Adapting the Attention of Cloud-Based Recognition Model to Client-Side Images without Local Re-Training

Yangwenjian Tan LIUAN18@SJTU.EDU.CN
Yikai Yan MIKU.MIKU.MIKU@SJTU.EDU.CN
Chaoyue Niu* RVINCE@SJTU.EDU.CN
Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Editors: Vu Nguyen and Hsuan-Tien Lin

Abstract

The mainstream workflow of image recognition applications is first training one global model on the cloud for a wide range of classes and then serving numerous clients. Images uploaded by each client typically come from a small subset of classes. From the cloud-client discrepancy on the range of image classes, the recognition model is desired to have strong adaptiveness, intuitively by focusing on each client's local dynamic class subset, while incurring negligible overhead. In this work, we propose to plug a new intra-client and inter-image attention (ICIIA) module into existing backbone recognition models, **requiring only one-time cloud-based training to be client-adaptive.** In particular, given an image to be recognized from a certain client, ICIIA introduces multi-head self-attention to retrieve relevant images from the client's local images, thereby calibrating the focus and the recognition result. We further identify the bottleneck of ICIIA's overhead being in linear projection, propose to group and shuffle the features before the projections, and allow increasing the number of feature groups to dramatically improve efficiency without scarifying much accuracy. We extensively evaluate ICIIA and compare its performance against several baselines, demonstrating effectiveness and efficiency. Specifically, for a partitioned version of ImageNet-1K with the backbone models of MobileNetV3-L and Swin-B, ICIIA improves the classification accuracy to 83.37% (+8.11%) and 88.86% (+5.28%), while adding only 1.62% and 0.02% of FLOPs, respectively. Source code is available in the supplementary materials.

Keywords: image recognition, model adaptation

1. Introduction

Nowadays, many computer vision (CV) models have been deployed to recognize client-side images in practice, such as identifying everything with Google Lens, categorizing images in Google Photos, and taking photos to search for products on Amazon Shopping. Let's examine the workflow of Google Lens, a mobile camera app with strong image recognition capabilities, in detail. A global recognition model is trained on the cloud using a large scale of labeled images, spanning a wide range of classes (e.g., various species of animals and plants, documents, commodities, etc.). Then, the recognition model will serve numerous app users once they take photos in daily life, and the photos taken by each user tend to involve a few classes.

* Corresponding author

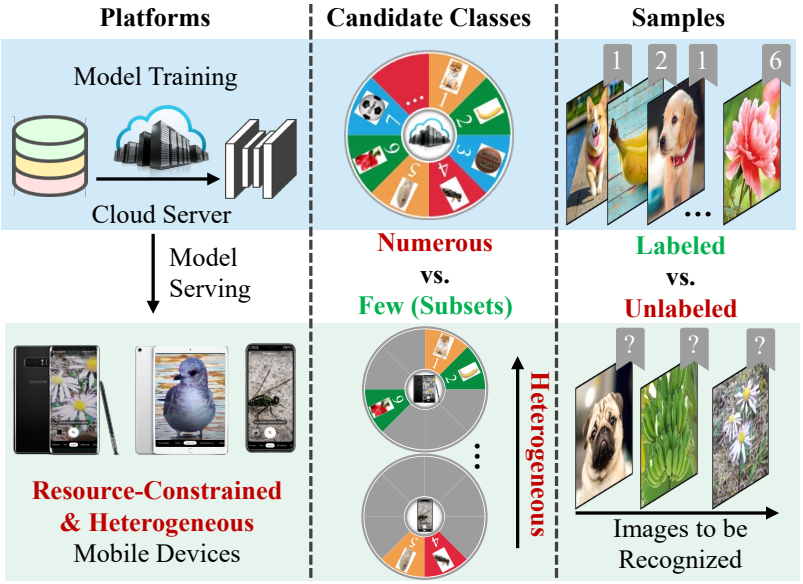


Figure 1: Image recognition from a new cloud-client view.

By probing image recognition from a new cloud-client perspective, we find several important but often neglected characteristics, as illustrated in Figure 1. (1) The recognition model is optimized for the full set of classes on the cloud, whereas the images to be recognized on each individual client normally come from a small subset of classes. Specifically, the class subsets of different clients differ from each other, and as a client collects new images over time, the client's class subset will change dynamically. (2) Different from the cloud with labeled images for training, the images on each client are unlabeled for real-time recognition. It is also practically infeasible to let non-expert users label images and take their annotations as the ground truth. More generally, the missing of labels on the side of clients is an atypical setting of CV scenarios, compared with natural language processing (NLP) and recommendation applications, which typically predict user interactive behaviors (e.g., next input word, click or browse) and take practical user feedback as labels.

The above cloud-client discrepancies raise the new requirement of strong adaptiveness on the image recognition model, from the cloud's full set of classes to each client's local subset, across different clients' heterogeneous class subsets, and over a certain client's dynamic image dataset. Intuitively, the focus of the recognition model was originally distributed on all the classes. When serving a specific client, if the focus can be concentrated on the client's class subset and further be dynamically adjusted as more local images are accumulated, the recognition accuracy can be significantly improved.

To achieve model adaptation effectively and efficiently, there still exist several practical challenges. From model architecture, the design should be model-agnostic. In particular, the design should be compatible with existing backbone networks for image recognition (e.g., convolutional neural network (CNN), transformer, etc.), thereby inheriting their strong representation abilities. In addition, no matter whether the recognition model was originally deployed on the cloud to serve a large scale of clients or offloaded to resource-constrained and heterogeneous clients for local serving, the design would better introduce only few

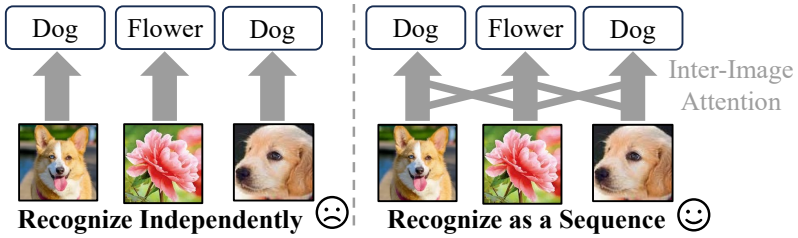


Figure 2: Comparison of traditional recognition paradigm with our framework. **All clients share the same attention module** which is regularly trained on the cloud.

parameters and negligible overhead to guarantee high efficiency. Of course, in the resource-rich context, scaling up the design for better accuracy should be an available option. From learning algorithm, although client-specific fine-tuning is model-agnostic and even does not need extra parameters, the missing of labels and the resource constraints of clients make such type of methods inapplicable to image recognition. Therefore, it is highly desirable, yet challenging, to adapt the model only once on the cloud and circumvent on-client re-training.

In this work, we propose an Intra-Client and Inter-Image Attention (ICIIA) module, which is plugable between the feature extractor and the classifier of an arbitrary backbone model for image recognition. After one-time training on the cloud as regular, ICIIA can well adapt to heterogeneous clients without local re-training. As shown in Figure 2, different from the conventional image recognition paradigm, which treats the images to be recognized independently, ICIIA instead adopts a sequence modeling method and exploits a client's historical images to benefit recognizing a certain image. In fact, sequence modeling of data is common and widely used in NLP and recommendation, where a sample is represented by a sequence of tokens (i.e., words in NLP or items in recommendation), and the dependence among the tokens is captured with attention mechanism. ICIIA also introduces multi-head self-attention Vaswani et al. (2017) to mine inter-image dependence for each individual client. In particular, when serving a client, given an image to be recognized, ICIIA retrieves the relevant images from the client's historically recognized images and enables the recognition model to concentrate on the local dynamic subset of classes, thereby calibrating the features and the recognition results. Through naturally inputting the personalized images from the client without manually inserting additional client-specific parameters, ICIIA can achieve adaptiveness in a desired one-for-all way. Further, as the client accumulates more images for attention, the performance of ICIIA will become better. However, when applying intra-client and inter-image attention, different from NLP and recommendation with a small token size, the dimension of each image's feature vector is high, and the linear projections pose a new bottleneck. ICIIA thus divides the features into several groups, linearly projects each group individually, and shuffles the features across groups, thereby reducing the overhead to the reciprocal of the number of feature groups. We summarize the key contributions as follows:

- From a new cloud-client perspective to view image recognition, we find the discrepancies and the variations on the range of candidate classes, raising the practical requirement of one-time model adaptation.

- We propose a plugable ICIIA module upon existing backbone image recognition models. To the best of our knowledge, ICIIA is the first to leverage the attention mechanism to mine intra-client and inter-image dependencies, thereby achieving strong model adaptiveness without local re-training. We further propose feature grouping and shuffling to sharply reduce the overhead of linear projections in ICIIA.
- We extensively evaluate ICIIA, with several baselines for comparison and variants for ablation study. Evaluation results reveal ICIIA's effective and efficient adaptation to heterogeneous clients, significant improvements over the original backbone models and tuning methods, and the necessity of each ingredient.

2. Related Work

Model adaptation. Adapting the global model trained on the cloud to serve heterogeneous clients is an important and practical problem. Rather than the cloud-to-client adaptation considered in this work, existing literature mainly considered task-to-task adaptation, either from upstream to downstream Devlin et al. (2019) (e.g., from masked auto-encoding to classification), from one domain to another Ye et al. (2022) (e.g., from daytime scenes to nighttime scenes), or across multiple modalities Radford et al. (2021) (e.g., natural language and vision). Among these work, some resorted to fine-tuning the pre-trained model on the labeled data of each target task and developed new pre-training techniques to facilitate the subsequent fine-tuning process Devlin et al. (2019). Other work proposed to modify the backbone model, by inserting learnable task-specific adapters Houlsby et al. (2019) or providing tunable prompts Li and Liang (2021); Lester et al. (2021). These methods consider model adaptation to a few target tasks normally with labeled samples. However, our cloud-to-client model adaptation involves a large number of target clients with only UNLABELED images, making previous designs inapplicable.

On-client training. Some work focused on recommendation or next-word prediction with the samples naturally labeled by each client and explored on-client model tuning over local data Yan et al. (2022). When extended to the setting of cloud-client collaborative training, an emerging federated learning McMahan et al. (2017) framework allows heterogeneous clients to jointly train a global model without uploading local data. Recent work further adapted the global model to each client's heterogeneous data Mansour et al. (2020). The designs mainly fall into the pattern of letting each client maintain a personalized model with client-specific parameters, by multi-task learning Smith et al. (2017), meta learning Chen et al. (2018), continual learning Yoon et al. (2021), hypernetworks Ma et al. (2022), knowledge distillation Zhu et al. (2021), etc. However, on-client training requires labeled samples, which are unavailable in the scenario of image recognition. Training on resource-constrained mobile devices also leads to problems like longer training time. ICIIA instead requires only cloud-based training as regular and does not need on-client re-training.

Attention mechanism. For NLP and recommendation tasks, each sample is represented by a sequence of tokens. The attention mechanism has been widely adopted to mine cross-token dependencies within each sample, such that the model can focus on the related tokens. In particular, Vaswani et al. Vaswani et al. (2017) proposed a novel architecture, called transformer, based on multi-head self-attention with residual connections He et al. (2016) and layer normalization Ba et al. (2016). Transformer has achieved state-of-the-art

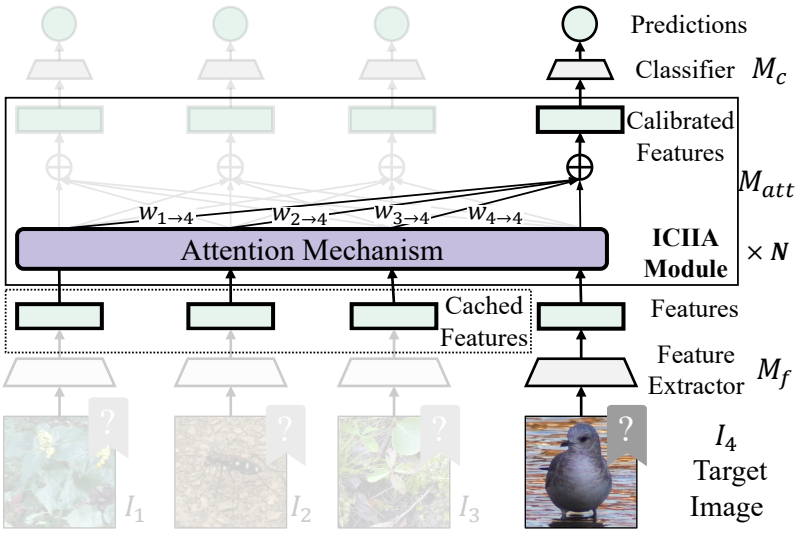


Figure 3: The plugged ICIIA module in an arbitrary image recognition backbone network. Features extracted from historical images are cached to assist the recognition of the current target image.

performance initially in NLP and later in CV Dosovitskiy et al. (2021), by capturing the INTRA-sample dependencies among words or image patches. At a totally different level, ICIIA captures the INTER-image dependencies for each client.

3. Design and Analysis

3.1. Overall Architecture

As illustrated in Figure 3, the ICIIA module is plugable between the feature extractor and the classifier of an arbitrary recognition backbone network. Same as the backbone, the parameters of this new module are shared by all clients and are trained on the cloud as regular. The entire model is either deployed for conventional cloud-based serving or offloaded to perform on-client inference. The key difference is that each batch of images to be recognized is fed as a sequence, rather than as individual samples. By default, the client-side images are recognized in a batch mode: the flattened feature vectors of the images in the same batch are calibrated with each other using self-attention. If the images are recognized one by one, then for a certain image, the features of the recent B unlabeled images are cached and can be used for attention.

3.2. Intra-Client and Inter-Image Attention

We adopt the standard transformer encoder layer Vaswani et al. (2017) as the building block of ICIIA to retrieve relevant images for calibrating the recognition of a certain image. Other architectures such as CNN are also compatible with our design, but according to the evaluation in Section 4.4, they are less effective than the transformer architecture.

Each ICIIA module consists of a multi-head self-attention (MSA) layer and a MLP layer with residual connections and LayerNorm (LN). Given the features of B image input to the l-th ICIIA module, denoted as a $B \times D$ matrix $I^l = (I_1^l \quad \dots \quad I_B^l)^T$, the output is computed as:

$$\hat{I}^l = \mathrm{LN}(I^l + \mathrm{MSA}(I^l, I^l, I^l)), I^{l+1} = \mathrm{LN}(\hat{I}^l + \mathrm{MLP}(\hat{I}^l)).$$

The $\mathrm{MSA}(Q, K, V)$ operation across images is defined as:

$$\sum_{h=1}^{H} \left(\left(\mathrm{softmax}(\frac{QW_q^h(KW_k^h)^T}{\sqrt{D/H}}) VW_v^h \right) W_o^h \right), \tag{1}$$

where H is the number of attention heads; $W_q^h, W_k^h, W_v^h \in \mathbb{R}^{D \times D/H}$, $W_o^h \in \mathbb{R}^{D/H \times D}$ are the weight matrices for the query, key, value, and output subspaces, respectively; and the bias terms are omitted for simplicity. With multi-head self-attention, the feature vector of each image is calibrated using the weighted average of the feature vectors of the images in the same batch, where the weights take the attention scores $\mathrm{softmax}(QW_q^h(KW_k^h)^T/\sqrt{D/H})$.

3.3. Model Training

Algorithm 1 Training process of ICIIA on the cloud

Input: The original recognition model backbone M, including the feature extractor M_f and the classifier M_c.

Output: The trained ICIIA module M_{att}.

1: Collect image datasets $\mathbb{D}_i'$ from each available client i.
2: Extract the features of each raw image using M_f and annotate the images on the cloud. The resulted feature-label pairs form the training datasets $\mathbb{D}_i$.
3: Initialize the ICIIA module M_{att}
4: **for** Each dataset $\mathbb{D}_i$, $i \in \{1, 2, \cdots, n\}$ **do**
5: **for** Each batch $\mathbb{B}$ in $\mathbb{D}_i$ **do**
6: Do the forward pass $M_c(M_{att}(\mathbb{B}))$.
7: Do backward pass by freezing the feature extractor M_f and **passing gradients through only the classifier M_c and the ICIIA module M_{att}.**
8: **end for**
9: **end for**

We describe the training process of ICIIA in Algorithm 1. It only additionally requires that each batch comes from the same client. All other procedures are identical to a standard cloud-based training process. Furthermore, if a well-trained backbone recognition model M is already available, it is possible to pre-compute the image features (Line 2) and skip both the forward and backward passes of the computationally expensive feature extractor M_f (Lines 6–7).

3.4. Feature Grouping and Shuffling

In standard NLP tasks, the transformer encoder layer involves a small token size and a large number of tokens Child et al. (2019). In contrast, ICIIA involves a large token size

(i.e., the feature dimension D) and a small number of tokens (i.e., the batch size B). As a result, when computing and applying the attention scores, the linear projections, whose parameter size is the square of the token size, dominate the overhead and become a new bottleneck.

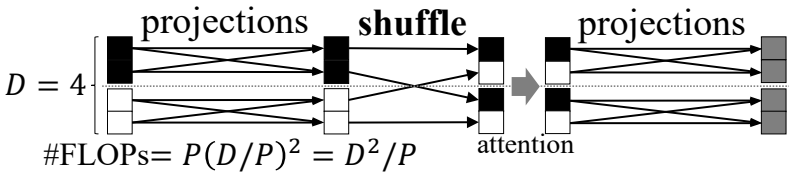


Figure 4: Linear projection with feature grouping and shuffling. Each D-dimensional input feature vector is divided into P D/P-dimensional groups (one black group and one white group in the figure), and each group is linearly projected individually. The features are then shuffled across groups before the attention heads.

To improve efficiency, as shown in Figure 4, we propose to divide the D-dimensional input features into P groups, each being D/P-dimensional, and create a small $D/P \times D/P$ linear projection for each group. Without feature grouping, a complete D-dim linear projection needs D^2 parameters and FLOPs, while by individually project each D/P-dim group, the parameter size and FLOPs are reduced to $P(D/P)^2 = D^2/P$, i.e., reduced by a factor of $1/P$. However, feature grouping incurs a side effect: the outputs from a certain group are derived from only the features in the same group. This isolates the information flow across feature groups and will degrade the model's representation ability. To handle this problem, we shuffle the features across different groups after each linear projection. In particular, given P feature groups with each group being D/P-dimensional, we simply transpose the $P \times D/P$ matrix and reshape it back into $P \times D/P$. From Figure 4, we can see that both the upper and the lower groups have both black and white features after feature shuffling, and the information are mixed in the next projection.

3.5. Overhead Analysis

Suppose N layers of the ICIIA module is adopted, and the attention scores are computed among a batch of B images. Each layer has 6 groups of linear projections, including 3 input projections for the query, key, and value vectors, 1 output projection, as well as 2 projections in the MLP layer, requiring $6D^2/P$ parameters and $6BD^2/P$ FLOPs. In addition, computing the attention scores requires B^2D FLOPs and applying the scores also requires B^2D FLOPs. Therefore, the total size of ICIIA-related parameters[1] is #Param. $= 6D^2/P \times N$. The ICIIA-related computational cost[1] of recognizing B images in a batch (e.g., in the scenario of Google Photos for categorizing one client's many local images) or recognizing one image given $B-1$ historical images is #FLOPs $= (6BD^2/P + 2B^2D) \times N$.

To give more intuitions about the efficiency of ICIIA, we introduce two different configurations: one is the base version without feature grouping (i.e., $P = 1$), called ICIIA-B; the

1. The parameter size and computational cost of the residual connections and layer normalization are negligible and thus omitted.

Table 1: The size of parameters and the number of FLOPs for ICIIA-B and ICIIA-T on *ImageNet-1K* with different backbone models. The #additional FLOPs are reported as the total computation for a sequence of B images (i.e., $(6BD^2/P + 2B^2D) \times N$). The percentages in the parentheses denote the relative ratios to the backbone model.

	Backbone		**ICIIA-B**		**ICIIA-T**	
	#Param.	#FLOPs	#Additional Param.	#Additional FLOPs	#Additional Param.	#Additional FLOPs
MobileNetV3-L	**5.5M**	**0.23G**	30M (538%)	0.47G (202%)	0.14M (**2.47%**)	3.8M (**1.62%**)
ResNet-152	60M	12G	76M (126%)	1.2G (10.4%)	0.33M (**0.54%**)	7.9M (**0.07%**)
EfficientNet-B4	19M	4.6G	58M (299%)	0.93G (20.2%)	0.25M (**1.32%**)	6.4M (**0.14%**)
Swin-B	88M	15G	19M (21.5%)	0.30G (1.97%)	0.09M (**0.10%**)	2.8M (**0.02%**)
ConvNeXt-L	198M	34G	43M (21.5%)	0.68G (1.98%)	0.19M (**0.10%**)	5.0M (**0.01%**)
EfficientNet-B7	66M	39G	118M (178%)	1.9G (4.86%)	0.50M (**0.76%**)	11M (**0.03%**)

Table 2: The practical inference latency and memory for recognizing one image with MobileNetV3-L. When plugging in the ICIIA module, we report the TOTAL overhead of recognizing one image with the assistance of the cached features of 15 historical images.

	Inference Latency	Memory
Original Backbone	5.0ms	1042MB
With ICIIA	5.2ms (**+4%**)	1042MB (**+0%**)

Table 3: The datasets and their client-level statistics. For the train-test split, "by client" means the training and testing datasets come from different clients, and "within client" means each client has some samples for training and the others for testing.

Dataset	**Client Partition**	**Train-Test Split**	**#Clients**	**#Samples**	**#Classes**	**Classes per Client**	
						mean	stdev
iNaturalist 2019	by user	by client	2,295	193,210	1,010	45.5	41.8
FEMNIST	by writer	within client	3,597	817,851	62	55.0	6.6
CelebA	by face ID	by client	9,343	200,288	N/A	N/A	N/A
ImageNet-1K	by class	within client	3,161	1,331,167	1,000	15.2	7.1
UCF101	by class	within client	121	13,320	101	22.8	9.2

other is the tiny version with $P = 256$ groups, called ICIIA-T. We take the 6 different backbone models to be evaluated over *ImageNet-1K*, set the number of ICIIA layers to $N = 3$, and set the number of images for computing attention scores to $B = 16$. Table 1 lists the number of parameters and FLOPs for ICIIA-B and ICIIA-T, as well as their relative ratios to the backbone. We find that compared with ICIIA-B, ICIIA-T sharply reduces the number of parameters and FLOPs via feature grouping, and meanwhile, introduces negligible overhead compared with the backbone. Even for MobileNetV3-L Howard et al. (2019), a lightweight network that can be deployed on mobile devices, the theoretical overhead is only 2.47% additional parameters and 1.62% additional FLOPs. We also measured the practical inference latency and memory consumption for MobileNetV3-L in Table 2, and observe only 4% additional latency and zero additional memory.

4. Evaluation

4.1. Setup

Datasets and recognition tasks. We extensively evaluate ICIIA on 5 representative datasets with 9 benchmark models, involving 3 different tasks. We list the statistics of the datasets in Table 3 and introduce them as follows.

iNaturalist 2019 Horn and Aodha (2019) contains images of 1,010 species of plants and animals collected by the users from iNaturalist[2], a citizen science website for naturalists. The task is to recognize the species of the images. We adopt the natural user partition of FedScale Lai et al. (2022), allocating each image to the corresponding user who holds the rights, while splitting the user pool into 1,901 users for training and 394 ones for testing. We leave out 20% of the users originally for training now for validation use. We take EfficientNet-B0 Tan and Le (2019) as the recognition model.

FEMNIST Caldas et al. (2018) is a dataset for recognizing hand-written digits and characters, and was built by LEAF Caldas et al. (2018) through partitioning EMNIST LeCun et al. (1998); Cohen et al. (2017) based on the writer. LEAF originally takes 90% and 10% of each writer's samples for on-client training and testing, respectively. We separate out 20% of the training samples for validation. We adopt the CNN architecture officially provided by LEAF.

CelebA Liu et al. (2015) contains face images of 10,177 celebrities, each with 40 binary attribute annotations. The task is to recognize the attributes. Since the attributes of being "male" or not and being "young" or not are normally unique and easy to be inferred for a certain user's images, we remove these 2 attributes and keep the other 38 attributes for recognition. We take LEAF's Caldas et al. (2018) natural user partition based on face ID, dividing the user pool into 8,408 users for training and 935 ones for testing. We still leave out 20% of the users originally for training now for validation. For multi-label classification, we replace the final layer of EfficientNet-B0 Tan and Le (2019) with multiple linear classifiers, one for each attribute.

As shown in Table 3, the local dataset of each client normally involves a small subset of all the classes, validating the starting point of the model adaptiveness requirement. For *ImageNet-1K* and *UCF101* which do not provide user ID for natural partitioning, we simulate this property as follows.

ImageNet-1K Deng et al. (2009) contains 1,331,167 images of 1,000 classes organized under the WordNet hierarchy Miller (1995) with 85 parent categories. We take the official train-test split and let each client hold only images from the same parent category. Each client holds roughly 324, 81, and 16 samples for training, validation, and testing, respectively. Regarding the backbone model, we adopt MobileNetV3-L Howard et al. (2019), ResNet-152 He et al. (2016), EfficientNet-B4 and -B7 Tan and Le (2019), Swin-B Liu et al. (2021), and ConvNeXt-L Liu et al. (2022).

UCF101 Soomro et al. (2012) contains 13,320 videos of 101 action classes from 5 general/parent action categories. Same as *ImageNet-1K*, we partition the clients according to the parent category. Each client takes roughly 62, 16, and 32 videos from a certain parent category for training, validation, and testing, respectively. We take C3D Tran et al. (2015) as the action recognition model.

2. www.inaturalist.org

Baselines and implementation details. We introduce 3 baselines, including the original backbone model, fine-tuning, and prompt tuning, for comparison.

Original backbone model takes the mainstream recognition network architectures without the ICIIA module. In addition, the backbone model is optimized over the global training dataset (i.e., a mixture of all the clients' training datasets). We initialize EfficientNets, the other models over *ImageNet-1K*, and C3D, by loading the weights pre-trained over *ImageNet-1K* from efficientnet-pytorch Melas-Kyriazi (2021), torchvision Paszke et al. (2019), and pytorch-video-recognition Zhang (2019), respectively. For *iNaturalist 2019*, *CelebA*, and *UCF101*, we further tune the pre-trained models over the global training dataset. For *FEMNIST*, we train the CNN model from scratch.

Fine-tuning is to let each client fine-tune the backbone model over its local training dataset and is a classical method for model adaptation. In the experiment, we fine-tune only the last layer of the backbone model (i.e., the classifier(s)) and freeze the other layers, which can function as a static feature extractor. We note that fine-tuning is not applicable to *iNaturalist 2019* or *CelebA*, because the clients in the testing dataset do not appear in the training dataset.

Prompt tuning, as introduced in Section 2, is an enhanced version of fine-tuning originally designed for NLP. We adapt this method to our context by associating each client with a prompt token and feed it as an extra input to the recognition models. We try and optimally set the dimension of the tokens to half of the feature dimension $D/2$. Similar to fine-tuning, prompt tuning is inapplicable in practice due to the lack of data annotation on the client side.

For the settings of our ICIIA module, we evaluate the two configurations of ICIIA-B and ICIIA-T introduced in Section 3.5. The detailed hyperparameter settings and hardware specifications are deferred to supplementary materials.

4.2. Adaptiveness to Heterogeneous Clients

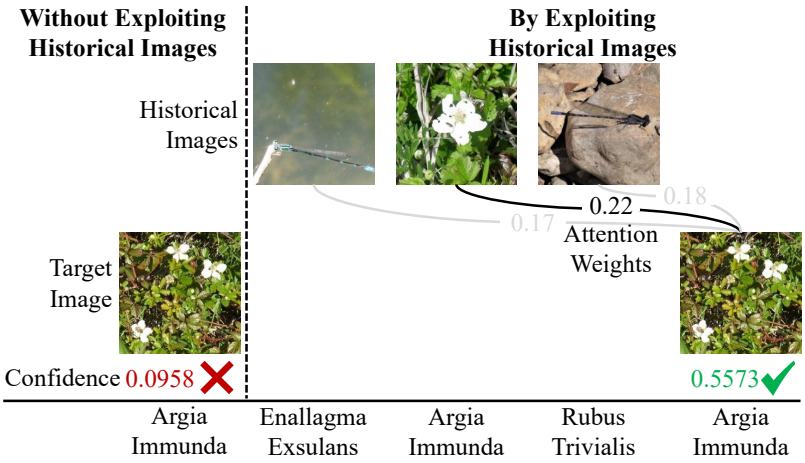


Figure 5: Adaptation to client #1919 in *iNaturalist 2019*.

Calibration from historical recognized images. We first visualize how ICIIA adapts to each client's local data distribution in the recognition phase. We pick client #1919

from *iNaturalist 2019*'s testing set and illustrate how ICIIA-B calibrates the classification from historical images in Figure 5. The starting point on the left is a production-ready model that can already accurately classify most images, but still misclassifies the image from "Argia Immunda". To calibrate the classification, ICIIA-B calculates the dependency between the target image and the historical images, pays the most attention to the second image from the same class (i.e., "Argia Immunda") with the target one, improves the confidence on the correct class, and classifies correctly.

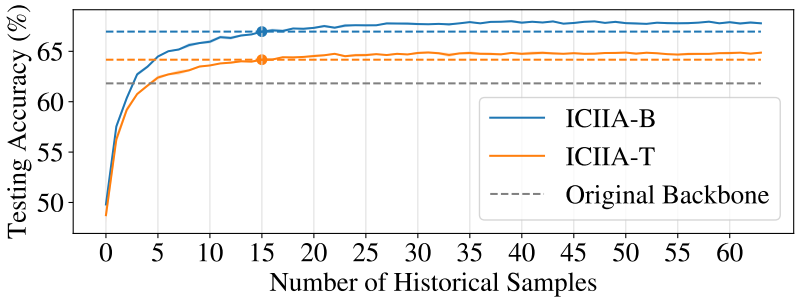


Figure 6: Testing accuracy of ICIIA over *iNaturalist 2019* by varying the number of historical samples. Dashed lines correspond to the training setting of 15 historical samples.

We also plot how the testing accuracy of ICIIA-B and ICIIA-T changes with the accumulation of historical images in Figure 6. One key observation is that ICIIA-B and ICIIA-T can quickly adapt to each client's local distribution and outperform the original backbone model using only 3 and 5 historical images, respectively. If a client has not accumulated enough historical images yet, it can trivially switch to the original backbone model, avoiding cold start. The second key observation is that as the size of historical images increases, ICIIA-B and ICIIA-T achieve better performance. Further considering the attention is performed on the batches of 16 images (15 historical images + 1 target image) in the training phase, while ICIIA-B and ICIIA-T continues to improve after accumulating 15 historical images in the prediction phase, we can derive that ICIIA generalizes well.

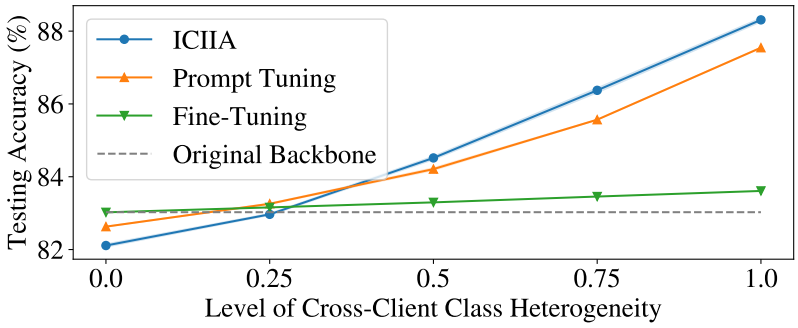
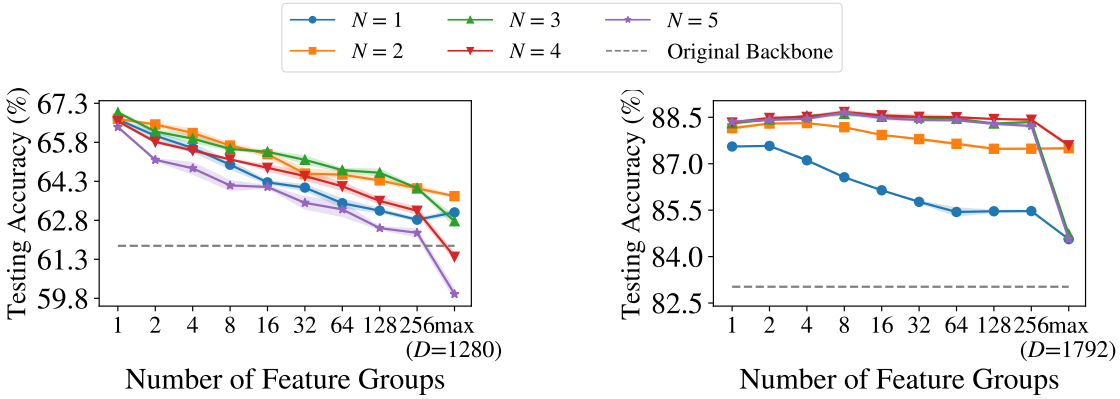


Figure 7: Testing accuracy of ICIIA-B and three baselines over *ImageNet-1K* with EfficientNet-B4, by varying the level of cross-client class heterogeneity, where the level "0" denotes the ideal case of homogeneity.



(*a*) *iNaturalist* (EfficientNet-B0) (*b*) *ImageNet* (EfficientNet-B4)

Figure 8: Testing accuracy of ICIIA with varying number of groups P and layers N. The "max" on x-axis denotes that P takes the feature dimension D. Results are averaged over 3 repeats, and the shaded region shows the standard error.

Adaptation to cross-client class heterogeneity. We next study how the heterogeneity of the image classes on different clients affects ICIIA. We use *ImageNet-1K* and vary client-level dataset splitting. In particular, the first group of clients now fetches the samples randomly from all the 85 parent categories, while the second group still fetches from a specific parent category. We view the ratio between the size of the second group and the size of all the clients as a metric of cross-client class heterogeneity. If the ratio is 0, all the clients fetches samples from all the parent categories, and cross-client class heterogeneity, as well as the cloud-client discrepancy, do not exist. We depict the testing accuracy of ICIIA-B and the three baselines in Figure 7. We can see that the advantage of ICIIA-B, fine-tuning, and prompt tuning over the original backbone model becomes larger as the level of cross-client class heterogeneity increases, and ICIIA-B performs the best. In the ideal case of no cross-client class heterogeneity, due to the disappearance of the device-cloud discrepancy, all the adaptive methods have no positive effect.

Adaptation of the number of feature groups P. We finally evaluate how ICIIA balances model performance and efficiency by taking different P. Figure 8 shows the evaluation results over *iNaturalist 2019* and *ImageNet-1K*, where the number of ICIIA layers N varies to demonstrate robustness. From Figure 8, we can see that as P increases, the parameter size is reduced to $1/P$ of ICIIA-B's size, and the testing accuracy of ICIIA will decrease. However, even when P reaches 256 (i.e., ICIIA-T), the testing accuracy of ICIIA-T is still higher than that of the original backbone. The advantage and the robustness of ICIIA are more evident over *ImageNet-1K*. We also observe a severe accuracy drop in the extreme case of $P = D$, because all the features are isolated from each other, and feature shuffling fails to work.

Table 4: ICIIA vs. the baselines in terms of testing accuracy. Although existing approaches require on-device training and are not applicable in practice due to the lack of data annotation, we try to simulate two typical methods, fine-tuning and prompt tuning, for comparison. The improvement over the original backbone model is shown in parentheses. The results are averaged over three repeats, each repeat randomly initializing the model parameters excluding the pre-trained weights. The standard errors of the mean are all below 0.24%.

Dataset	Backbone	Original	Fine-Tuning	Prompt Tuning	ICIIA-B	ICIIA-T
iNaturalist 2019	EfficientNet-B0	61.82%	N/A	N/A	66.70% (+**4.88%**)	64.03% (+**2.21%**)
FEMNIST	CNN	88.48%	90.03%	88.47%	91.94% (+**3.46%**)	91.38% (+**2.89%**)
CelebA	EfficientNet-B0	90.84%	N/A	N/A	91.70% (+**0.85%**)	91.58% (+**0.74%**)
ImageNet-1K	MobileNetV3	75.26%	79.06%	83.93%	84.01% (+**8.75%**)	83.37% (+**8.11%**)
	ResNet-152	82.31%	83.54%	86.85%	88.16% (+**5.84%**)	87.90% (+**5.58%**)
	EfficientNet-B4	83.03%	83.65%	87.54%	88.31% (+**5.28%**)	88.35% (+**5.32%**)
	Swin-B	83.58%	84.45%	87.00%	89.23% (+**5.65%**)	88.86% (+**5.28%**)
	ConvNeXt-L	84.42%	84.84%	87.59%	89.31% (+**4.89%**)	89.14% (+**4.72%**)
	EfficientNet-B7	84.58%	84.98%	88.71%	89.36% (+**4.78%**)	89.37% (+**4.79%**)
UCF101	C3D	79.94%	79.96%	80.01%	81.09% (+**1.15%**)	80.87% (+**0.93%**)

Table 5: Drop of testing accuracy after removing or replacing different ingredients of ICIIA.

Dataset	ICIIA-B → CNN	ICIIA-B -attention	ICIIA-T -shuffling
iNaturalist 2019	-5.06%	-4.78%	-0.20%
ImageNet-1K	-3.14%	-6.50%	-0.28%

4.3. Comparison with Baselines

We compare ICIIA with the baselines and report the testing accuracy in Table 4. We can observe consistent and significant improvements of ICIIA over all the baselines on all the datasets and recognition models. (1) By comparing ICIIA-B with the original backbone model, we can draw that ICIIA indeed improves model performance by exploiting each client's class heterogeneity from the local samples to be recognized; (2) by comparing ICIIA-B with fine-tuning and prompt tuning, we can derive that the intra-client inter-image attention mechanism has the strongest adaptiveness, even without on-client re-training; (3) by comparing ICIIA with the original backbone on *iNaturalist 2019* and *CelebA*, where the clients in the testing dataset do not appear in the training dataset, we validate ICIIA's generalization ability to be trained on the cloud with only public data and adapt to the local data of unseen clients; and (4) by comparing ICIIA-T with ICIIA-B and the baselines, we can draw that although ICIIA-T compromises model accuracy for efficiency, it still significantly outperforms the baselines.

4.4. Ablation Study

Impact of intra-client and inter-image attention. We first replace the attention operation of ICIIA-B with a convolution operation across images, and reserve all the linear projections in both the original multi-head self-attention and the MLP layers. As shown in the first column of Table 5, the testing accuracy sharply drops on both datasets, demon-

strating that attention is indeed more effective for mining the intra-client and inter-image dependencies. We then remove the attention operation, and from the second column, we still observe a significant drop of the testing accuracy, validating the necessity of attention in ICIIA.

Impact of feature shuffling. We remove the feature shuffling operation from ICIIA-T and observe a slight drop of testing accuracy, as shown in the third column of Table 5. We also report the accuracy of ICIIA with and without feature shuffling by varying the number of feature groups P in supplementary materials, and see that the drop generally becomes larger for a larger P. These results validate the necessity of feature shuffling, especially together with a large P.

5. Conclusion

In this work, we study the ubiquitous image recognition applications from a new cloud-client perspective. We have proposed a plugable ICIIA module to adapt the backbone recognition model from the cloud's full set of classes to each individual client's local dynamic subset of classes, while preserving the mainstream cloud-based training workflow without on-client re-training. ICIIA captures intra-client and inter-image dependencies with multi-head self-attention and further improves efficiency by feature grouping and shuffling. We have extensively evaluated ICIIA, revealing effectiveness, efficiency, and superiority.

Acknowledgments

This work was supported by National Key R&D Program of China (No. 2022ZD0119100), China NSF grant No. 62025204, No. 62202296, and No. 62272293, Alibaba Innovation Research (AIR) Program, and SJTU-Huawei Explore X Gift Fund. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings, 2018.

Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication, 2018.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers, 2019.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Grant Van Horn and Oisin Mac Aodha. iNaturalist 2019. https://sites.google.com/view/fgvc6/competitions/inaturalist-2019, 2019.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *ICML*, 2019.

Andrew Howard, Ruoming Pang, Hartwig Adam, Quoc V. Le, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, and Yukun Zhu. Searching for mobilenetv3. In *ICCV*, 2019.

Fan Lai, Yinwei Dai, Sanjay Sri Vallabh Singapuram, Jiachen Liu, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. Fedscale: Benchmarking model and system performance of federated learning at scale. In *ICML*, 2022.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791. URL https://ieeexplore.ieee.org/document/726791.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP (1)*, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP (1)*, 2021.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. Layer-wised model aggregation for personalized federated learning. In *CVPR*, 2022.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

Luke Melas-Kyriazi. EfficientNet-PyTorch. https://github.com/lukemelas/EfficientNet-PyTorch, 2021.

George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. In *NeurIPS*, 2017.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild, 2012.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Yikai Yan, Chaoyue Niu, Renjie Gu, Fan Wu, Shaojie Tang, Lifeng Hua, Chengfei Lyu, and Guihai Chen. On-device learning for model personalization with large-scale cloud-coordinated domain adaption. In *KDD*, 2022.

Junjie Ye, Changhong Fu, Guangze Zheng, Danda Pani Paudel, and Guang Chen. Unsupervised domain adaptation for nighttime aerial tracking. In *CVPR*, 2022.

Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *ICML*, 2021.

Jianfeng Zhang. Pytorch-Video-Recognition. https://github.com/jfzhang95/pytorch-video-recognition, 2019.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *ICML*, 2021.