HyperFeel: An Efficient Federated Learning Framework Using Hyperdimensional Computing

Haomin Li (Speaker)

Fangxin Liu, Yichi Chen, and Li Jiang*

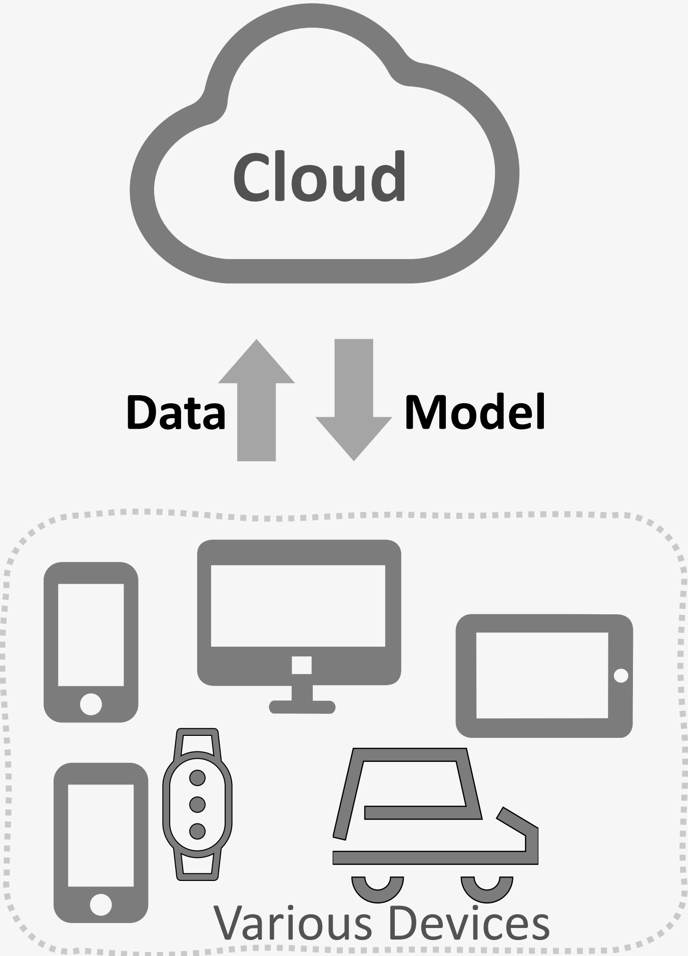
Shanghai Jiao Tong University

饮 水 思 源 • 爱 国 荣 校

Outline

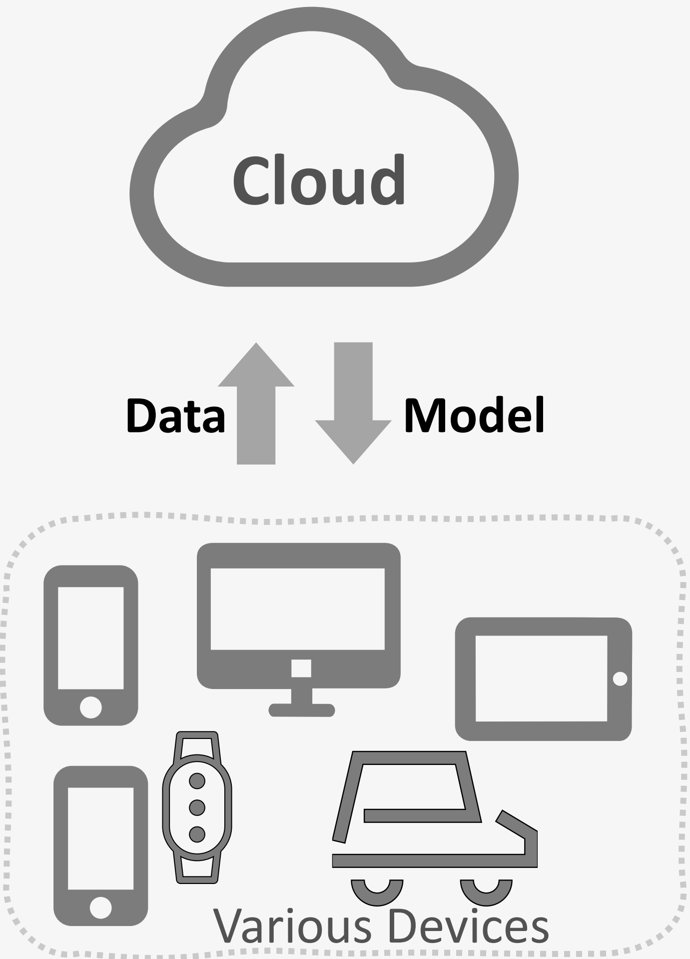
- Background and motivation
- Proposal: Efficient Federated Learning Framework Using Hyperdimensional Computing
- Design and implementation details
- Experimental results
- Conclusion

Federated Learning for Privacy computing

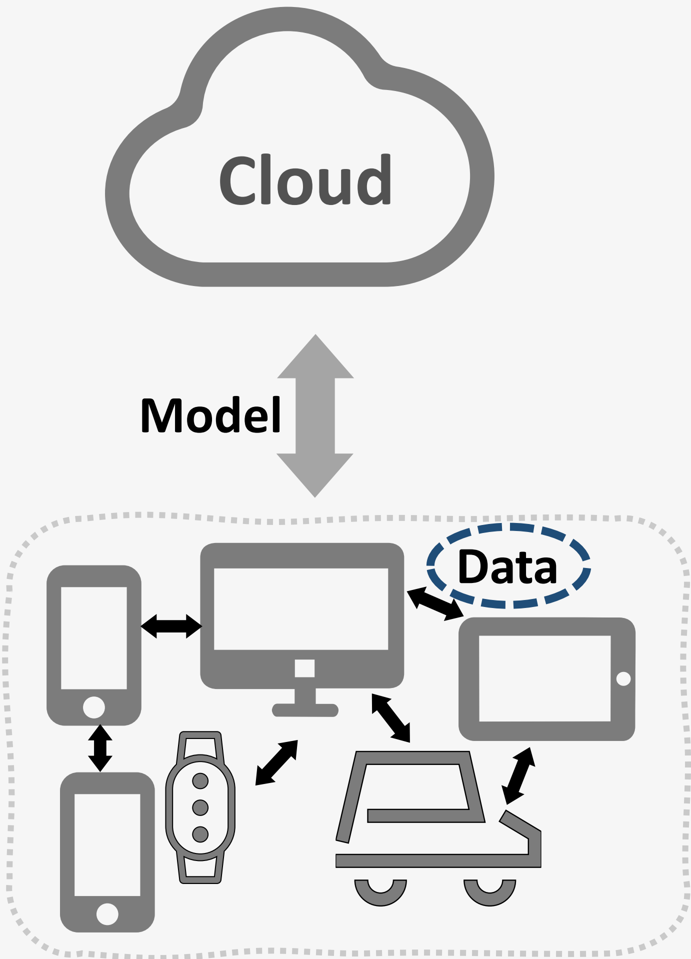


Totally Cloud

Federated Learning for Privacy computing

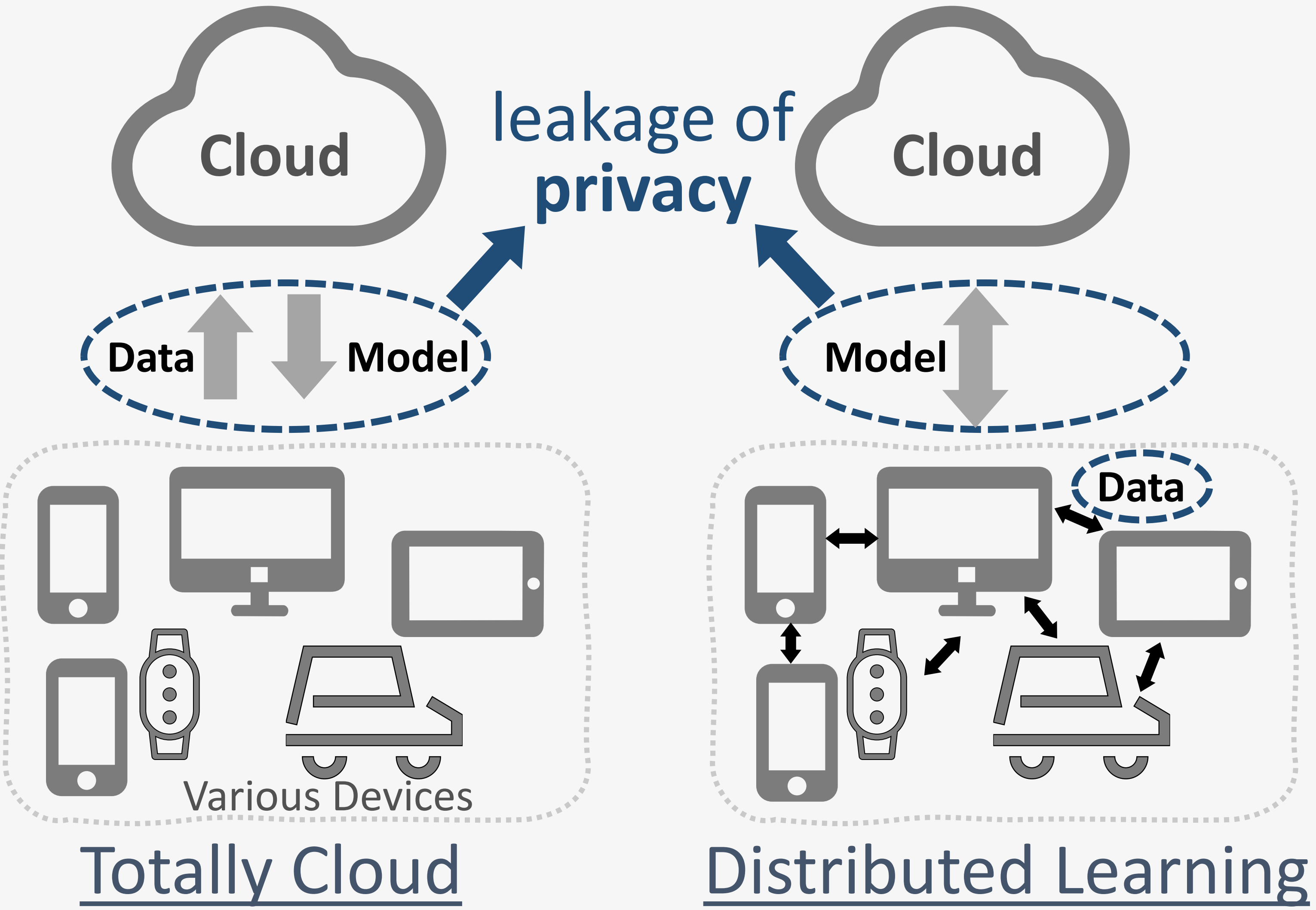


Totally Cloud

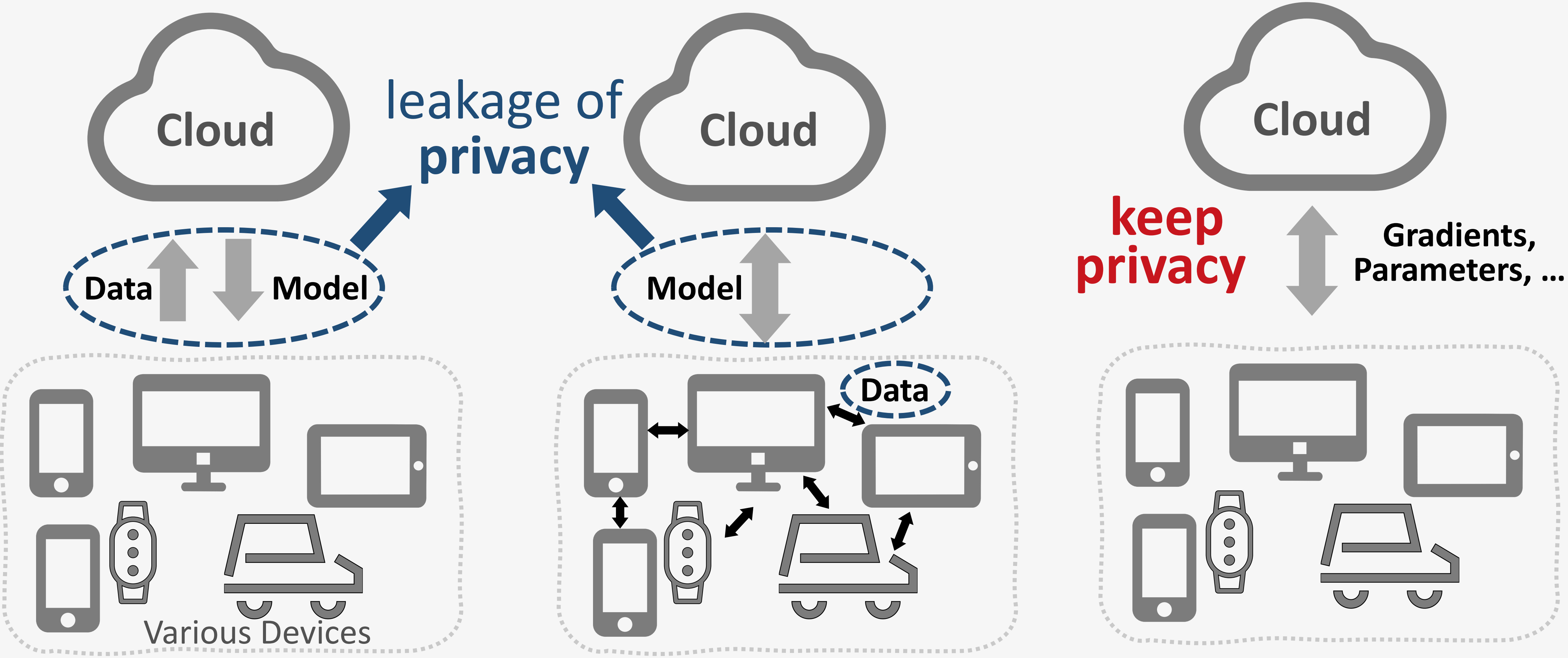


Distributed Learning

Federated Learning for Privacy computing

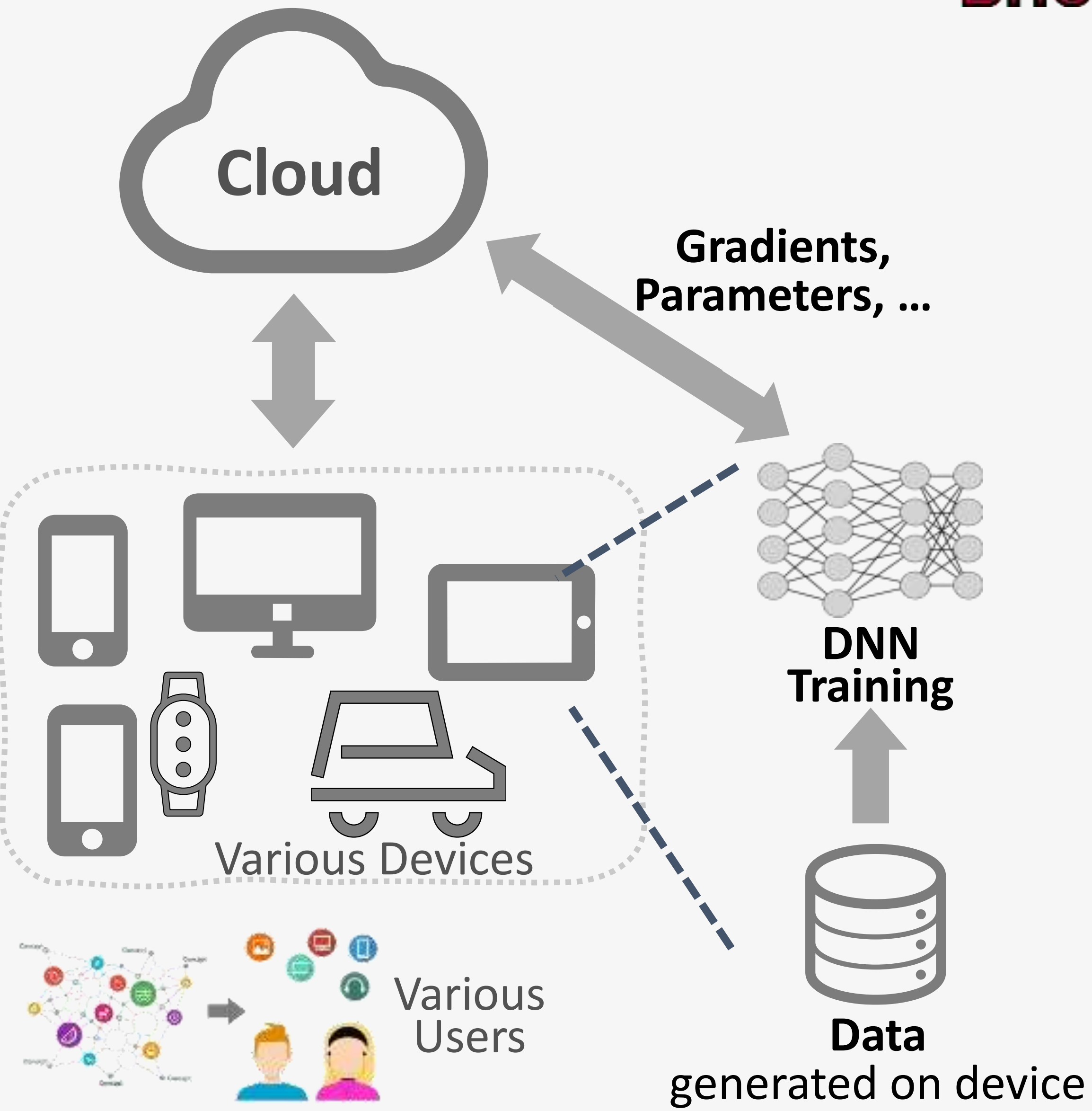


Federated Learning for Privacy computing

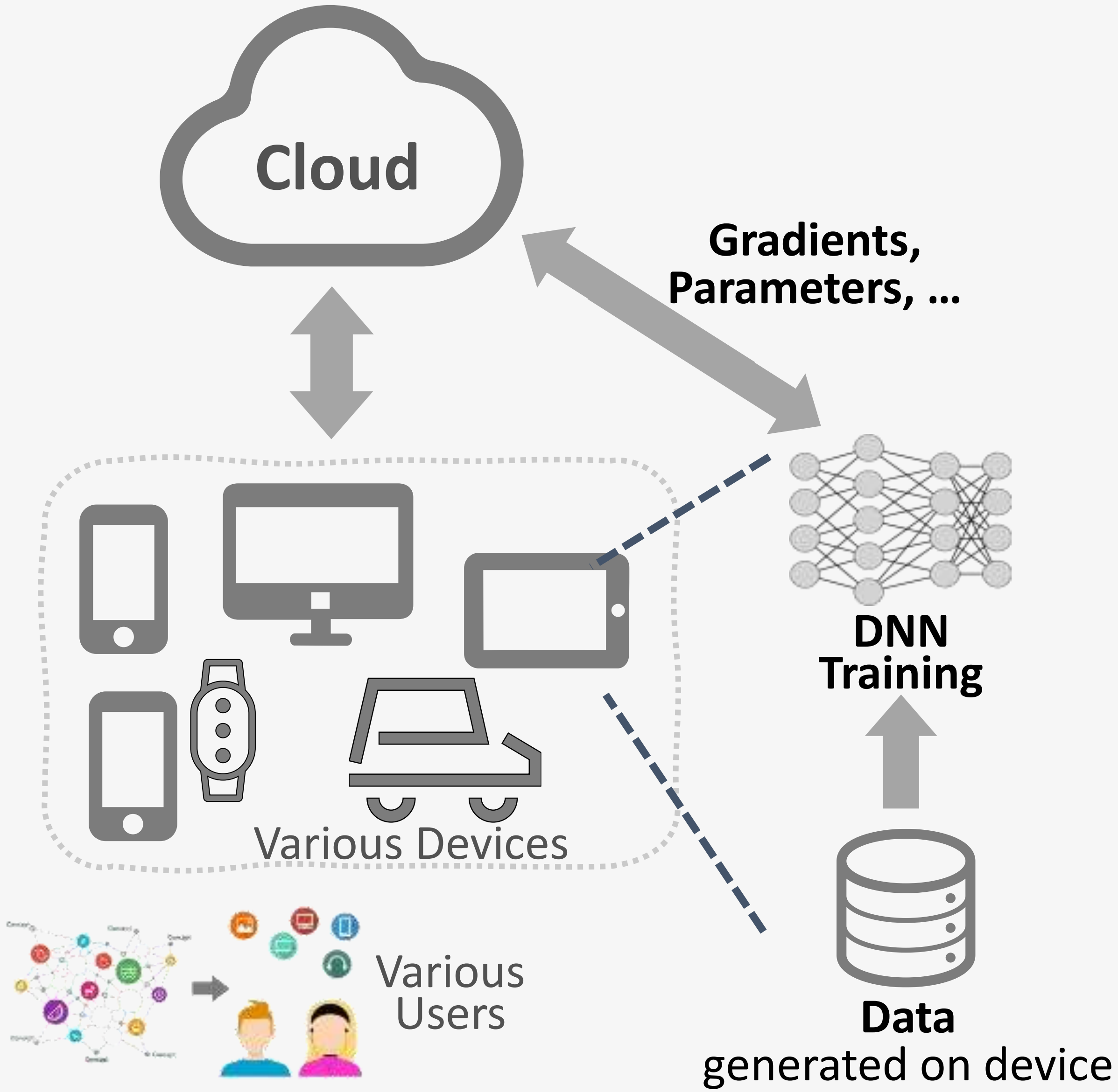


Totally Cloud   Distributed Learning   **Federated Learning**

Federated Learning

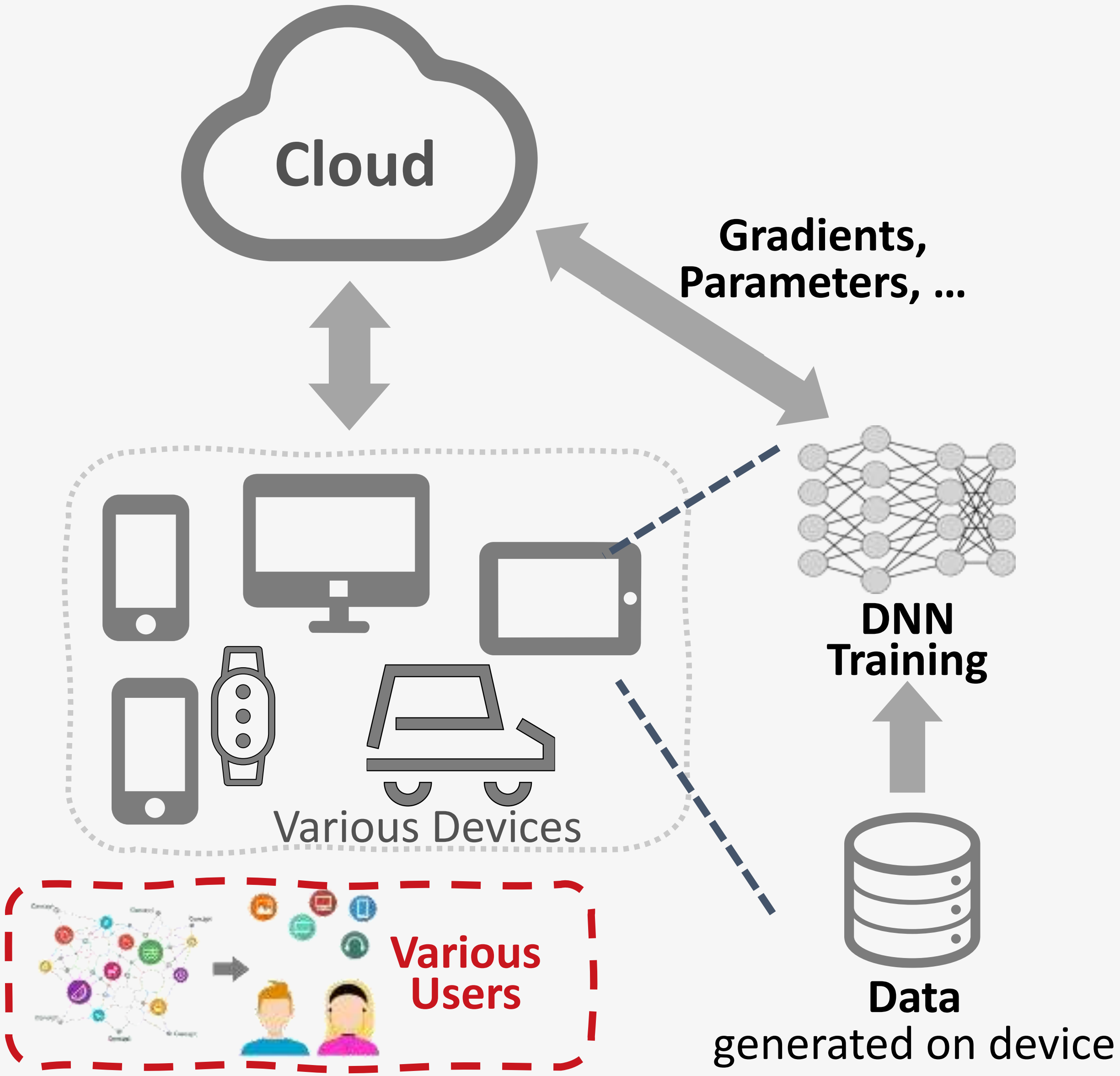


Federated Learning



FL Limitations

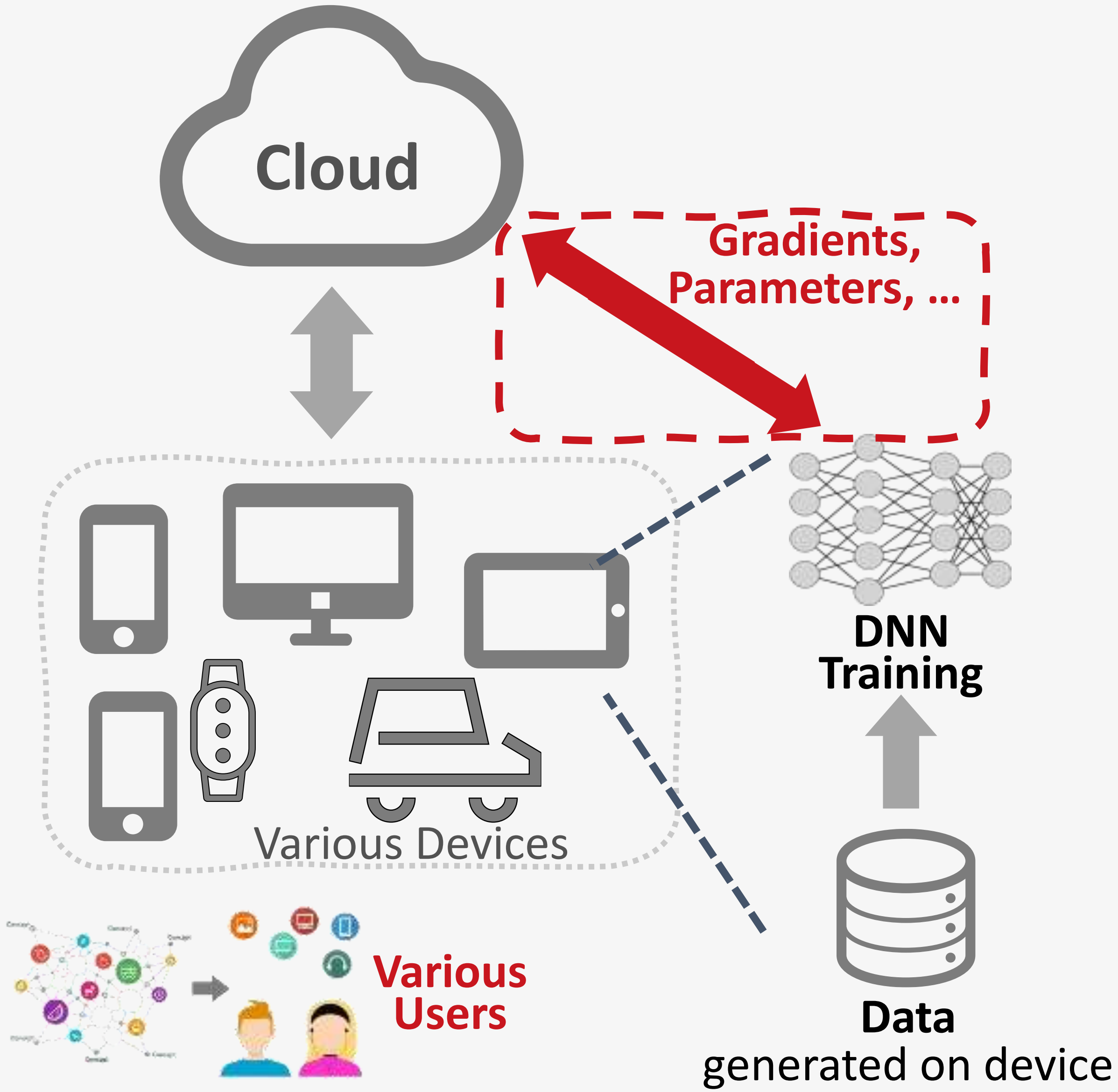
Federated Learning



FL Limitations

- Non-IID Data

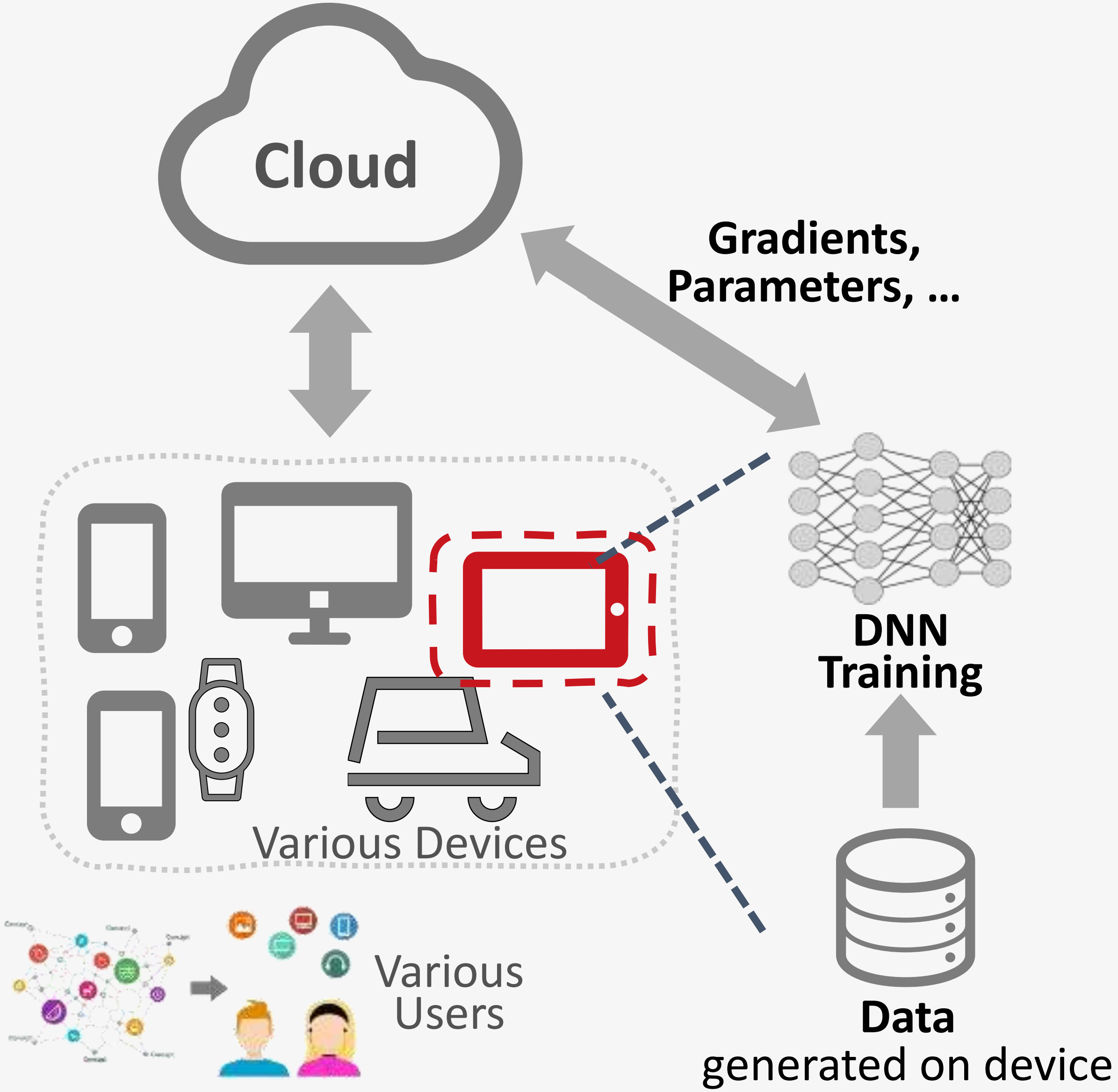
Federated Learning



FL Limitations

- Non-IID Data
- Massive Communication

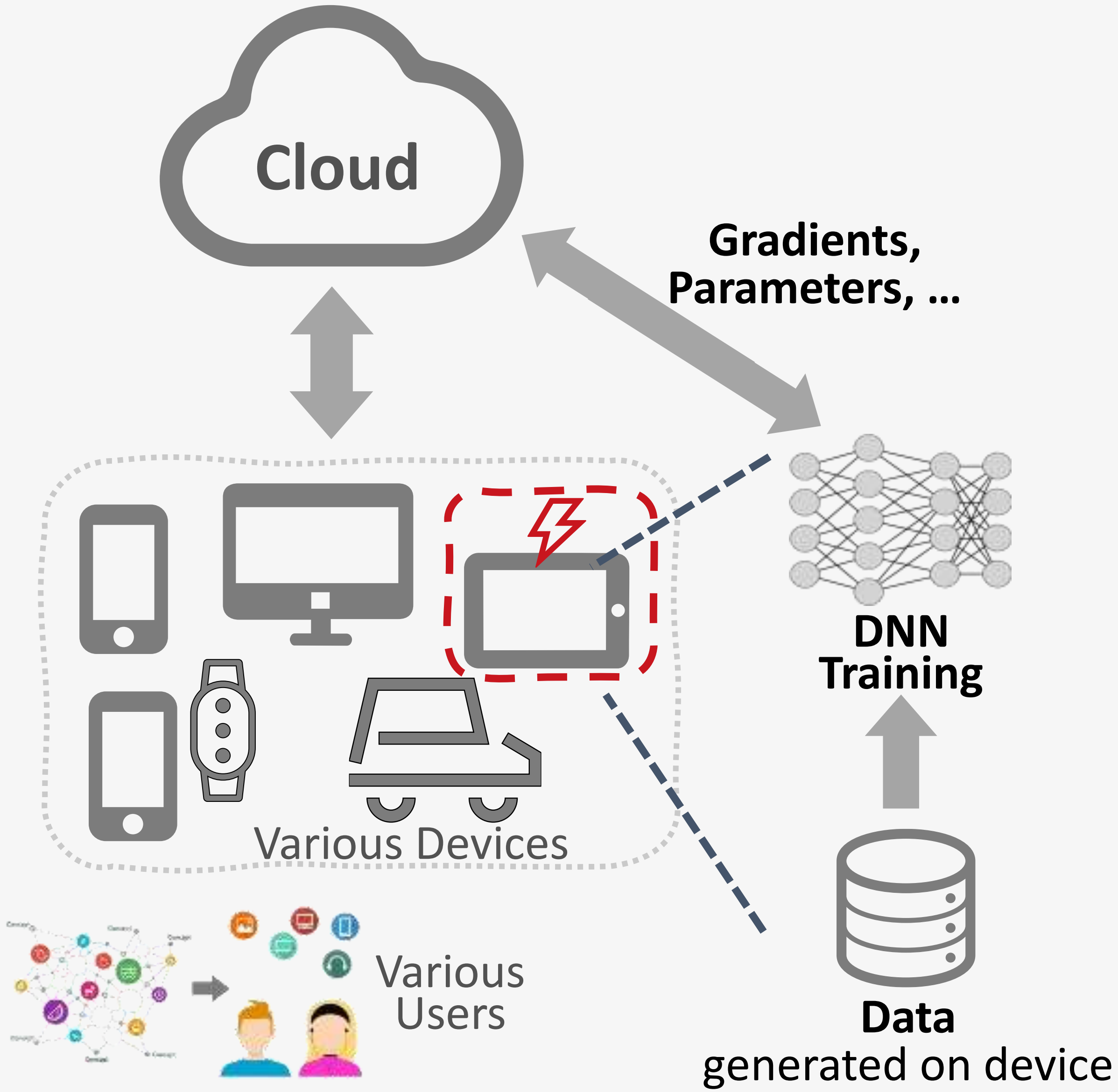
Federated Learning



FL Limitations

- Non-IID Data
- Massive Communication
- Limited Resource

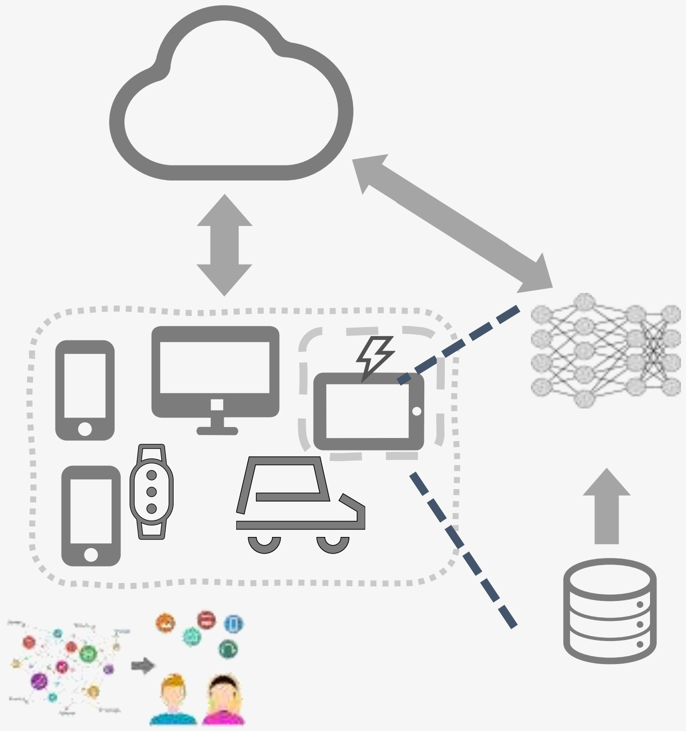
Federated Learning



FL Limitations

- Non-IID Data
- Massive Communication
- Limited Resource
- Disturbance in edge scenes
- ……

Federated Learning



FL Limitations

- Non-IID Data
- Massive Communication
- Limited Resource
- Disturbance in edge scenes
- ……

?

Hyper-Dimensional Computing

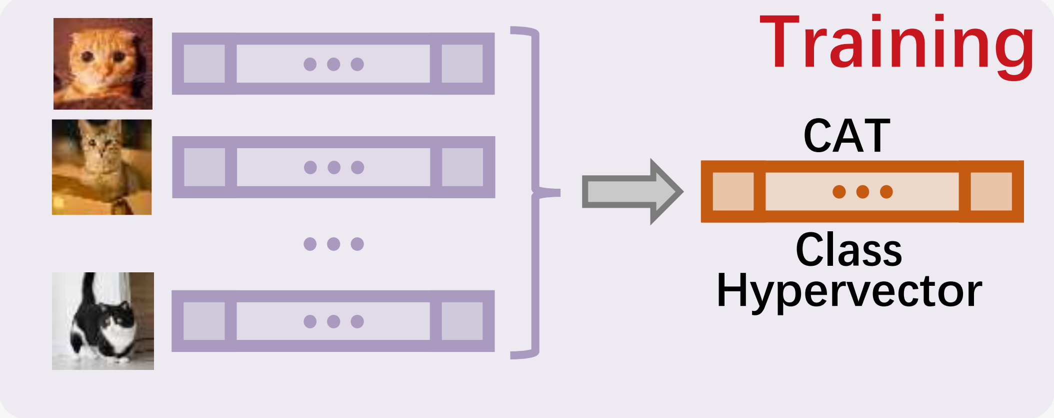
- **Prototype**
- **Light-weight**
- **Efficient**
- **Robustness**

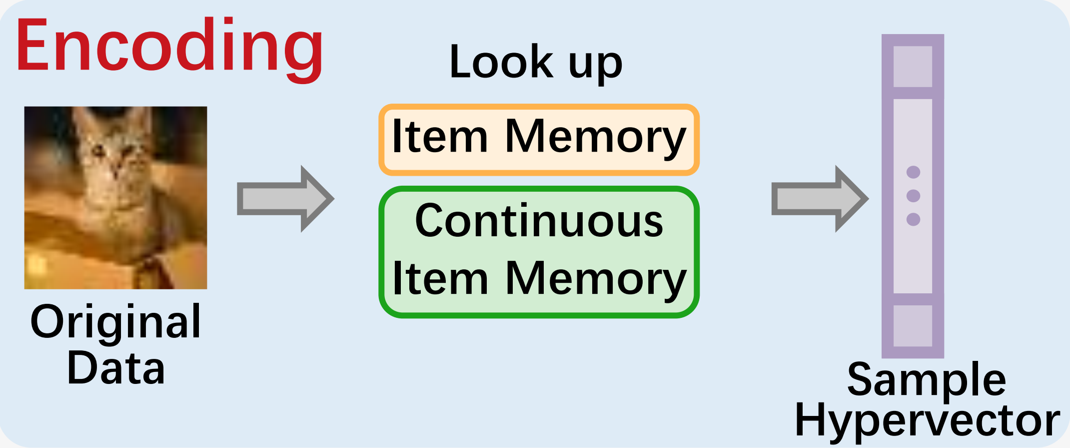
Hyper-Dimensional Computing(HDC)

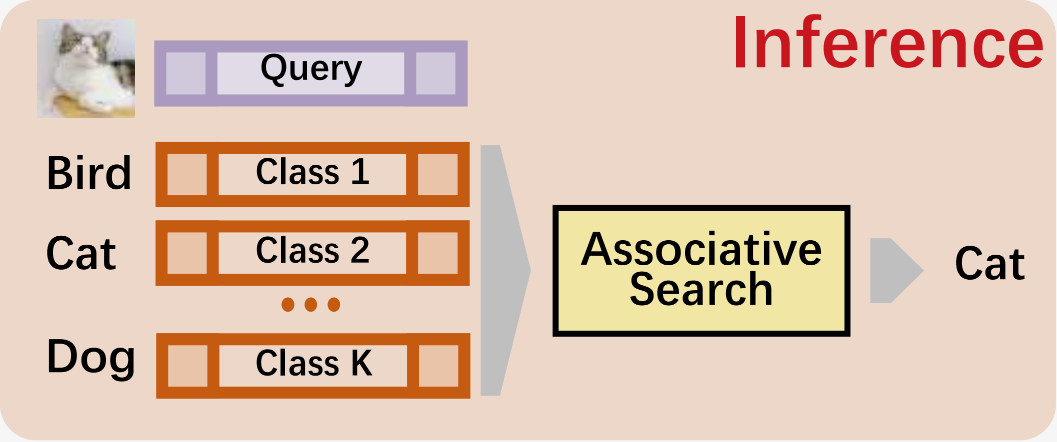
Hyper-Dimensional Space

Pseudo-Orthogonal Property

Bundle $\oplus$	element-wise addition
Bind $\otimes$	element-wise multiplication
Permutation ρ	cyclic right-shift

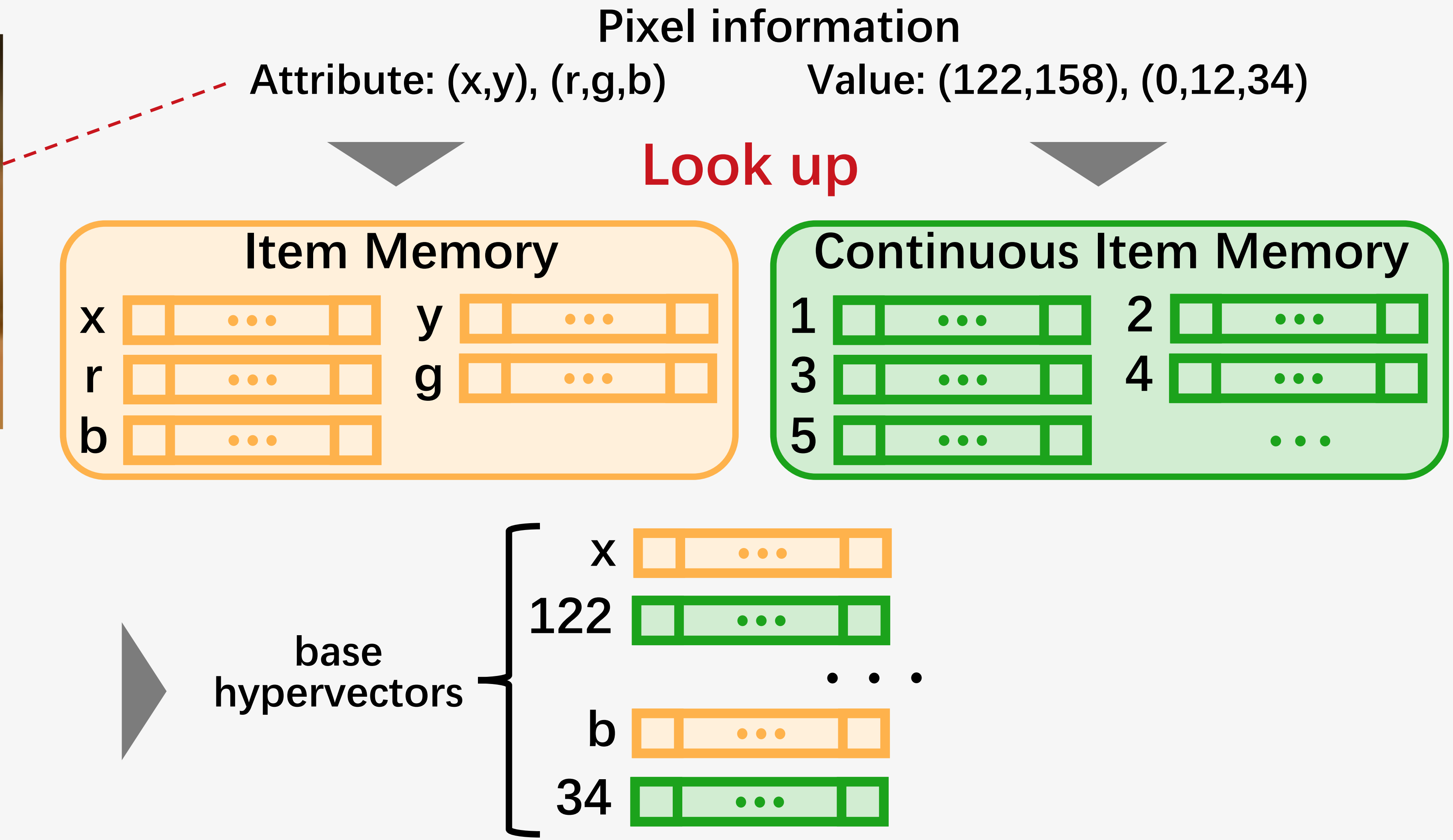




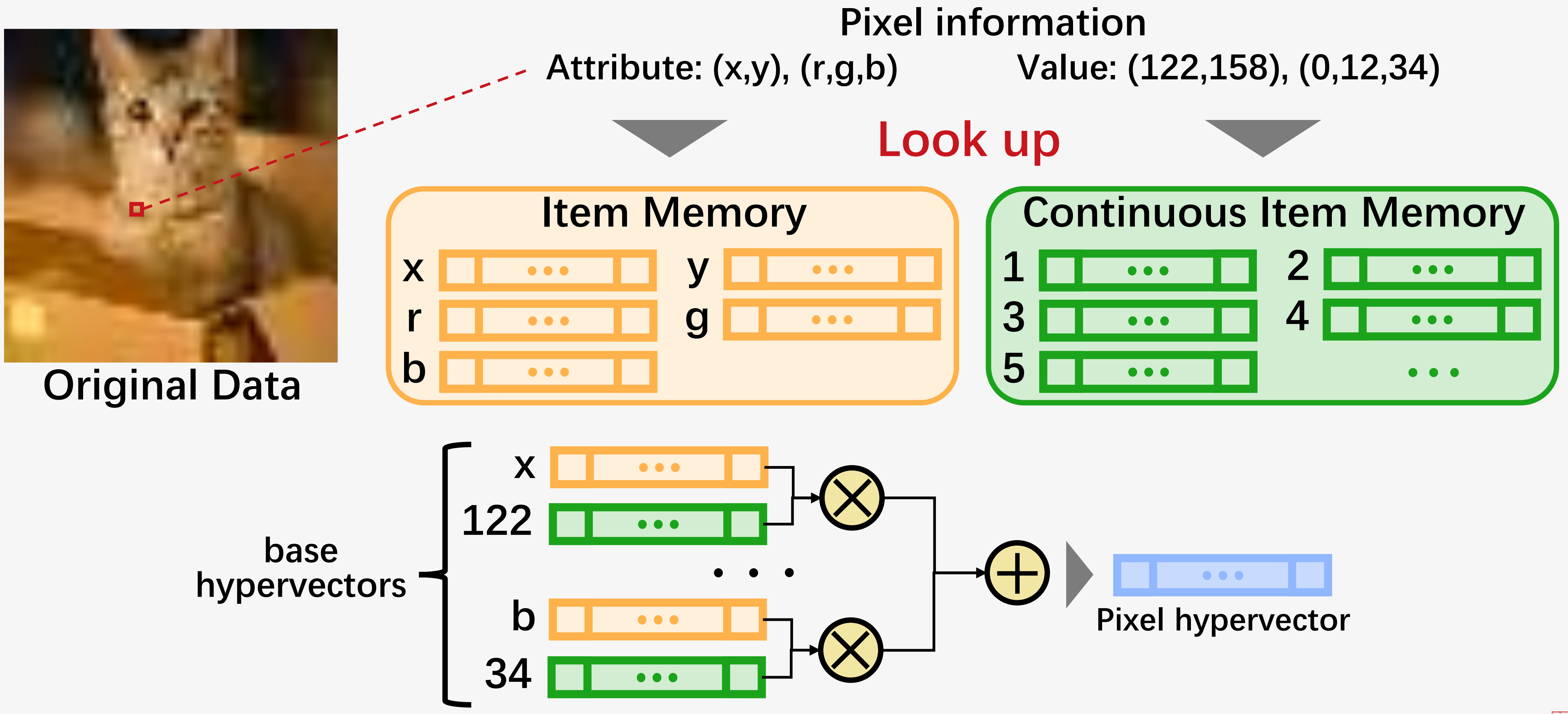


HDC: Encoding

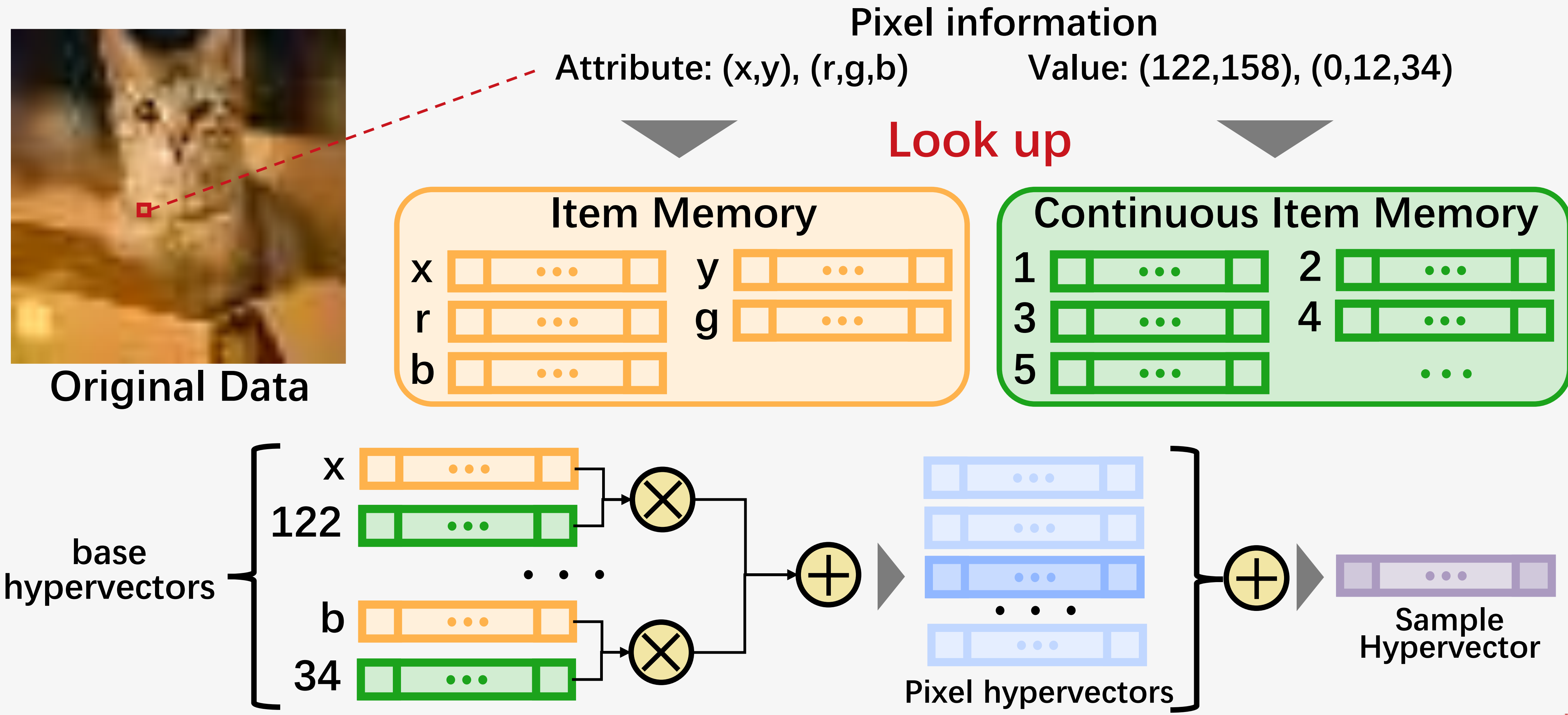
Original Data



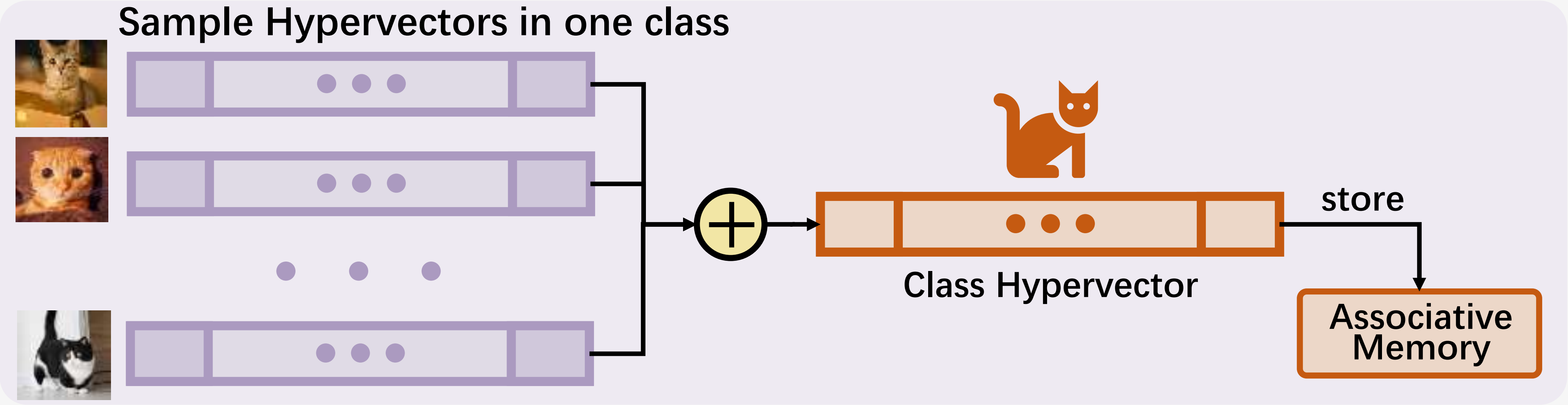
HDC: Encoding



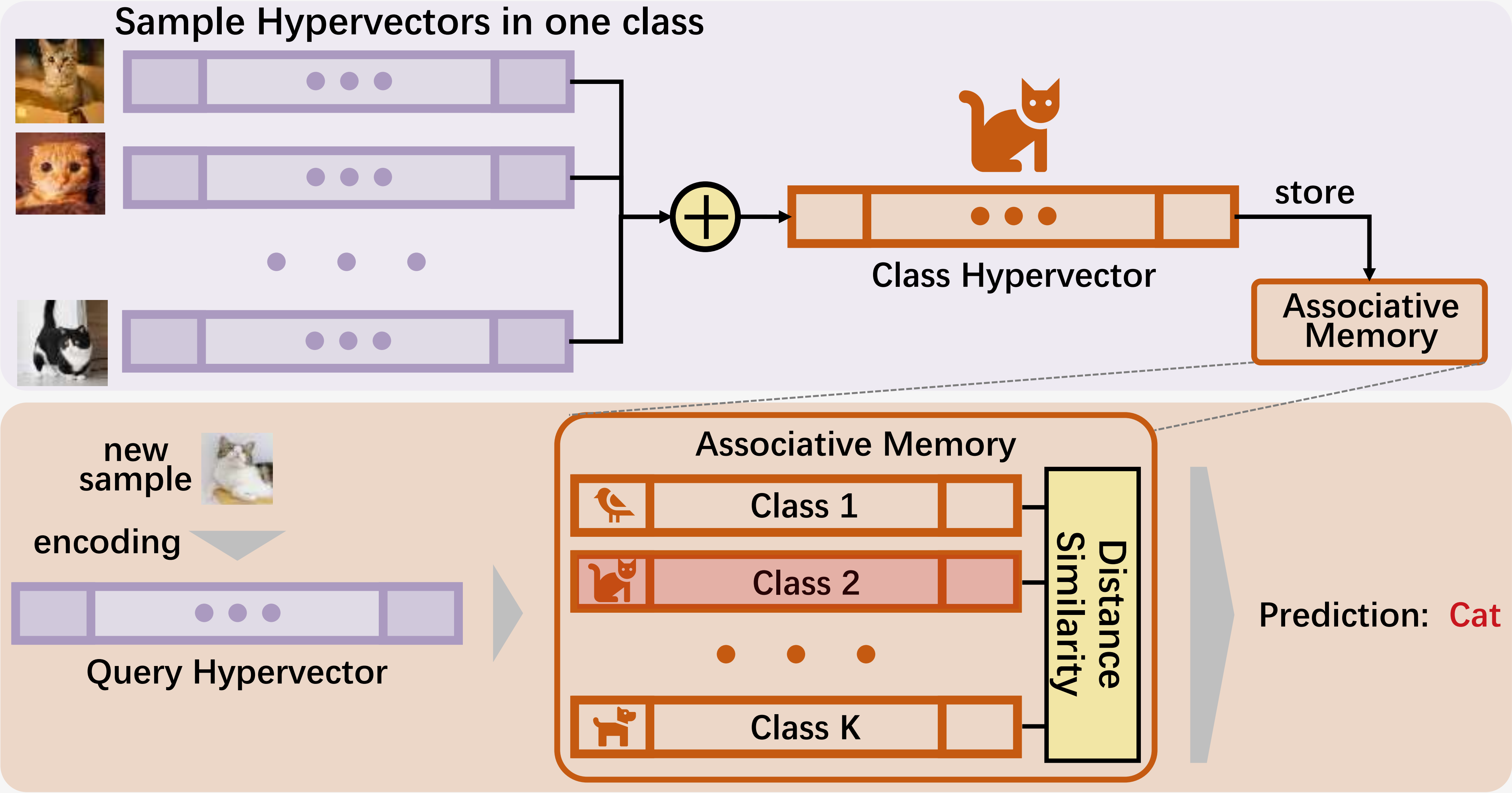
HDC: Encoding



HDC: Training & Inference



HDC: Training & Inference



Challenges

Existing HDC works for FL		Limitation Analysis
FL-HDC[1] • Binary model on device **FHDnn[2]** • Apply NN as feature extractor	▶	**FL-HDC** • Notable **accuracy loss** **FHDnn** • Massive **computation overhead**

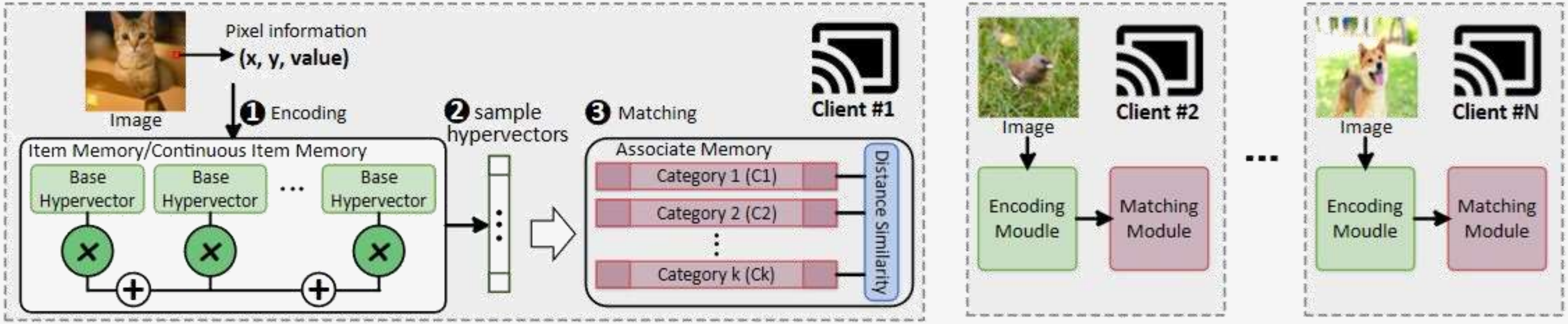
An efficient and accurate HDC framework for FL is still missing

[1] C.-Y . Hsieh *et al.*, “Fl-hdc: Hyperdimensional computing design for the application of federated learning,” in AICAS, 2021.
[2] R. Chandrasekaran *et al.*, “FHDnn: communication efficient and robust federated learning for AIoT networks,” in DAC, 2022.

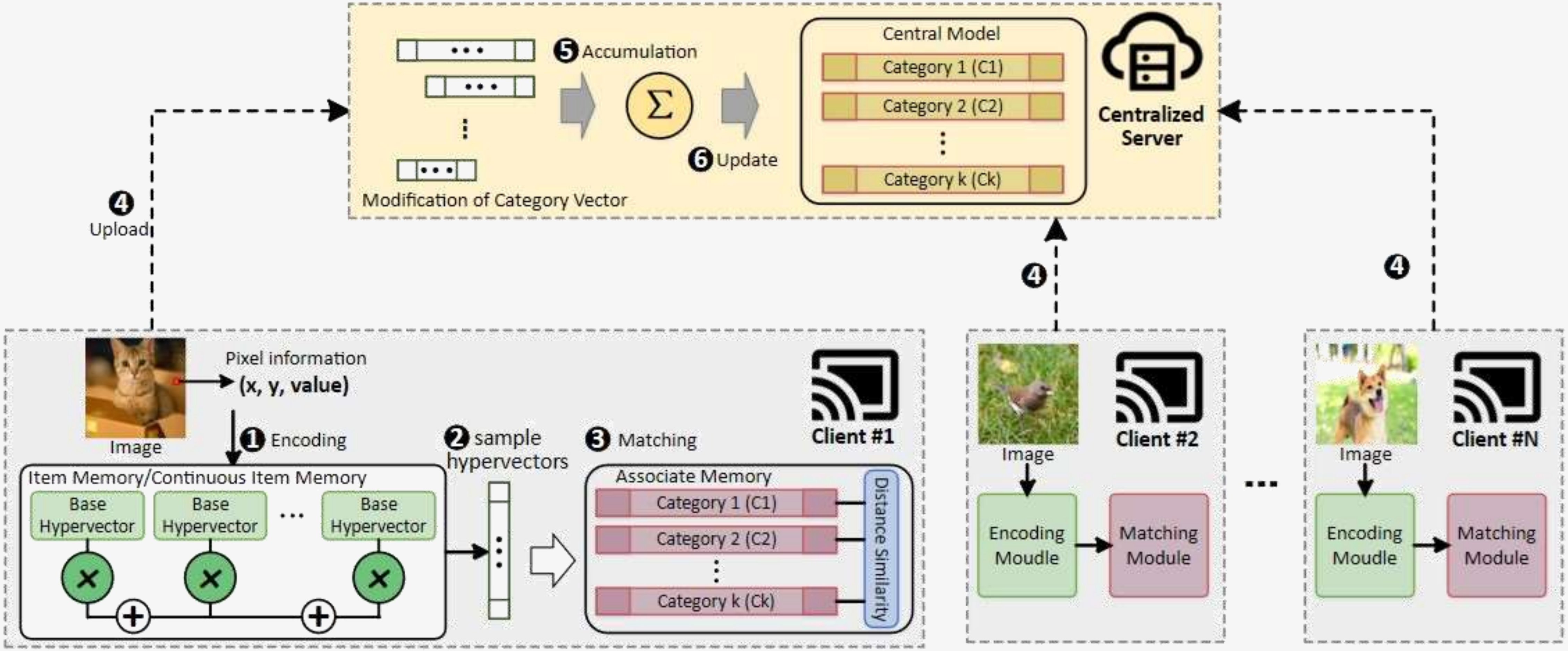
What to consider for designing the framework

- How to design **efficient** training mechanism?
  - **Low Communication**
  - **Little Computation**
- How to get **accurate** models?
  - **Reduce influence from non-iid data**
- How to construct a **general** FL framework?
  - **Support horizontal FL**
  - **Extend to Vertical FL**

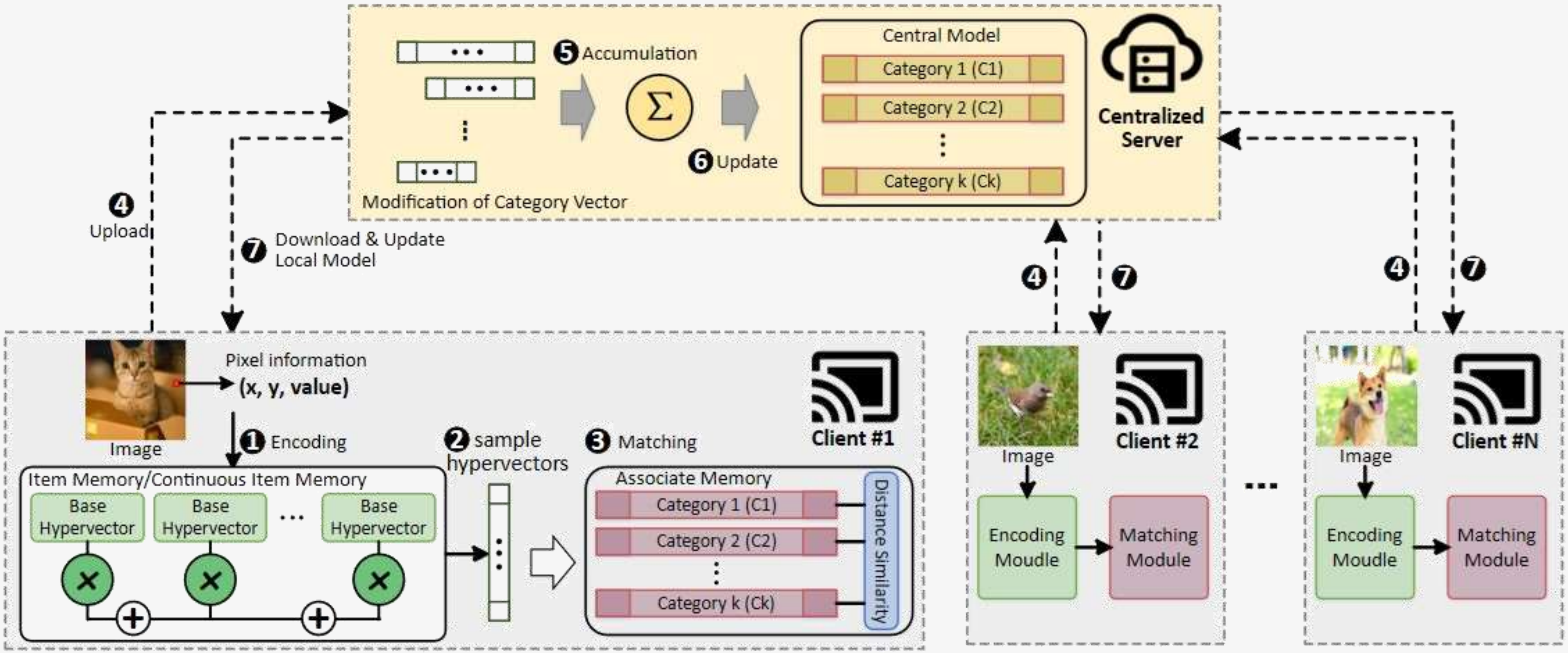
Overview of HyperFeel



Overview of HyperFeel

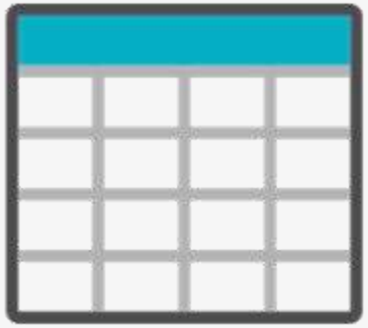


Overview of HyperFeel



HyperFeel Encoding

Tabular Data Encoding

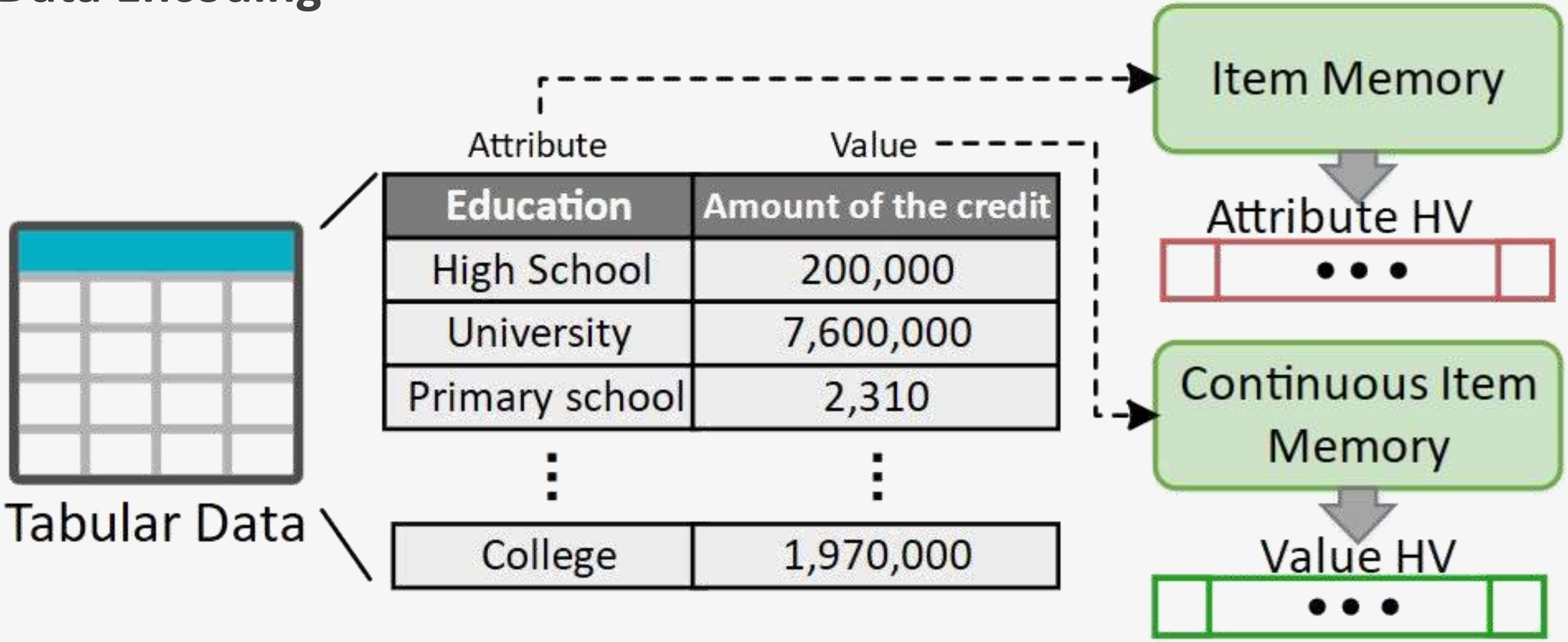


Tabular Data

Attribute	Value
Education	**Amount of the credit**
High School	200,000
University	7,600,000
Primary school	2,310
⋮	⋮
College	1,970,000

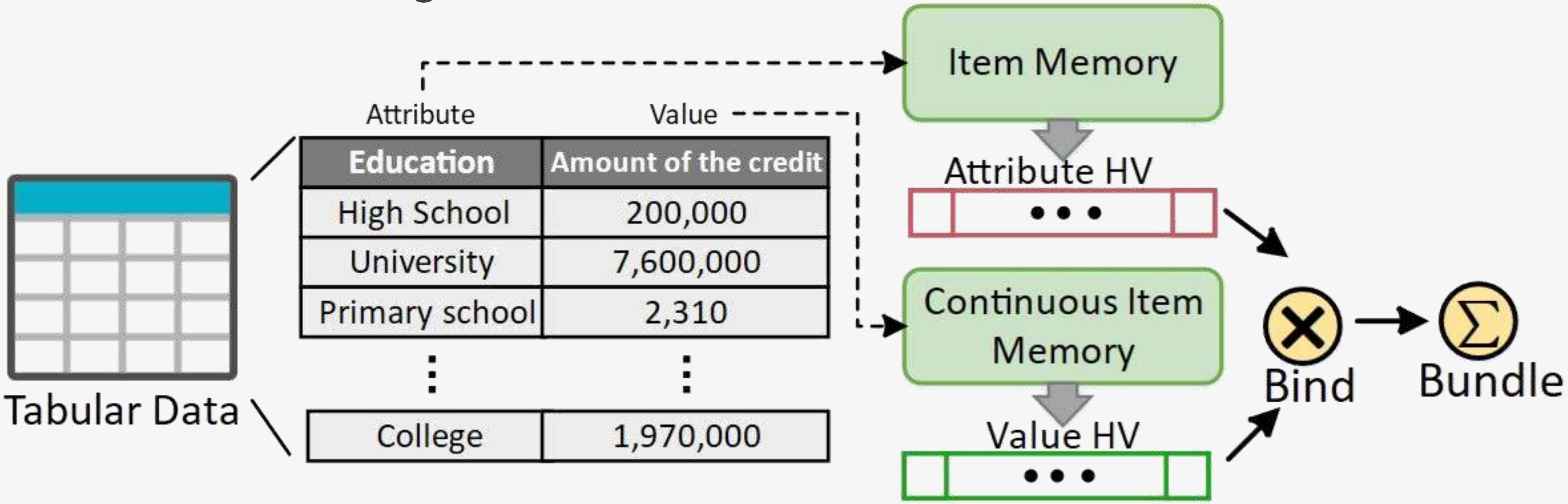
HyperFeel Encoding

Tabular Data Encoding



HyperFeel Encoding

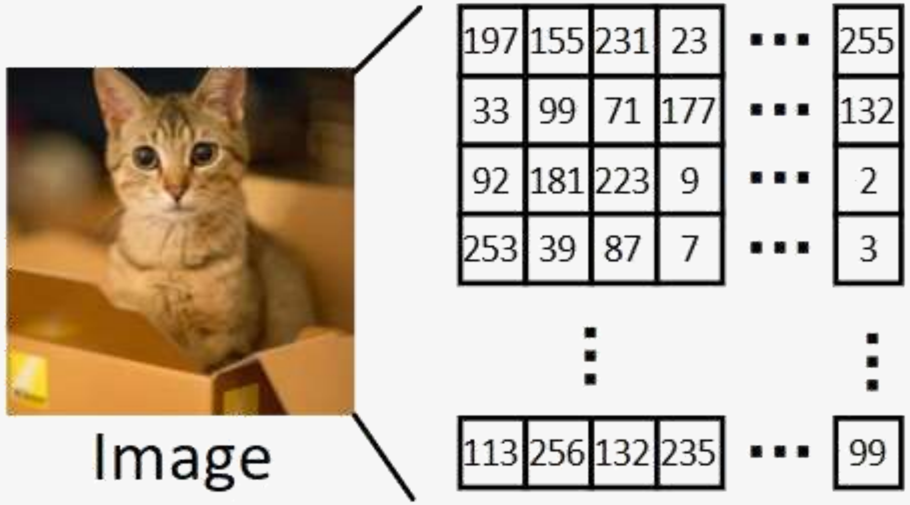
Tabular Data Encoding



HyperFeel Encoding

Image Encoding

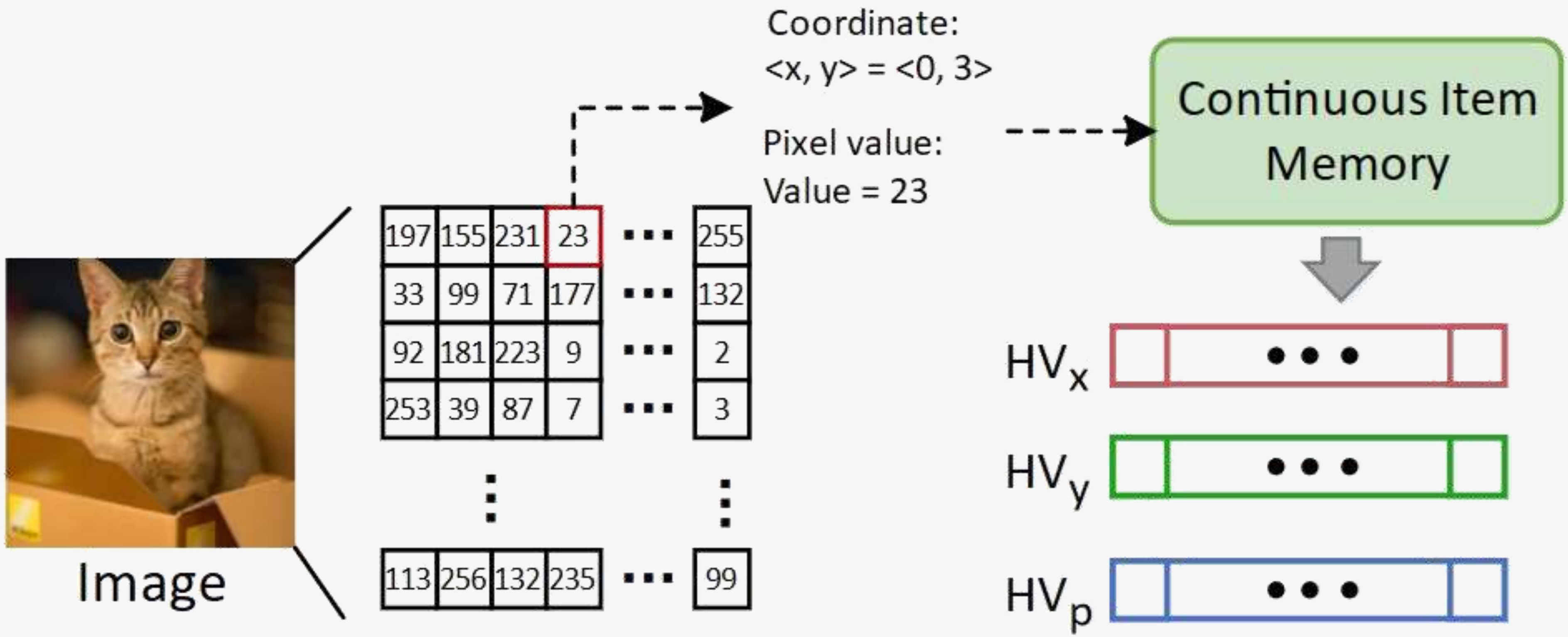
- M pixels: $\vec{F} = \{< f_x^1, f_y^1, f_p^1 >, < f_x^2, f_y^2, f_p^2 >, \cdots, < f_x^M, f_y^M, f_p^M >\}$



HyperFeel Encoding

Image Encoding

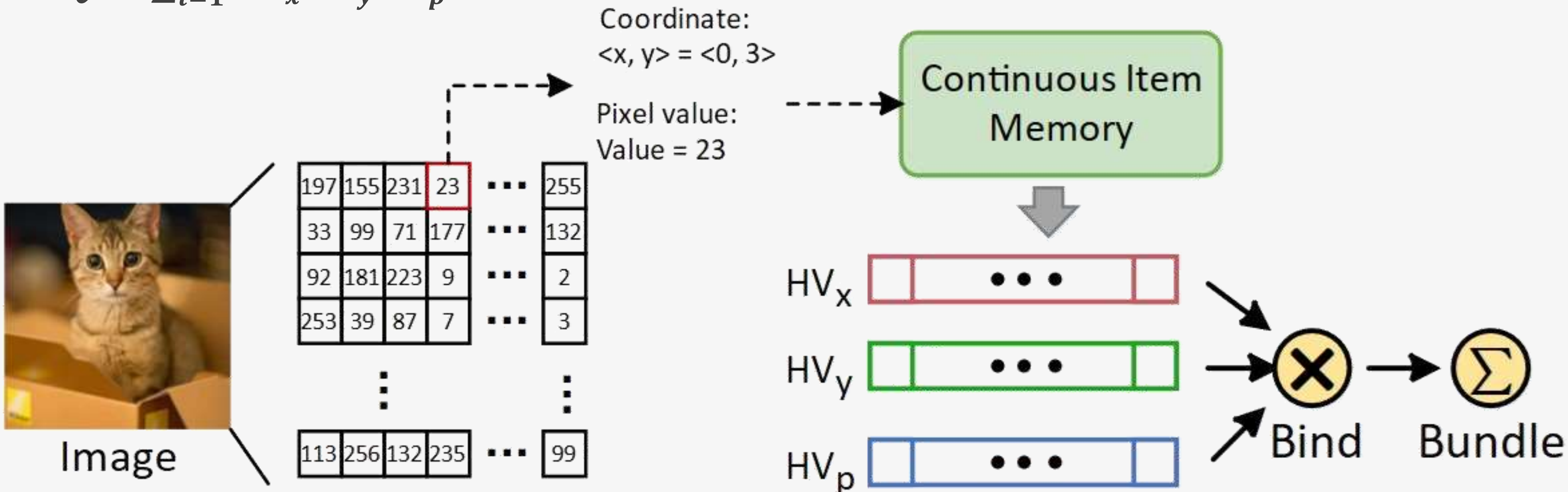
- M **pixels:** $\vec{F} = \{< f_x^1, f_y^1, f_p^1 >, < f_x^2, f_y^2, f_p^2 >, \cdots, < f_x^M, f_y^M, f_p^M >\}$



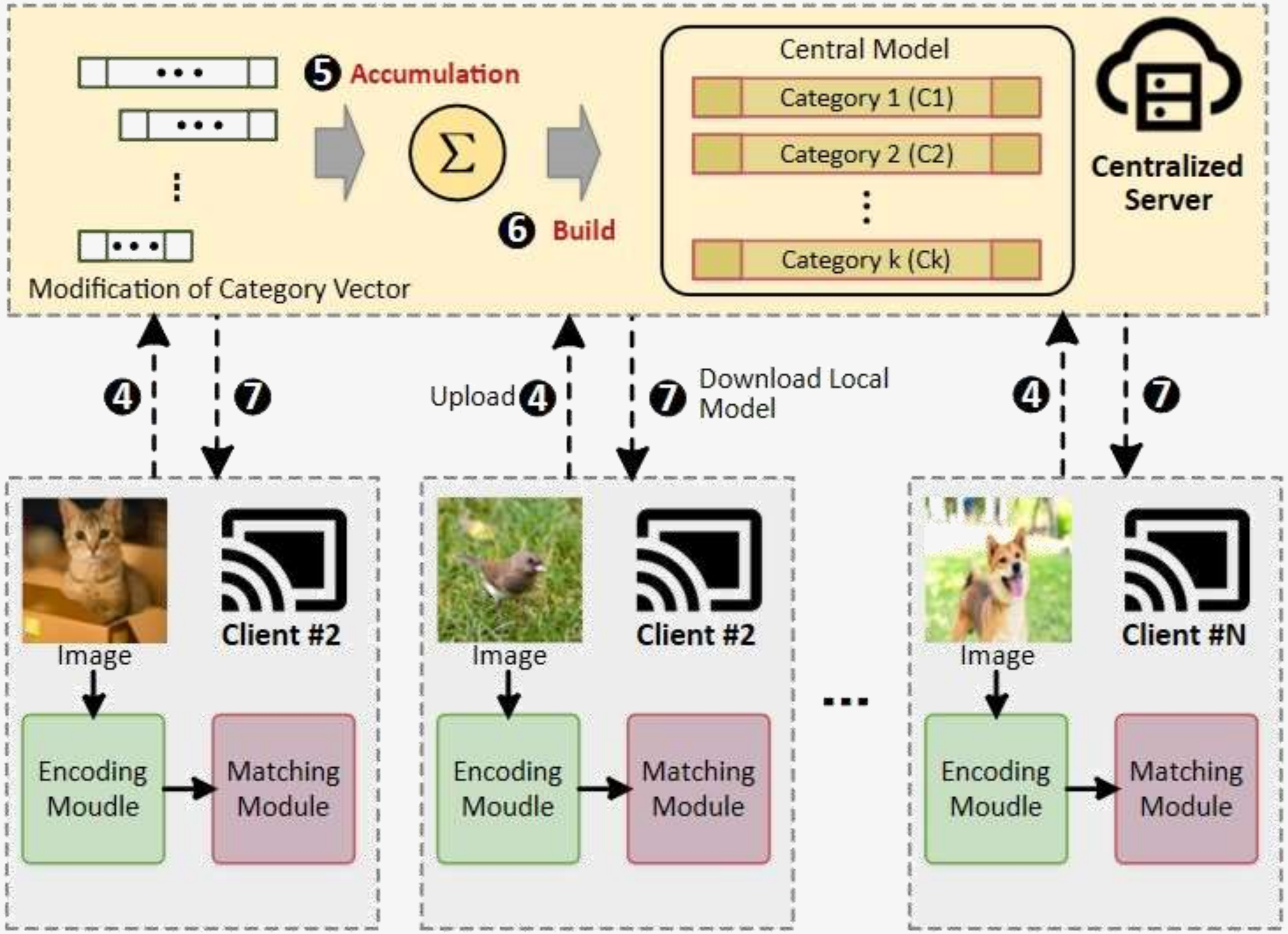
HyperFeel Encoding

Image Encoding

- M **pixels:** $\vec{F} = \{< f_x^1, f_y^1, f_p^1 >, < f_x^2, f_y^2, f_p^2 >, \cdots, < f_x^M, f_y^M, f_p^M >\}$
- $Q = \sum_{i=1}^{M} HV_x^i HV_y^i HV_p^i$



Training

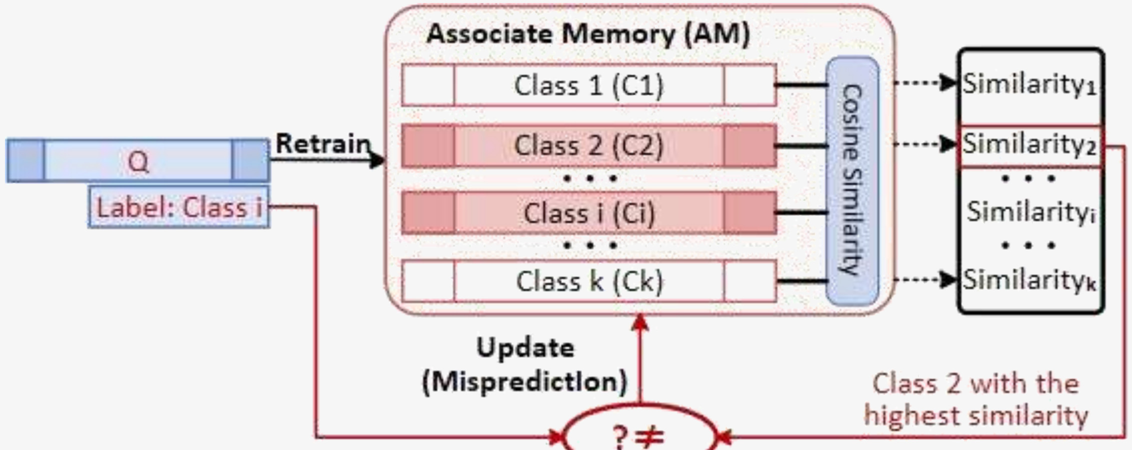


5 Accumulation
6 Build
Modification of Category Vector
Central Model
Category 1 (C1)
Category 2 (C2)
Category k (Ck)
Centralized Server
4
7
Upload 4
7 Download Local Model
Image
Client #2
Client #2
Client #N
Encoding Moudle
Matching Module

Retraining

In Client i

- For encoded local data Q
  - Update local AM
    - $C_{ix} = C_{ix} + lr \cdot Q$; $\quad C_{iy} = C_{iy} - lr \cdot Q$
  - Accumulate modified class vector
    - $\delta_{ix} = \delta_{ix} + lr \cdot Q$; $\quad \delta_{iy} = \delta_{iy} - lr \cdot Q$
- Upload δ_i



Algorithm 1: Retraining Strategy

Data: number of retrain rounds R, number of clients n, number of classes k
Result: models $C_1, \dots, C_n$

for $r \leftarrow 1$ *to* R **do**
  for $i \leftarrow 1$ *to* n ***in parallel*** **do**
    Client$_i$ does:
    $\Delta \leftarrow$ download$_{Server \rightarrow Client_i}(\Delta)$;
    for $j \leftarrow 1$ *to* k **do**
      $C_{ij} \leftarrow$ Personalization Update(C_{ij}, Δ_j);
    end
    for $sample, label_{real} \in local\ data$ **do**
      $Q \leftarrow encode(sample)$;
      $label_{predict} \leftarrow AssociativeSearch(Q, C_i)$;
      if $label_{predict} \neq label_{real}$ **then**
        $C_i \leftarrow$ Accumulation(C_i, Q, $label_{real}$, $label_{predict}$);
        $\delta_i \leftarrow$ Accumulation(δ_i, Q, $label_{real}$, $label_{predict}$);
      end
    end
    upload$_{Client_i \rightarrow Server}(\delta_i)$;
  end
  Server does:
  gather$_{Client_i \rightarrow Server}(\delta_i)$, $i = 1, 2, \dots, n$;
  for $j \leftarrow 1$ *to* k **do**
    $\Delta_j \leftarrow \sum_{i=1}^{N} \delta_{ij}$;
  end
  broadcast$_{Server \rightarrow Client_i}(\Delta)$, $i = 1, 2, \dots, n$;
end

Retraining

In Client i

- For encoded local data Q
  - Update local AM
    - $C_{ix} = C_{ix} + lr \cdot Q$; $C_{iy} = C_{iy} - lr \cdot Q$
  - Accumulate modified class vector
    - $\delta_{ix} = \delta_{ix} + lr \cdot Q$; $\delta_{iy} = \delta_{iy} - lr \cdot Q$
- Upload δ_i

In Server

- Gather δ_i from clients
- Generate Δ
  - $\Delta_j = \sum_{i=1}^{N} \delta_{ij}$, $j = 1, 2, \dots, k$
- Broadcast

Algorithm 1: Retraining Strategy

Data: number of retrain rounds R, number of clients n, number of classes k

Result: models $C_1, \dots, C_n$

for $r \leftarrow 1$ *to* R **do**
  for $i \leftarrow 1$ *to* n ***in parallel*** **do**
    Client$_i$ does:
    $\Delta \leftarrow \text{download}_{Server \to Client_i}(\Delta)$;
    for $j \leftarrow 1$ *to* k **do**
      $C_{ij} \leftarrow$ Personalization Update(C_{ij}, Δ_j);
    end
    for $sample, label_{real} \in local\ data$ **do**
      $Q \leftarrow encode(sample)$;
      $label_{predict} \leftarrow AssociativeSearch(Q, C_i)$;
      if $label_{predict} \neq label_{real}$ **then**
        $C_i \leftarrow$ Accumulation(C_i, Q, $label_{real}$, $label_{predict}$);
        $\delta_i \leftarrow$ Accumulation(δ_i, Q, $label_{real}$, $label_{predict}$);
      end
    end
    $\text{upload}_{Client_i \to Server}(\delta_i)$;
  end
  Server does:
  $\text{gather}_{Client_i \to Server}(\delta_i),\ i = 1, 2, \dots, n$;
  for $j \leftarrow 1$ *to* k **do**
    $\Delta_j \leftarrow \sum_{i=1}^{N} \delta_{ij}$;
  end
  $\text{broadcast}_{Server \to Client_i}(\Delta),\ i = 1, 2, \dots, n$;
end

Retraining

- **In Client i**
  - Download Δ from Server
  - Personalization Update
    - $C_{ij} = C_{ij} + \frac{error_{ij}}{cnt_{ij}} \cdot lr \cdot \Delta_j$
    - cnt_{ij}
      - total number of samples of class j on client i
    - $error_{ij}$
      - number of samples with errors in one-round retraining

Algorithm 1: Retraining Strategy

Data: number of retrain rounds R, number of clients n, number of classes k

Result: models $C_1, \ldots, C_n$

for $r \leftarrow 1$ *to* R **do**
  for $i \leftarrow 1$ *to* n ***in parallel*** **do**
    Client$_i$ does:
    $\Delta \leftarrow$download$_{Server \rightarrow Client_i}(\Delta)$;
    for $j \leftarrow 1$ *to* k **do**
      $C_{ij} \leftarrow$Personalization Update(C_{ij}, Δ_j);
    end
    for $sample, label_{real} \in local\ data$ **do**
      $Q \leftarrow encode(sample)$;
      $label_{predict} \leftarrow AssociativeSearch(Q, C_i)$;
      if $label_{predict} \neq label_{real}$ **then**
        $C_i \leftarrow$Accumulation(C_i, Q, $label_{real}$, $label_{predict}$);
        $\delta_i \leftarrow$Accumulation(δ_i, Q, $label_{real}$, $label_{predict}$);
      end
    end
    upload$_{Client_i \rightarrow Server}(\delta_i)$;
  end
  Server does:
  gather$_{Client_i \rightarrow Server}(\delta_i)$, $i = 1, 2, \ldots, n$;
  for $j \leftarrow 1$ *to* k **do**
    $\Delta_j \leftarrow \sum_{i=1}^{N} \delta_{ij}$;
  end
  broadcast$_{Server \rightarrow Client_i}(\Delta)$, $i = 1, 2, \ldots, n$;
end

Extension to Vertical FL

- Vertical FL
  - Different clients have different attribute sets
- Vertical FL assumption
  - “the server is honest and does not collude with clients, but all the clients are honest but curious to each other”

Extension to Vertical FL

- Vertical FL
  - Different clients have different attribute sets
- Vertical FL assumption
  - “the server is honest and does not collude with clients, but all the clients are honest but curious to each other”
- participating clients create local IMs
- All clients share a CIM for attribute value hypervectors
- Data alignment between clients
  - employ Privacy Set Intersection (PSI) approach

Overhead Analysis

HyperFeel vs. FedAvg

- N: #clients
- M: #parameters (FedAvg)
- C: fraction of clients performing computation in each round
- #dimensions D, #classes K (HyperFeel)

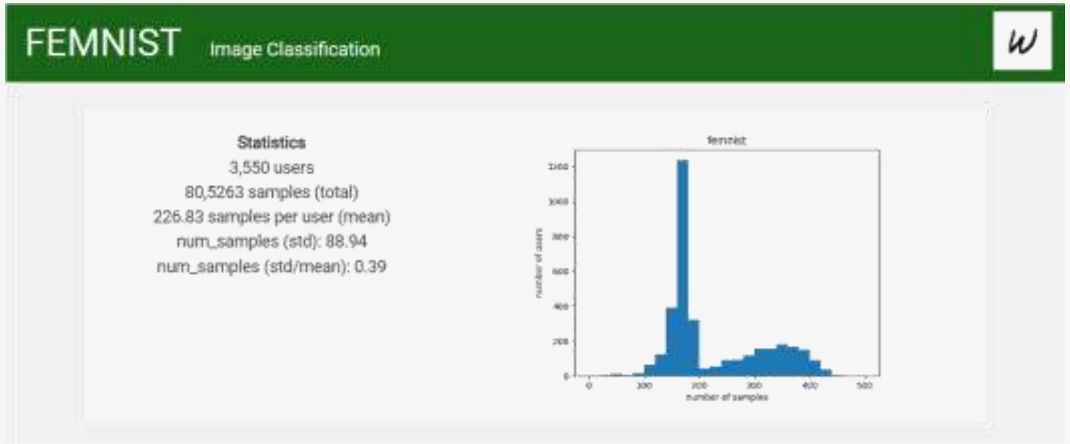
Method	Storage Cost	Comm. Cost(per round)
FedAvg	N×M	C×N×M
HyperFeel	D×N×K	D×N×K

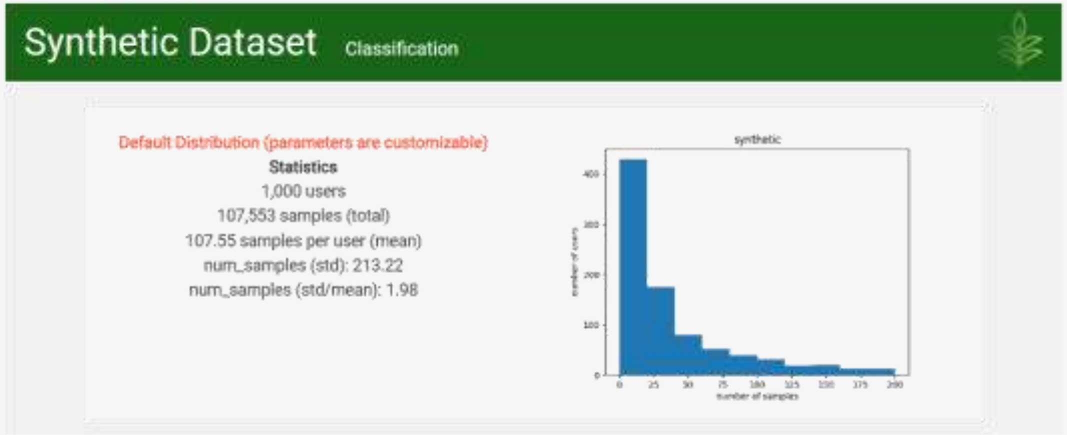
$M >> D \times K$ $\qquad$ $C \times M >> D \times K$

Experimental Setup

Dataset

- Horizonal FL
  - FEMNIST & Synthetic
- Vertical FL
  - credit card customer default probability prediction
- Dataset Process & Partition
  - LEAF [1]





[1] S. Caldas *et al.*, "Leaf: A benchmark for federated settings," arXiv, 2018.

Experimental Setup

Dataset

- FEMNIST, Synthetic, credit card customer default probability prediction
- Process & Partition: LEAF

Baselines

- Horizon FL
  - FedAvg, FedAvg with additional Pipeline, FL-HDC
- Vertical FL
  - KNN, LR, Parsimonious Bayesian, NN

Configurations

- Dimension $D = 1000$
- #Clients $N = 30$

Performance Comparison

Method	Storage Cost	Commun. Cost	Acc.
	FEMNIST DataSet		
FedAvg [5]	6.35MB	5.58GB	74.72%
Additional pipeline [20]	-	-	80.24%
FL-HDC [15]	2.37MB	2.08GB	48.99%
HyperFeel (ours)	**249.63KB**	**70.95MB**	**68.54%**
	Synthetic DataSet		
FedAvg [5]	1.19KB	1.17MB	71.89%
Additional pipeline [20]	-	-	87.34%
FL-HDC [15]	195.31KB	183.11MB	79.69%
HyperFeel (ours)	**27.47KB**	**6.10MB**	**90.13%**

26× Storage Reduction

80.5× Communication Reduction

Performance Comparison

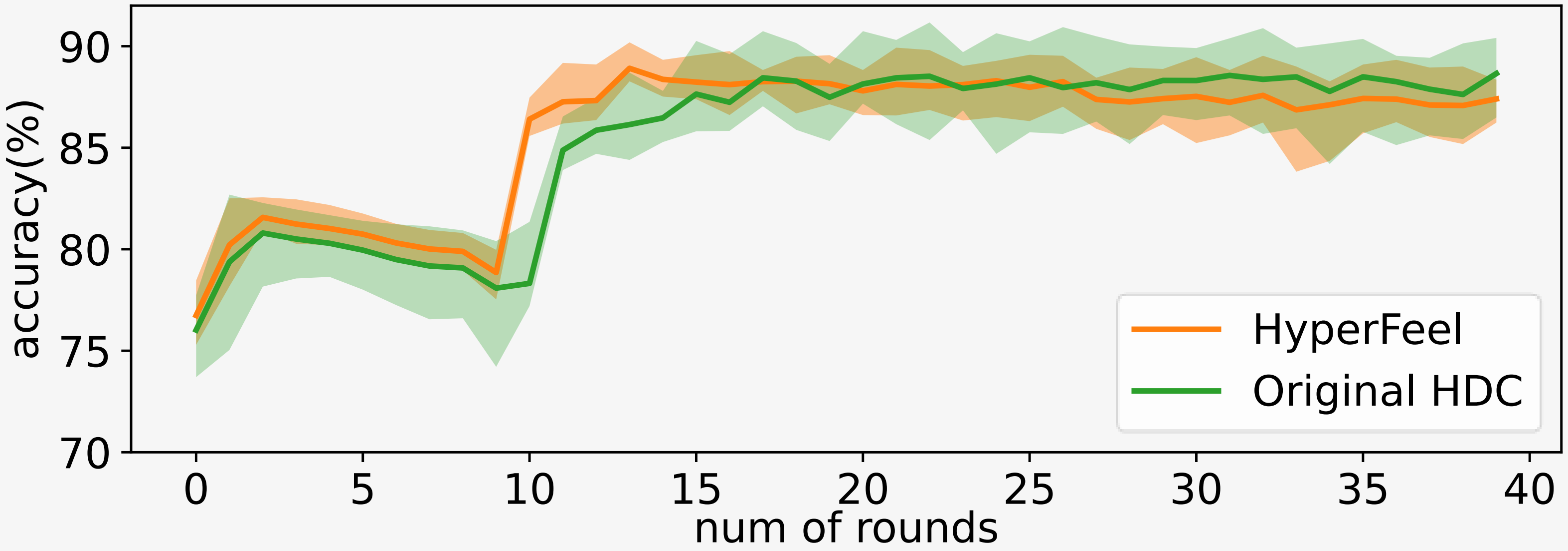
Method	Accuracy	Storage	Communication
KNN	84%	182,169KB	182,169KB
LR	82%	-	-
Bayesian	79%	813KB	813KB
Neural Network	83%	2,412KB	2,412KB
HyperFeel (ours)	**82%**	**430.91KB**	**39.06KB**

A little accuracy loss

113× Storage Reduction

1257× Communication Reduction

Impact of Online Update & Robustness

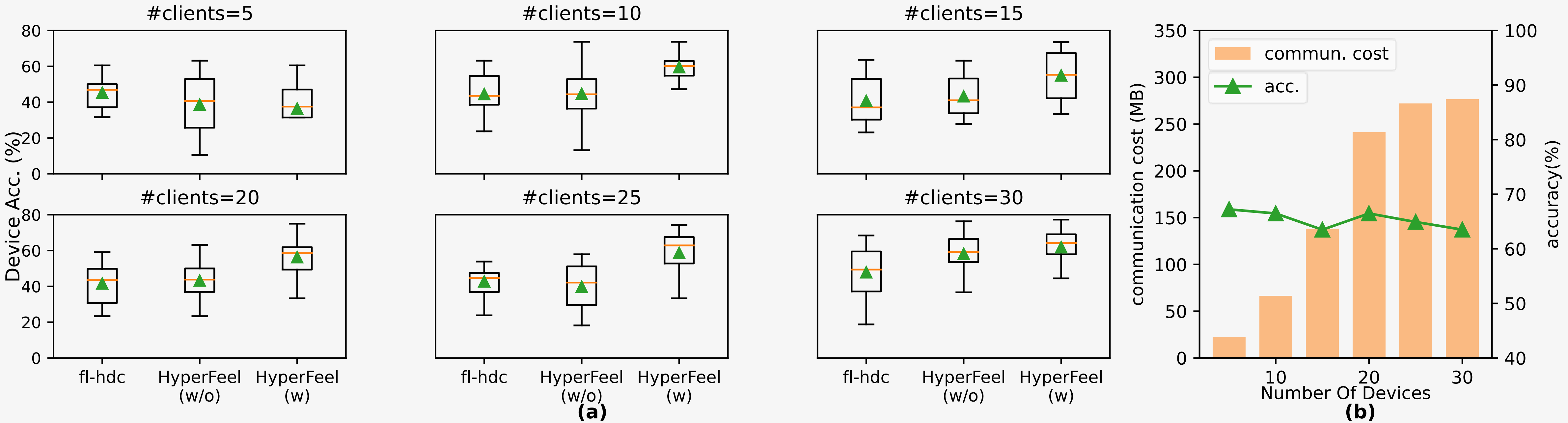


Converge Faster

bit-flip rate	1%	5%	10%	20%	30%
Accuracy	88.37%	88.00%	87.82%	87.80%	86.41%

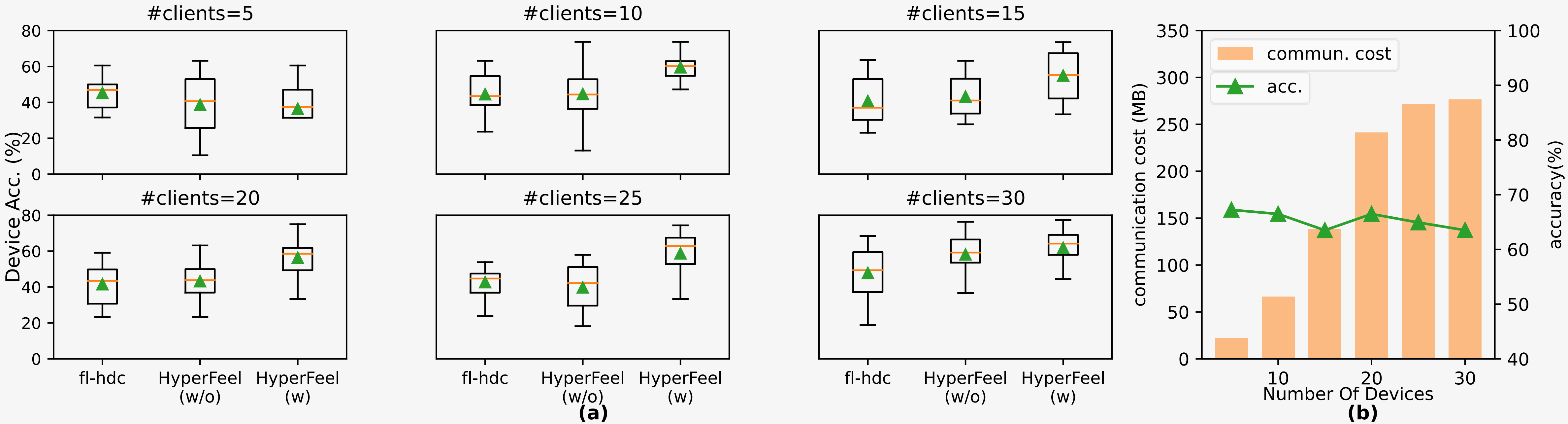
High Robustness

Impact of non-IID data & #clients



FL-HDC ***vs.*** **HyperFeel (without personalization)** ***vs.*** **HyperFeel (with personalization)**

Impact of non-IID data & #clients



Accuracy of HyperFeel is barely affected by varying the number of client.

Average communication per client: 8MB

Average Storage per client: 240KB

Conclusion

- A novel FL framework based on the HDC
- Efficient
  - Reduce storage cost
  - Reduce communication cost
- Accurate
  - Personalization update strategy for non-iid data
- General
  - Fit for both horizon FL and vertical FL

Thank you!

HyperFeel: An Efficient Federated Learning Framework Using Hyperdimensional Computing

Haomin Li (Speaker)
Fangxin Liu, Yichi Chen, and Li Jiang*
Shanghai Jiao Tong University