HETEROSWITCH: CHARACTERIZING AND TAMING SYSTEM-INDUCED DATA HETEROGENEITY IN FEDERATED LEARNING

Gyudong Kim [1] **Mehdi Ghasemi** [2] **Soroush Heidari** [2] **Seungryong Kim** [1]
Young Geun Kim [1] **Sarma Vrudhula** [2] **Carole-Jean Wu** [3]

ABSTRACT

Federated Learning (FL) is a practical approach to train deep learning models collaboratively across user-end devices, protecting user privacy by retaining raw data on-device. In FL, participating user-end devices are highly fragmented in terms of hardware and software configurations. Such fragmentation introduces a new type of data heterogeneity in FL, namely *system-induced data heterogeneity*, as each device generates distinct data depending on its hardware and software configurations. In this paper, we first characterize the impact of system-induced data heterogeneity on FL model performance. We collect a dataset using heterogeneous devices with variations across vendors and performance tiers. By using this dataset, we demonstrate that *system-induced data heterogeneity* negatively impacts accuracy, and deteriorates fairness and domain generalization problems in FL. To address these challenges, we propose HeteroSwitch, which adaptively adopts generalization techniques (i.e., ISP transformation and SWAD) depending on the level of bias caused by varying HW and SW configurations. In our evaluation with a realistic FL dataset (FLAIR), HeteroSwitch reduces the variance of averaged precision by 6.3% across device types.

1 INTRODUCTION

Federated learning (FL) enables mobile devices to collaboratively train a shared machine learning (ML) model while keeping all the raw data on device (McMahan et al., 2017; Kairouz et al., 2021). As only the model gradients are shared with the cloud servers for updating the shared global model, FL has been considered as a practical way to prevent the privacy leakage (Kairouz et al., 2021; Huba et al., 2022). Although FL has gained much attention in various applications, such as computer vision (Li et al., 2022), voice recognition (Guliani et al., 2021), health monitoring (Brisimi et al., 2018), and recommender system (Hejazinia et al., 2022), the following key challenges make FL deployment less practical: high degree of system and data heterogeneity causing unstable and degraded performance of the global model (Kim & Wu, 2021; Li et al., 2020).

Data heterogeneity: In FL, the size and distributions of training data can be significantly heterogeneous depending on the geographical locations, cultural backgrounds, personal habits, and device usage patterns of participating users (Kairouz et al., 2021). For example, in a handwriting recognition scenario, users who write the same words exhibit differences in stroke width and slant. Such data heterogeneity can lead to imbalanced and biased model updates (Wu & Wang, 2022), eventually degrading the overall model performance (Zhao et al., 2018) or resulting in disparate accuracy across devices (Li et al., 2019). Thus, many prior works have tried to mitigate the data heterogeneity by adopting a regularization method to local models (Li et al., 2020), sharing a small amount of public data across local clients (Zhao et al., 2018; Mansour et al., 2020), or employing weighted averaging for model aggregation (Li et al., 2019).

System-induced data heterogeneity: In FL, the hardware of client devices can also be highly diverse: there are more than ten thousand devices with different SoCs, sensors, and software systems in the market (Wu et al., 2019; Kim & Wu, 2020). Such system heterogeneity introduces a new type of data heterogeneity, namely *system-induced data heterogeneity*, in FL. For example, in case of vision tasks, heterogeneous sensor hardware can produce noticeably different training data (i.e., images) even for the same scene or object, due to the respective attributes such as focal length, aperture, and other variables (see Figure 1). The difference across the training data can get even more severe due to the fragmented image signal processing (ISP) algorithms employed onto the devices. This new type of data heterogeneity eventually contributes to bias across the local models, re-

[1]Korea University. [2]Arizona State University. [3]Meta (work done while at ASU). Correspondence to: Young Geun Kim <younggeun_kim@korea.ac.kr>.

Proceedings of the 5^{th} MLSys Conference, Santa Clara, CA, USA, 2024.

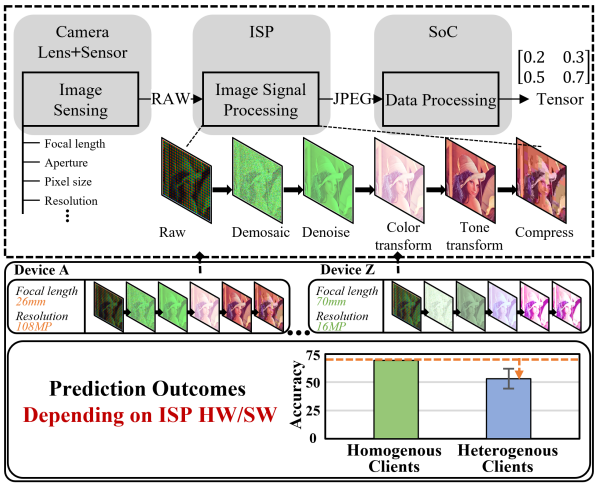

Figure 1. End-to-end ISP pipeline for a vision DNN. Devices generate *heterogeneous data* due to the *SW and HW variations*, degrading model performance significantly.The accuracy of Homogeneous Client is obtained when FL devices are the same, whereas Heterogeneous Client is obtained with different device types.

sulting in significant model performance degradation (by 23.5%, on average, as shown in Fig. 1).

Despite the significant impact, system-induced data heterogeneity has been overlooked in FL research. Most prior works have primarily focused on label distribution skew, where a non-IID dataset is formed by partitioning a flat existing dataset based on the labels (Li et al., 2020; Zhao et al., 2018; Mansour et al., 2020; Li et al., 2019; McMahan et al., 2017; Karimireddy et al., 2020). Hence, they cannot consider the characteristics of cross-client heterogeneity which should be inherently included in the real-world FL datasets significantly degrading the model performance.

Furthermore, such system-induced data heterogeneity deteriorates two notable problems in FL: fairness and domain generalization. In FL, fairness involves ensuring that the global-model fairly represents the learning from all devices and is not biased towards any particular group of devices (Pessach & Shmueli, 2022; Maeng et al., 2022). Domain generalization (DG), meanwhile, involves designing machine learning models that can generalize well to new, unseen domains (Wang et al., 2022). In the context of FL, the domain can be a particular type of devices, each displaying unique characteristics due to system-induced data heterogeneity.

In this paper, we first characterize the impact of system-induced data heterogeneity for FL. We collect a vision dataset using a collection of devices of heterogeneous hardware[1]. By using the collected dataset, we analyze the variations and inconsistencies in the global model that arise due to system-induced data heterogeneity originated from hardware (i.e., sensors) and software (i.e., ISP algorithms). We also investigate fairness and domain generalization implications which can stem from system-induced data heterogeneity in realistic FL environments. Based on the characterization results, we propose HeteroSwitch — a selective generalization method that combines the ISP transformation and SWAD — which counteracts the effects of system-induced data heterogeneity in FL. Compared with the baseline, HeteroSwitch reduces the variance of accuracy across device types by 79.5%, while improving the worst out-of-distribution (OOD) accuracy by 5.8%.

Key Contributions: We summarize the main contributions of this work as follows:

- We create a new dataset that takes into account client devices of heterogeneous hardware to independently identify the impact of system-induced data heterogeneity. We attempt to discern the most influential factors contributing to system-induced data heterogeneity in FL models during the data generation process, and examine the impact of *system-induced heterogeneity* on the FL model.
- We investigate the fairness and domain generalization implications which can be deteriorated by system-induced data heterogeneity.
- Based on the analysis, we propose HeteroSwitch. By switching the use of generalization techniques in an FL environment where each device possesses heterogeneous data, HeteroSwitch effectively mitigates the model performance degradation caused by system-induced data heterogeneity.

2 Background

2.1 Federated Learning

Federated Learning (FL) provides a privacy-preserving alternative to conventional machine learning, allowing for collaborative model training across various devices while keeping the data locally stored (McMahan et al., 2017). Given N local client devices, a server first initializes a global deep learning model and its global parameters by specifying the number of local training epochs E, the local training mini-batch size B, and the number of participant devices K. (B, E, K) is determined by FL-based services (McMahan et al., 2017). In each training round, the server selects K devices from the total N devices. The global model is then broadcast to the selected devices. Each selected device locally trains the model using its data samples with the batch size B over E epochs. Once training is completed, the devices send the model gradients back to the server. The server averages these gradients to update the global model. This process is repeated until the desired accuracy is achieved.

[1]The dataset is available at `https://github.com/CASL-KU/HeteroSwitch`.

2.2 ISP Pipeline

In FL, images collected by each client are used for local-model training. Each device produces different images depending on a wide range of hardware and software. Fig. 1 depicts the process from image capturing to the formation of a tensor to be used by the training of a DNN or for inference. (1) At the beginning, the image sensor records the light signal as RAW data based on its properties like focal length, aperture, pixel size, and resolution. (2) After that, a series of image signal processing (ISP) stages (i.e., from Demosaic to Compress in Fig. 1) are applied to the RAW data to produce a human visible image (Buckler et al., 2017; Hansen et al., 2021). The ISP stages include:

- **Demosaicing** converts RAW data into a color image.
- **Denoising** removes noise from the image.
- **Color transformation** (i.e., White Balance adjustments) corrects the colors in the image to make them natural (Wyszecki & Stiles, 2000).
- **Color transformation** (Gamut mapping) converts the colors of the image to a standard gamut.
- **Tone transformation** adjusts the brightness and contrast of the image.
- **Image compression** reduces file size while attempting to maintain image quality.

(3) Finally, the images are transformed to a tensor to be used by model training or inference.

In FL, the heterogeneous combination of sensors and ISP algorithms from the large collection of client devices cause system-induced data heterogeneity.

3 System-induced Data Heterogeneity in FL

This section characterizes the impact of system-induced data heterogeneity on FL. We create a novel image dataset by collecting images using a collection of representative devices of different system hardware characteristics (Section 3.1). By using the dataset, we analyze how system-induced data heterogeneity introduces bias on the global model in FL (Section 3.2). We also attempt to discern the most influential factors contributing to system-induced data heterogeneity during the data generation process (Section 3.3 and 3.4).

3.1 Dataset Creation

To isolate and examine the impact of system-induced data heterogeneity on FL, we create a custom dataset by using a total of nine smartphones (Table 1). We select three smartphones from each of the three different vendors — Samsung, LG, and Google, representing high-end (H), mid-end (M), and low-end (L) categories with at least two-year gaps between their release dates (Kim & Wu, 2020; 2021; Wu et al., 2019) — by introducing variations across vendors and performance tiers, we emulate a realistic device composition for FL and assess how device heterogeneity contributes to system-induced data heterogeneity.

Table 1. Mobile devices used in dataset creation, categorized by performance tiers and vendors. (N%) indicates the market share of respective device. H, M, and L represents high-end, mid-end, and low-end devices, respectively.

Level	Vendor		
	Samsung	LG	Google
H	GalaxyS22 (12%)	VELVET (2%)	Pixel5 (1%)
M	GalaxyS9 (27%)	G7 (5%)	Pixel2 (3%)
L	GalaxyS6 (38%)	G4 (8%)	Nexus5X (4%)

Each device is fixed with tripod, and used to capture images displayed on a monitor in a dark room — to isolate the impact of system-induced data heterogeneity and to prevent a potential introduction of the other types of feature distribution skew, we controlled other external factors (e.g., lighting, position, and object being photographed) that may affect the captured images (Cidon et al., 2021; Zhang et al., 2016). For the images, we use 12 non-overlapping ImageNet (Deng et al., 2009) classes from 12 higher-level categories for the images displayed: Chihuahua, Altar, Cock, Abaya, Ambulance, Loggerhead, Timber Wolf, Tiger Beetle, Accordion, French Loaf, Barber Chair, and Orangutan. This dataset is compact, yet sufficiently challenging to learn with a higher number of labels and larger image sizes compared to datasets used in previous FL works (LeCun, 1998; Krizhevsky et al., 2009; Cohen et al., 2017; Scheuerman et al., 2021). To separate the impact of hardware (HW) and software (SW) differences, we collected both RAW data (i.e. unprocessed and uncompressed data directly obtained from the image sensor without employing the ISP algorithms) and images processed by the default camera application of each device.

3.2 Data Heterogeneity across Device type

In FL, system-induced data heterogeneity can create significant bias across different device types. Table 2 shows quality degradation of the global model across various device types, compared to when the model is tested on the same device type it was trained on. The quality of the model is measured by its accuracy on each deployed device. Each row represents the device type used by the clients for training the global model, while each column shows the device type used for testing the trained global model. The Mean Others of each device refers to the average model quality degradation on all devices except the device itself — the degradation indicates how easily the model is affected by *system-induced heterogeneity* alone. The highest accuracy

Table 2. Model quality degradation in model when deployed to various device types, compared to the training device type.

Train on	Test on (Model Quality Degradation)									Mean Others
	Pixel5	Pixel2	Nexus5X	VELVET	G7	G4	S22	S9	S6	
Pixel5	-	5.7%	21.3%	10.0%	20.0%	21.2%	22.4%	12.9%	24.6%	17.3%
Pixel2	1.0%	-	11.4%	5.2%	11.8%	14.1%	19.0%	10.5%	15.3%	11.0%
Nexus5X	28.1%	19.6%	-	24.9%	6.7%	33.9%	43.0%	20.2%	19.8%	24.5%
VELVET	9.4%	10.8%	15.6%	-	11.0%	11.9%	20.6%	4.3%	18.7%	12.8%
G7	32.7%	21.3%	12.5%	17.7%	-	19.1%	42.7%	11.6%	17.9%	21.9%
G4	14.0%	17.3%	16.7%	13.5%	15.8%	-	26.8%	12.0%	13.6%	16.2%
S22	25.8%	21.4%	25.9%	18.4%	20.5%	26.4%	-	21.9%	35.8%	24.5%
S9	29.6%	25.5%	14.7%	10.6%	11.1%	29.4%	50.7%	-	16.4%	23.5%
S6	29.2%	24.9%	15.5%	24.7%	9.8%	17.2%	43.7%	17.2%	-	22.8%
Mean Others	21.2%	18.3%	16.7%	15.6%	13.3%	21.7%	33.6%	13.8%	20.3%	19.4%

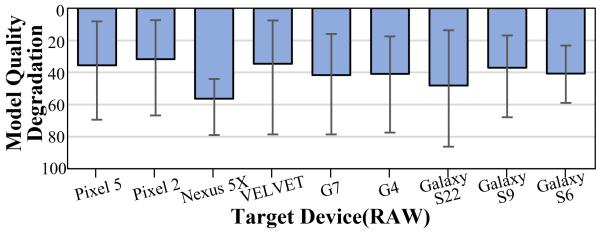

Figure 2. Model quality degradation in model when deployed to various device types using RAW data.

is always achieved when the model is tested on the same device type it was trained on, and there is an observable drop (1.0% $\sim$ 50.7%) in accuracy when the model is tested on the others.

In Table 2, the model quality degradation depends on the device type. For example, when the model trained on a G7 is tested on Pixel5, the accuracy degrades by 32.7% — the accuracy degrades by 20.0% for the reverse case. This is because the two devices are using significantly different hardware (i.e., camera sensors) and software (i.e., ISP algorithms). On the other hand, Pixel 5 and Pixel 2, which have the smallest differences in HW and SW, show the least model quality degradation when tested on each other (5.7% and 1.0%, respectively). These results demonstrate the unique feature of the *system-induced heterogeneity* which depends on the heterogeneous sensor hardware and ISP algorithms.

To better understand the system-induced data heterogeneity effect, we further investigate its two primary sources in the next subsection: 1) HW (i.e., lens and sensor) variations and 2) SW (i.e., ISP algorithm) variations.

3.3 Deeper Look at Heterogeneity: HW

In this subsection, we further investigate the impact of HW variations on the data and the trained model performance. To focus on the impact of HW, especially the type of sensors used, we exclude the impact of SW by training the model with RAW data (i.e. unprocessed and uncompressed data directly obtained from the image sensor without employing the ISP algorithms).

The image sensor heterogeneity is a significant source of the data discrepancy. Fig. 2 represents the model quality degradation on each target device, when the model is trained with RAW data of the other devices. The x-axis represents the target device, while the y-axis represents the model quality degradation and the error bars indicate the minimum and maximum deviations across the trained devices. Compared to the results obtained using post-processed images (i.e., the last row in Table 2), the degradation is more significant, averaging between 31.74% and 56.41%, when we use RAW data. This result implies that this severe heterogeneity among RAW data files should be carefully handled in FL.

3.4 Deeper Look at Heterogeneity: SW

ISP algorithms are applied to RAW data to produce human visible images (Hansen et al., 2021). To quantify the contributions of the ISP algorithm variations on system-induced data heterogeneity, we divide the ISP process into six primary stages, from demosaicing to compression, creating distinct images at each stage (Buckler et al., 2017). Table 3 lists the algorithms used at each stage. The impact of each stage on model accuracy is assessed by either omitting a specific stage or applying another algorithm for each stage[2] — we train the global model using data processed by the Baseline column in Table 3, and test the model while adopting either Option 1 or Option 2 for each stage.

Although the ISP algorithms are known to reduce the HW variations (Ebner, 2007; Morovič, 2008), variations in certain ISP methods still introduce unmanageable heterogeneity in images that have a detrimental impact on model performance.Fig. 3 depicts the model quality degradation due to

[2]In case of demosaicing, which is a prerequisite for subsequent stages, we use two different methods rather than omitting the stage.

Table 3. ISP algorithms applied to each stage. '-' means we omit this stage.

ISP stage	Baseline	Option 1	Option 2
Denoising	FBDD (Gozdz, 2010)	-	Wavelet-bayesShrink (Chipman et al., 1997)
Demosaicing	PPG (Lin, 2003)	Pixel binning (Cannistra)	AHD (Hirakawa & Parks, 2005)
Color transformation	Gray world (Ebner, 2007)	-	White patch (Ebner, 2007)
Gamut mapping	srgb	-	Prophoto
Tone transformation	srgb gamma correction (Stokes, 1996)	-	srgb gamma correction +tone Equalization
Image compression	JPEG (Quality=85) (Cidon et al., 2021)	-	JPEG(Quality=50) (Cidon et al., 2021)

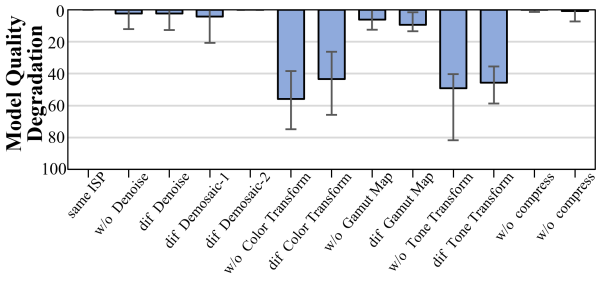

Figure 3. Model quality degradation in model when tested with adjustment of ISP algorithms at every stage.

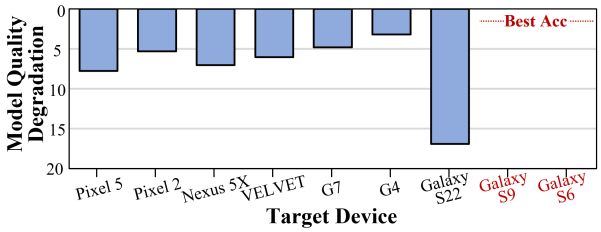

Figure 4. Model quality degradation over the highest accuracy achieved by dominant devices, Galaxy S9 & S6, showing the bias in the global model towards dominant devices. The participant devices used for training followed the ratio specified in Table 1.

the various ISP algorithms. The x-axis represents the stage omitted (or modified) from the baseline. As shown in Fig. 3, the model quality degrades when we omit (or modify) each stage of the ISP algorithms. Notably, omitting or modifying the color and tone transformation algorithms result in significant declines in model quality, regardless of the device type — the accuracy is degraded by 56.0% and 49.2% when we exclude Color (specifically, White Balancing) and Tone transformation from ISP stages, respectively, which is even more severe compared to that we observe in RAW data. This result implies that each stage of ISP algorithms, especially the color and tone transformation stages, significantly contributes to system-induced data heterogeneity and thus there is a demand for new solutions to tackle ISP heterogeneity across the devices in FL.

4 Fairness and Domain Generalization Issues

In a real-world FL environment, a global model is trained with a number of client devices. Considering our observations in Section 3.2, the diverse types of client devices can contribute differently to the global model due to their unique data characteristics stemming from HW (Section 3.3) and SW variations (Section 3.4). As a result, certain characteristics affected by system heterogeneity can make the global model learn a bias towards those characteristics. This potential issue can be interpreted through the lens of two well-established fields in machine learning: fairness and domain generalization.

4.1 Fairness

In FL, the number of client devices is not always same for all device types. This uneven distribution makes the fairness problems[3] in FL (Mohri et al., 2019; Maeng et al., 2022). For example, if a majority of training data comes from a particular group of devices with the similar HW and SW configurations, the global model may become more tailored towards those devices, degrading accuracy for the rest of the devices. To quantify the fairness problem in a realistic manner, we allocate device types of clients according to the vendor market share (StatCounter, 2022) (summarized in Table 1) and system performance distribution (Wu et al., 2019). Note, in our analysis, we refer the device types with the highest percentage of participation as the dominant devices (i.e., Galaxy S9 and S6 in Table 1) — these can be seen as a privileged group in the context of fairness, benefiting from biases in the global model (Pessach & Shmueli, 2022). To assess the fairness of FL across various device types, we compute the model quality degradation on each remaining device compared to the accuracy on the dominant devices.

[3]In the realm of FL, fairness ensures the global model performs adequately across all participating devices, rather than being skewed towards a certain type or group of devices.

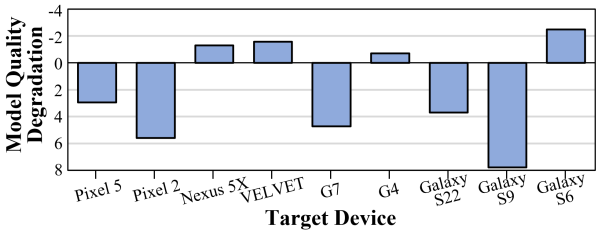

Figure 5. Model quality degradation when a device type is excluded from training, illustrating the complex relationship of DG and accuracy with various device types in FL.

Fig. 4 shows the model quality degradation for each device compared to the dominant devices — the x-axis represents the device type the model is deployed. As shown in Fig. 4, the accuracy on the 7 less dominant devices is 3.2% to 16.9% lower than that on the dominant devices. This implies that the global model has a bias toward the HW specification and SW algorithms of the dominant devices. On the other hand, although Galaxy S22 is the third most used for training, deploying the model to it yields the lowest accuracy among the devices. This also implies that a higher participation rate does not always guarantee a high accuracy, meaning that there exist the system features, such as the advanced ISP algorithms, which can smooth out (or deteriorate) the bias.

4.2 Domain Generalization

In the context of FL, deploying the model to a new, unseen device type is closely related to the concept of domain generalization (DG) [4]. This concept involves designing ML models that can effectively generalize to new, unseen domains (Wang et al., 2022), such as unique device types. In FL, domains can be characterized by system-induced data heterogeneity, where each device exhibits distinct features. To emulate the DG problem in FL, we consider each device as an unseen domain — note we analyze the ability of the model to generalize across different device types by training the global model with all other devices and subsequently testing it on the selected unseen device.

Fig. 5 shows the model quality degradation of global model accuracy when the device is excluded from training, compared to when all device types equally participate in FL (this practical is widely used in DG (Dou et al., 2019; Segu et al., 2023)). The x-axis represents the device type that was excluded from training, while the y-axis shows the model quality degradation on the excluded (i.e., unseen) device. In DG, it is typically expected that a domain excluded from training would show lower accuracy, since the global model lacks exposure to its specific characteristics (Dou et al., 2019; Segu et al., 2023). Interestingly, excluding a device from training does not consistently affect accuracy as they generate different details for samples, such as color, texture, and contextual details (Tommasi et al., 2017). For example, the accuracy for S9 drops when it is not part of the training. Conversely, older devices like S6, Nexus5X and G4, which have lower resolutions and simpler ISP algorithms, commonly exhibit increased accuracy, even though they were excluded from training. This inconsistent result demonstrates system-induced data heterogeneity can deteriorate the complexity of DG in FL.

Our observations highlight the importance of considering system-induced data heterogeneity and its potential impact on fairness and DG in FL. There is a need for a careful strategy that ensures stable performance of global model for various device types.

5 Proposed Design: HeteroSwitch

As we observe in Section 4, system-induced data heterogeneity causes significant accuracy degradation of the deployed DNNs on the typical device types. To mitigate the problem, we propose HeteroSwitch — a selective generalization technique. Fig. 6 shows the overview of HeteroSwitch. As shown in Fig. 6, HerteroSwitch incrementally applies generalization techniques to the clients that have biased data due to system-induced data heterogeneity. For each round, HeteroSwitch measures the amount of bias in the data of participating clients, by comparing their initial loss L_{init} with the Exponential Moving Average (EMA) of the aggregated loss L_{EMA}.(Section 5.1). Based on the measured bias, HeteroSwitch determines whether it applies the generalization techniques to the client data or not (Section 5.2). The algorithm detail is explained in Algorithm 1.

5.1 Bias Measurement

In the context of FL, every client has a unique dataset — *each data differently contributes to the global model*. Moreover, The dynamic nature of FL causes the datasets used for training not to remain constant every round (McMahan et al., 2017). With this constraint, applying a "one-size-fits-all" approach may not yield the best results. For example, applying the same level of generalization technique to all clients could negatively affect the learning process for those using less common device types, leading to unnecessary performance degradation for them. To mitigate these effects, HeteroSwitch selectively employs the generalization techniques to the data of participating clients based on the training loss. We use the Exponential Moving Average (EMA) loss from previous communication rounds or the validation loss as the criteria for switching the generalization techniques on the client side.

[4]Given that more than 500 new smartphones are released every year (Arena Com, Ltd., 2022), it is common to see unseen devices for FL-based services.

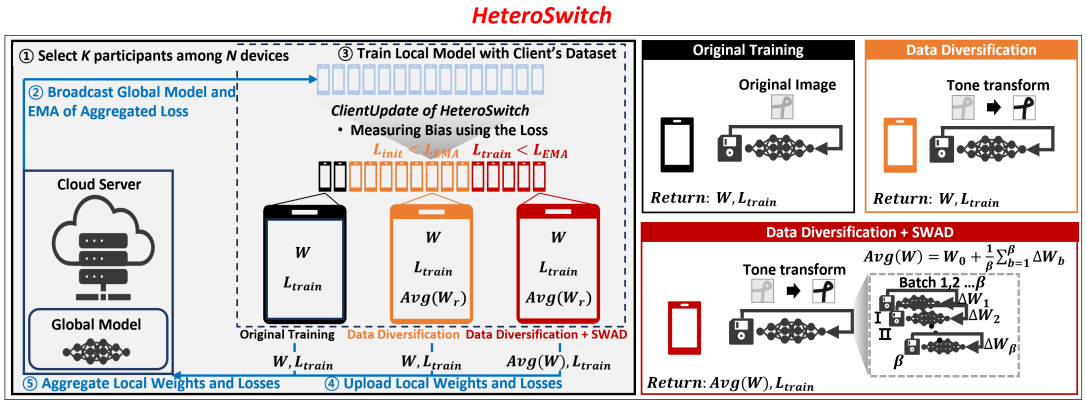

Figure 6. Overall Design of HeteroSwitch

EMA of Previous Train Loss L_{EMA}:

$$L_{EMA,t+1} \leftarrow \alpha * L_{cur} + (1-\alpha) * L_{EMA,t} \quad (1)$$

If the test loss on dataset of a client before updating the local model is lower than the aggregated training loss calculated during the previous round, the dataset is considered to have a potential bias. This indicates that the characteristics of the dataset are already learned by the global model.

5.2 Generalization Techniques

Fairness and DG share similar objectives, as discussed in (Creager et al., 2021). To solve the fairness problem, it is important to elevate the accuracy of devices that underperform. On the other hand, the domain generalization problem seeks to improve the accuracy for unseen devices. In essence, if consistent accuracy is achieved across all devices, regardless of their presence during training, it may be sufficient to address both fairness and DG problems. To achieve this, we propose a two-pronged method focusing on the dataset diversification and the model generalization. The method is used on the client-side during the training, adapting to system-induced data heterogeneity.

Dataset Diversification with ISP Adjustment: Given heterogeneity across the data from different device types, we propose to expand the diversity of the data via *random transformation* during training — such transformation, including geometric, color adjustments, and random erasing, does not change the inherent label or meaning of the image, encouraging the model to learn from more varied range of data (Shorten & Khoshgoftaar, 2019).

Motivated by our observations in Section 3.4, which highlighted the need for addressing Tone and Color (especially White Balance (Wyszecki & Stiles, 2000)) variations, we temporarily adopt random White Balance and tone transformations (via random gamma application (Wyszecki & Stiles, 2000)) to the dataset of each client during local training.

Random WB:

$$\text{With } r_1, r_2, r_3 \sim U(1-degree, 1+degree),$$
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_{out} = \begin{bmatrix} r_1 & 0 & 0 \\ 0 & r_2 & 0 \\ 0 & 0 & r_3 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{in} \quad (2)$$

Random Gamma:

$$\text{With } \gamma \sim U(1-degree, 1+degree),$$
$$\text{Img}_{out} = \text{Img}_{in}^{\gamma} \quad (3)$$

However, data diversification does not necessarily need to be applied to every data. Given distinct ISP methods employed by each device, some data may pass through ISP stages that are not learned by the global model. To make the global model learn diverse characteristics of the ISP stages, we do not employ the data diversification in such a case.

Model Generalization through SWAD: Even with data diversification that accounts for ISP adjustments, biases may still arise to the global model due to extreme differences between devices. We employ a weight averaging method (Izmailov et al., 2018; Cha et al., 2021) for further generalization. Fig. 7[5] shows the comparative robustness of two different weight averaging methods, Stochastic Weight Averaging Densely (SWAD) (Cha et al., 2021) and conventional SWA (Izmailov et al., 2018), for three training scenarios: applying only data random transformation, applying SWAD with transformation, and applying conventional

[5]For this experiment, we use the original 12-class ImageNet dataset that we utilized for dataset creation (Section 3.1). In each training scenario, we train the model for 10 epochs, applying a random data transformation at a low degree (degree=0.3). After training, we measure the accuracy on the original dataset. Then, we compare this accuracy with the accuracies on the transformed datasets (degrees ranging from 0.3 to 0.9) to assess the robustness of the model.

Algorithm 1 ClientUpdate of HeteroSwitch

Input: Model Parameters W, Client's Dataset D, EMA of Aggregated Loss L_{EMA}(eq. 1)
Parameter: Learning Rate η, # of Epochs E, Batch Size B
Output: Updated Model Parameters W_{return}, Train Loss L_{train}
ClientUpdate(W, L_{EMA})**:**

$Switch_1, Switch_2 = False$
Calculate $L_{init} = L(D, W)$
if $L_{init} < L_{EMA}$ **then**
  $Switch_1 = True$
end if
if $Switch_1 == True$ **then**
  Random Transformation on D(eq. 2, eq. 3)
end if
$\beta \leftarrow$ (split D into batches of size B)
Initialize W_{SWA} as copy of W
Initialize batch index $Idx_b = 0$, Train Loss $L_{train} = 0$
for epoch $e = 1$ to E **do**
  for batch $b \in \beta$ **do**
    $L_{train} \leftarrow \frac{L(b,W) * Idx_b + W}{Idx_b + 1}$
    $W \leftarrow W - \eta \nabla L(b, W)$
    if $Switch_1 == True$ **then**
      $W_{SWA} \leftarrow \frac{W_{SWA} * Idx_b + W}{Idx_b + 1}$
    end if
    $Idx_b \leftarrow Idx_b + 1$
  end for
end for
if $Switch_1 == True$ and $L_{train} < L_{EMA}$ **then**
  $Switch_2 = True$
end if
if $Switch_2 == True$ **then**
  $W_{return} = W_{SWA}$
else
  $W_{return} = W$
end if
return W_{return}, L_{train} to server

SWA with transformation. The x-axis shows different random transformation methods.

Utilizing diverse data transformations with SWAD considerably enhances the model robustness. As depicted in Figure 7, SWAD with data transformations consistently exhibits superior robustness across all transformations. This is highlighted by observed improvements in metrics for Affine, Gaussian noise, WB, and Gamma by 14.0%, 0.4%, 14.6%, and 3.26%, respectively, over models trained only with transformations. The application of SWA also provides better resilience against the Affine transformation, showing a 12.0% improvement over models using only transformations without weight averaging. However, transformations with SWA shows increased vulnerability to transformations such as Gaussian noise, WB, and Gamma, resulting in greater quality degradation. Consequently, SWAD shows better generalization performance compared to SWA.

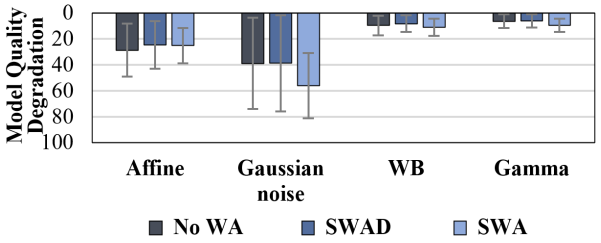

Figure 7. Comparison of Model Quality Degradation for different training methods and transformations

SWA averages the model weights per every epoch while SWAD averages the model weights per every batch which is coarser than epoch. Hence, the integration of SWAD with random transformation, designed to average out significant variability with fewer samples (Dekking et al., 2005), enables DNN models to have enhanced adaptability to both geometric (e.g., Affine) and appearance-based (e.g., WB & Gamma) variations during training.

6 Evaluation Results and Analysis

To assess the effectiveness of HeteroSwitch, we evaluate its impact on both fairness and domain generalization in FL by using the test accuracy of the global model (Shi et al., 2023) on each deployed device type. For fairness, we measure the average variance of accuracy across devices — the portion of participation of each device type follows the market share summarized in Table. 1. For DG, we use the worst-case accuracy across devices as the metric to ensure a minimum required performance across all devices (Sagawa et al., 2019; Krueger et al., 2021).

We configure FL experiments with total devices $N = 100$, minibatch size $B = 10$, selected devices for training for each round $K = 20$, local epoch $E = 1$, and number of rounds $T = 1000$. For a DNN model, we use MobileNetv3-small (Howard et al., 2019), which is widely used in mobile execution environments. We implement HeteroSwitch using PyTorch, and compare it with a baseline method, FedAvg (McMahan et al., 2017), and prior works for data heterogeneity: q-FedAvg (Li et al., 2019), FedProx (Li et al., 2020), and Scaffold (Karimireddy et al., 2020).

We also evaluate HeteroSwitch with a realistic FL dataset, Flair (Song et al., 2022) — this dataset includes images collected by real end-users with more than one thousand device types. Since Flair targets multi-label classification, we compare the averaged-precision across device types and variance of HeteroSwtich with a baseline of FedAvg and prior works (i.e., q-FedAvg and FedProx).

Table 4. Evaluation of HeteroSwitch on fairness and DG

Method	DG	Fairness	
	Acc (%) (worst case)	Variance	Acc (%) (average)
(Baseline) FedAvg	61.17	8.63	64.01
ISP Transformation	63.83	2.58	66.81
+ SWAD	64.50	2.74	66.29
HeteroSwitch	**64.71**	**1.77**	**67.38**
q-FedAvg	59.38	5.40	64.95
FedProx	61.50	8.43	65.16
Scaffold	57.50	7.53	66.18

6.1 HeteroSwitch Evaluation

HeteroSwitch shows a significant variance and accuracy improvement in both fairness and DG as depicted in Table 4. Notably, the use of our proposed generalization methods - ISP (WB&Tone) transformation and SWA per every update - in a incremental manner leads to the improvement compared to the one-fits-all counterparts. As the ISP transformation adds further diversity into the dataset, it reduces the 4.4% higher worst-case accuracy over the baseline, handling the DG problem with *system-induced heterogeneity*. It also mitigates the difference between learned device types due to ISP, improving variance and average accuracy by 68.2% and 4.4%, respectively. per-batch SWA further improves the worst-case accuracy by 1.0% showing better generalization capability — it is designed for stronger generalization technique by averaging updates with different ISP transformation. However, this degrades the variance and average accuracy by 6.2% and 0.8%, respectively, compared to ISP transformation alone. This is because a one-size-fits-all excessive generalization can cause the unnecessary performance degradation for marginalized devices during training.

HeteroSwitch overcomes this limitation by selectively applying the generalization techniques based on the loss comparison with the previous training rounds. As a result, it shows the highest worst-case accuracy, i.e., 5.8% over the baseline, demonstrating the highest generalization capability. HeteroSwitch also demonstrates its ability to handle marginalized devices, showing the best variance and average accuracy (79.5% and 5.3% over the baseline, respectively).

6.2 Comparison with Prior Works

We compare HeteroSwitch with three prominent FL prior works: q-FedAvg (Li et al., 2019), FedProx (Li et al., 2020), and Scaffold (Karimireddy et al., 2020).

q-FedAvg aims to minimize the accuracy variance among clients owing to data heterogeneity, by assigning weights to each client dataset based on the loss. However, q-FedAvg does not consider system-induced data heterogeneity across different device types in FL. As a result, it fails to generalize to unseen device type with 2.9% decrease in worst-case accuracy; it improves the variance and average accuracy by 37.4% and 1.5%, respectively, over the baseline though.

FedProx mitigates data heterogeneity in FL by adjusting the magnitude of local updates with an additional L2 regularization. Similar to q-FedAvg, this method does not focus on system-induced data heterogeneity. Hence, although it improves the worst-case accuracy, variance and average accuracy by 0.5%, 2.3%, and 1.8%, respectively, over the baseline, it shows lower performance compared to HeteroSwitch.

Scaffold, which employs client control variates, to adjust global update directions, addresses non-IID data variance. Still, it faces a similar challenge to q-FedAvg in generalizing to unseen devices, with a 6.0% drop in worst-case accuracy.

HeteroSwitch, which is designed to effectively handle system-induced data heterogeneity, shows better variance, average accuracy, and worst-case accuracy, compared to all the above methods. This distinction highlights the unique contribution of HeteroSwitch in addressing heterogeneous device environments of FL, providing a more balanced and generalized solution.

6.3 Effectiveness to DNN Models

In the previous section, we primarily used MobileNetv3-small, which is widely used in mobile execution environment (Howard et al., 2019; Qian et al., 2021). Here, we conduct additional experiments with ShuffleNet (Ma et al., 2018) and SqueezeNet (Iandola et al., 2016), which are also mobile friendly light-weight models. Table 5 shows the worst-case accuracy for DG and the variance and average accuracy for fairness with various models. As shown in Table 5, HeteroSwitch always shows better worst-case accuracy compared to FedAvg, demonstrating its robustness to domain generalization problem. Although ShuffleNet exhibits a decline in average accuracy with HeteroSwitch, it shows improvement in variance. Conversely, SqueezeNet, which initially fails to learn with FedAvg, showing performance equivalent to random guessing, experiences a dramatic increase in accuracy with HeteroSwitch. These results suggest that, with some adjustments, Shufflenet could work even better with our method if the observed effects are not inherent to the model structure. In short, HeteroSwitch can be used to a variety of mobile-friendly models dealing with the problems caused by system-induced data heterogeneity.

6.4 Impact on Realistic FL Dataset

We evaluate HeteroSwitch with a realistic FL dataset, Flair (Song et al., 2022). The distribution of averaged precision (AP) across device types is shown in Table 6. Because of the increased complexity of system-induced data hetero-

Table 5. Evaluation of HeteroSwitch with different Model architectures

Models	FedAvg			HeteroSwitch		
	DG	Fairness		DG	Fairness	
	Acc (%) (worst case)	Variance	Acc (%) (average)	Acc (%) (worst case)	Variance	Acc (%) (average)
MobileNetV3-small	61.17	8.63	64.01	64.71	1.77	67.38
Shufflenet_v2_x0_5	62.21	10.92	67.09	62.67	4.24	64.76
Squeezenet1_1	8.75	0.00	8.33	16.13	1.70	31.86

Table 6. Evaluation of HeteroSwitch with Flair

Method	Averaged Precision (%)	Variance
(Baseline) FedAvg	47.72	265.79
HeteroSwitch	**47.80**	**249.02**
q-FedAvg	47.25	257.32
FedProx	47.07	313.67

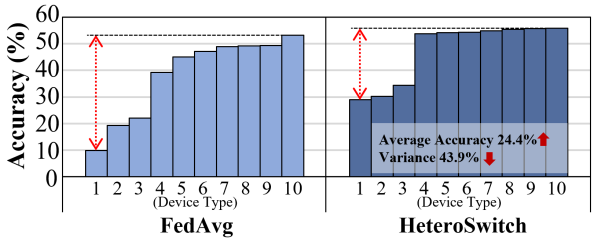

Figure 8. Model accuracy on 10 different synthetically generated device types with CIFAR-100

geneity, FedAvg exhibits a diverse AP distribution across device types.

As the problem becomes more complex, q-FedAvg and FedProx fail to handle the high variance properly. q-FedAvg reduces variance by 3.2% at the expense of averaged precision compared to the baseline. FedProx performs worse with a 1.4% decline in averaged precision and an 18% increase in variance compared to the baseline, indicating its insufficiency to address the problem. In contrast, HeteroSwitch reduces the variance by 6.3% improving the averaged precision by 0.2%. This indicates that HeteroSwitch still mitigates system-induced data heterogeneity even in the presence of more diverse device types in the wild.

6.5 Impact on Synthetic Dataset

In alignment with our findings on collected dataset, we further explore the impact of system-induced heterogeneity using the synthetic CIFAR-100 dataset (Krizhevsky et al., 2009). To emulate the diverse characteristics observed in real device data (Section 3.4), we inject the heterogeneity into the CIFAR-100 dataset by implementing 10 different randomized settings for contrast, brightness, saturation, and hue. A simple CNN model is used to quantify the impact of these modifications in FL setting. The distribution of accuracy across synthetic device types with FedAvg and HeteroSwitch is illustrated in Figure 8.

Under synthetic condition, FedAvg achieves an average accuracy of 38.34% but exhibits a variance of 212.97 across the synthetic devices. This scenario underscores the challenges posed by system-induced data heterogeneity, as certain device types show decreased accuracy. In contrast, HeteroSwitch significantly outperforms FedAvg, enhancing accuracy by 24.4% and reducing variance by 43.9%, demonstrating its effectiveness in managing device heterogeneity.

6.6 Impact on Non-vision Dataset

We expand our evaluation to non-vision data, an Electrocardiogram (ECG) dataset. This dataset comprises data from four distinct sensor types, each introducing unique noise patterns and thereby contributing to heterogeneity in the dataset (Vollmer et al., 2022). A simple DNN model, designed to calculate heart rates from ECG signals, is used to quantify the impact of sensor heterogeneity.

Using FedAvg, heart rate predictions from the same individual ECG data show a significant divergence, with an average deviation of 31.8% due to sensor variability. In contrast, HeteroSwitch, equipped with a Random-Gaussian Filter, notably reduces this deviation to 18.3%. This demonstrates that system-induced data heterogeneity is not confined to vision data and highlights the capability of HeteroSwitch to mitigate such heterogeneity across diverse data types.

7 Related Work

System Heterogeneity in ML: In Centralized Training, several research have shown that neural networks can be vulnerable to system heterogeneity. — e.g., they may suffer quality distortions like blurring and noise (Dodge & Karam, 2016), be affected by simulated camera parameters such as pixel size and exposure (Liu et al., 2020) or display unstable inference with different devices if trained with ImageNet (Cidon et al., 2021). However, no previous work has thoroughly explored which specific steps within the signal processing process introduce biases into the ML training using real devices. Moreover, we are the first to highlight the potential challenges this could pose within a FL context, where a variety of devices can participate.

Data Heterogeneity in FL: Federated Learning, with an increased number of clients, inherently produces diverse data, reflecting variations in class distribution, participant geography, culture, and device usage patterns (Kairouz et al., 2021), leading to data heterogeneity. Prior works have tried to mitigate this user-induced heterogeneity by adopting a regularization method to local models (Li et al., 2020), sharing a small amount of public data across local clients (Zhao et al., 2018; Mansour et al., 2020), or employing weighted averaging for model aggregation (Li et al., 2019). Despite these efforts, differences between devices remain evident even after addressing the user-induced heterogeneity. Given that different causes of data heterogeneity result in unique characteristics in the feature space (where each potentially requires own distinct solution) we focus on the system-induced heterogeneity in this work. To the best of our knowledge, our work is the first to analyze the problems caused by system-induced data heterogeneity in FL where a range of device types can participate in the learning process and propose a solution that addresses the aforementioned challenges while still preserving the data privacy in FL.

8 Conclusion

In this paper, we first introduce *system-induced data heterogeneity* in FL. By using the dataset that we created, we show that system-induced data heterogeneity can negatively affect the accuracy of FL models. We also demonstrate the fairness and domain generalization problems which can stem from system-induced data heterogeneity in realistic execution scenarios of FL. To mitigate system-induced data heterogeneity, we propose HeteroSwitch, a selective generalization technique based on the ISP transformation and SWAD. Our evaluation results demonstrate that HeteroSwitch reduces the variance of accuracy by 79.5% and improves the worst out-of-distribution (OOD) accuracy by 5.8%, compared to the baseline. We believe our work will pave the path forward by enabling future work on system-induced data heterogeneity in a variety of realistic FL execution environment for a practical deployment of FL.

Acknowledgements

This work was supported in part by National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (2021R1C1C1008617 and RS-2023-00212711), ICT Creative Consilience Program through the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (IITP-2024-2020-0-01819), and ITRC (Information Technology Research Center) support program through the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (IITP-2024-RS-2023-00260091).

References

Arena Com, Ltd. Gsmarena. `https://www.gsmarena.com/`, 2022. Accessed: 2023-07-11.

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., and Shi, W. Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67, 2018.

Buckler, M., Jayasuriya, S., and Sampson, A. Reconfiguring the imaging pipeline for computer vision. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 975–984, 2017.

Cannistra, S. Pixel binning. `http://www.starrywonders.com/binning.html`. Accessed: 2023-08-06.

Cha, J., Chun, S., Lee, K., Cho, H.-C., Park, S., Lee, Y., and Park, S. Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems*, 34:22405–22418, 2021.

Chipman, H. A., Kolaczyk, E. D., and McCulloch, R. E. Adaptive bayesian wavelet shrinkage. *Journal of the American Statistical Association*, 92(440):1413–1421, 1997.

Cidon, E., Pergament, E., Asgar, Z., Cidon, A., and Katti, S. Characterizing and taming model instability across edge devices. *Proceedings of Machine Learning and Systems*, 3:624–636, 2021.

Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Creager, E., Jacobsen, J.-H., and Zemel, R. Environment inference for invariant learning. In *International Conference on Machine Learning*, pp. 2189–2200. PMLR, 2021.

Dekking, F. M., Kraaikamp, C., Lopuhaä, H. P., and Meester, L. E. *A Modern Introduction to Probability and Statistics: Understanding why and how*, volume 488. Springer, 2005.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dodge, S. and Karam, L. Understanding how image quality affects deep neural networks. In *2016 eighth international*

conference on quality of multimedia experience (QoMEX), pp. 1–6. IEEE, 2016.

Dou, Q., Coelho de Castro, D., Kamnitsas, K., and Glocker, B. Domain generalization via model-agnostic learning of semantic features. *Advances in neural information processing systems*, 32, 2019.

Ebner, M. *Color constancy*, volume 7. John Wiley & Sons, 2007.

Gozdz, J. Fbdd denoising. `https://valelab4.ucsf.edu/svn/"#"micromanager2/trunk/DeviceAdapters/"#"TetheredCam/LibRaw/internal/dcb_demosaicing.c`, 2010. Accessed: 2023-08-06.

Guliani, D., Beaufays, F., and Motta, G. Training speech recognition models with federated learning: A quality/cost framework. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3080–3084. IEEE, 2021.

Hansen, P., Vilkin, A., Krustalev, Y., Imber, J., Talagala, D., Hanwell, D., Mattina, M., and Whatmough, P. N. Isp4ml: The role of image signal processing in efficient deep learning vision systems. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2438–2445. IEEE, 2021.

Hejazinia, M., Huba, D., Leontiadis, I., Maeng, K., Malek, M., Melis, L., Mironov, I., Nasr, M., Wang, K., and Wu, C.-J. Fel: High capacity learning for recommendation and ranking via federated ensemble learning. *arXiv preprint arXiv:2206.03852*, 2022.

Hirakawa, K. and Parks, T. W. Adaptive homogeneity-directed demosaicing algorithm. *Ieee transactions on image processing*, 14(3):360–369, 2005.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1314–1324, 2019.

Huba, D., Nguyen, J., Malik, K., Zhu, R., Rabbat, M., Yousefpour, A., Wu, C.-J., Zhan, H., Ustinov, P., Srinivas, H., et al. Papaya: Practical, private, and scalable federated learning. *Proceedings of Machine Learning and Systems*, 4:814–832, 2022.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Kim, Y. G. and Wu, C.-J. Autoscale: Energy efficiency optimization for stochastic edge inference using reinforcement learning. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 1082–1096. IEEE, 2020.

Kim, Y. G. and Wu, C.-J. Autofl: Enabling heterogeneity-aware energy efficient federated learning. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 183–198, 2021.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Zhang, D., Le Priol, R., and Courville, A. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pp. 5815–5826. PMLR, 2021.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Li, T., Sanjabi, M., Beirami, A., and Smith, V. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Li, Z., Xu, X., Cao, X., Liu, W., Zhang, Y., Chen, D., and Dai, H. Integrated cnn and federated learning for covid-19 detection on chest x-ray images. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2022.

Lin, C.-k. Pixel grouping for color filter array demosaicing, 2003.

Liu, Z., Lian, T., Farrell, J., and Wandell, B. A. Neural network generalization: The impact of camera parameters. *IEEE Access*, 8:10443–10454, 2020.

Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 116–131, 2018.

Maeng, K., Lu, H., Melis, L., Nguyen, J., Rabbat, M., and Wu, C.-J. Towards fair federated recommendation learning: Characterizing the inter-dependence of system and data heterogeneity. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 156–167, 2022.

Mansour, Y., Mohri, M., Ro, J., and Suresh, A. T. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Mohri, M., Sivek, G., and Suresh, A. T. Agnostic federated learning. In *International Conference on Machine Learning*, pp. 4615–4625. PMLR, 2019.

Morovič, J. *Color gamut mapping*. John Wiley & Sons, 2008.

Pessach, D. and Shmueli, E. A review on fairness in machine learning. *ACM Computing Surveys (CSUR)*, 55(3):1–44, 2022.

Qian, S., Ning, C., and Hu, Y. Mobilenetv3 for image classification. In *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, pp. 490–497, 2021. doi: 10.1109/ICBAIE52039.2021.9389905.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Scheuerman, M. K., Hanna, A., and Denton, E. Do datasets have politics? disciplinary values in computer vision dataset development. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW2):1–37, 2021.

Segu, M., Tonioni, A., and Tombari, F. Batch normalization embeddings for deep domain generalization. *Pattern Recognition*, 135:109115, 2023.

Shi, Y., Yu, H., and Leung, C. Towards fairness-aware federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Song, C., Granqvist, F., and Talwar, K. Flair: Federated learning annotated image repository. *Advances in Neural Information Processing Systems*, 35:37792–37805, 2022.

StatCounter. Mobile vendor market share. `https://gs.statcounter.com/vendor-market-share/mobile/united-states-of-america/2021`, 2022. Accessed: 2023-07-11.

Stokes, M. A standard default color space for the internet-srgb. *http://www. w3. org/Graphics/Color/sRGB. html*, 1996.

Tommasi, T., Patricia, N., Caputo, B., and Tuytelaars, T. A deeper look at dataset bias. *Domain adaptation in computer vision applications*, pp. 37–55, 2017.

Vollmer, M., Bläsing, D., Reiser, J., Nisser, M., and Buder, A. Simultaneous physiological measurements with five devices at different cognitive and physical loads. *Physionet*, 101(23):215–220, 2022.

Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., Chen, Y., Zeng, W., and Yu, P. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Wu, C.-J., Brooks, D., Chen, K., Chen, D., Choudhury, S., Dukhan, M., Hazelwood, K., Isaac, E., Jia, Y., Jia, B., et al. Machine learning at facebook: Understanding inference at the edge. In *2019 IEEE international symposium on high performance computer architecture (HPCA)*, pp. 331–344. IEEE, 2019.

Wu, H. and Wang, P. Node selection toward faster convergence for federated learning on non-iid data. *IEEE Transactions on Network Science and Engineering*, 9(5): 3099–3111, 2022.

Wyszecki, G. and Stiles, W. S. *Color science: concepts and methods, quantitative data and formulae*, volume 40. John wiley & sons, 2000.

Zhang, H., Li, L., Qiao, K., Wang, L., Yan, B., Li, L., and Hu, G. Image prediction for limited-angle tomography via deep learning with convolutional neural network. *arXiv preprint arXiv:1607.08707*, 2016.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

A Experimental Details

A.1 Hardware & Software

We implement FL with PyTorch and TensorFlow, and emulate it with our dataset and Flair, respectively, on our servers. **The following is the experimental environment used for our dataset (Section 3.1).**

- GPU/CPU models: No GPU, Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz
- Amount of memory: 128GiB
- Operating system: Ubuntu 20.04.5
- Version of PyTorch: 1.12.0

The following is the experimental environment used for the Flair (Song et al., 2022).

- GPU/CPU models: NVIDIA Tesla V100 GPU, Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz
- Amount of memory: 128GiB
- Operating system: Ubuntu 20.04.3
- Version of TensorFlow: 2.9.0
- Version of TensorFlow-Federated: 0.20.0

A.2 FL Parameters

We set the hyperparameters of FL based on the sensitivity analysis. **The hyperparameters that we test are summarized as follows:**

- learning rate $\eta \in \{0.001, 0.01, 0.1\}$
- Minibatch size $B \in \{1, 10, 20\}$
- Local epochs $E \in \{1, 3, 5\}$
- Number of communication rounds $T \in \{100, 500, 1000\}$

Fig. 9 shows the sensitivity analysis result on the learning rate, minibatch size, local epoch, and number communication rounds. Based on results presented in Fig. 9, we select 0.1, 10, 1, and 1000 for the learning rate, minibatch size, local epoch, and the number of communication rounds, respectively, for the rest of our experiments (i.e., characterization and evaluation in Section 3 and 6, respectively).

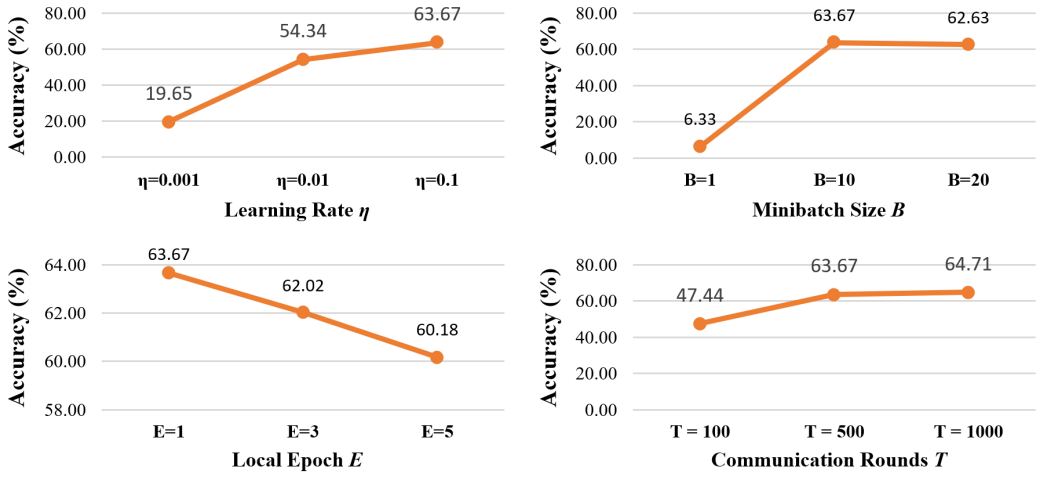

Figure 9. Selection of Learning Rate, Minibatch Size, Local Epoch, and Communication Rounds

For a fair comparison with q-FedAvg and FedProx, we also configure the hyperparameters of q-FedAvg and FedProx based on the sensitivity analysis. **Below are the hyperparameter grids that we searched on for previous works(Section 6.2).**

- q-FedAvg $q \in \{1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$
- FedProx $\mu \in \{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$

we use $q = 1e - 6$ and $\mu = 1e - 1$, for q-FedAvg, and FedProx, respectively.

Below are the hyperparameter grids that we searched on for HeteroSwitch

- Random WB $degree \in \{0.001, 0.01, 0.1, 0.5, 0.9\}$
- Random Gamma $degree \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$

We set the smoothing factor for Exponential Moving Average $\alpha = 0.9$ for L_{EMA}, Random WB $degree = 0.001$ and Random Gamma $degree = 0.9$.