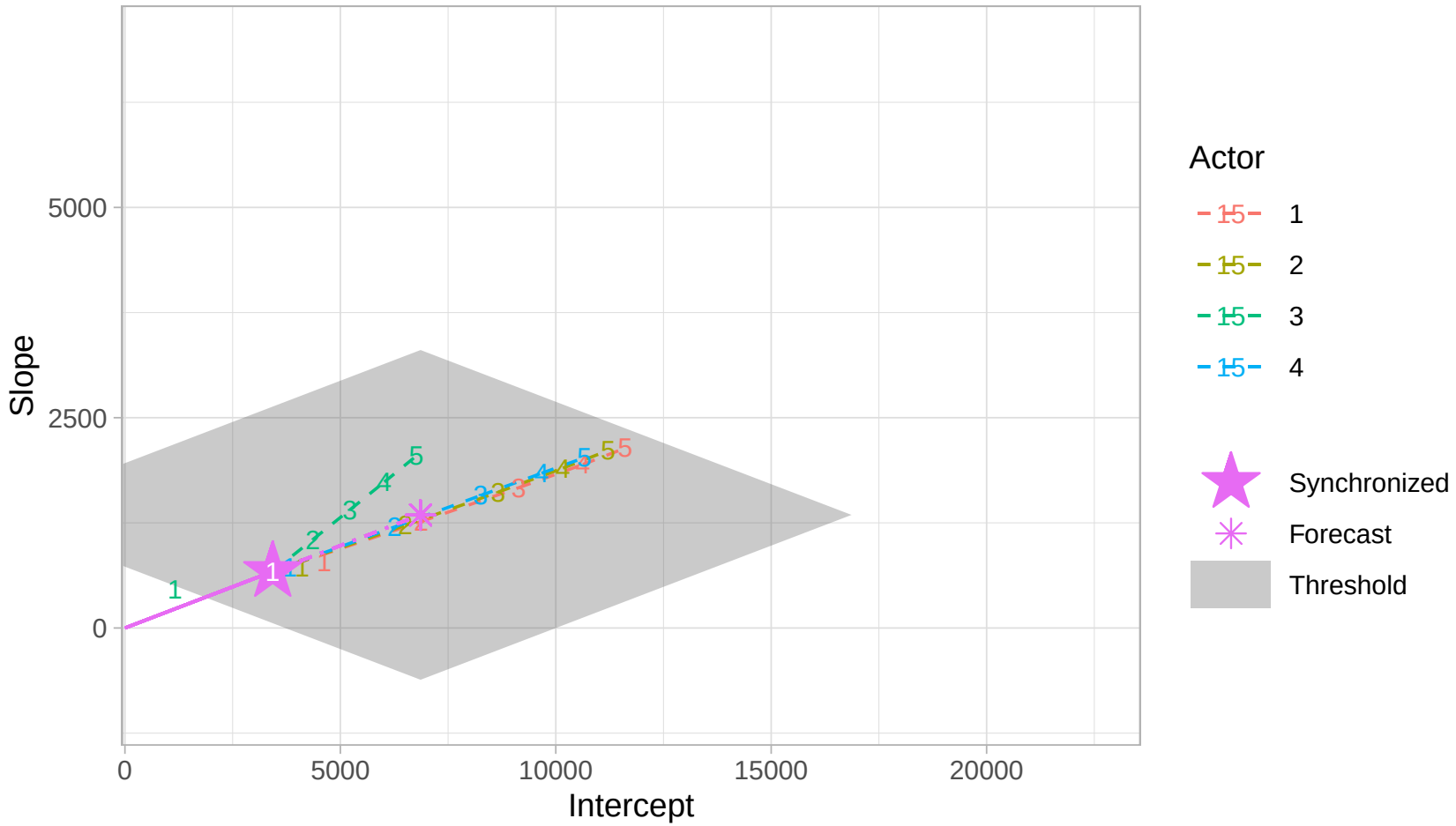
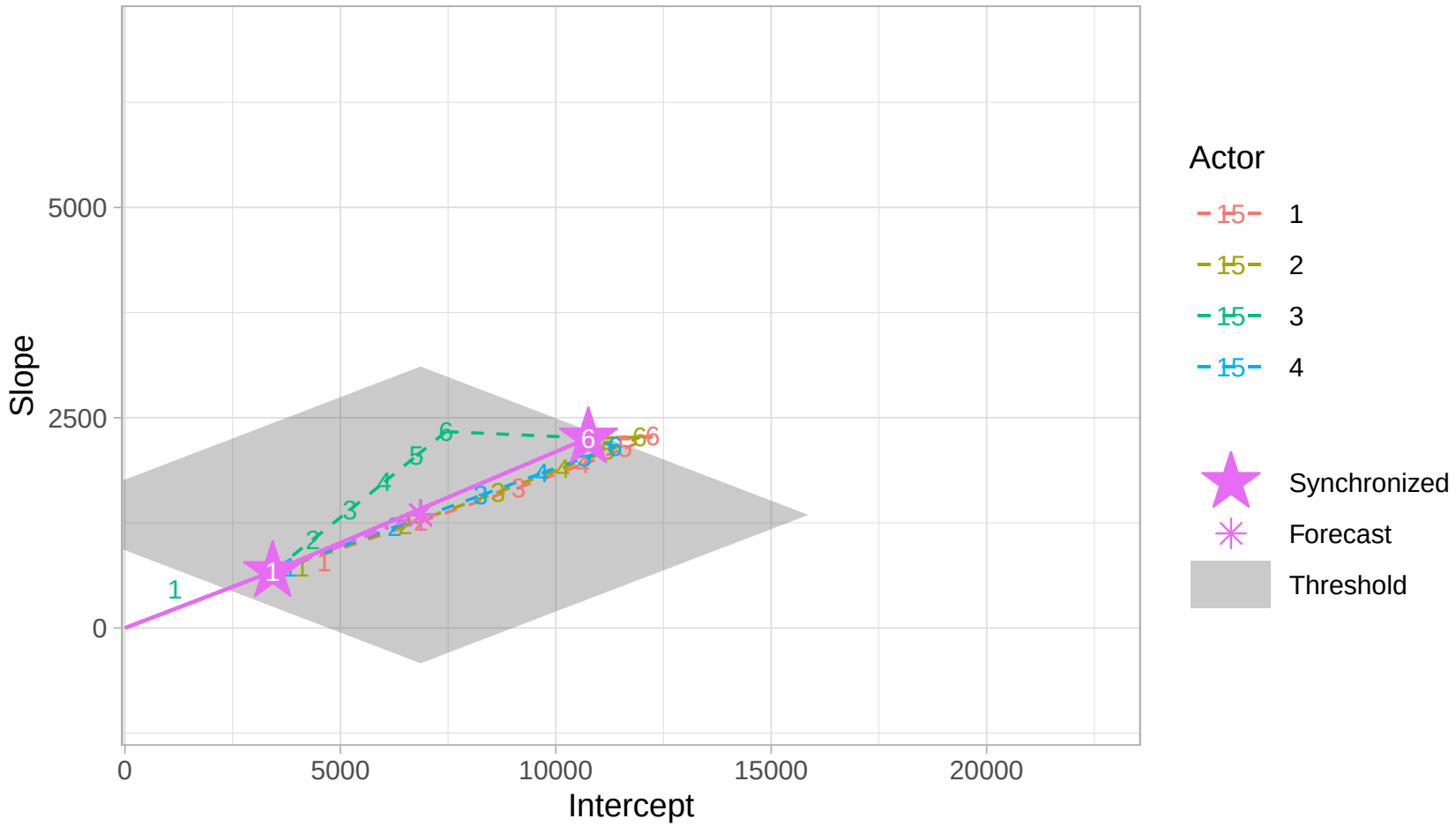


Illustration – Epoch 5



David Weissteiner, Know Center Research GmbH

Illustration – Epoch 6



David Weissteiner, Know Center Research GmbH

Illustration – Epoch 7

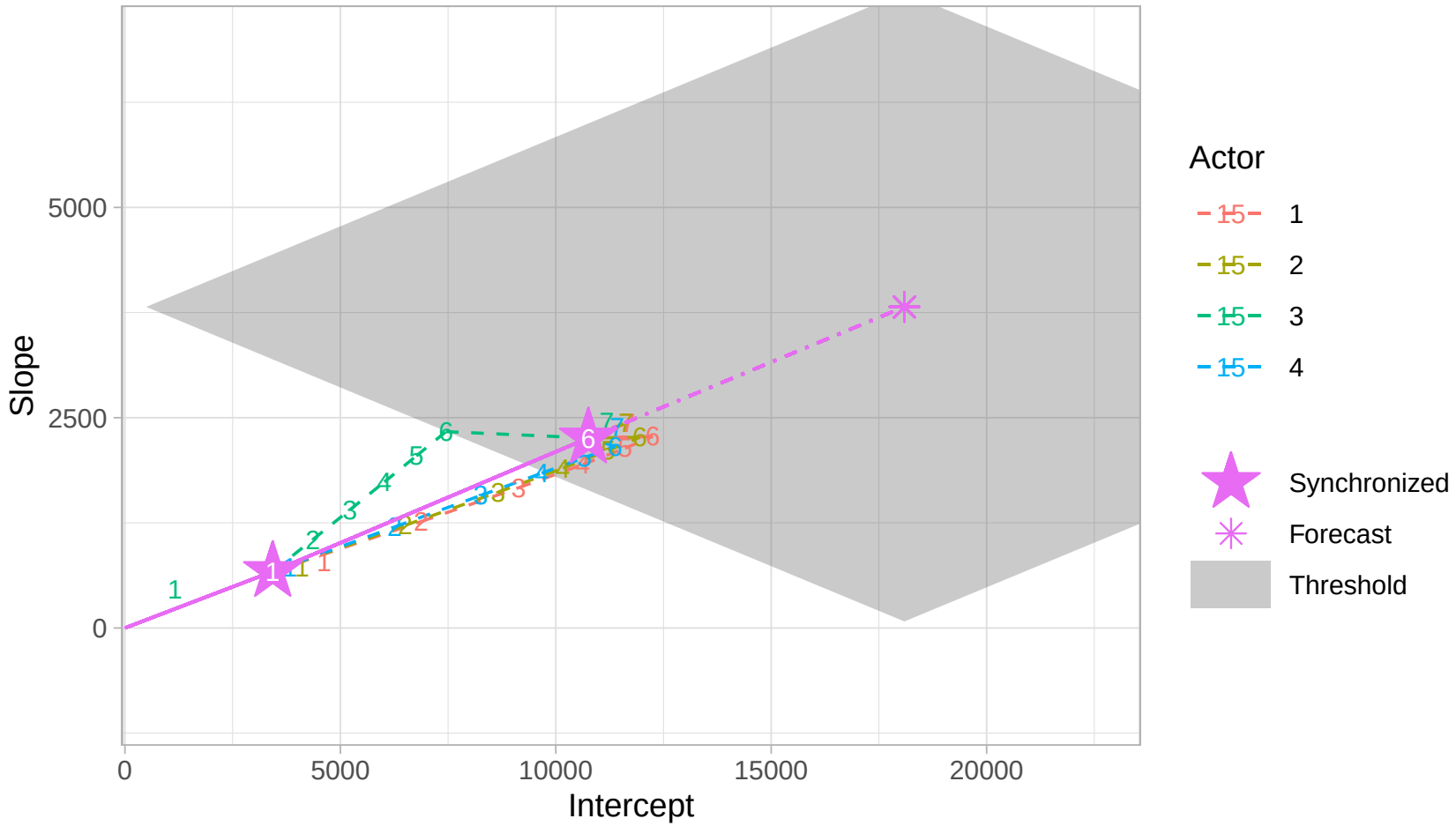


Illustration – Epoch 8

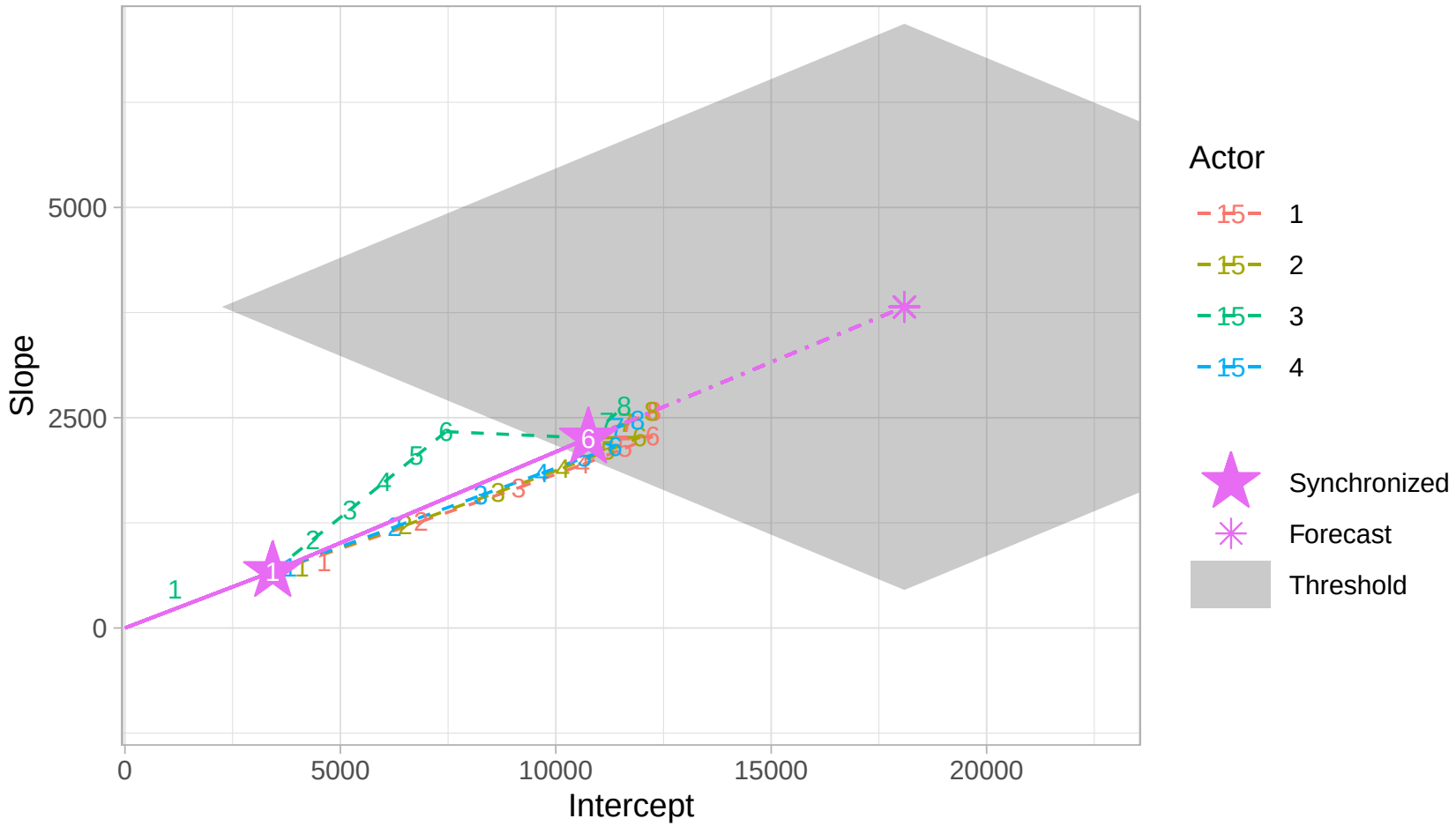


Illustration – Epoch 9

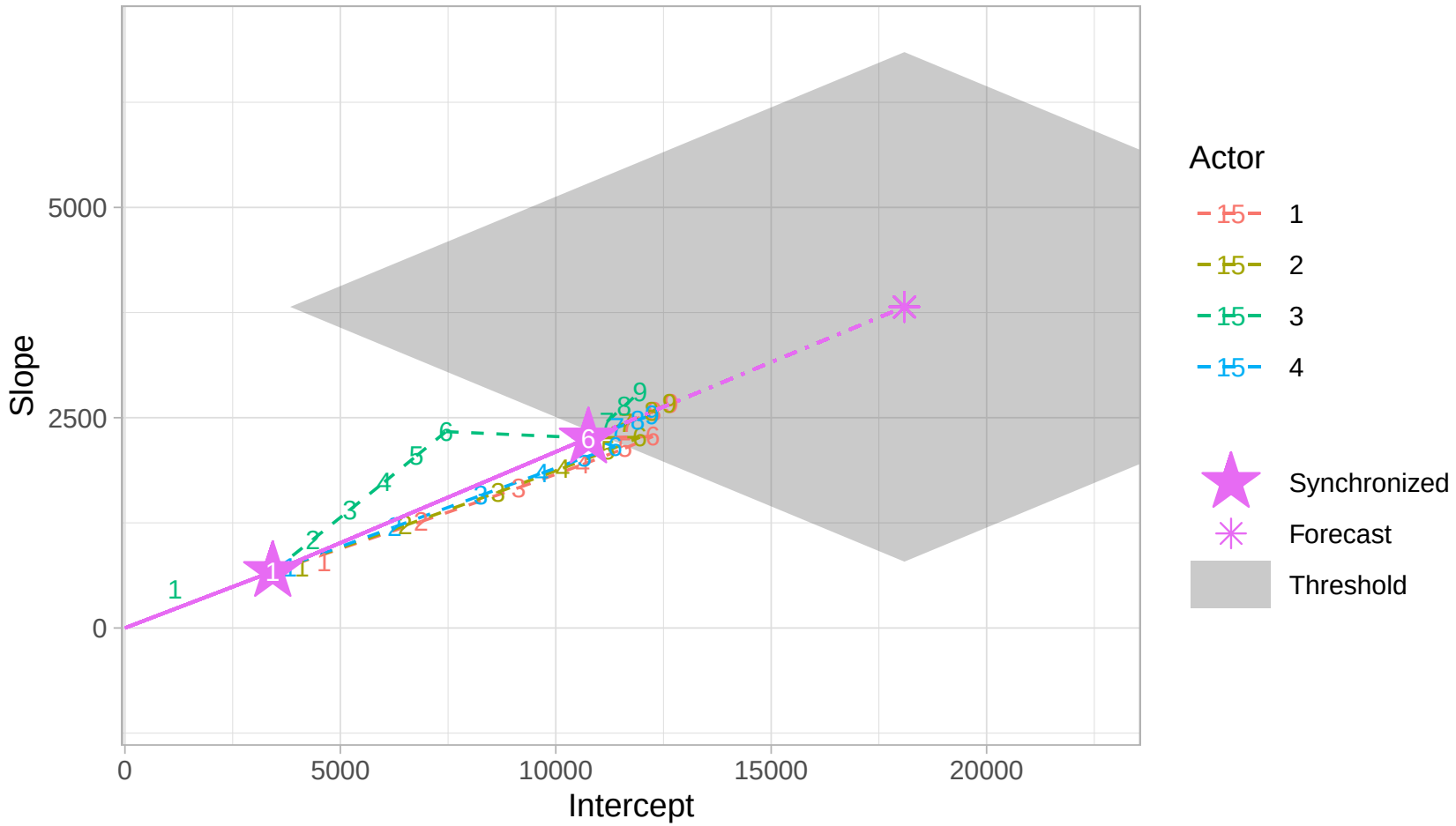


Illustration – Epoch 10

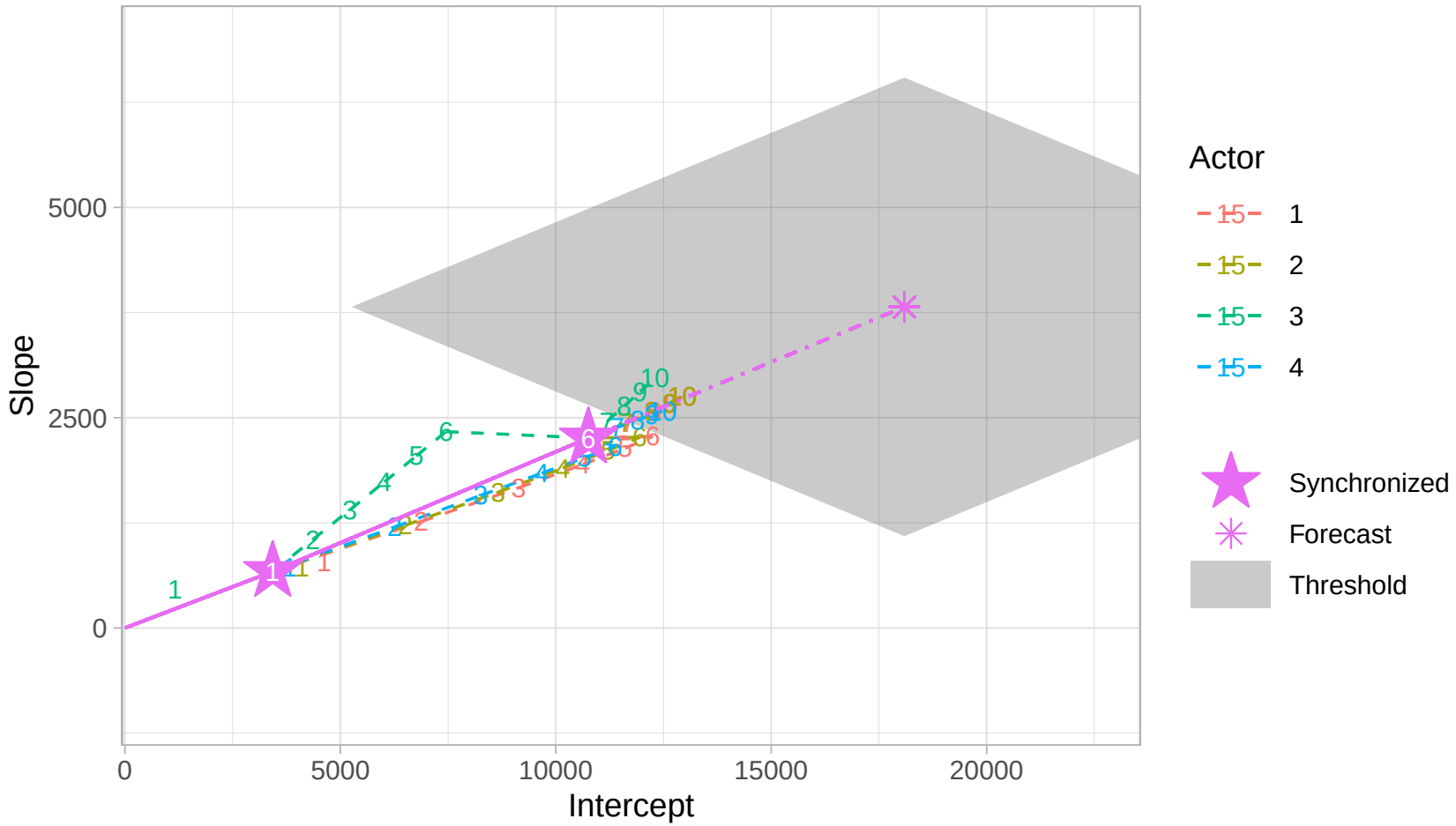


Illustration – Epoch 11

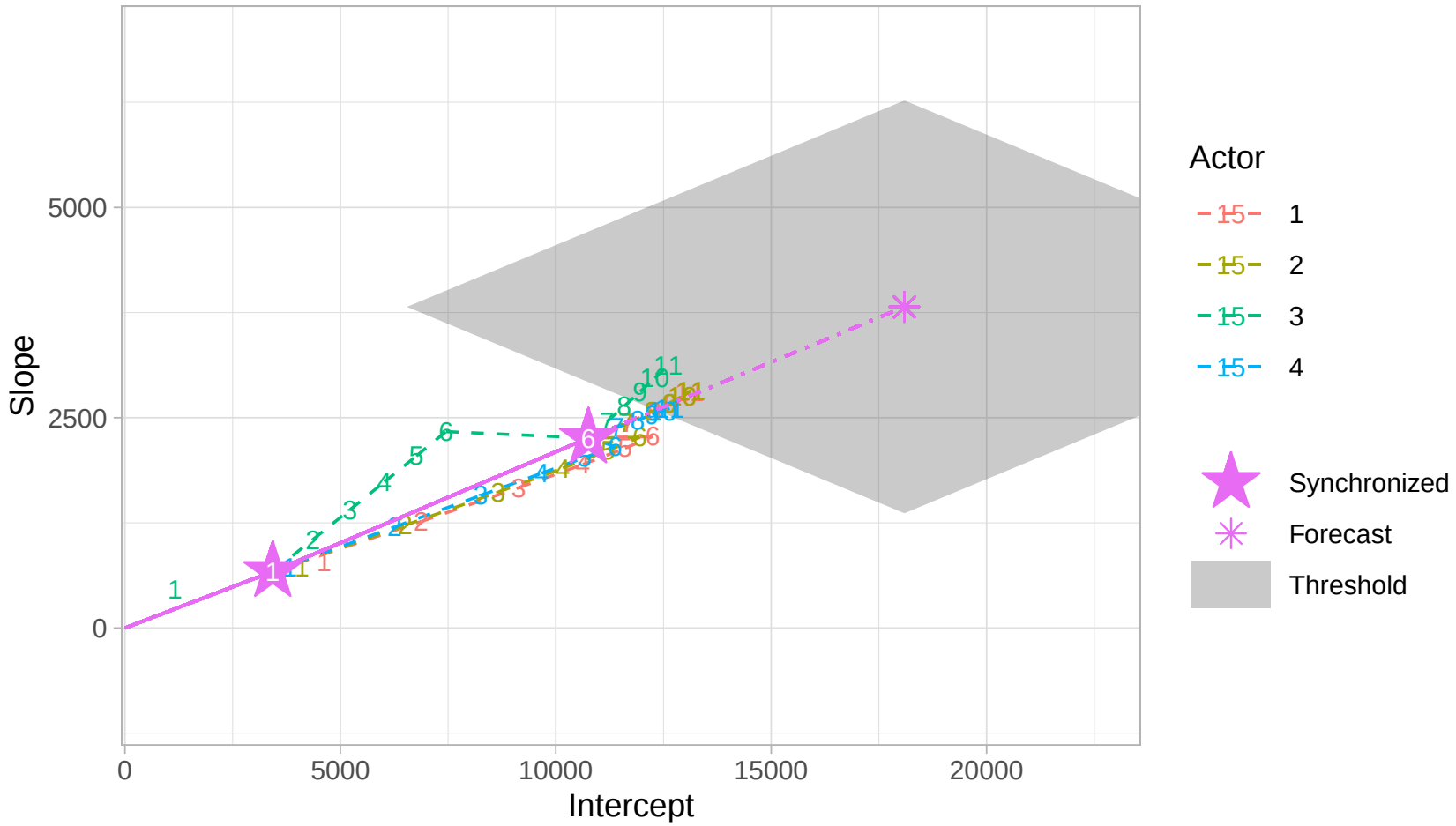


Illustration – Epoch 12

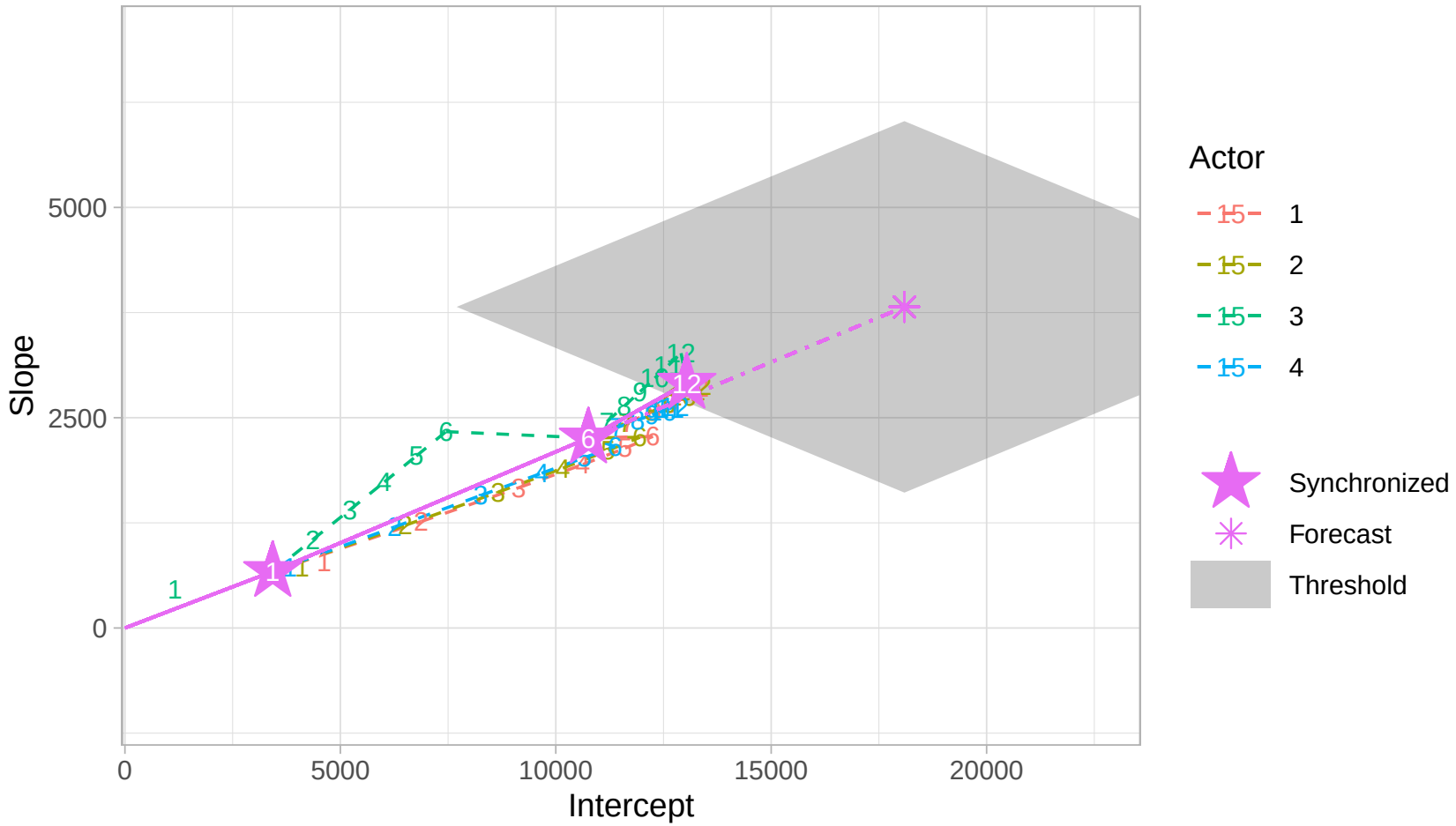


Illustration – Epoch 13

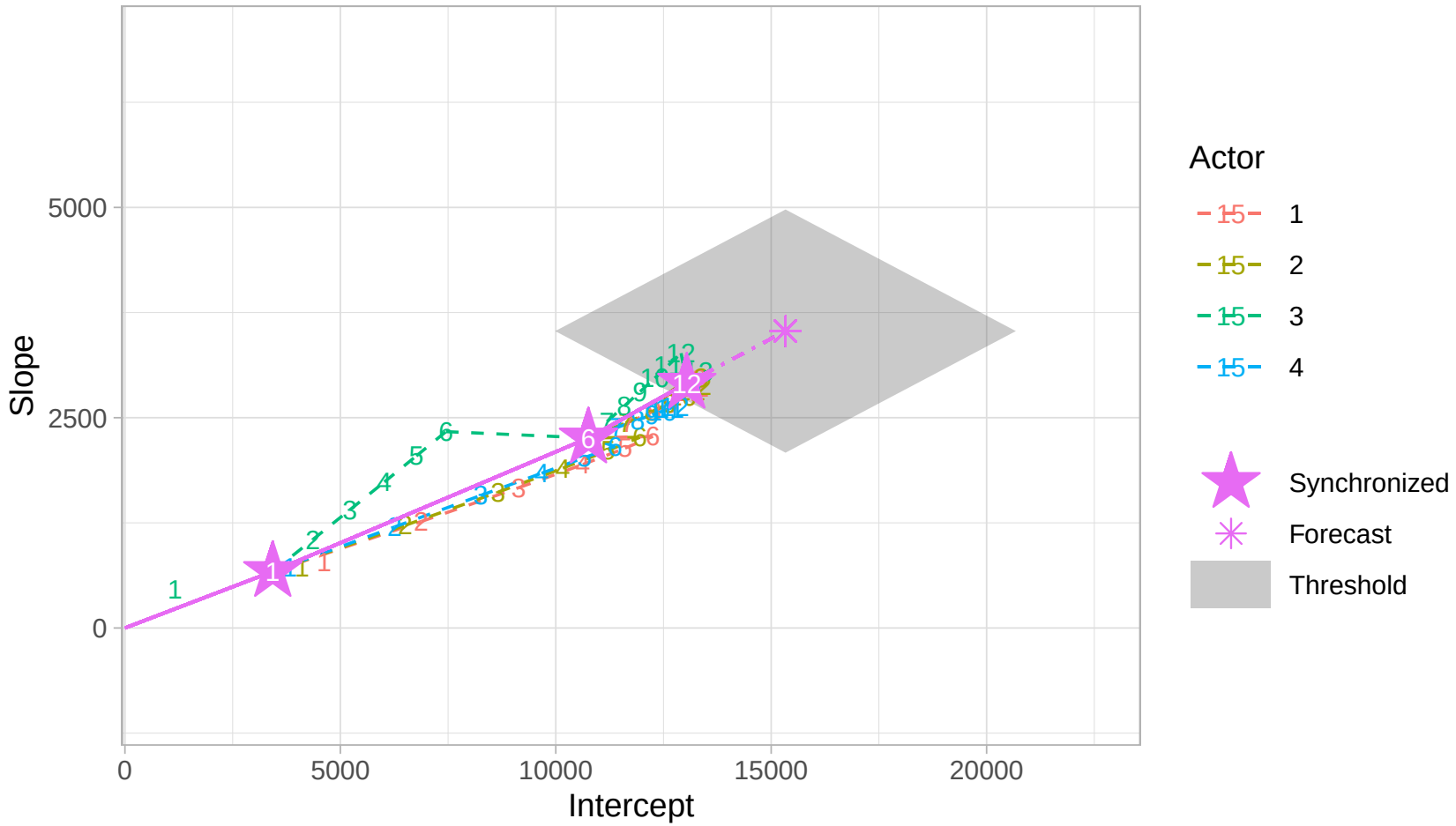


Illustration – Epoch 14

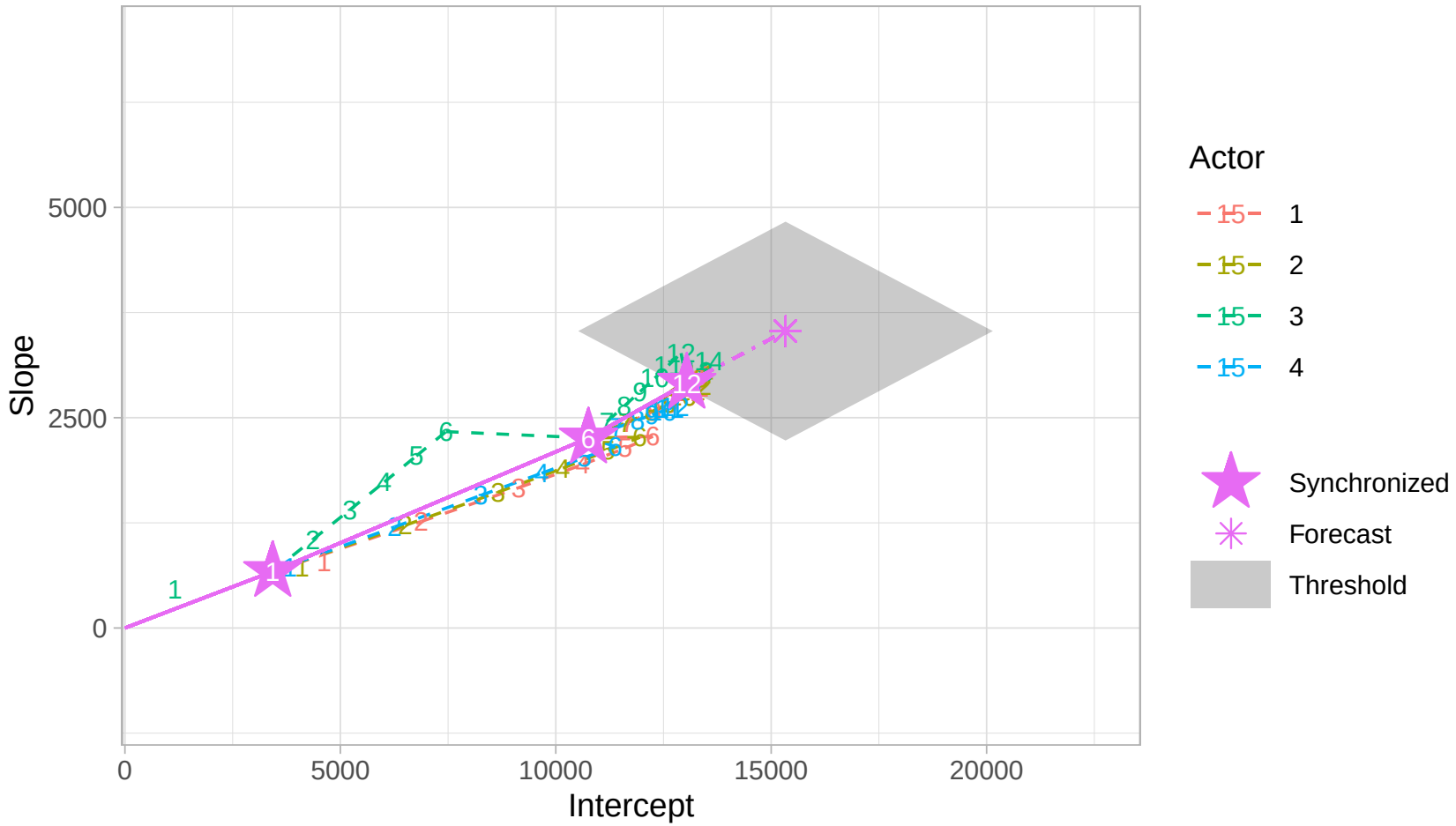
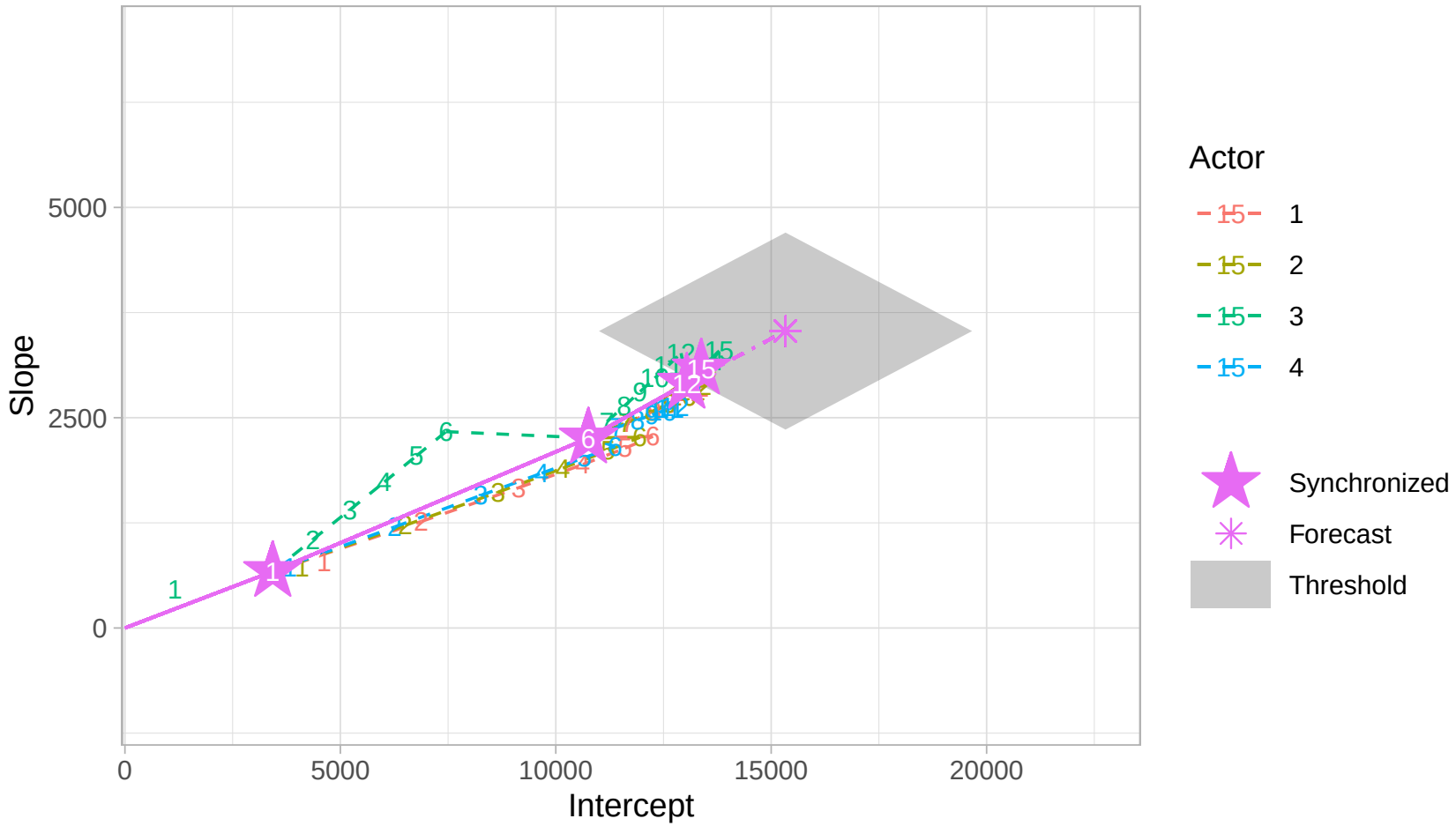
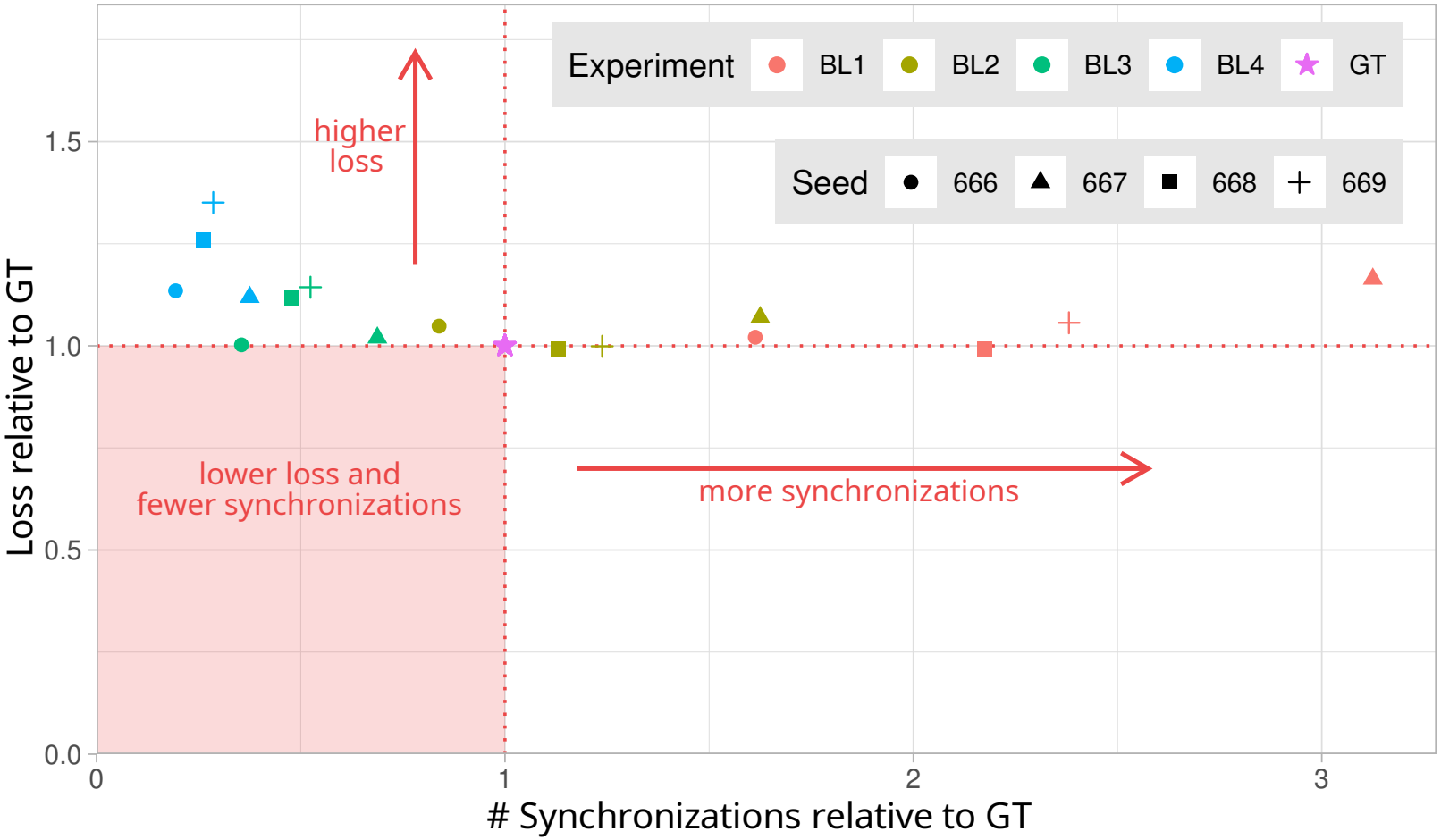


Illustration – Epoch 15



Experiments



Master's Thesis: Decentralized Federated Learning

Summary

- MoDeFL
  - github.com/ywcb00/MoDeFL
- Survey on Communication Efficiency
  - Submitted to Journal Artificial Intelligence Review
- Gradient Thresholding
  - github.com/ywcb00/GradientThresholding

Finale

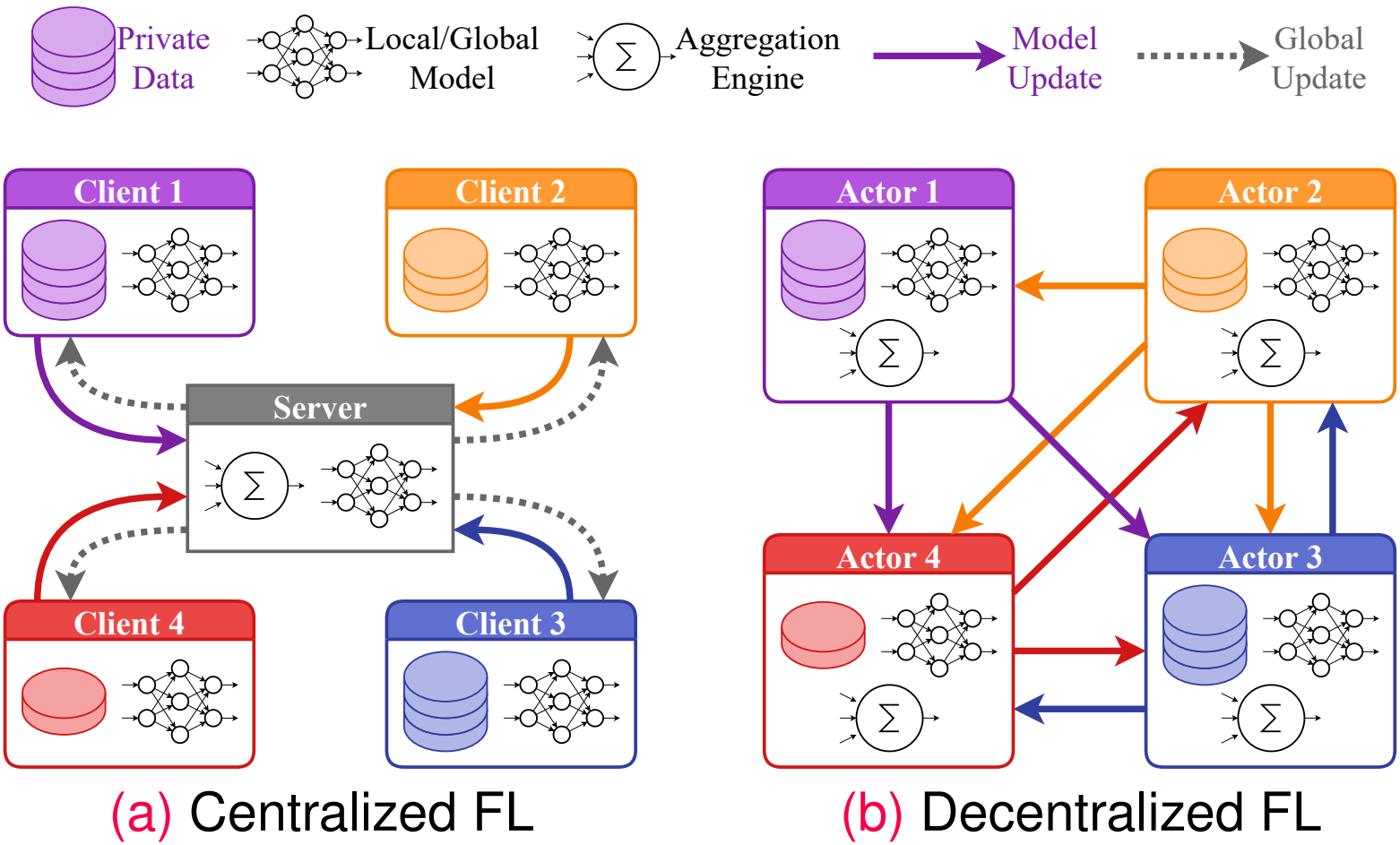
- Know Center Research GmbH
- FFG funded project “Pro‘K’ress”
- FFG funded project “QUICHE”
- ELSA Workshop on Privacy-Preserving ML

Finale

- Know Center Research GmbH
- FFG funded project “Pro‘K’ress”
- FFG funded project “QUICHE”
- ELSA Workshop on Privacy-Preserving ML

Thank You

CFL vs. DFL



(a) Centralized FL

(b) Decentralized FL

Privacy

Can You Really Backdoor Federated Learning?

Ziteng Sun*
Cornell University
zs335@cornell.edu

Peter Kairouz
Google
kairouz@google.com

Ananda Theertha Suresh
Google
theertha@google.com

H. Brendan McMahan
Google
mcmahan@google.com

Abstract

The decentralized nature of federated learning makes detecting and defending against adversarial attacks a challenging task. This paper focuses on backdoor attacks in the federated learning setting, where the goal of the adversary is to reduce the performance of the model on targeted tasks while maintaining a good performance on the main task. Unlike existing works, we allow non-malicious clients to have correctly labeled samples from the targeted tasks. We conduct a comprehensive study of backdoor attacks and defenses for the EMNIST dataset, a real-life, user-partitioned, and non-iid dataset. We observe that in the absence of defenses, the performance of the attack largely depends on the fraction of adversaries present and the "complexity" of the targeted task. Moreover, we show that norm clipping and "weak" differential privacy mitigate the attacks without hurting the overall performance. We have implemented the attacks and defenses in TensorFlow Federated (TFF), a TensorFlow framework for federated learning. In open sourcing our code, our goal is to encourage researchers to contribute new attacks and defenses and evaluate them on standard federated datasets.

Attack of the Tails: Yes, You Really Can Backdoor Federated Learning

Hongyi Wang[w] Kartik Sreenivasan[w] Shashank Rajput[w] Harit Vishwakarma[w] Saurabh Agarwal[w]
Jy-yong Sohn[k] Kangwook Lee[w] Dimitris Papailiopoulos[w]

[w] University of Wisconsin-Madison
[k] Korea Advanced Institute of Science and Technology

Abstract

Due to its decentralized nature, Federated Learning (FL) lends itself to adversarial attacks in the form of backdoors during training. The goal of a backdoor is to corrupt the performance of the trained model on specific sub-tasks (*e.g.*, by classifying green cars as frogs). A range of FL backdoor attacks have been introduced in the literature, but also methods to defend against them, and it is currently an open question whether FL systems can be tailored to be robust against backdoors. In this work, we provide evidence to the contrary. We first establish that, in the general case, robustness to backdoors implies model robustness to adversarial examples, a major open problem in itself. Furthermore, detecting the presence of a backdoor in a FL model is unlikely assuming first order oracles or polynomial time. We couple our theoretical results with a new family of backdoor attacks, which we refer to as *edge-case backdoors*. An edge-case backdoor forces a model to misclassify on seemingly easy inputs that are however unlikely to be part of the training, or test data, *i.e.*, they live on the tail of the input distribution. We explain how these edge-case backdoors can lead to unsavory failures and may have serious repercussions on fairness, and exhibit that with careful tuning at the side of the adversary, one can insert them across a range of machine learning tasks (*e.g.*, image classification, OCR, text prediction, sentiment analysis).

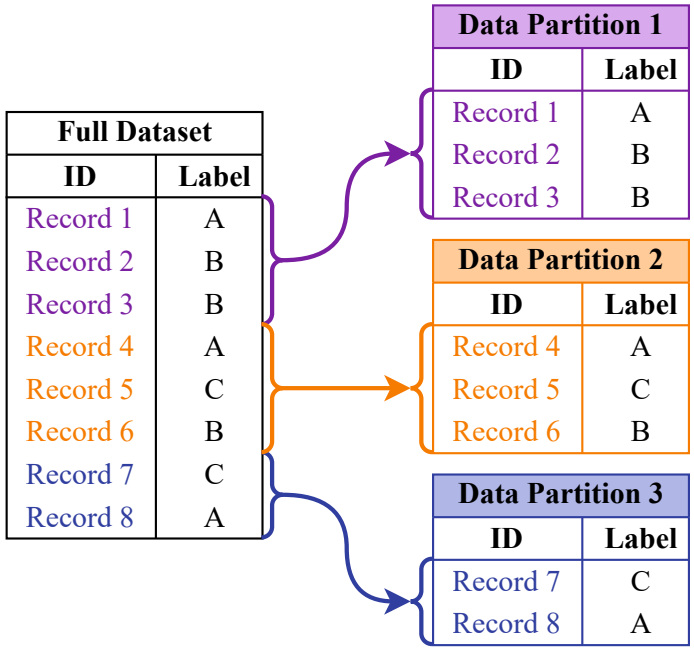
Sun, Z., Kairouz, P., Suresh, A.T., & McMahan, H.B. (2019). Can You Really Backdoor Federated Learning? ArXiv, abs/1911.07963. [3]
Wang, H., Sreenivasan, K.K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J., Lee, K., & Papailiopoulos, D. (2020). Attack of the Tails: Yes, You Really Can Backdoor Federated Learning. ArXiv, abs/2007.05084. [4]

Comparability of DFL Methods

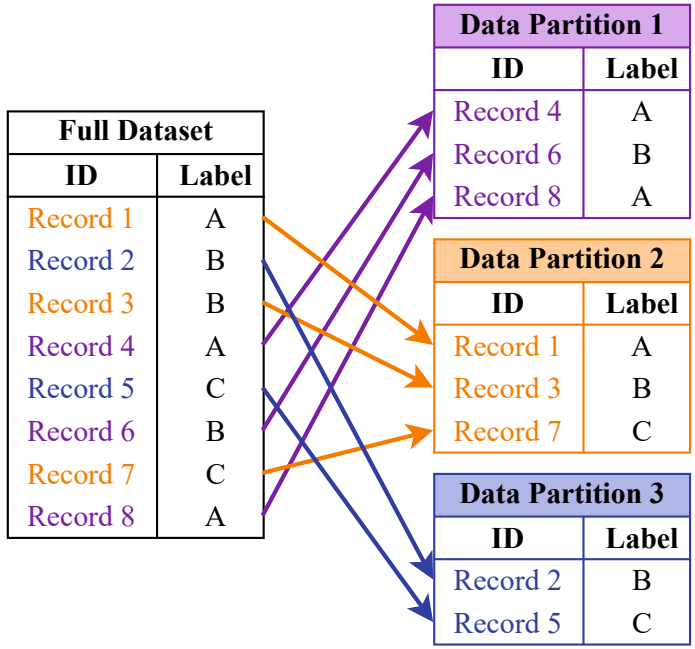
Exemplary DFL methods from literature

Reference	Model Aggregation	Sync. Method	Network Topologies	Compression	Local Updates	Partial Device Participation
Wang et al. SparSFA [5]	wAvg	sync.	fully, chain, random	novel sparsification	5	None
Gupta et al. TravellingFL [6]	incAvg	sync.	directed graph	None	1, 2, 5, 10	None
Liu et al. AEDFL [7]	wAvg	async.	exponential graph	sparsification	implicit	random one

Built-in Data Partitioning

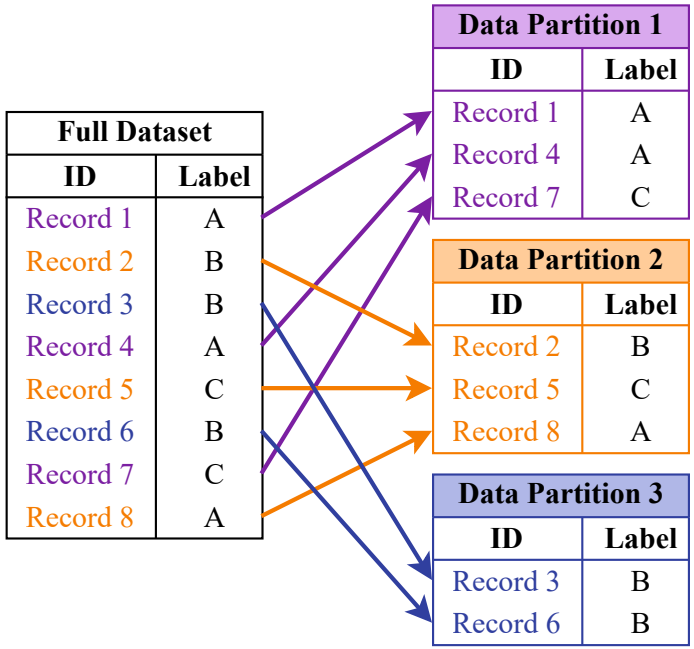


(a) Range

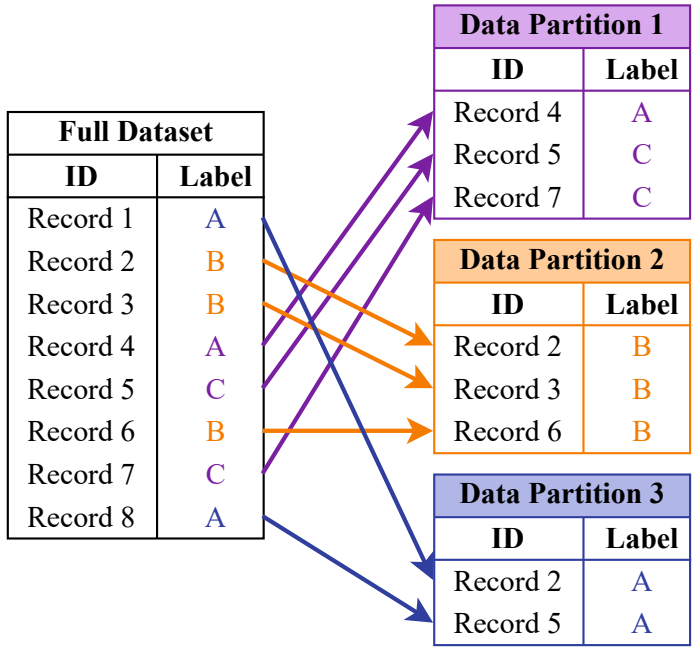


(b) Random

Built-in Data Partitioning contd.

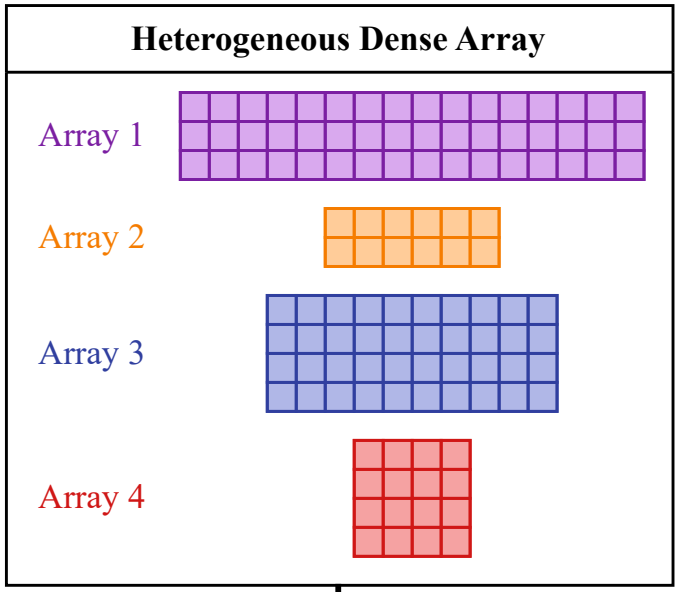


(c) Round-Robin



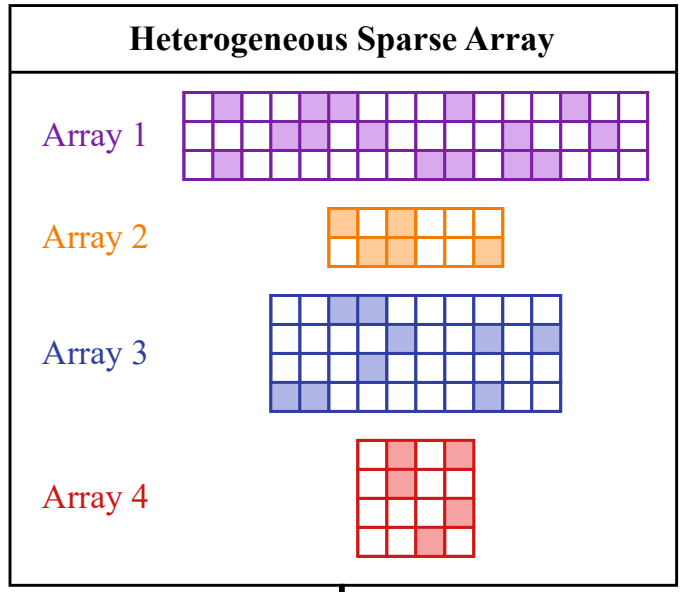
(d) Dirichlet

Model Parameter Serialization



serialize()

	Serialized List		
A1	Values	16×3	*float32*
A2	Values	6×2	*float32*
A3	Values	10×4	*float32*
A4	Values	4×4	*float32*

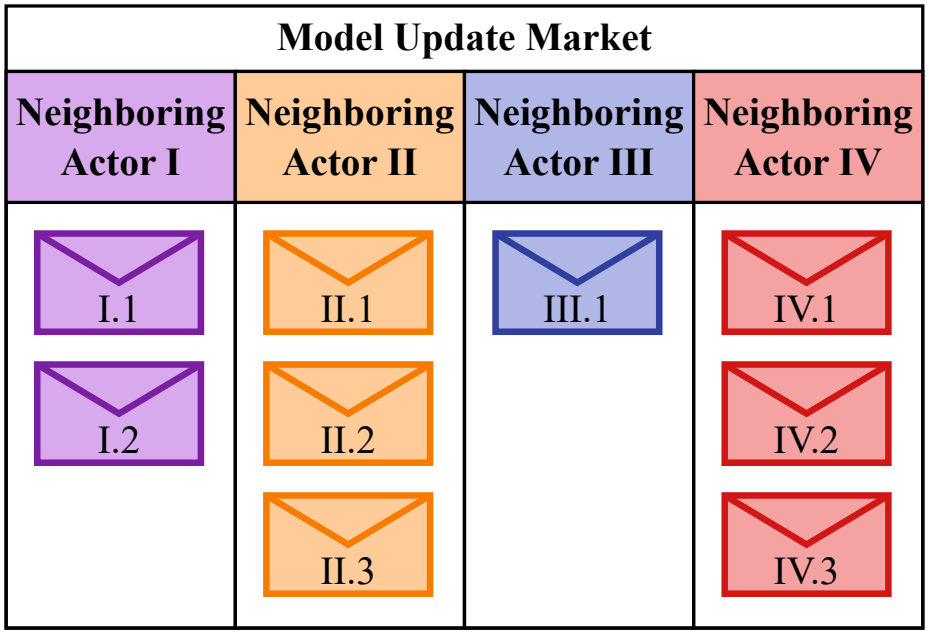


serialize()

	Serialized List			
A1	Coordinates	Values	16×3	*float32*
A2	Coordinates	Values	6×2	*float32*
A3	Coordinates	Values	10×4	*float32*
A4	Coordinates	Values	4×4	*float32*

Model Update Market

X-th Model Update from Actor A



One from each	
get()	1 1 1 1
get()	wait for actor III 2

One from each with timeout (2 sec.)	
get()	1 1 1 1
get()	wait for 2 sec., then 2 2 2
get()	wait for 2 sec., then 3 3

Model Update Market contd.

X-th Model Update from Actor A

Model Update Market			
Neighboring Actor I	**Neighboring Actor II**	**Neighboring Actor III**	**Neighboring Actor IV**
I.1	II.1	III.1	IV.1
I.2	II.2		IV.2
	II.3		IV.3

Available / min. one from each / minimum k ($k \leq 9$)	
get()	1 1 1 1 2 2 2 3 3

One from minimum percentage 75%	
get()	1 1 1 1
get()	2 2 2
get()	wait for 3 or 2

Configuration File

Argument	Default value	Description
`seed`	13	Random seed for reproducibility
`num_threads_server`	# logical CPUs	Number of threads for the gRPC service
`log_level`	`DEBUG`	Logging level for the console output

Table: Common configurations

Argument	Default value	Description
`log_performance_flag`	`True`	Flag indicating whether to log training and validation metrics or not
`log_communication_flag`	`True`	Flag indicating whether to log the data volume transferred between actors
`log_dir`	`./log`	Path to the directory for storing log files

Table: Actor configurations

Configuration File contd.

Argument	Default value	Description
`dataset_id`	`Mnist`	Dataset to be used for training
`partitioning_scheme`	`ROUND_ROBIN`	Scheme for partitioning the dataset
`partitioning_alpha`	2.5	Common value for the concentration parameters of the Dirichlet partitioning scheme
`addr_file`	`./addr.txt`	Path to the address file containing the actor addresses
`adj_file`	`./adj.txt`	Path to the adjacency matrix file specifying the network topology
`num_fed_epochs`	5	Number of communication rounds to be performed
`num_local_epochs`	1	Number of local training epochs in every communication round
`lr`	Model dependent	Learning rate applied for local training at the actors
`lr_global`	Model dependent	Global learning rate considered in certain aggregation methods
`learning_type`	`DFLv1`	Learning strategy to perform
`sync_strategy`	`ONE_FROM_EACH`	Synchronization strategy for retrieving the model updates from the market

Table: Initiator configurations I

Configuration contd.

Argument	Default value	Description
`sync_strat_percentage`	0.5	Percentage attribute for synchronization strategies
`sync_strat_amount`	2	Amount attribute for synchronization strategies
`sync_strat_timeout`	3	Timeout in seconds considered in certain synchronization strategies
`sync_strat_allowempty`	`False`	Flag indicating whether to count an empty update in the synchronization strategy or not
`compression_type`	`NoneType`	Compression method applied to the model updates
`compression_k`	100	Amount of non-zero values for sparsification methods
`compression_percentage`	0.2	Percentage of non-zero values for sparsification methods
`compression_precision`	8	Precision in bits for quantization methods
`pdp_strategy`	`NoneStrategy`	Strategy for partial device participation
`pdp_k`	2	Number of neighboring actors for partial device participation

Table: Initiator configurations II

Related Surveys

Survey	Model Aggregation	Synchronization Method	Network Topology	Compression	Local Computation	Partial Device Participation
[2]	∼	✓	✓	∼	×	×
[8]	×	✓	✓	×	×	✓
[9]	✓	×	×	✓	×	×
[10]	×	∼	✓	×	×	×
[11]	✓	∼	∼	∼	×	✓
[12]	×	×	∼	∼	×	✓
[13]	×	∼	∼	✓	×	∼
Ours	✓	✓	✓	✓	✓	✓

Table: Coverage of communication aspects in surveys;
× : does not comprise communication;
∼ : mentions communication;
✓ : elaborates on communication.

Communication-influencing Components

Reference	Model Aggregation	Synchronization Method	Network Topology
[14]	wAvg	Synchronous	Fully-connected, Ring, Torus
[15]	KT/KD	Synchronous	Fully-connected
[16]	wAvg	Synchronous	Random
[17]	wAvg	Synchronous	Fully-connected
[5]	wAvg	Synchronous	Fully-connected, Random, Other
[18]	wAvg	Synchronous	N/A
[19]	ADMM	Semi-synchronous	Random
[20]	wAvg	Synchronous	Fully-connected, Ring
[21]	wAvg	Synchronous	Torus
[7]	wAvg	Asynchronous	Other
[6]	incAvg	Synchronous	Other
[22]	wAvg	Asynchronous	Random
[23]	wAvg	Synchronous	Grid, Hybrid

Model Aggregation

wAvg

$$m_{k+1,i} = \sum_{j \in \mathcal{N}_i} (W_{i,j} \Delta_{k,j}) \tag{1}$$

incAvg

$$m_i = m_p - \eta \nabla f_i(m_p, x_i) \tag{2}$$

ADMM

$$\lambda_i^{k+1} = \lambda_i^k + c \left(|\mathcal{N}_i| m_i^k - \sum_{j \in \mathcal{N}_i} m_j^k \right)$$

$$m_i^{k+1} = \left(\nabla f_i(m_i^k, x_i) + 2c|\mathcal{N}_i| I \right)^{-1} \left(c|\mathcal{N}_i| m_i^k + c \sum_{j \in \mathcal{N}_i} m_j^k - \lambda_i^{k+1} \right) \tag{3}$$

Knowledge Transfer / Knowledge Distillation

$$m_s = m_s - \eta_s \frac{\partial \text{Loss}_s(m_s, B_l, P_{r,l})}{\partial m_s}$$

$$m_r = m_r - \eta_r \frac{\partial \text{Loss}_r(m_r, B_l, P_{s,l})}{\partial m_r} \tag{4}$$

Model Aggregation – wAvg

Reference	Mixing Weights	Update Selection	Update Content
[14]	Equal	None	Estimated model delta
[16]	Equal (others)	None (PDP)	Model params.
[17]	Equal (both)	None (PDP)	Model params.
[5]	Trust-score (dynamic)	None	Model params.
[18]	Network topology	None	Model delta
[20]	Degree (global)	None (PDP)	Model params.
[21]	Data size	None	Model params.
[7]	Staleness (dynamic)	RL-based (cached)	Model params.
[22]	Degree	None (PDP)	Model params., scalar
[23]	Equal	None (PDP)	Model params.

Model Aggregation – wAvg contd.

Reference	Dir.	Additional Communication
[16]	$\rightharpoonup$	Matching decompositions, seed
[5]	$\rightleftharpoons$	Data size, data variance, connectivity
[21]	$\rightleftharpoons$	Data size

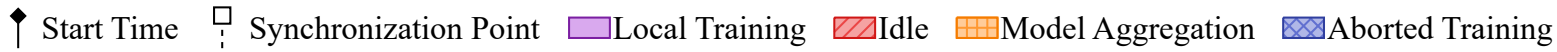
(a) Once for initialization.

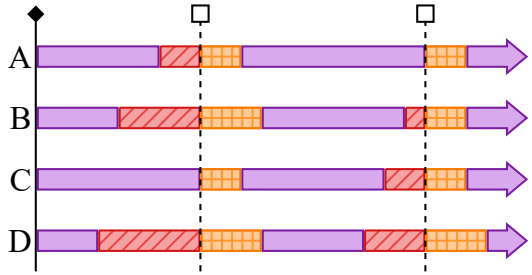
Reference	Dir.	Additional Communication
[17]	$\rightharpoonup$ $\leftharpoondown$	Seed, mixing weights matrix, round index Bandwidth, loss, accuracy
[20]	$\rightharpoonup$ $\leftharpoondown$	Set of neighbors, compression ratio Consensus distance, communication capacity
[22]	$\rightleftharpoons$	Training loss, local iteration index
[23]	$\rightleftharpoons$	Parameters of last layer

(b) Every communication round.

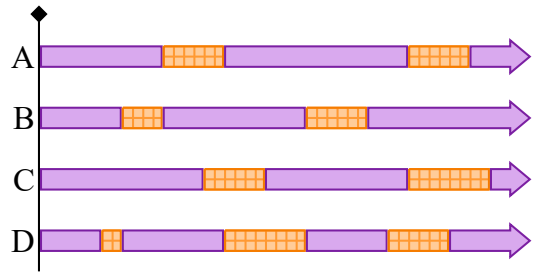
Table: Communicated information in addition to the model update content; $\rightharpoonup$ from coordinator to actors; $\leftharpoondown$ from actors to coordinator; $\rightleftharpoons$ among actors.

Synchronization Method

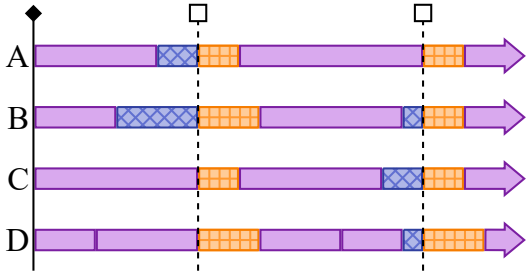




(a) Synchronous

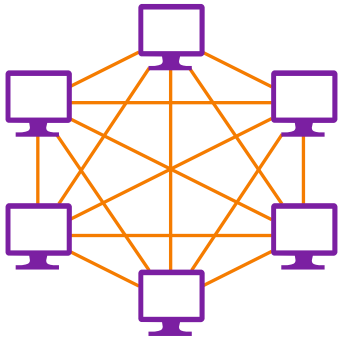


(b) Asynchronous

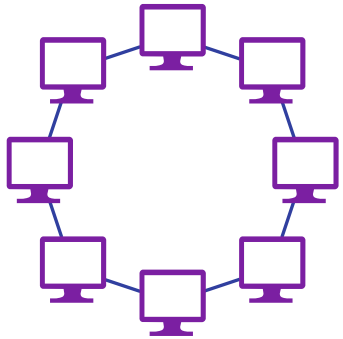


(c) Semi-synchronous

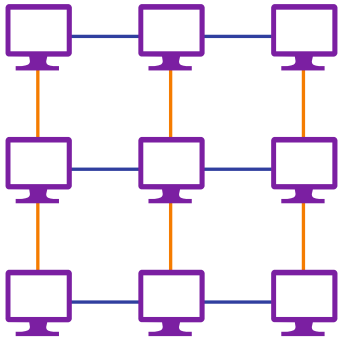
Network Topology



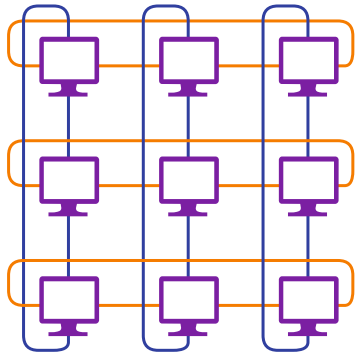
(a) Fully-connected



(b) Ring



(c) Grid



(d) Torus

Communication Optimization Techniques

Reference	Compression	Local Computation	Partial Device Participation
[14]	Quantization, Sparsification	None	Random (not in experiments)
[15]	None	1, 10 Updates	Random 20%
[16]	None	Implicitly (PDP)	Selective (connectivity)
[17]	Sparsification	None	Selective (bandwidth), single
[5]	Sparsification	5 Updates	None
[18]	Quantization	4 Updates	None
[19]	1BCS	Random in [5, 15]	Random 20%, 50%, 80%
[20]	Sparsification	50 Updates	Selective (consensus)
[21]	None	None	None
[7]	Sparsification	Implicitly (PCP)	Random, single
[6]	None	1, 2, 5, 10 Updates	None
[22]	None	2 Updates	Selective (loss)
[23]	None	5 Updates	Selective (similarity), single

Local Computation

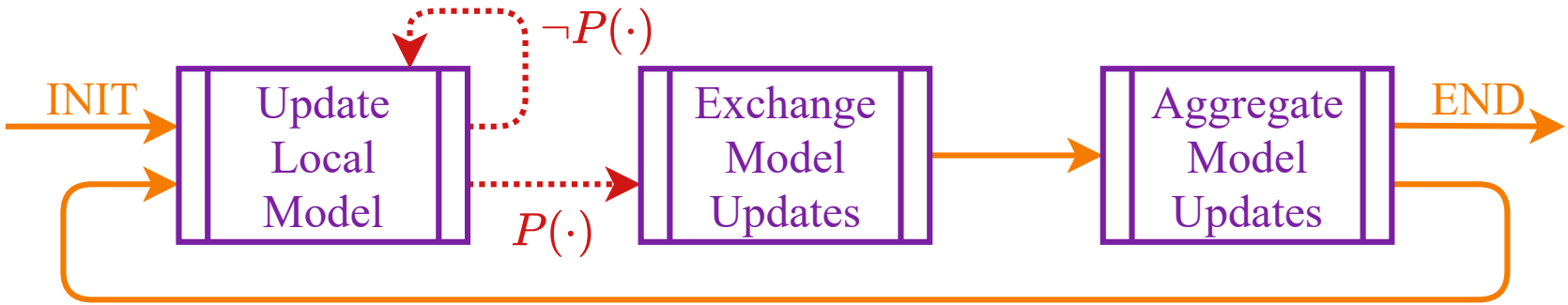
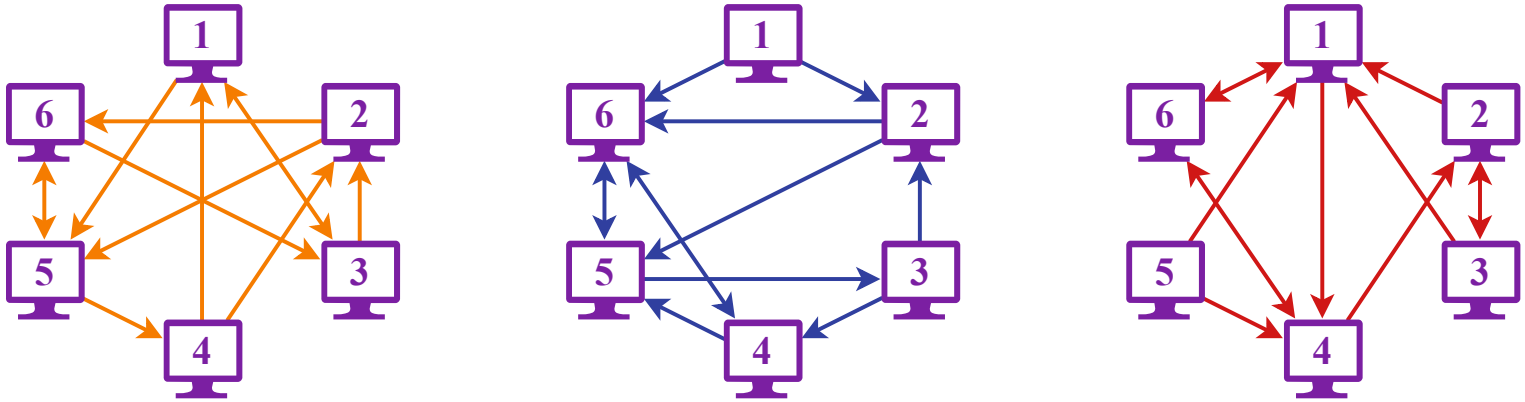


Figure: High-level DFL process with Local Computation through the predicate function $P(\cdot)$.

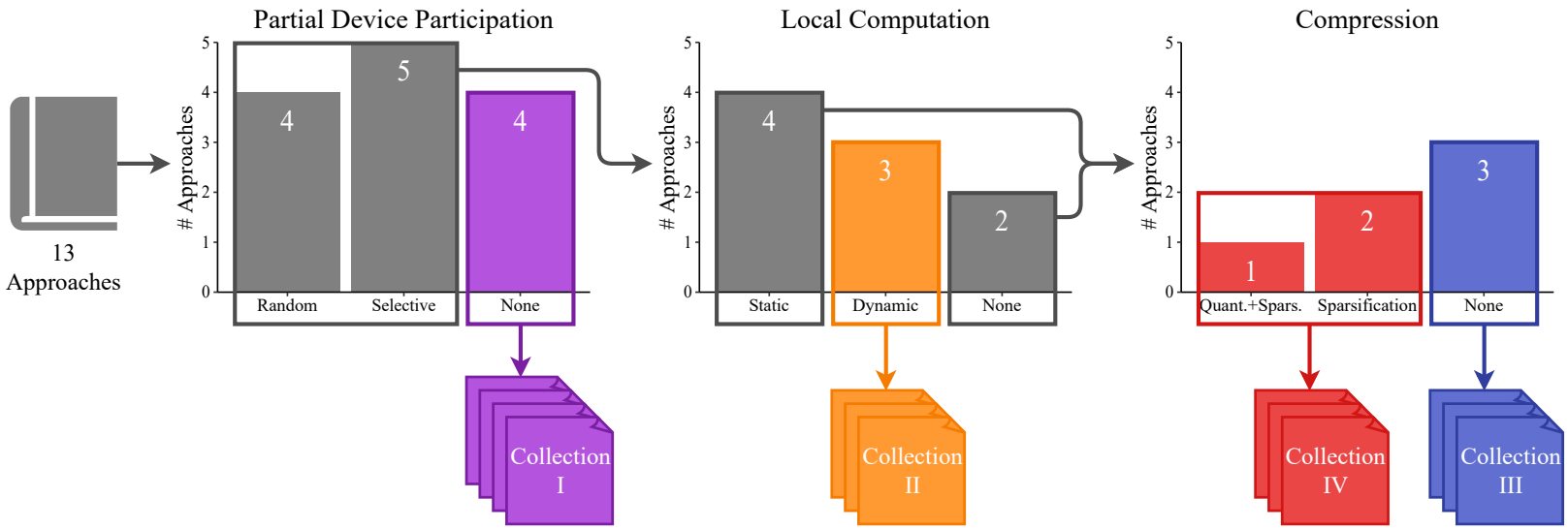
Partial Device Participation



(a) Epoch 1 (b) Epoch 2 (c) Epoch 3

Figure: Exemplary Partial Device Participation with random selection of 40% (i.e., 2/5) neighboring actors over three epochs.

Literature



- Collection I: Full device participation
- Collection II: Dynamic number of local updates
- Collection III: Perfect parameter transmission
- Collection IV: Sparsification for compression

Literature contd.

Collection	Commonality	Publications
I	Full device participation	[5], [6], [18], [21]
II	Dynamic number of local updates	[7], [16], [19]
III	Perfect parameter transmission	[15], [22], [23]
IV	Sparsification for compression	[14], [17], [20]

Table: Grouping of publications into four distinct collections according to commonalities in implemented optimization techniques.

Related Work

Chen et al., “Lag” [24]	C3	⚠ Central server
Zhang et al., “ASP” [25]	C3	(⚠ Central server)
Kamp, Adilova, et al. [26]	C2	⚠ Central server
Theologitis et al., “FDA” [27]	C3	⚠ Central server
Local Computation	C2	

- Challenge 2 (C2): Divergence detection
- Challenge 3 (C3): Limited communication

Threshold Region

- Threshold Center (forecast)
  - Defines direction and influences size
- Threshold Extent (cylindrical)
  - Projected distance and projection distance
- Weighted Distance
  - Account for different sensitivity of parameters
- Threshold Decay
  - Decrease the threshold width in every epoch

Threshold Region contd.

Threshold Center

$$\widetilde{\mathcal{G}}_{(k)} = \begin{cases} \overline{\mathcal{G}^{\Sigma}_{(1)}} \\ \frac{\left\|\overline{\mathcal{G}^{\Sigma}_{(k)}}\right\|_2}{\left\|\theta_\beta \frac{\overline{\mathcal{G}^{\Sigma}_{(k)}}}{\left\|\overline{\mathcal{G}^{\Sigma}_{(k)}}\right\|_2} + (1-\theta_\beta) \frac{\widetilde{\mathcal{G}}_{(k-1)}}{\left\|\widetilde{\mathcal{G}}_{(k-1)}\right\|_2}\right\|_2} \left(\theta_\beta \frac{\overline{\mathcal{G}^{\Sigma}_{(k)}}}{\left\|\overline{\mathcal{G}^{\Sigma}_{(k)}}\right\|_2} + (1-\theta_\beta) \frac{\widetilde{\mathcal{G}}_{(k-1)}}{\left\|\widetilde{\mathcal{G}}_{(k-1)}\right\|_2}\right) \end{cases} \tag{5}$$

Threshold Extent

$$\begin{aligned} \left\|\mathrm{proj}_{[\widetilde{\mathcal{G}}]}(\mathcal{G}) - \widetilde{\mathcal{G}}\right\|_2 &\leq \left(1 + \frac{1}{k - k_s}\right) \left\|\widetilde{\mathcal{G}}\right\|_2 \\ \left\|\mathcal{G} - \mathrm{proj}_{[\widetilde{\mathcal{G}}]}(\mathcal{G})\right\|_W &\leq \theta_\rho \left(1 + \frac{1}{k - k_s}\right) \left\|\widetilde{\mathcal{G}}\right\|_W \end{aligned} \tag{6}$$

Weighted Distance

$$W = \max_{\circ}\left(\widetilde{\mathcal{G}}^{|\cdot|}, \mathrm{med}\left(\widetilde{\mathcal{G}}^{|\cdot|}\right)\right)^{\circ -1} \tag{7}$$

Gradient Thresholding Protocol

Input: number of epochs $\mathcal{T}$, learning rate η, threshold extent factor θ_ρ, threshold extent decay θ_α

Initialize reference model parameters $\mathcal{M}_R$ to initial state.
for all $i \in \mathcal{N}$ **do in parallel**
    $\mathcal{M}_i \leftarrow \mathcal{M}_R$; $\mathcal{G}_i^\Sigma \leftarrow \mathbf{0}$; $\widetilde{\mathcal{G}} \leftarrow \mathbf{0}$; $t_s \leftarrow -1$; $\rho \leftarrow 0$ ▷ Initialize.
end for
for $t = 1$ **to** $\mathcal{T}$ **do**
    for all $i \in \mathcal{N}$ **do in parallel**
        $\mathcal{G}_i \leftarrow \text{ObtainGradient}(\mathcal{M}_i, \mathcal{D}_i)$; $\mathcal{M}_i \leftarrow \text{ApplyGradient}(\mathcal{M}_i, \mathcal{G}_i, \eta)$
        $\mathcal{G}_i^\Sigma \leftarrow \mathcal{G}_i^\Sigma + \mathcal{G}_i$
        if $t == 1$ OR RequiresSynchronization $\left(\widetilde{\mathcal{G}}, \mathcal{G}_i^\Sigma, \rho, \theta_\rho\right)$ **then**
            $\mathcal{M}_R, \widetilde{\mathcal{G}} \leftarrow \text{Synchronize}(\mathcal{M}_R, \mathcal{G}_i^\Sigma, \widetilde{\mathcal{G}}, \eta)$
            $\mathcal{M}_i \leftarrow \mathcal{M}_R$
            $\mathcal{G}_i^\Sigma \leftarrow \mathbf{0}$; $\rho \leftarrow \left(1 + \frac{1}{t - t_s}\right)$; $t_s \leftarrow t$ ▷ Reset.
        else
            $\rho \leftarrow \theta_\alpha \rho$ ▷ Decay threshold extent.
        end if
    end for
end for

Gradient Thresholding Synchronization Rule

Input: forecast gradient $\widetilde{\mathcal{G}}$, local accumulated gradient $\mathcal{G}_i^{\Sigma}$, threshold extent ρ, threshold extent factor θ_ρ
Output: boolean flag indicating if synchronization is needed

function REQUIRESSYNCHRONIZATION($\widetilde{\mathcal{G}}$, $\mathcal{G}_i^{\Sigma}$, ρ, θ_ρ)

$\text{proj}_{\left[\widetilde{\mathcal{G}}\right]}\left(\mathcal{G}_i^{\Sigma}\right) \leftarrow \frac{\mathcal{G}_i^{\Sigma} \cdot \widetilde{\mathcal{G}}}{\widetilde{\mathcal{G}} \cdot \widetilde{\mathcal{G}}} \widetilde{\mathcal{G}}$ ▷ Projection onto forecast gradient line.

$d_{\widetilde{\mathcal{G}}}^{\max} \leftarrow \rho \cdot \left\| \widetilde{\mathcal{G}} \right\|_2$ ▷ Max. distance to forecast gradient.

$d_{\widetilde{\mathcal{G}}} \leftarrow \left\| \text{proj}_{\left[\widetilde{\mathcal{G}}\right]}\left(\mathcal{G}_i^{\Sigma}\right) - \widetilde{\mathcal{G}} \right\|_2$ ▷ Projected distance to forecast gradient.

if $d_{\widetilde{\mathcal{G}}} > d_{\widetilde{\mathcal{G}}}^{\max}$ **then**

  return true ▷ Projected distance exceeds threshold.

end if

$W \leftarrow \text{ComputeDistanceWeights}(\widetilde{\mathcal{G}})$

$d_L^{\max} \leftarrow \theta_\rho \cdot \rho \cdot \left\| \widetilde{\mathcal{G}} \right\|_W$ ▷ Max. distance to forecast gradient line.

$d_L \leftarrow \left\| \mathcal{G}_i^{\Sigma} - \text{proj}_{\left[\widetilde{\mathcal{G}}\right]}(\mathcal{G}_i^{\Sigma}) \right\|_W$ ▷ Projection distance to forecast gradient line.

if $d_L > d_L^{\max}$ **then**

  return true ▷ Projection distance exceeds threshold.

end if

return false ▷ Accumulated gradient is within threshold region.

end function

Gradient Thresholding Distance Weights

Input: estimated gradient $\widetilde{\mathcal{G}}$
Output: distance weights W

function ComputeDistanceWeights($\widetilde{\mathcal{G}}$)
    $W \leftarrow \text{abs}(\widetilde{\mathcal{G}})$
    for $l = 1$ **to** #layers **do** ▷ For each layer.
        for all $w_i \in \text{layer}(l, W)$ **do** ▷ For each value of the layer.
            $w_i \leftarrow \max(\text{med}(\text{layer}(l, W)), w_i)$ ▷ Cap weights by layer median.
        end for
    end for
    $W \leftarrow 1 \,/\, W$
    return W
end function

Gradient Thresholding Synchronization

Input: reference model parameters $\mathcal{M}_R$, local accumulated gradient $\mathcal{G}_i^{\Sigma}$, forecast gradient $\widetilde{\mathcal{G}}$, learning rate η, exponential moving average coefficient θ_β

Output: $\mathcal{M}_R$ averaged model parameters, $\widetilde{\mathcal{G}}$ forecast gradient

function SYNCHRONIZE($\mathcal{M}_R$, $\mathcal{G}_i^{\Sigma}$, $\widetilde{\mathcal{G}}$, η, θ_β)
  Send: $\mathcal{G}_i^{\Sigma}$ to other devices
  Receive: $\{\mathcal{G}_j^{\Sigma}\}_{j \in \mathcal{N} \setminus \{i\}}$ from other devices
  $\overline{\mathcal{G}^{\Sigma}} \leftarrow \frac{1}{|\mathcal{N}|} \sum_{j \in \mathcal{N}} \mathcal{G}_j^{\Sigma}$
  $\mathcal{M}_R \leftarrow \text{ApplyGradient}\left(\mathcal{M}_R, \overline{\mathcal{G}^{\Sigma}}, \eta\right)$
  if $\widetilde{\mathcal{G}} == \mathbf{0}$ **then**
    $\widetilde{\mathcal{G}} \leftarrow \overline{\mathcal{G}^{\Sigma}}$
  else
    $$\widetilde{\mathcal{G}} \leftarrow \frac{\left\|\overline{\mathcal{G}^{\Sigma}}\right\|_2}{\left\|\theta_\beta \frac{\overline{\mathcal{G}^{\Sigma}}}{\left\|\overline{\mathcal{G}^{\Sigma}}\right\|_2} + (1-\theta_\beta) \frac{\widetilde{\mathcal{G}}}{\left\|\widetilde{\mathcal{G}}\right\|_2}\right\|_2} \left(\theta_\beta \frac{\overline{\mathcal{G}^{\Sigma}}}{\left\|\overline{\mathcal{G}^{\Sigma}}\right\|_2} + (1-\theta_\beta) \frac{\widetilde{\mathcal{G}}}{\left\|\widetilde{\mathcal{G}}\right\|_2}\right)$$
  end if
  return $\mathcal{M}_R$, $\widetilde{\mathcal{G}}$
end function

Iris

- Dataset: 150 instances; 81% train; 9% validation; 10% test; Dirichlet $\alpha = 0.05$; [28]
- Task: Three-class classification
- 100 epochs; SGD; Categorical cross-entropy; 0.1 learning rate; 32 batch size; 8 actors
- Sensitivity parameter $\theta_\rho = 2$

Iris – Model

- Model: Adopted from Jason Browniee [29]

Layer Type	Output Shape	# Parameters	Configuration
Input layer	4	0	None
Densely-connected layer	8	40	ReLU activation
Densely-connected layer	3	27	Softmax activation

Mnist

- Dataset: 70000 instances; 77% train;
  9% validation; 14% test; Dirichlet $\alpha = 0.25$; [30]
- Task: Image classification w/ 10 classes
- 20 epochs; SGD; Categorical cross-entropy;
  0.2 learning rate; 256 batch size; 4 actors
- Sensitivity parameter $\theta_\rho = 1.75$

Mnist – Model

- Model:

Layer type	Output Shape	# Parameters	Configuration
Input layer	26×26	0	None
2D convolution layer	$26 \times 26 \times 32$	320	3×3 kernel ReLU activation
2D max pooling operation	$13 \times 13 \times 32$	0	2×2 pool
2D convolution layer	$11 \times 11 \times 64$	18496	3×3 kernel ReLU activation
2D max pooling operation	$5 \times 5 \times 64$	0	2×2 pool
Input flattening	1600	0	None
Dropout layer	1600	0	50% drop rate
Densely-connected layer	10	16010	Softmax activation

SVHN

- Dataset: 99289 instances; 66% train; 7% validation; 26% test; Dirichlet $\alpha = 0.25$; [31]
- Task: Image digit recognition w/ 10 classes
- 50 epochs; ADAM; Sparse categorical cross-entropy; 0.0002 learning rate; 128 batch size; 4 actors
- Sensitivity parameter $\theta_\rho = 2$

SVHN – Model I

- Model adopted from Dimitrios Roussis [32]:

Layer type	Output Shape	# Parameters	Configuration
Input layer	$32 \times 32 \times 3$	0	None
2D convolution layer	$32 \times 32 \times 32$	896	3×3 kernel ReLU activation
Batch normalization	$32 \times 32 \times 32$	128	None
2D convolution layer	$32 \times 32 \times 32$	9248	3×3 kernel ReLU activation
2D max pooling operation	$16 \times 16 \times 32$	0	2×2 pool
Dropout layer	$16 \times 16 \times 32$	0	30% drop rate
2D convolution layer	$16 \times 16 \times 64$	18496	3×3 kernel ReLU activation
Batch normalization	$16 \times 16 \times 64$	256	None
2D convolution layer	$16 \times 16 \times 64$	36928	3×3 kernel ReLU activation
2D max pooling operation	$8 \times 8 \times 64$	0	2×2 pool
Dropout layer	$8 \times 8 \times 64$	0	30% drop rate
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$

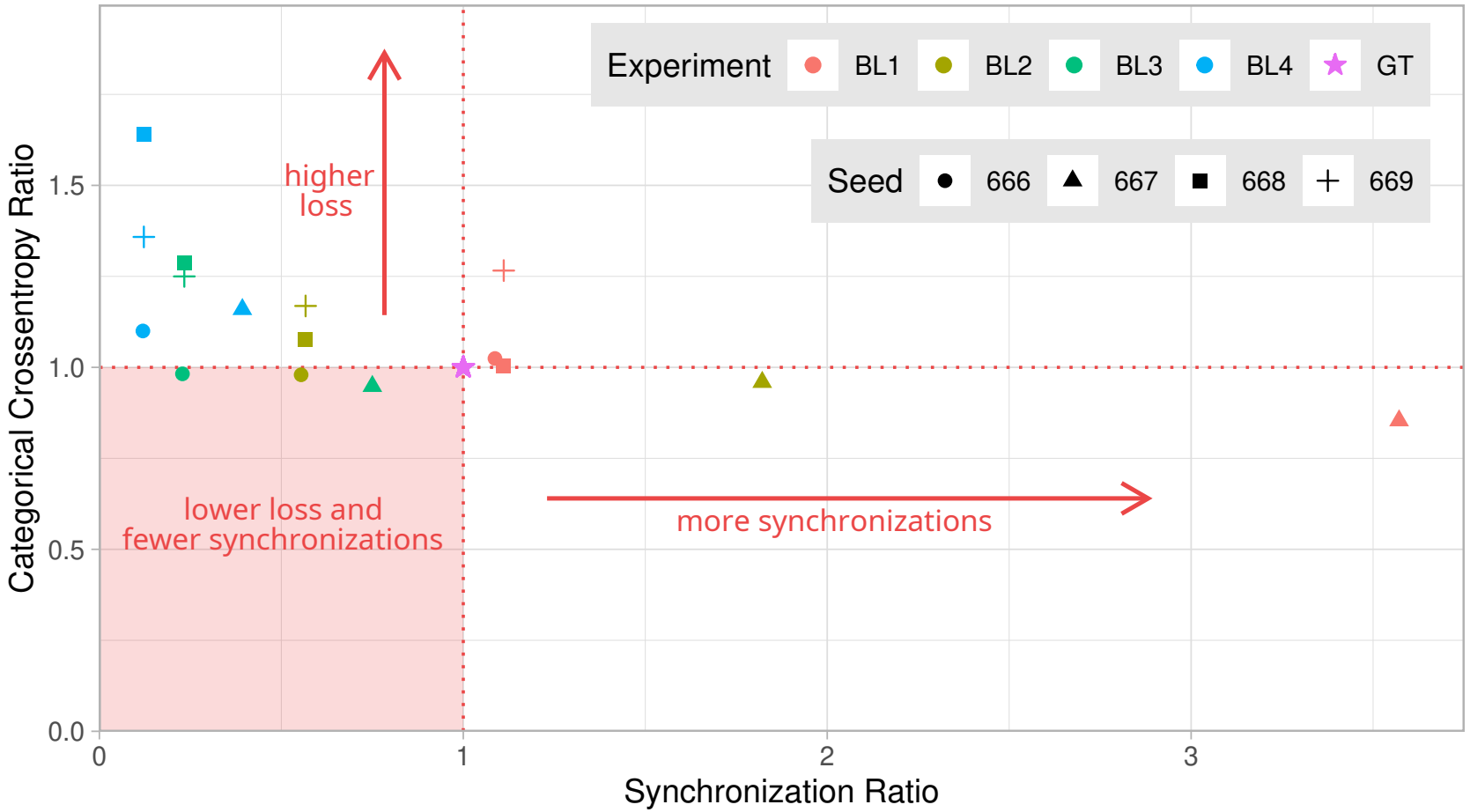
SVHN – Model II

Layer type	Output Shape	# Parameters	Configuration
↑	↑	↑	↑
2D convolution layer	$8 \times 8 \times 128$	73856	3×3 kernel ReLU activation
Batch normalization	$8 \times 8 \times 128$	512	None
2D convolution layer	$8 \times 8 \times 128$	147584	3×3 kernel ReLU activation
2D max pooling operation	$4 \times 4 \times 128$	0	2×2 pool
Dropout layer	$4 \times 4 \times 128$	0	30% drop rate
Input flattening	2048	0	None
Densely-connected layer	128	262272	ReLU activation
Dropout layer	128	0	40% drop rate
Densely-connected layer	10	1290	Softmax activation

Baseline Comparison – Iris

Exp.	SEED 666		SEED 667		SEED 668		SEED 669	
	Loss	Sync.	Loss	Sync.	Loss	Sync.	Loss	Sync.
BL1	0.539029	100	0.501504	100	0.474516	100	0.487596	100
BL2	0.515485	51	0.563500	51	0.508835	51	0.450143	51
BL3	0.516858	21	0.556961	21	0.609173	21	0.481423	21
BL4	0.578848	11	0.681120	11	0.776382	11	0.523183	11
GT	0.526247	92	0.587449	28	0.473216	90	0.385178	90

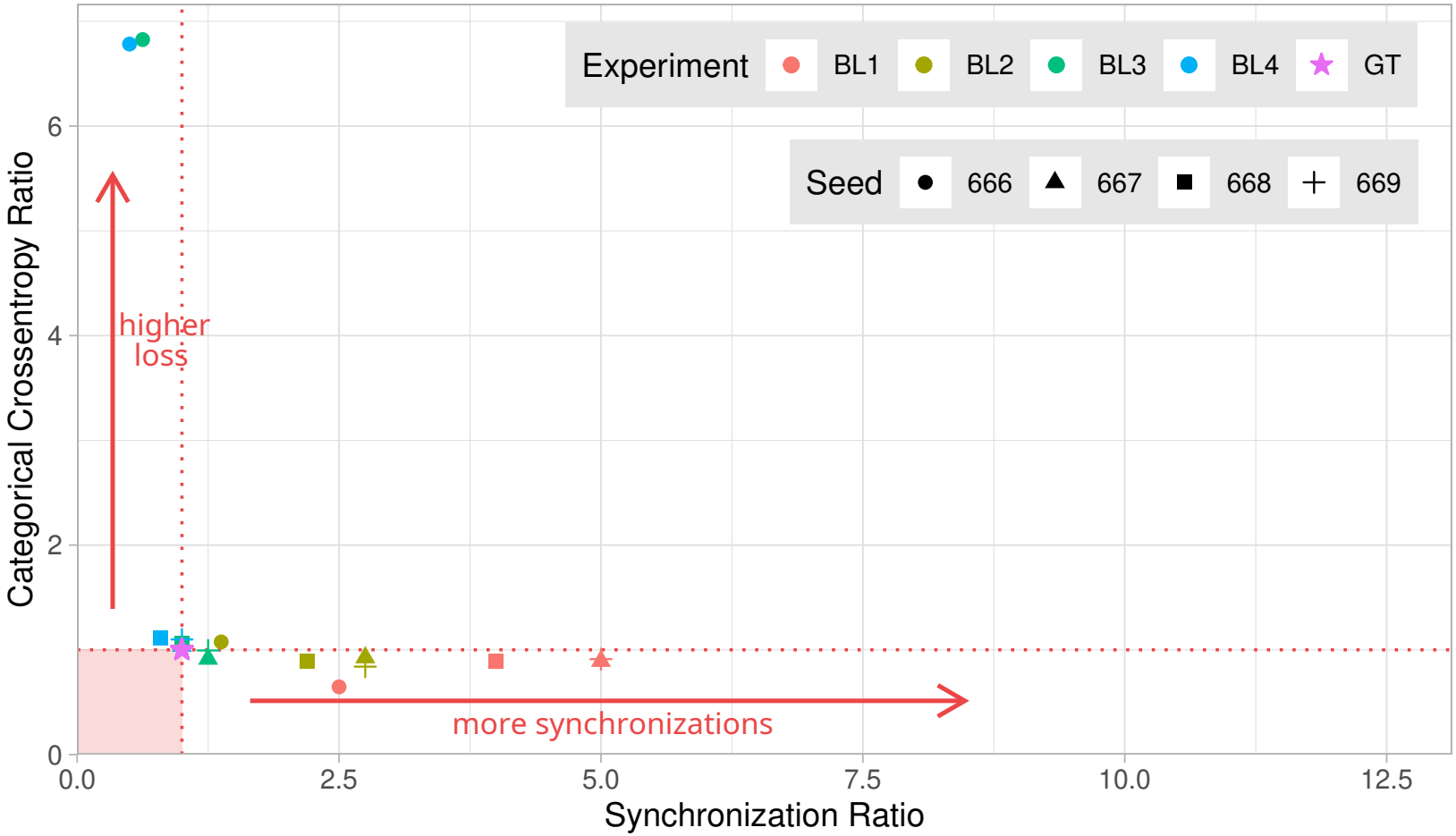
Baseline Comparison – Iris contd.



Baseline Comparison – Mnist

Exp.	SEED 666		SEED 667		SEED 668		SEED 669	
	Loss	Sync.	Loss	Sync.	Loss	Sync.	Loss	Sync.
BL1	0.232185	20	0.285152	20	0.205982	20	0.497485	20
BL2	0.386331	11	0.296058	11	0.206932	11	0.458328	11
BL3	2.451042	5	0.291849	5	0.246449	5	0.542560	5
BL4	2.435904	4	0.331246	4	0.258262	4	0.599970	4
GT	0.359094	8	0.319479	4	0.231651	5	0.545842	4

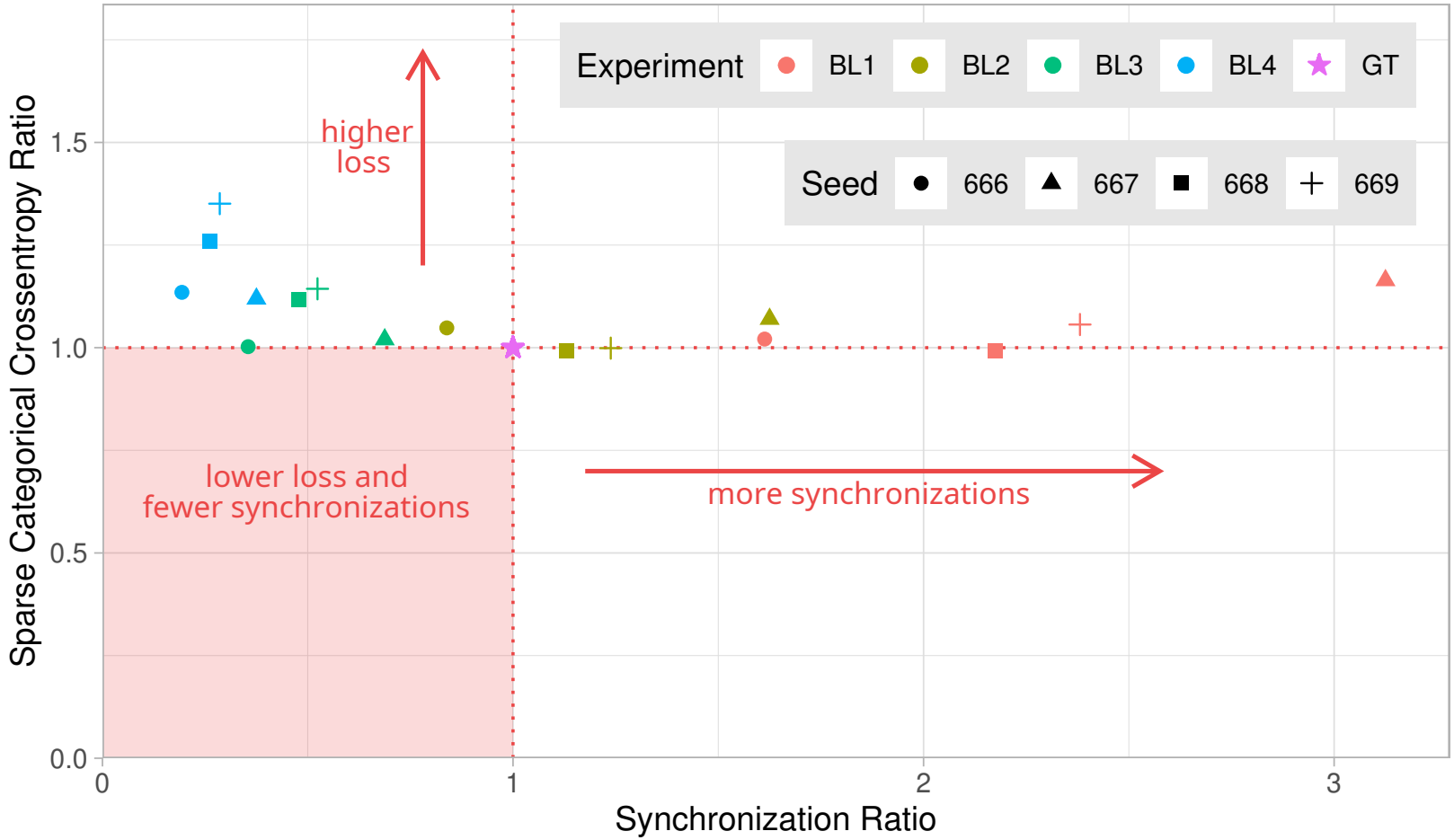
Baseline Comparison – Mnist contd.



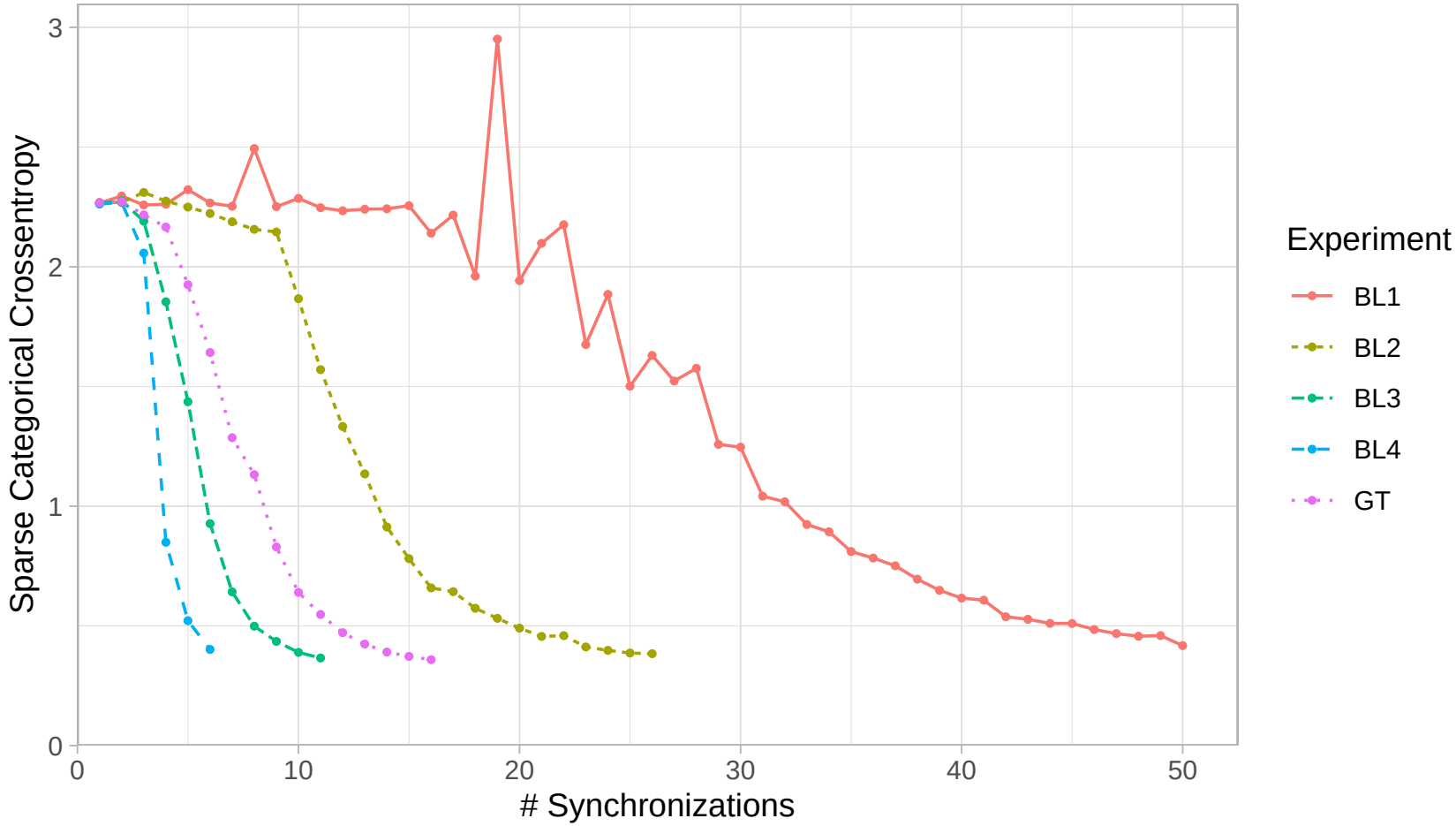
Baseline Comparison – SVHN

Exp.	SEED 666		SEED 667		SEED 668		SEED 669	
	Loss	Sync.	Loss	Sync.	Loss	Sync.	Loss	Sync.
BL1	0.426264	50	0.417600	50	0.378228	50	0.395136	50
BL2	0.437644	26	0.383815	26	0.378151	26	0.373628	26
BL3	0.418537	11	0.365922	11	0.425989	11	0.427693	11
BL4	0.473795	6	0.401534	6	0.479799	6	0.505273	6
GT	0.417503	31	0.358765	16	0.380986	23	0.374048	21

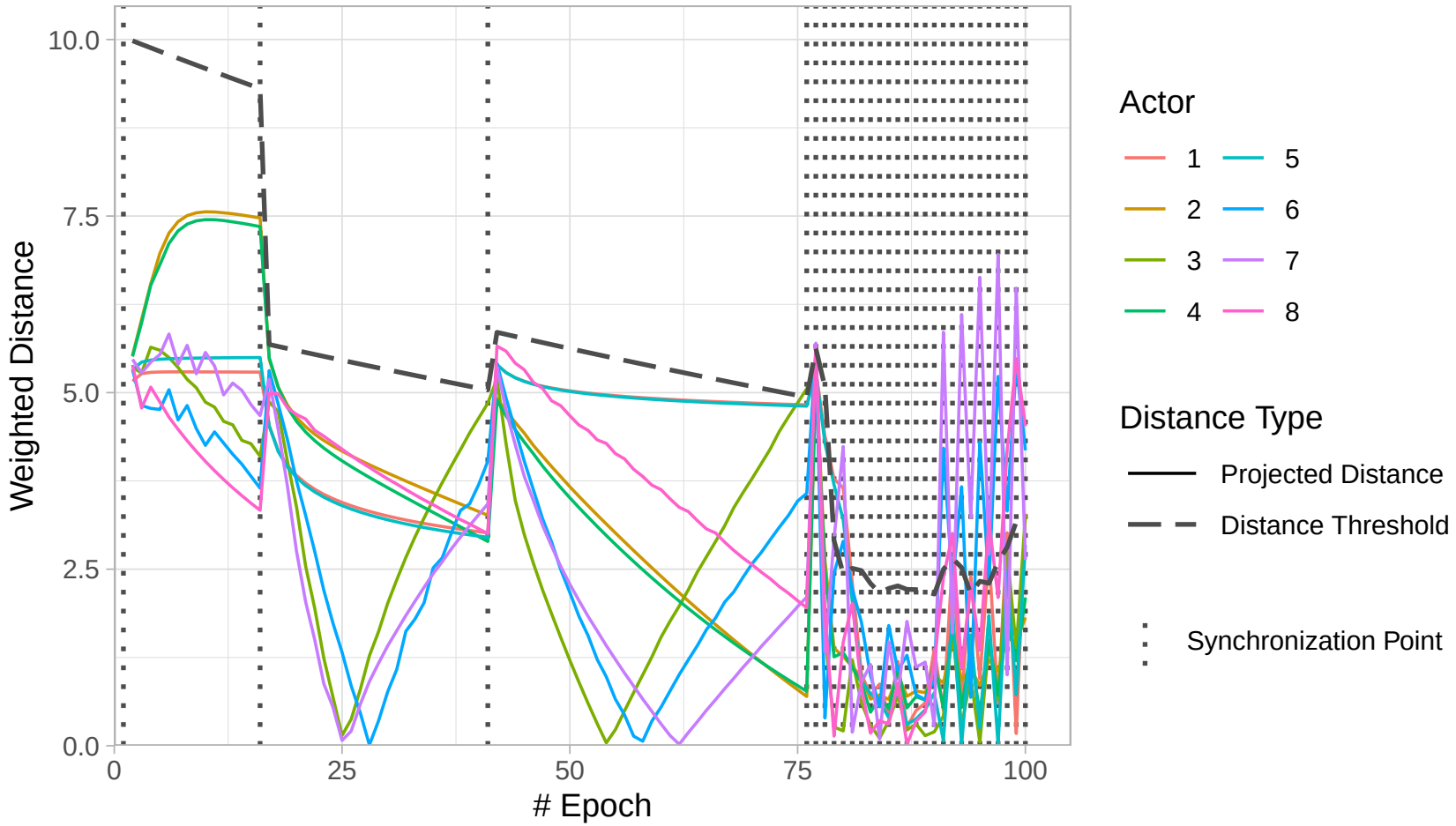
Baseline Comparison – SVHN contd.



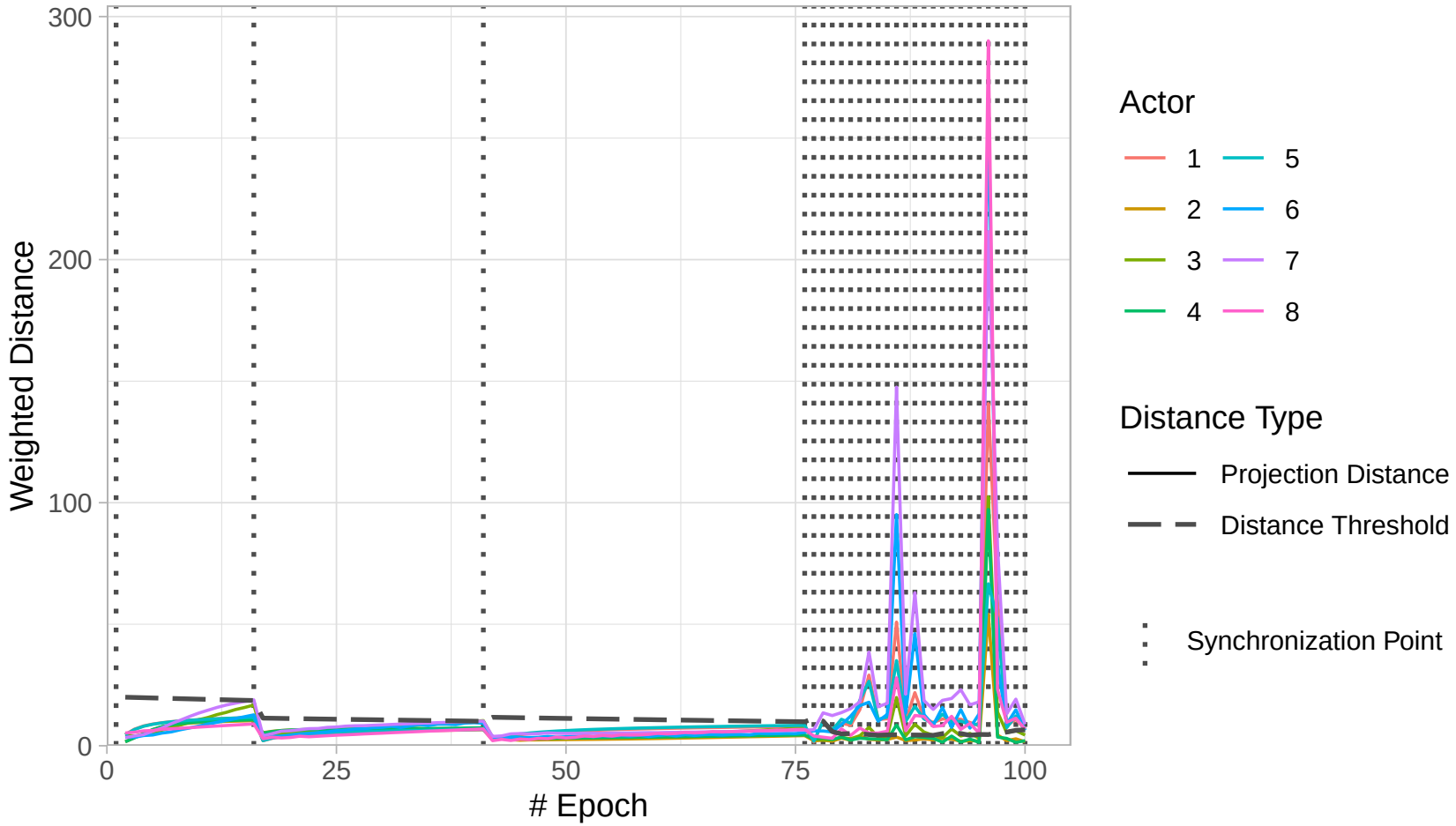
Baseline Comparison – SVHN contd.



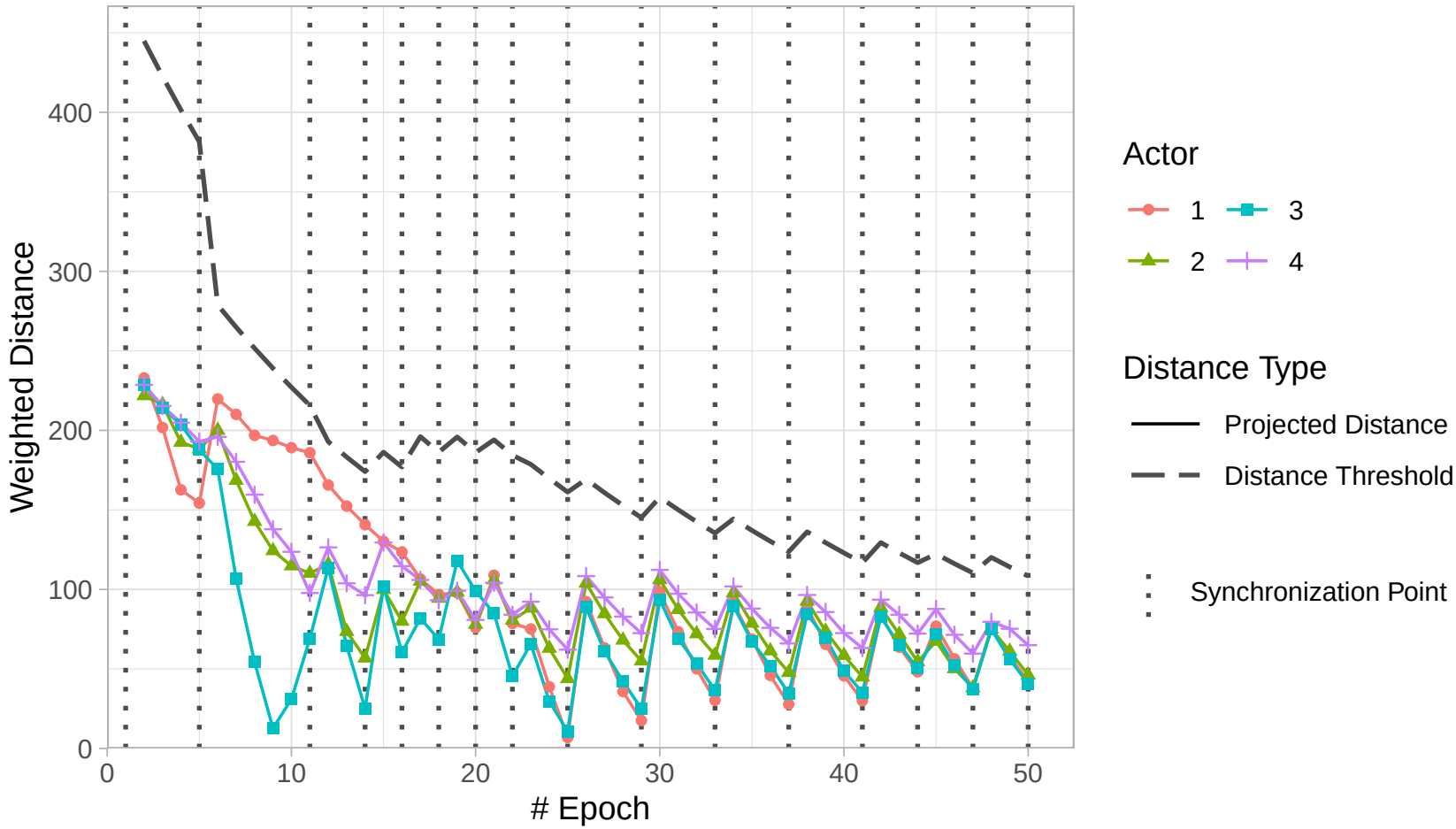
Threshold Distance – Iris



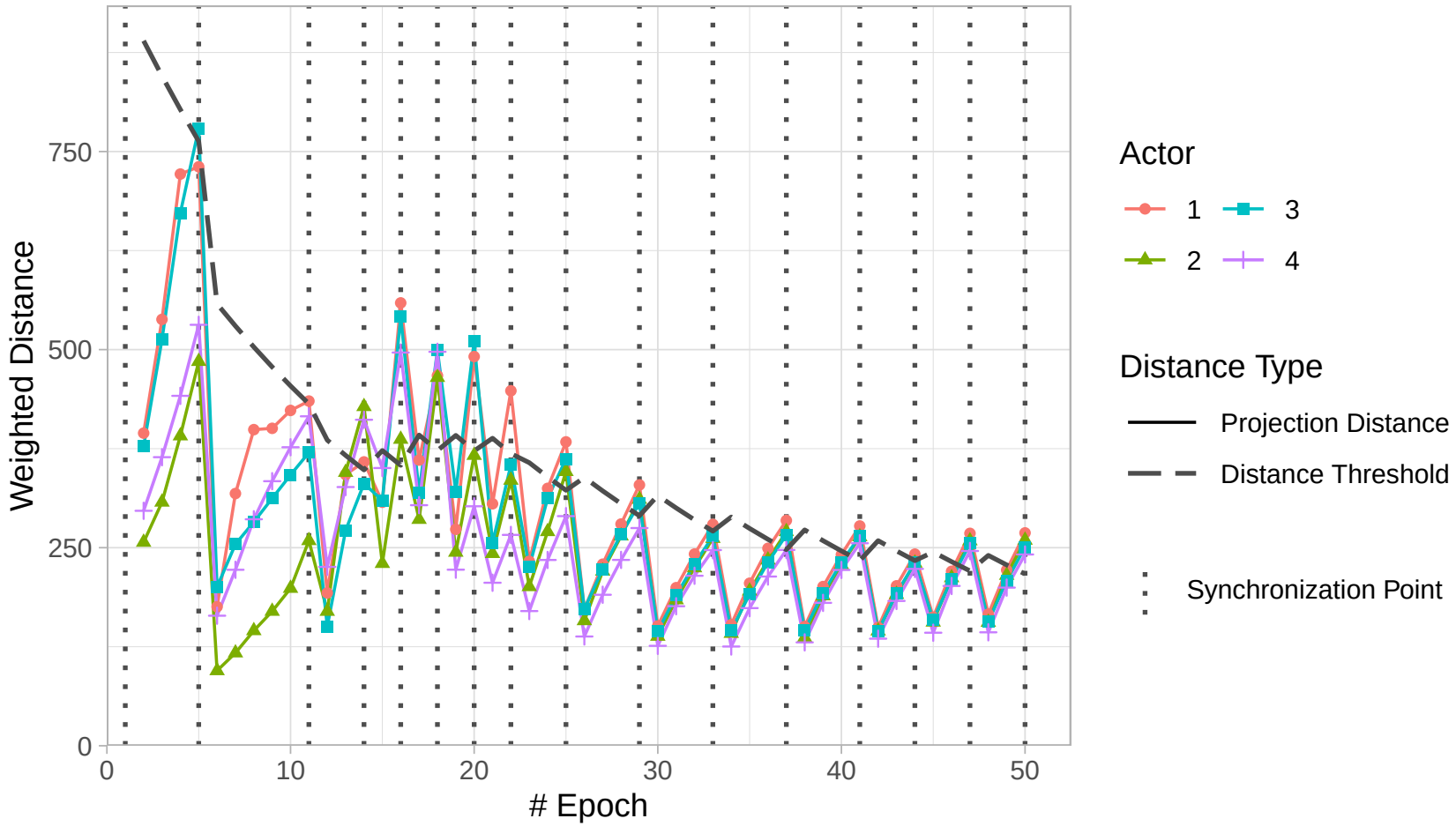
Threshold Distance – Iris contd.



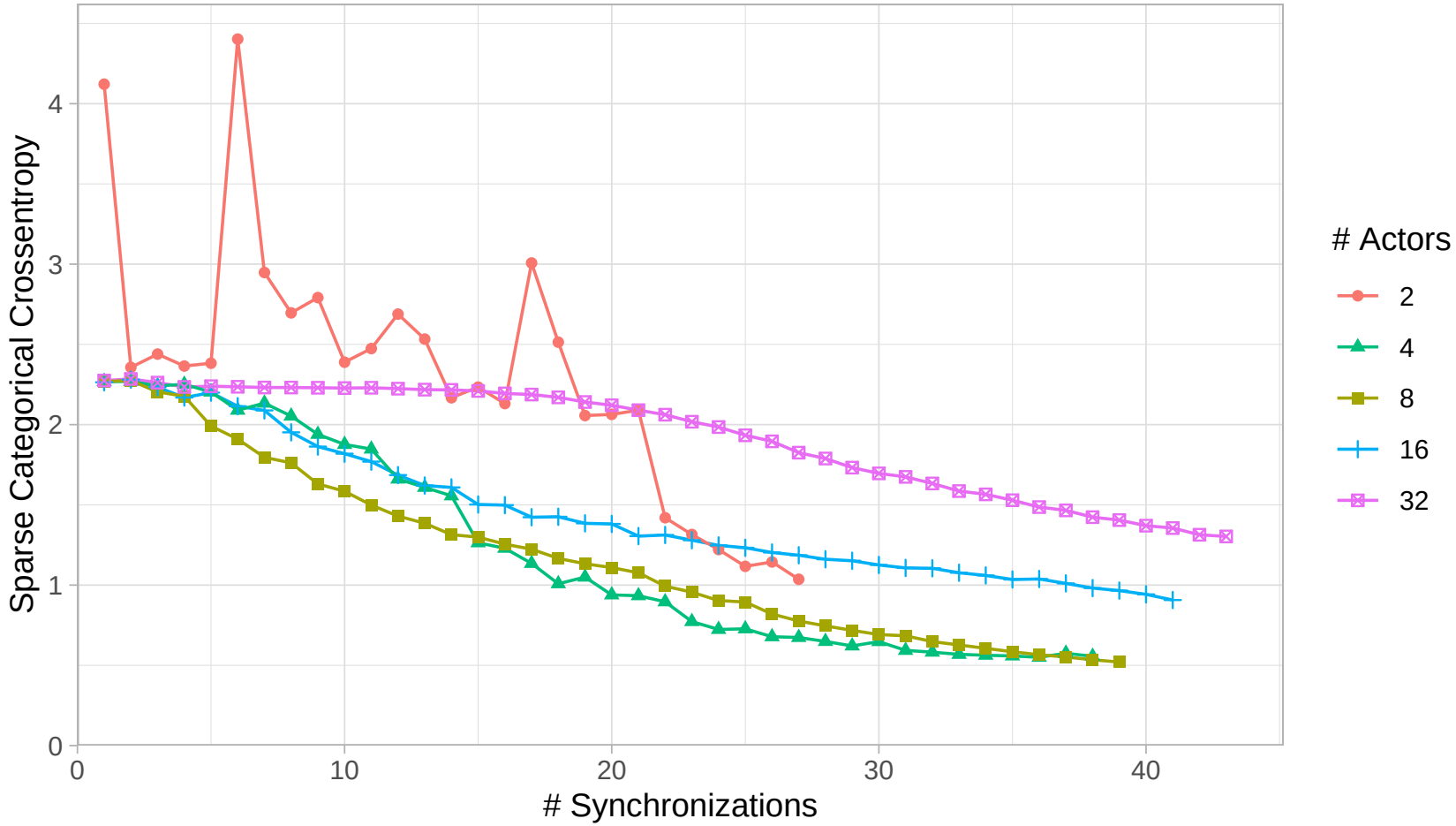
Threshold Distance – SVHN



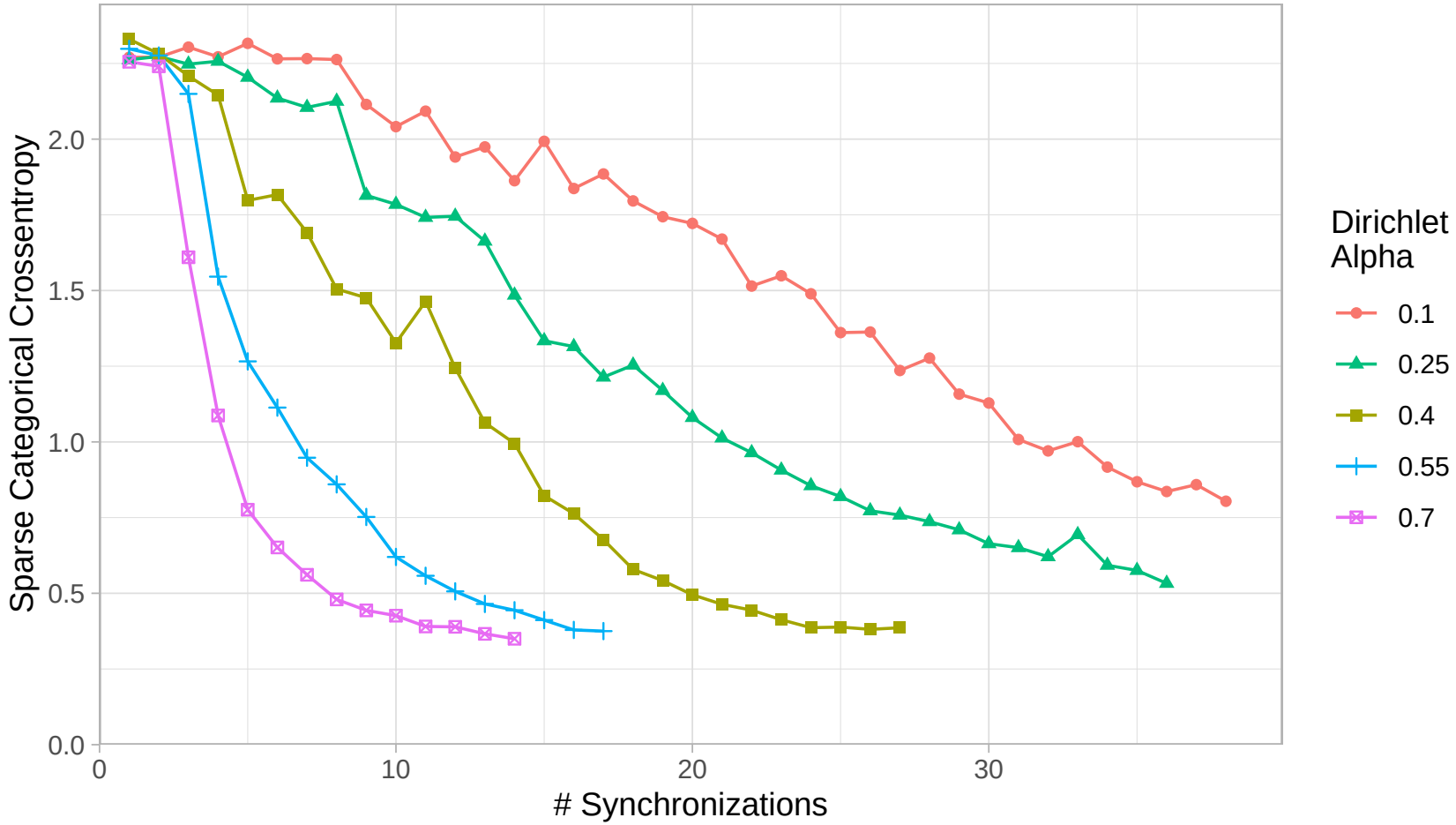
Threshold Distance – SVHN contd.



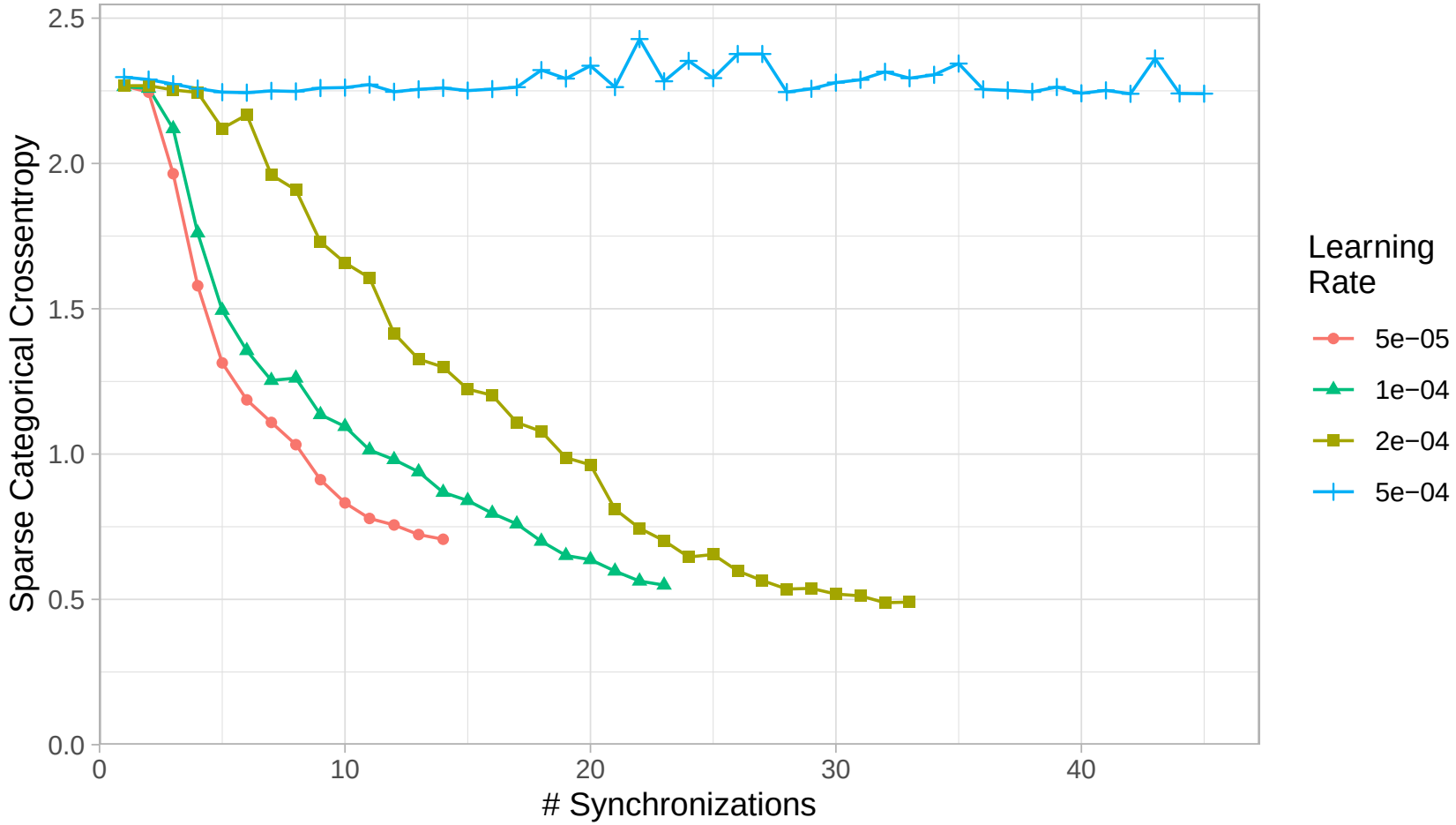
Variability Test – SVHN



Variability Test – SVHN contd.



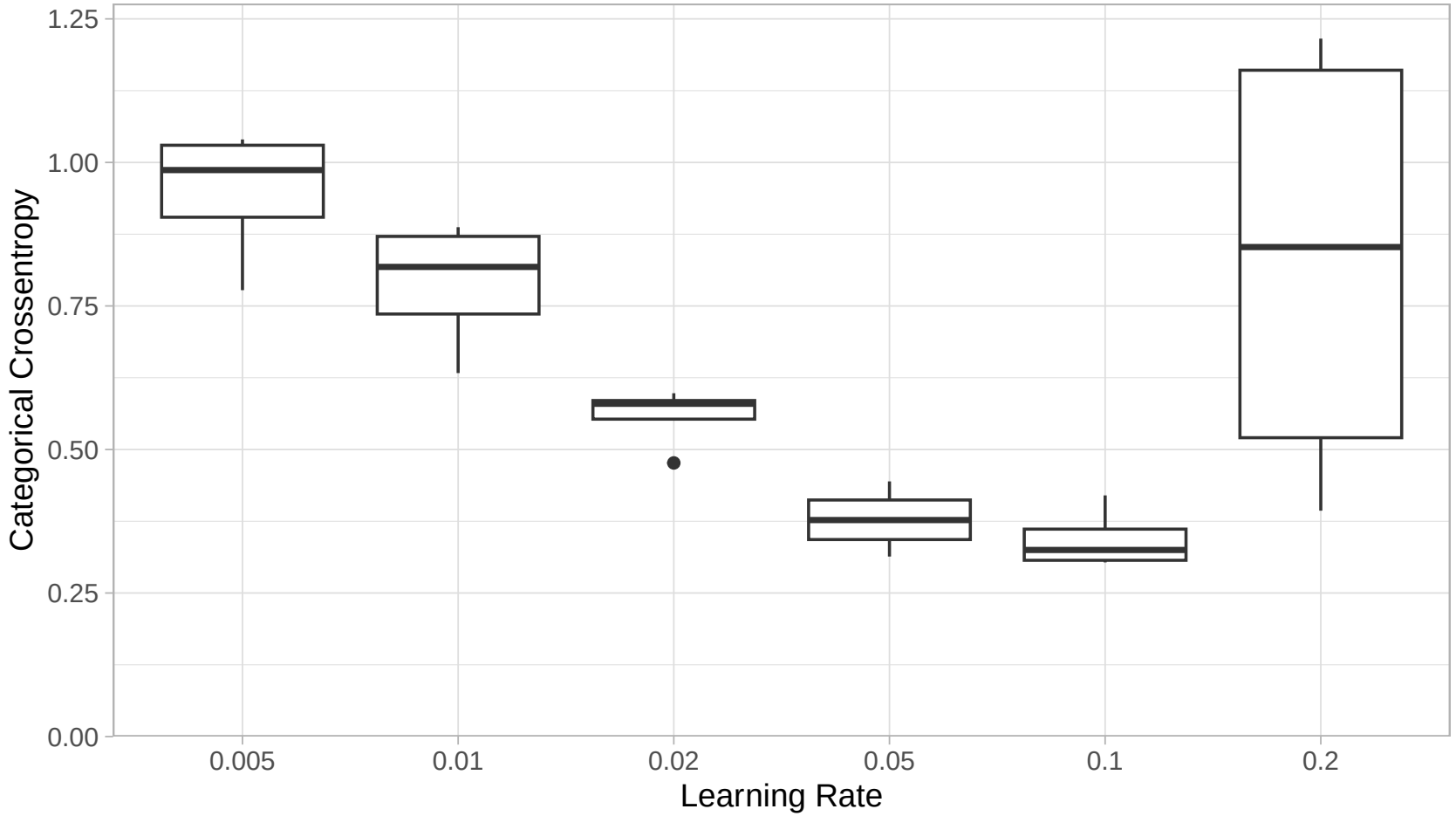
Variability Test – SVHN contd.



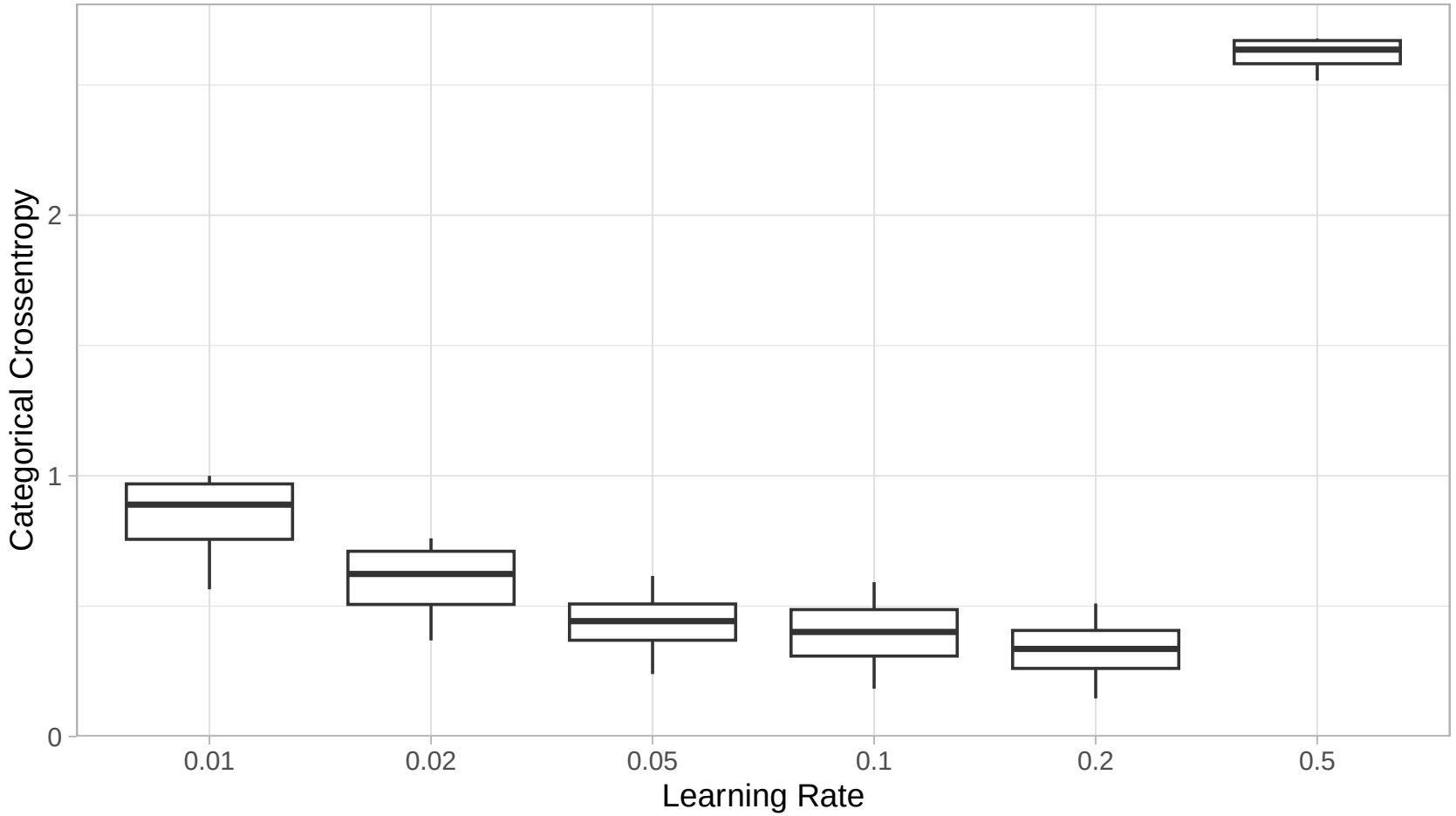
Future Work

- Theoretical foundation
  - Intuition $\rightarrow$ theoretical basis
- Adaptation of sensitivity
  - Dynamic adjustment of θ_ρ
  - (Stochastic random walk model)
- Arbitrary network topology
  - Personalized threshold regions (multiple)

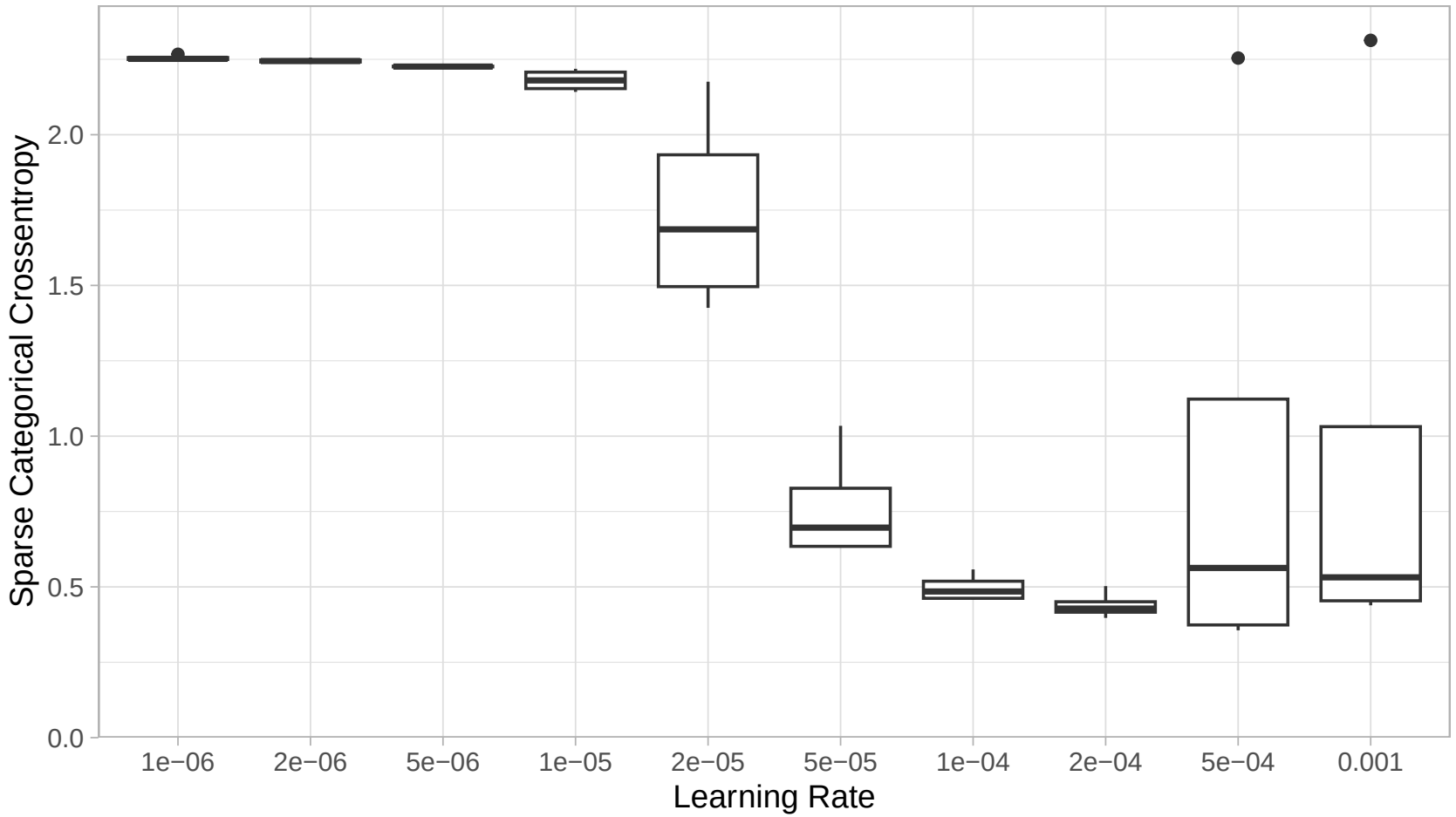
Learning Rate Grid Search – Iris



Learning Rate Grid Search – Mnist



Learning Rate Grid Search – SVHN



Choice of Sensitivity Parameter – Iris

Experiment		SEED 666		SEED 667		SEED 668		SEED 669	
		Loss	Sync.	Loss	Sync.	Loss	Sync.	Loss	Sync.
Baseline 1		0.4827	100	0.5441	100	0.5110	100	0.2893	100
Baseline 2		0.3913	51	0.6352	51	0.5380	51	0.2934	51
Baseline 3		0.3654	21	0.6212	21	0.5930	21	0.3458	21
Baseline 4		0.4108	11	0.7794	11	0.6974	11	0.4022	11
GT	$\theta_\rho = 1$	0.4653	98	0.6615	95	0.5090	96	0.2901	99
	$\theta_\rho = 2$	0.4656	92	0.6642	28	0.5093	90	0.2066	90
	$\theta_\rho = 3$	0.2886	82	0.7854	5	0.5538	78	0.4632	5
	$\theta_\rho = 4$	0.3076	42	0.8461	4	0.5129	73	0.5235	4
	$\theta_\rho = 5$	0.3059	46	0.8254	4	0.5684	61	0.5077	4

Choice of Sensitivity Parameter – Mnist

Experiment		SEED 666		SEED 667		SEED 668		SEED 669	
		Loss	Sync.	Loss	Sync.	Loss	Sync.	Loss	Sync.
Baseline 1		0.2470	20	0.2774	20	0.2265	20	0.5016	20
Baseline 2		0.3999	11	0.2893	11	0.2296	11	0.4557	11
Baseline 3		2.4647	5	0.2899	5	0.2745	5	0.5348	5
Baseline 4		2.4486	4	0.3286	4	0.2894	4	0.5919	4
GT	$\theta_\rho = 1$	0.3237	20	0.2681	16	0.2274	16	0.4624	17
	$\theta_\rho = 1.25$	0.2491	18	0.3772	14	0.2308	13	0.4695	10
	$\theta_\rho = 1.5$	2.3760	17	0.2688	5	0.2332	9	0.5544	5
	$\theta_\rho = 1.75$	0.3748	8	0.3147	4	0.2617	5	0.5346	4
	$\theta_\rho = 2$	0.3847	11	0.2920	4	0.2942	4	0.5816	4
	$\theta_\rho = 2.25$	2.5542	7	0.3223	3	0.2937	4	0.5848	3
	$\theta_\rho = 2.5$	2.4151	6	0.2991	3	0.3092	4	0.6917	3
	$\theta_\rho = 2.75$	2.3923	6	0.3308	4	0.2974	4	0.7008	3
	$\theta_\rho = 3$	1.8258	8	0.2741	3	0.2797	3	0.6897	3

Choice of Sensitivity Parameter – SVHN

Experiment		SEED 666		SEED 667		SEED 668		SEED 669	
		Loss	Sync.	Loss	Sync.	Loss	Sync.	Loss	Sync.
Baseline 1		0.3995	50	0.4202	50	0.3748	50	0.4186	50
Baseline 2		0.4215	26	0.3932	26	0.3769	26	0.3997	26
Baseline 3		0.4110	11	0.3721	11	0.4341	11	0.4433	11
Baseline 4		0.4701	6	0.4070	6	0.4729	6	0.5275	6
GT	$\theta_\rho = 1$	0.4046	48	0.4319	50	0.3787	48	0.4309	48
	$\theta_\rho = 2$	0.4088	31	0.3642	16	0.3832	23	0.3972	21
	$\theta_\rho = 3$	0.4300	9	0.3876	7	0.4237	9	0.4537	8
	$\theta_\rho = 4$	0.4458	8	0.3991	7	0.4190	8	0.4614	7
	$\theta_\rho = 5$	0.4698	7	0.4212	7	0.4209	7	0.4855	7

References I

[1] H. B. McMahan et al. **Communication-Efficient Learning of Deep Networks from Decentralized Data.** International Conference on Artificial Intelligence and Statistics. 2016. URL: https://api.semanticscholar.org/CorpusID:14955348.

[2] Enrique Tomás Martínez Beltrán et al. **Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges.** IEEE Communications Surveys & Tutorials 25 (2022), pp. 2983–3013. URL: https://api.semanticscholar.org/CorpusID:253523208.

[3] Ziteng Sun et al. **Can You Really Backdoor Federated Learning?** ArXiv abs/1911.07963 (2019). URL: https://api.semanticscholar.org/CorpusID:208157929.

[4] Hongyi Wang et al. **Attack of the Tails: Yes, You Really Can Backdoor Federated Learning.** ArXiv abs/2007.05084 (2020). URL: https://api.semanticscholar.org/CorpusID:220487047.

[5] Han Wang et al. **SparSFA: Towards robust and communication-efficient peer-to-peer federated learning.** Comput. Secur. 129 (2023), p. 103182. URL: https://api.semanticscholar.org/CorpusID:257566786.

[6] Vansh Gupta et al. **TravellingFL: Communication Efficient Peer-to-Peer Federated Learning.** IEEE Transactions on Vehicular Technology 73 (2024), pp. 5005–5019. URL: https://api.semanticscholar.org/CorpusID:265225957.

References II

[7] Ji Liu et al. **AEDFL: Efficient Asynchronous Decentralized Federated Learning with Heterogeneous Devices.** ArXiv abs/2312.10935 (2023). URL: https://api.semanticscholar.org/CorpusID:266359500.

[8] Jiajun Wu et al. **Topology-aware Federated Learning in Edge Computing: A Comprehensive Survey.** ACM Computing Surveys 56 (2023), pp. 1–41. URL: https://api.semanticscholar.org/CorpusID:256616242.

[9] Hui Chen et al. **Advancements in Federated Learning: Models, Methods, and Privacy.** ACM Computing Surveys (2023). URL: https://api.semanticscholar.org/CorpusID:257078992.

[10] Qazi Waqas Khan et al. **Decentralized Machine Learning Training: A Survey on Synchronization, Consolidation, and Topologies.** IEEE Access 11 (2023), pp. 68031–68050. URL: https://api.semanticscholar.org/CorpusID:259815892.

[11] Liangqi Yuan et al. **Decentralized Federated Learning: A Survey and Perspective.** IEEE Internet of Things Journal 11 (2023), pp. 34617–34638. URL: https://api.semanticscholar.org/CorpusID:259064130.

[12] Edoardo Gabrielli, Giovanni Pica, and Gabriele Tolomei. **A Survey on Decentralized Federated Learning.** ArXiv abs/2308.04604 (2023). URL: https://api.semanticscholar.org/CorpusID:260735643.

References III

[13] Saqr Khalil Saeed Thabet et al. **Towards Efficient Decentralized Federated Learning: A Survey.** International Conference on Advanced Data Mining and Applications. 2024. URL: https://api.semanticscholar.org/CorpusID:275280274.

[14] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi. **Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication.** International Conference on Machine Learning. 2019. URL: https://api.semanticscholar.org/CorpusID:59553565.

[15] Chengxi Li, Gang Li, and Pramod K. Varshney. **Decentralized Federated Learning via Mutual Knowledge Transfer.** IEEE Internet of Things Journal 9 (2020), pp. 1136–1147. URL: https://api.semanticscholar.org/CorpusID:229371278.

[16] Jianyu Wang et al. **Matcha: A Matching-Based Link Scheduling Strategy to Speed up Distributed Optimization.** IEEE Transactions on Signal Processing 70 (2022), pp. 5208–5221. URL: https://api.semanticscholar.org/CorpusID:253461410.

[17] Zhenheng Tang et al. **GossipFL: A Decentralized Federated Learning Framework With Sparsified and Adaptive Communication.** IEEE Transactions on Parallel and Distributed Systems 34 (2023), pp. 909–922. URL: https://api.semanticscholar.org/CorpusID:255046867.

References IV

[18] Wei Liu et al. **Communication-Efficient Design for Quantized Decentralized Federated Learning.** IEEE Transactions on Signal Processing 72 (2023), pp. 1175–1188. URL: https://api.semanticscholar.org/CorpusID:257532612.

[19] Shenglong Zhou, Kaidi Xu, and Geoffrey Y. Li. **Communication-Efficient Decentralized Federated Learning via One-Bit Compressive Sensing.** 2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring) (2023), pp. 1–5. URL: https://api.semanticscholar.org/CorpusID:261395891.

[20] Lun Wang et al. **Accelerating Decentralized Federated Learning in Heterogeneous Edge Computing.** IEEE Transactions on Mobile Computing 22 (2023), pp. 5001–5016. URL: https://api.semanticscholar.org/CorpusID:249144039.

[21] Ruixing Zong et al. **Fedcs: Efficient communication scheduling in decentralized federated learning.** Inf. Fusion 102 (2023), p. 102028. URL: https://api.semanticscholar.org/CorpusID:262156336.

[22] Yunming Liao et al. **Asynchronous Decentralized Federated Learning for Heterogeneous Devices.** IEEE/ACM Transactions on Networking 32 (2024), pp. 4535–4550. URL: https://api.semanticscholar.org/CorpusID:271236198.

References V

[23] Behnaz Soltani et al. **DFLStar: A Decentralized Federated Learning Framework with Self-Knowledge Distillation and Participant Selection.** International Conference on Information and Knowledge Management. 2024. URL: https://api.semanticscholar.org/CorpusID:273497250.

[24] Tianyi Chen et al. **LAG: Lazily Aggregated Gradient for Communication-Efficient Distributed Learning.** Neural Information Processing Systems. 2018. URL: https://api.semanticscholar.org/CorpusID:44061071.

[25] Jilin Zhang et al. **An Adaptive Synchronous Parallel Strategy for Distributed Machine Learning.** IEEE Access 6 (2018), pp. 19222–19230. URL: https://api.semanticscholar.org/CorpusID:5039234.

[26] Michael Kamp et al. **Efficient Decentralized Deep Learning by Dynamic Model Averaging.** ArXiv abs/1807.03210 (2018). URL: https://api.semanticscholar.org/CorpusID:49655200.

[27] Michail Theologitis et al. **Communication-Efficient Distributed Deep Learning via Federated Dynamic Averaging.** International Conference on Extending Database Technology. 2024. URL: https://api.semanticscholar.org/CorpusID:270199927.

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

References VI

[29] Jason Browniee. **Multi-Class Classification Tutorial with the Keras Deep Learning Library.** https://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/. Accessed: 2025-11-11.

[30] Yann LeCun, Corinna Cortes, and CJ Burges. **MNIST handwritten digit database.** ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist 2 (2010).

[31] Yuval Netzer et al. **Reading Digits in Natural Images with Unsupervised Feature Learning.** NIPS workshop on deep learning and unsupervised feature learning. Vol. 2011. 5. 2011, p. 7. URL: https://api.semanticscholar.org/CorpusID:16852518.

[32] Dimitrios Roussis. **SVHN Classification with CNN (Keras - 96% Acc).** https://www.kaggle.com/code/dimitriosroussis/svhn-classification-with-cnn-keras-96-acc. Accessed: 2025-11-11.