实验报告

实验结果一览（完整算法流程在下面）

首先是实验参数，采用mnist数据集，共十个客户端，用自己改的CNN模型，遗忘0号与6号客户端，在强non-iid环境下运行，选取SVD 主方向数为6，Fisher 估计样本数为5，擦除强度0.1

python3 main.py --exp_name mnist_small --dataset mnist --total_num_clients 10 --num_training_iterations 40 --forget_clients 0 --model smallcnn --device cuda --num_workers 0 --lr 0.005 --client_data_distribution dirichlet --num_participating_clients -1 --seed 42 --num_local_epochs 1 --baselines fair_vue --fair_rank_k 8 --fair_tau_mode median --fair_fisher_batches 5 --fair_erase_scale 0.10 --fair_vue_debug --skip_retraining

结论是遗忘效果可观，但可能存在些许过度损害全局模型的情况

如下是具体实验数据

这是数据在客户端的分布情况，这里类别具体指的是mnist数据集的0-9十个数字

客户端	类别0	类别1	类别2	类别3	类别4	类别5	类别6	类别7	类别8	类别9	合计
0	0	0	1114	0	4541	144	21	2388	0	0	**8208**
1	0	0	8	27	0	0	71	2034	5476	0	**7616**
2	6	379	250	1705	0	0	13	20	0	3862	**6235**
3	1	0	0	16	188	75	0	1681	0	81	**2042**
4	138	3208	929	0	1112	0	5659	0	0	0	**11046**
5	2	56	0	4314	0	4	29	0	314	4	**4723**
6	7	13	0	6	0	5197	1	141	0	1983	**7348**
7	3187	3083	0	0	0	0	0	0	0	0	**6270**
8	113	0	3	63	1	1	124	0	61	19	**385**
9	2469	3	3654	0	0	0	0	1	0	0	**6127**
类别总样本数	**5923**	**4745**	**5958**	**6131**	**5842**	**5426**	**5918**	**6265**	**5851**	**7949**	**60008**

首先是遗忘客户端6的情况，这里直接贴表格：

阶段	Test Accuracy	忘却客户端6精度
Training	**0.8083 (80.83%)**	76.85%
Retraining	**0.7013 (70.13%)**	21.39%
FAIR-VUE	**0.6548 (65.48%)**	14.07%

类别	Training	Retraining	FAIR-VUE
0	0.9776	0.9602	0.9949
1	0.9912	0.9947	0.9683
2	0.9680	0.9729	0.9864
3	0.6931	0.8218	0.1931
4	0.9053	0.9318	0.9104
5	**0.7444**	**0.2298**	**0.0000**
6	0.9729	0.9697	0.9269
7	0.9844	0.9591	0.9757
8	0.0021	0.0041	0.0000
9	0.7948	0.0694	0.4718

客户端	Training	Retraining	FAIR-VUE
0	93.01	93.26	91.93
1	27.82	28.14	27.39
2	76.26	36.66	44.68
3	96.38	91.14	91.38
4	97.46	97.93	94.33
5	63.43	75.42	22.30
6	**76.85**	**21.39**	**14.07**
7	98.39	97.77	97.34
8	75.06	75.06	68.05

客户端	Training	Retraining	FAIR-VUE
9	96.95	96.75	98.94

接下来是遗忘客户端0的情况，调整了遗忘强度：

阶段	Test Accuracy	忘却客户端0自有数据精度
Training	0.8606（86.06%）	95.46%
Retraining	0.6747（67.47%）	41.37%
FAIR-VUE	0.7540（75.40%）	39.23%

类别	Training	Retraining	FAIR-VUE
0	0.9898	0.9959	0.9612
1	0.9956	0.9965	0.9604
2	0.9777	0.9564	0.9157
3	0.9525	0.8733	0.8545
4	0.9318	0.0000	0.0000
5	0.8520	0.0123	0.9574
6	0.9666	0.9186	0.9739
7	0.9874	0.9562	0.8930
8	0.0000	0.0452	0.0000
9	0.9108	0.8454	0.9861

客户端	Training	Retraining	FAIR-VUE
0	95.46	41.37	39.23
1	27.98	29.82	25.12
2	92.75	83.79	93.87
3	97.75	84.23	80.56
4	97.94	85.76	86.45

客户端	Training	Retraining	FAIR-VUE
5	87.55	78.28	75.61
6	87.30	25.68	96.07
7	99.27	99.63	95.69
8	80.78	77.92	77.66
9	98.07	96.26	91.06

总结：

核心目标达成：忘却成功

FAIR-VUE 成功大幅**抹除了目标客户端的知识贡献**，效果甚至比完整重训练还强，达到了“遗忘”目标，但整体精度下降，且可能影响其他客户端

子空间分解分析

```
[FV-DBG] top singular values=[0.50, 0.37, 0.19, 0.13, 0.10, 0.07, 0.06, 0.04]
[FV-DBG] rho stats: tau=6.42e-02
[FV-DBG] |V_spec|=4, |V_comm|=4
[FV-DBG] ||spec_total||_2=2.036e+00
```

- **奇异值**衰减合理，主方向信息集中；
- **特异子空间**4维，占比 50%，说明目标客户端在 4 个方向上与其他客户端存在明显差异；
- **擦除量** `||spec_total||_2=2.036` ，相比前一次 `1.113` ，**提高了 82%**，说明擦除操作更强、更有效；
- 阈值 τ≈0.064，合理地划分出特异维度。

为什么整体精度下降

> test_acc 从 0.80 降到 0.65，是因为：

1. 特异空间擦除较强，部分共享知识被连带削弱；
2. 未做 fine-tune 纠偏；
3. 没有使用共通空间重新投影（V_comm重构）；
4. 擦除比例 (`erase_scale`) 可能偏高。

FAIR-VUE 算法流程说明

总体流程（入口到评测）

1. **准备数据与模型**
   - 读取数据集，按 `dirichlet` 或 `iid` 划分到各客户端，创建各自 DataLoader；测试集也创建好。
   - 构建初始全局模型 `global_model`（支持 AllCNN / ResNet18 / SmallCNN（这个是自制的模型））。
2. **联邦训练**
   - 运行 `fed_train` 完整训练 `num_training_iterations` 轮，保存每一轮的各客户端权重到 `experiments/<exp_name>/full_training/iteration_t/client_i.pth`。
   - 评测训练后整体性能与各客户端精度。
3. **重训练（baseline）**
   - 从**保留客户端**（剔除要遗忘的客户端）数据再次训练得到 `retrained_global_model`；
   - 打印整体与遗忘客户端精度。
4. **FAIR-VUE（主算法）**
   - 在 `--baselines fair_vue` 打开时执行，详见下文。
5. **其他 baseline（可选）**
   - 代码还保留了 `PGA / FedEraser / FedFIM` 的钩子；
   - 旧的 `Legacy Unlearn` 已通过开关关闭（默认不跑）。

FAIR-VUE：实现细节（按代码执行顺序）

A. 解析快照、构造逐轮增量 Δ

- 逐轮读取 `full_training/iteration_t/client_i.pth`，并构造每轮每个客户端的更新：

$$\Delta_i^{(t)} = w_i^{(t)} - \bar{w}^{(t-1)}$$

  其中 $\bar{w}^{(t-1)}$ 是上一轮所有客户端权重均值。
- 将**目标客户端**的历史增量收集为 `target_deltas_list`，其余客户端拼到 `other_deltas_list`。

> 这样每个增量都相对于“上一轮全局”，方向稳定，便于比较。

B. 只在“参数空间”操作，屏蔽 Buffers

- 取 `param_keys = [name for name, p in fair_model.named_parameters() if p.requires_grad]`
- 后续所有展平/回写只对这些 key 生效；**BN running_mean/var 不改**，避免崩溃。

C. 目标数据上估计 对角 Fisher

- 在目标客户端 DataLoader 上开启求导，计算经验 Fisher 对角矩阵；
- 作为每个参数维度的权重，突出敏感方向，抑制噪声维。

D. Fisher 加权 + 低秩 PCA（SVD）

- 构造加权矩阵 $X_w \in \mathbb{R}^{T \times D_p}$，对每条 Δ 乘 `sqrt(Fisher)`；
- 仅使用参数向量（不含 buffers）拼接；
- 对 X_w 做中心化 SVD，取前 `k=--fair_rank_k` 个右奇异向量：

$$V \in \mathbb{R}^{D_p \times k}$$

即目标客户端跨轮更新的主方向。

E. 方向“特异性”打分 ρ 与阈值 τ

- 对每个基向量 v，计算在其他客户端增量上的平均投影幅度：

$$\rho(v) = \mathrm{mean}_{j \in \mathrm{others}} |\langle \Delta_j, v \rangle|$$

- 选取 `--fair_tau_mode`（`median` / `mean`）为阈值 τ，划分：
  - 特异子空间 $V_{\mathrm{spec}} = \{v | \rho(v) \le \tau\}$
  - 通用子空间 V_{comm}
- 若 V_{spec} 为空，兜底选 1–2 个最小 ρ 方向。

F. 低内存投影（避免 OOM，爆显存）

- 对 V_{spec} QR 分解得正交基 $Q \in \mathbb{R}^{D_p \times r}$；
- 用公式
  - 特异分量：`spec(x) = Q @ (Q.T @ x)`
  - 通用分量：`keep(x) = x - spec(x)`
- 不显式构造大矩阵 $P = I - QQ^\top$，仅使用矩阵乘法。

G. 累计特异分量并擦除

- 初始化 `spec_total = 0`；
- 对每轮目标客户端增量：

$$\mathrm{spec}^{(r)} = Q(Q^\top \Delta^{(r)})$$

并累加；
- 用强度 `erase_scale` 从当前模型参数中扣除：

$$\theta_{\text{new}} = \theta_{\text{now}} - \texttt{erase\_scale} \cdot \text{spec\_total}$$

- 仅回写参数向量，buffers 保留。

H. 评测与调试

- 统一评测整体/客户端/分类别精度；
- 打印遗忘客户端精度；
- 若加 `--fair_vue_debug` ，打印：
  - 轮数、Δ 规模、Fisher 统计；
  - X_w 形状、奇异值；
  - ρ 分布、τ、$|V_{\text{spec}}|$；
  - Q 形状、 `||spec_total||` 等。

关键设计与理由

模块	设计目的
Δ 定义	相对于上一轮全局均值，方向稳定，可比较
Fisher 加权	突出敏感维度，抑制噪声
特异/通用分离	通过其他客户端投影度量共享性，自适应阈值
仅操作参数空间	避免改坏 BN 统计量
低秩实现	用 $Q(Q^\top \cdot)$ 代替 $P\cdot$，节省显存
只擦除目标特异分量	避免误伤通用方向
可调强度	通过 `--fair_erase_scale` 控制影响范围

可调参数

参数	作用
`--fair_rank_k`	SVD 主方向数
`--fair_tau_mode`	阈值模式 `median/mean`

参数	作用
`--fair_fisher_batches`	Fisher 估计样本数
`--fair_erase_scale`	擦除强度 (0.2~0.6 推荐)
`--fair_vue_debug`	打印详细中间量

一句话总结

> FAIR-VUE 通过 Fisher 加权 PCA 找出目标客户端独有的更新方向，仅在参数空间低秩擦除这些方向上的累计分量，不重训练即可有效遗忘目标知识，同时尽量保持其他客户端与总体性能。