CollabAR: Edge-assisted Collaborative Image Recognition for Mobile Augmented Reality

Zida Liu
Duke University
zida.liu@duke.edu

Guohao Lan
Duke University
guohao.lan@duke.edu

Jovan Stojkovic
University of Belgrade
sj160151d@student.etf.bg.ac.rs

Yunfan Zhang
Duke University
yunfan.zhang@duke.edu

Carlee Joe-Wong
Carnegie Mellon University
cjoewong@andrew.cmu.edu

Maria Gorlatova
Duke University
maria.gorlatova@duke.edu

ABSTRACT

Mobile Augmented Reality (AR), which overlays digital content on the real-world scenes surrounding a user, is bringing immersive interactive experiences where the real and virtual worlds are tightly coupled. To enable seamless and precise AR experiences, an image recognition system that can accurately recognize the object in the camera view with low system latency is required. However, due to the pervasiveness and severity of image distortions, an effective and robust image recognition solution for mobile AR is still elusive. In this paper, we present CollabAR, an edge-assisted system that provides *distortion-tolerant image recognition* for mobile AR with *imperceptible system latency*. CollabAR incorporates both *distortion-tolerant* and *collaborative* image recognition modules in its design. The former enables distortion-adaptive image recognition to improve the robustness against image distortions, while the latter exploits the 'spatial-temporal' correlation among mobile AR users to improve recognition accuracy. We implement CollabAR on four different commodity devices, and evaluate its performance on two multi-view image datasets. Our evaluation demonstrates that CollabAR achieves over 96% recognition accuracy for images with severe distortions, while reducing the end-to-end system latency to as low as 17.8ms for commodity mobile devices.

CCS CONCEPTS

• **Computing methodologies** → **Distributed algorithms**; **Mixed / augmented reality**.

KEYWORDS

Edge computing, collaborative augmented reality, image recognition.

1 INTRODUCTION

Mobile augmented reality (AR), which overlays digital content with the real world around a user, has recently jumped from the pages of science fiction novels into the hands of consumers. To enable seamless contextual AR experience, an effective image recognition system that can *accurately recognize objects* in the camera view of the mobile device with *imperceptible latency* is required [1, 2]. While this may come across as a solved problem given the recent advancements in deep neural networks (DNNs) [3, 4] and mobile offloading [1, 5, 6], there are three practical aspects that have been largely overlooked in existing solutions.

First, in real-world mobile AR scenarios, a large proportion of images taken by the smartphone or the head-mounted AR device contains distortions. For instance, images frequently contain motion blur caused by the motion of the user [1, 7]. Gaussian blur appears when the camera is de-focusing, or is used in an underwater AR scenario [8]. Gaussian white noise is also inevitable in low illumination conditions [9]. Indeed, using two different commodity mobile AR devices, our measurement study (Section 3.3) indicates that *over 70% of the images can be corrupted by distortions* in different practical scenarios. Given the high distortion proportion, simply filtering out the distorted images, as has been suggested in prior work [1, 2, 10], can hardly solve the problem.

Second, for image recognition, applying DNNs trained on large scale datasets, e.g., ImageNet [11] and Caltech-256 [12], directly to mobile AR images that contain severe multiple distortions is difficult, and often results in dramatic performance degradation [9, 13, 14]. This deficiency results from the *domain adaptation problem*, where distorted images fail to share the same feature distribution with the clear training images [13, 15, 16]. Indeed, when testing distorted images with a pre-trained MobileNetV2 [4] network on the Caltech-256 dataset, we observe that even a small amount of distortion in the images can significantly affect the recognition accuracy (Section 3.2). Although deblurring methods and spatial filters [17] can be used to reduce distortions, they also remove the fine-scaled image details that are useful for image recognition.

Lastly, to achieve imperceptible recognition latency for resource-constrained mobile AR devices, a number of works have explored computation offloading [1, 5, 18] and approximate computation reuse [6, 19] to reduce the system overhead. Although the state-of-the-art solutions have reduced the end-to-end system latency to below 33ms [5, 18], such performance is achieved in distortion-free or distortion-modest environments. The challenge in mitigating the trade-off between recognition accuracy and system latency for mobile AR with severe multiple distortions has not been addressed.

To fill this the gap, we present CollabAR, an edge-assisted Collaborative image recognition framework for mobile Augmented Reality. Due to recent advances in AR development platforms, AR applications have become accessible on a wide range of mobile devices. We have seen many dedicated AR devices, such as Microsoft HoloLens [20] and Magic Leap [21], as well as Google ARCore [22] and Apple ARKit [23] SDKs that work on a wide range of mobile phones and tablets. The pervasive deployment of mobile AR will offer numerous opportunities for multi-user collaboration [10, 24]. Moreover, mobile users that are in close proximity usually exhibit

978-1-7281-5497-8/20/$31.00 ©2020 IEEE
DOI 10.1109/IPSN48710.2020.00-26

a strong correlation in both device usage and spatial-temporal context [25]. The requested image recognition tasks are usually correlated temporally or spatially. Thus, unlike conventional image recognition systems where individual users independently complete their tasks to recognize the same object [1, 5, 18], CollabAR embraces the concept of *collaborative image recognition* in its design, and exploits the *temporally and spatially correlated* images captured by the users to improve the image recognition accuracy.

Bringing this high-level concept into a holistic system requires overcoming several challenges. First, we need to address the domain adaptation problem [15] caused by the image distortions. As DNNs can adapt to a particular distortion type, but not multiple types at the same time [9, 15], we need to adaptively select a DNN that has been fine-tuned to recognize a specific type of distorted images in runtime. We introduce the *distortion-tolerant image recognizer* which incorporates the *image distortion classifier* and *a set of dedicated recognition experts*. CollabAR employs the image distortion classifier to identify the most significant distortion contained in the image, and triggers one of the dedicated recognition experts to handle the distorted image. This distortion-adaptive selection of a recognizer ensures a lower feature domain mismatch between the image and the DNN, and a better robustness against distortions.

Second, to enable collaborative image recognition, we need to identify a set of spatially and temporally correlated images that contain the same object. Prior works rely on image feature vectors [19] or feature maps [6] to identify temporally correlated images (e.g., continuous frames in a video). However, image features are vulnerable to distortions, and result in high lookup errors in practical AR scenarios. CollabAR introduces the *anchor-based pose estimation* and the *spatial-temporal image lookup module* to efficiently identify correlated images.

Third, as the correlated images suffer from various distortions with different severity levels, they lead to heterogeneity in the recognition confidence and accuracy. Conventional multi-view image recognition systems [26, 27] lack a reliable metric to differentiate the heterogeneity, and are not able to optimally aggregate the multi-view results [28]. In this work, we propose the Auxiliary-assisted Multi-view Ensemble Learning (AMEL) to dynamically aggregate the heterogeneous recognition results of the correlated images.

Lastly, having all the system building blocks in mind, we aim to provide imperceptible system latency (i.e., below 33ms to ensure 30fps continuous recognition), without sacrificing accuracy. The targeted low system latency can leave more time and space for many other computation-intensive tasks (e.g., virtual object rendering) that are running on mobile AR devices. We explore several design options (i.e., selection of DNN models, what to offload and what to process locally) and conduct a comprehensive system profiling to guide the final optimized implementation. We propose an edge-assisted system framework to mitigate the trade-off between recognition accuracy and system latency.

In this paper, we presents CollabAR, a collaborative image recognition system for mobile AR. Our main contributions are:

- We propose the *distortion-tolerant image recognizer* to resolve the domain adaptation problem caused by image distortions. Compared to conventional methods, the proposed solution improves the recognition accuracy by 20% to 60% for different settings.
- To further boost the recognition accuracy, CollabAR embraces collaboration opportunities among mobile AR devices, and exploits the spatial-temporal correlation among images for multi-view image recognition. CollabAR introduces the *anchor-based image lookup module* to effectively identify the spatially and temporally correlated images, and the *auxiliary-assisted multi-view ensemble learning framework* to dynamically aggregate the heterogeneous multi-view results. Compared to the single-view-based recognition, the proposed method can improve the recognition accuracy by 16% to 20% in severe multiple distortions scenario.
- We implement CollabAR on four different commodity devices, and evaluate its performance on two multi-view image datasets, one public and one collected by ourselves. The evaluation demonstrates that CollabAR achieves over 96% recognition accuracy for images that contain severe multiple distortions, while reducing the end-to-end system latency to as low as 17.8ms.

The research artifacts, including our own collected multi-view multiple distortions image dataset (Section 7), and the source codes for both the distortion-tolerant image recognizer and the auxiliary-assisted multi-view ensemble learning framework, are available at https://github.com/CollabAR-Source/.

2 RELATED WORK

As a holistic image recognition system for multi-user mobile AR, CollabAR builds on a body of prior work in mobile AR, distortion-tolerant image recognition, and distributed image recognition.

Image Recognition for Mobile AR. Before overlaying the rendered virtual objects on the view of a user, mobile AR systems leverage image recognition to identify the object appearing in the view. For this purpose, image retrieval [1, 18], image localization [2], and DNN-based image recognition methods [5] are widely used in prior work. Image retrieval-based solutions, such as OverLay [1] and Jaguar [18], exploit salient feature descriptors to match image in with the annotated image stored in the database. However, salient descriptors are vulnerable to image distortions; a large proportion of images cannot be correctly recognized but is simply filtered out [1]. Image localization-based methods require the server to maintain the annotated image dataset of the service area [29], which is computationally heavy and involves a large volume of data [2]. Lastly, DNN-based solutions [5] perform well with pristine images, but even a small amount of distortion can lead to a dramatic performance drop [9, 13]. Instead of filtering out the distorted images, CollabAR incorporates the distortion-tolerant image recognizer to enable image recognition for mobile AR systems.

Distortion-tolerant Image Recognition: Recent efforts have been made by the computer vision community in resolving the negative impacts of image distortion on DNNs [9, 13, 14, 16]. Dodge et al. [9] propose a mixture-of-experts-based model, in which recognition experts are fine-tuned to handle a specific image distortion, and their outputs are aggregated using a gating network. In [13], Ghosh et al. propose a similar master-slave CNN architecture for distorted image recognition. Brokar et al. [14] propose an objective metric to identify the most distortion-susceptible convolutional filters in the CNNs. Correction units are added in the network to correct the outputs of the identified filters. However, prior works mainly focus on the single-view scenario, and fail when the image contains

multiple distortions or the distortion level is high [9, 13, 14]. In contrast, CollabAR can dynamically aggregate the spatially and temporally correlated images to improve recognition accuracy in multiple distortions scenario.

Distributed Image Recognition: Using distributed information collected from multi-camera setups for image and object recognition is a widely studied topic. For instance, in DDNNs [27], distributed deep neural networks are deployed over cloud, edge, and end devices for joint object recognition. By leveraging the geographical diversity of a multi-camera network, DDNNs improves the object recognition accuracy. To improve the accuracy for distortion-tolerant pill image recognition, MobileDeepPill [30] relies on a multi-CNN model to collectively capture the shape, color, and imprint characteristics of pills. Kestrel [31] incorporates a video analytics system that detects and tracks vehicles across heterogeneous multi-camera networks. These existing systems only perform in distortion-free [27] or distortion-modest environments [30]. They are not able to recognize the image if it contains severe or multiple distortions. Moreover, many of them suffer from high end-to-end latency (i.e., 200ms) [31].

3 BACKGROUND AND CHALLENGES

3.1 Image Distortions

In this paper, we consider three different types of image distortion that commonly appear in mobile AR scenarios: motion blur, Gaussian blur, and Gaussian noise.

Motion blur (MBL): images taken by the smartphone or the head-mounted AR set camera frequently contain motion blur caused by the motion of the user [1]. Given a pristine image $F(x, y)$ with pixel coordinates x and y, the corresponding motion-blurred image, $F_{MBL}(x, y)$, can be modeled as $F_{MBL}(x, y) = F(x, y) * H(l, a)$ [16], where $H(l, a)$ is the non-uniform motion blur kernel that characterizes the length (l) and orientation (a) of the motion when the camera shutter is open, and $*$ is the convolution operator. The distortion level of $F_{MBL}(x, y)$ is determined by the length of the blur kernel l, while the angle a defines the direction in which the motion blur affects the image. Figure 1(a) shows the pristine image (l=0) and four motion-blurred images with different kernel lengths (the angle a is randomly selected). Motion blur dramatically reduces the sharpness of the image and distorts its spatial features.

Gaussian blur (GBL): Gaussian blur appears when the camera is de-focusing or the image is taken underwater (e.g., in underwater AR [8]) or in a foggy environment [17]. For a pristine image $F(x, y)$, the corresponding Gaussian-blurred image, $F_{GBL}(x, y)$, can be modeled as $F_{GBL}(x, y) = F(x, y) * G_\sigma(x, y)$ [17], where $G_\sigma(x, y)$ is the two-dimensional circularly symmetric centralized Gaussian kernel. The distortion level of $F_{GBL}(x, y)$ is determined by the aperture size k of the circular Gaussian kernel. Figure 1(b) shows an example of images containing different levels of Gaussian blur. Note that k must be odd to maintain the circularly symmetric property.

Gaussian noise (GN): noise is also inevitable in images. Possible sources of noise include poor illumination conditions [9], digital zooming, and the use of a low quality image sensor [32]. In most cases, noise can be modeled as zero-mean additive Gaussian noise [32]. Given a pristine image $F(x, y)$, the corresponding noisy image is expressed as $F_{GN}(x, y) = F(x, y) + N(x, y)$, where $N(x, y)$ is the additive noise which follows a zero-mean Gaussian distribution. The distortion level of $F_{GN}(x, y)$ is decided by the standard deviation σ_{GN}. As shown in Figure 1(c), the Gaussian noise adds random variation in both brightness and color of the image.

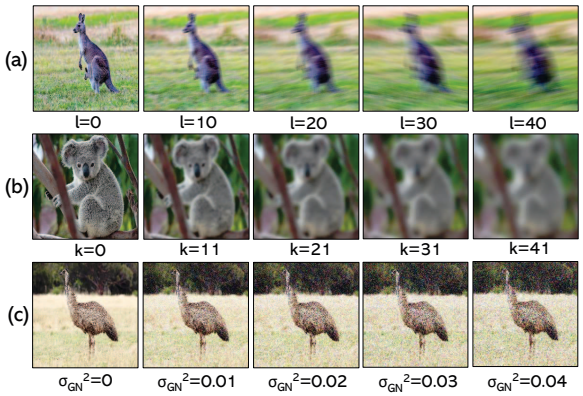


Figure 1: Examples of images containing different types of distortions at different severity levels: (a) motion blur with blur kernel length l, (b) Gaussian blur with aperture size k, and (c) Gaussian noise with variance σ^2_{GN}.

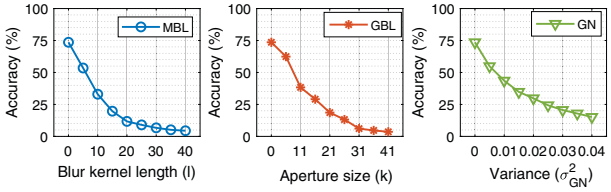


Figure 2: Impacts of image distortion on the recognition performance of MobileNetV2 with the Caltech-256 dataset.

3.2 Impact of Distortion on Image Recognition

Deep neural networks (DNN), such as the AlexNet [3] and the MobileNetV2 [4], have shown state-of-the-art performance in image recognition. However, as DNN models are usually trained and tested on the images that are pristine, e.g., ImageNet [11], directly applying them to mobile AR images that contain severe multiple distortions often leads to dramatic performance degradation [14, 17, 33]. This deficiency results from the domain adaptation problem where real world distorted images fail to share the same feature distribution with the pristine training images [13, 15, 16].

As a case study to quantify this impact, we investigate the image recognition accuracy of MobileNetV2 on distorted images. We pre-train the MobileNetV2 using the images from the Caltech-256 [12] dataset, and synthesize distorted images using the methods introduced in Section 3.1. For each type of distortion, we adjust the distortion level by configuring the corresponding parameter. The synthesized images are used as the test dataset. The results are shown in Figure 2. We can observe that *even a small amount of distortion in the images could significantly affect the recognition accuracy*. For instance, the accuracy of MobileNetV2 drops from 73.6% to 33.0% with a blur kernel length of 10, even though the distortion at this level does not hinder the human eyes' ability to recognize the object [34] (e.g., see an example in Figure 1(a)).

Table 1: Measurements of image distortions in real-world mobile AR scenarios. A large proportion of images suffers from severe distortions.

Distortion	Setting	Hardware	
		Nokia 7.1	Magic Leap One
Gaussian noise ($\sigma^2_{GN} \geqslant 0.003$)	Dark room (7lux)	617/1558=39.6%	infeasible
	Camera zoom-in (509lux)	1755/2185=97.2%	infeasible
Motion blur ($l \geqslant 5$)	Corridor (178lux)	2681/3452=77.6%	3760/3776=99.6%
	Sunny outdoor (9873lux)	16/2687=0.5%	957/2766=34.6%
Gaussian blur ($k \geqslant 5$)	Foggy	762/935=81.6%	infeasible
	Underwater	1250/1524=82.0%	infeasible

3.3 Image Distortion for Real-world Mobile AR

Below, we quantify the extent of image distortion in real-word mobile AR scenarios. We use two commodity AR devices, the Nokia 7.1 smartphone and the Magic Leap One head-mounted AR set, to record videos (at 30 frames per second) in different environments, and measure the proportion of recorded frames that suffer from severe image distortions.

First, to quantify *Gaussian noise*, we use the Nokia 7.1 to record videos in environments with poor illumination conditions, i.e., indoor dark room (with 7lux). In addition, to study Gaussian noise resulting from digital zooming, we conduct an experiment in an indoor environment at daytime (with 509lux), and turn on the built-in digital zoom function of the Nokia 7.1 to record videos of objects that are three meters away. We cannot record video using Magic Leap One as its image and video recording service is unavailable when the light condition is low, and it does not support image zooming. Second, to quantify *motion blur*, we ask one user to hold the Nokia 7.1 in her hand and ask another user to wear the Magic Leap One. They record videos using the device front-camera in both indoor corridor (with 178lux) and sunny outdoor (with 9873lux) environments when walking at a normal pace. Lastly, to quantify *Gaussian blur*, we put the Nokia 7.1 in a waterproof case, and record videos in both underwater and foggy environments.[1] The collected images comprise the MobileDistortion dataset, which is available at https://github.com/CollabAR-Source/Distortion.

To determine if an image suffers from severe distortion, we use distortion levels $\sigma^2_{GN} = 0.003$, $l = 5$, and $k = 5$ as the thresholds, for the three types of distortion, respectively. The thresholds correspond to the distortion levels that lead to a 10% accuracy drop in the experiments shown in Figure 2. The results are summarized in Table 1. The illumination of the environment is measured using the smartphone light sensor that is adjacent to the smartphone camera. First, we apply the image spatial quality estimator [35] to quantify Gaussian noise. In poor illumination conditions, over 39% of the images recorded by the Nokia 7.1 suffer from severe Gaussian noise. Moreover, when using digital zooming, 97% of the images suffer from severe Gaussian noise due to the additive noise introduced in digital image processing. Second, we apply the partial-blur image detector [36] to quantify motion blur. As shown in Table 1, for Magic Leap One, 99.6% and 34.6% of the recorded images suffer from severe motion blur in the corridor and the outdoor environments, respectively. For Nokia 7.1, 77.6% and 0.5% of the images contain severe motion blur in these two environments. We can notice that there is less motion blurring in the sunny outdoor environment (9873lux) than in the corridor (178lux): as the camera exposure time is shorter with a higher illumination, and the impact of motion on the image is proportional to the exposure time, there is less motion blurring in high illumination environment. Lastly, we apply the local binary pattern-based blur detector [37] to detect Gaussian blur. The results show that 82% and 81.6% of the images suffer from severe Gaussian blur in the underwater and foggy environments, respectively. Our measurements highlight the pervasiveness and severity of image distortions in real-world scenarios. Given the high distortion proportion, simply filtering out the distorted images, as has been suggested in prior work [1, 2, 10], can hardly solve the distortion problem for practical AR applications.

[1]We do not conduct this experiment with a Magic Leap One as we are unaware of the availability of waterproof cases for it.

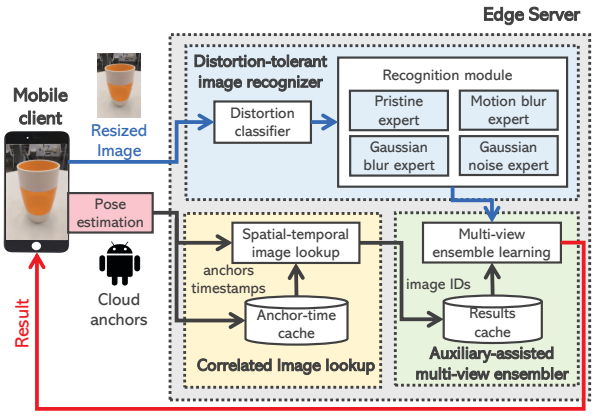


Figure 3: System architecture of CollabAR.

4 SYSTEM OVERVIEW

The architecture of CollabAR is shown in Figure 3, which incorporates three major components: (1) the distortion-tolerant image recognizer, (2) the correlated image lookup module, and (3) the auxiliary-assisted multi-view ensembler. They are deployed on the edge server to mitigate the trade-off between recognition accuracy and end-to-end system latency (detailed system measurements are given in Section 8.4). In addition to the server, the client is running an anchor-based pose estimation module which leverages the Cloud anchors provided by the Google ARCore [22] to track the location and orientation of the mobile device.

When the user takes an image of the object, the mobile client sends the image along with the IDs of the anchors in the camera view to the server. Having received the data, the server marks the image with a unique image ID and a timestamp. The image is used as the input for the distortion-tolerant image recognizer, while the anchor IDs and the timestamp are used as references for spatial-temporal image lookup. In CollabAR, anchor IDs and timestamps are stored in the *anchor-time cache* in the format of $< \text{imageID}, \{\text{anchorIDs}\}, \text{timestamp} >$, while the recognition results of the images are stored in the *results cache* in the format of $< \text{imageID}, \text{inferenceResult} >$ for future reuse and aggregation. In our design, we do not share information among the clients. Instead, clients communicate directly with the edge server to ensure high recognition accuracy and low end-to-end latency.

Distortion-tolerant image recognizer: the distortion image recognizer incorporates an image distortion classifier (Section 5.1) and four recognition experts (Section 5.2) to resolve the domain

adaptation problem [15] caused by the image distortions. As DNNs can adapt to a particular distortion, but not multiple distortions at the same time [9, 15], we need to identify the most significant distortion in the image, and adaptively select a DNN that is dedicated to the detected distortion. As shown in Figure 3, for a new image received from the client, the distortion classifier detects if the image is pristine or if it contains any type of distortion (i.e., motion blur, Gaussian blur, or Gaussian noise). Then, based on the detected distortion type, one of the four recognition experts is triggered for the recognition. Each of the recognition experts is a Convolutional Neural Network (CNN) that has been fine-tuned to recognize a specific type of distorted images. The inference result of the expert is sent to the auxiliary-assisted multi-view ensembler for aggregation, and is stored in the results cache for reuse.

Correlated image lookup: to enable collaborative image recognition, the correlated image lookup module (Section 6.1) searches the Anchor-time cache to find previous images that are spatially and temporally correlated with the new one. The anchor IDs are used as the references to identify the spatial correlation, while the timestamps are used to determine the temporal correlation. The correlated images could be the images that are captured by different users or the continuous image frames that are obtained by a single device. The IDs of the correlated images are forwarded to the multi-view ensembler.

Auxiliary-assisted multi-view ensembler: based on the identified image IDs, the multi-view ensembler retrieves the inference results of the correlated images from the results cache. Together with the recognition result of the new image, the retrieved results are aggregated by the multi-view ensemble learning module to improve the recognition accuracy. Specifically, as the correlated images suffer from various distortions with different severity levels, they are unequal in quality and lead to heterogeneity in the recognition accuracy. CollabAR leverages the Auxiliary-assisted Multi-view Ensemble Learning (AMEL) (Section 6.2) to dynamically aggregate the heterogeneous results provided by the correlated images. Below, we describe the design of CollabAR in detail.

5 DISTORTION-TOLERANT RECOGNIZER

5.1 Image Distortion Classifier

Different types of distortion have distinct impacts on the frequency domain features of the original images. For instance, as discussed in Section 3.1, Gaussian blur can be considered as having a two-dimensional circularly symmetric centralized Gaussian convolution on the original image in the spatial domain. This is equivalent to applying a Gaussian-weighted, circularly shaped low pass filter on the image, which filters out the high frequency components in the original image. Similarly, motion blur can be considered as a Gaussian-weighted, directional low pass filter that smooths the original image along the direction of the motion [7]. Lastly, the Fourier transform of additive Gaussian noise is approximately constant over the entire frequency domain, whereas the original image contains mostly low frequency information [38]. Hence, an image with severe Gaussian noise will have higher signal energy in the high frequency components.

Figure 4 gives the Fourier spectrum of images containing different distortions. The spectrum is shifted such that the DC-value is

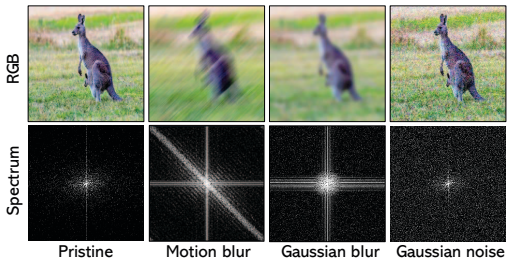


Figure 4: The Fourier spectrum of images containing different distortions. Different distortions have distinct impacts on the frequency domain features of the images. The patterns are independent of the semantic content in the image.

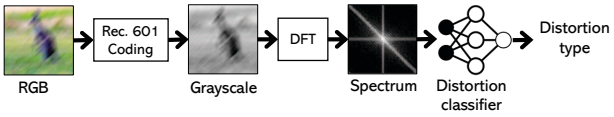


Figure 5: The pipeline of the image distortion classification.

displayed in the center of the image. For any point in the spectrum, the farther away it is from the center, the higher is its corresponding frequency. The brightness of the point in the spectrum indicates the energy of the corresponding frequency component in the image. We can see that the pristine image contains components of all frequencies, and most of the high energy components are in the lower frequency domain. Differently, Gaussian blur removes the high frequency components of the image in all directions. The remaining two dominating lines in the spectrum, one passing vertically and one passing horizontally through the center of the spectrum, correspond to the vertical and horizontal patterns in the original image. Similarly, motion blur removes the high frequency components in the image along the direction of motion. Thus, in addition to the vertical and horizontal lines, we can see a strong diagonal line which is perpendicular to the direction of the motion blur. Lastly, Gaussian noise introduces more high energy components in the high frequency domain. We leverage these distinct frequency domain patterns for distortion classification.

Figure 5 shows the pipeline of the image distortion classification. First, we convert the original RGB image into grayscale using the standard Rec. 601 luma coding method [39]. Then, we apply the two-dimensional discrete Fourier transform (DFT) to obtain the Fourier spectrum of the grayscale image, and shift the zero-frequency component to the center of the spectrum. The centralized spectrum is used as the input for the distortion classifier. As shown in Table 2, the distortion classifier adopts a shallow CNN architecture which consists of three convolutional layers (*conv1*, *conv2*, and *conv3*), two pooling layers (*pool1* and *pool2*), one flatten layer, and one fully connected layer (*fc*). This shallow design avoids over-fitting and ensures low computation latency.

5.2 Recognition Experts

Based on the output of the distortion classifier, one of the four dedicated recognition experts is selected for the image recognition. We consider either the lightweight MobileNetV2 or the AlexNet as the base DNN model for the recognition experts. Although deeper models (e.g., ResNet [40] and VGGNet [41]) can achieve higher

Table 2: The architecture of the image distortion classifier.

Layer	Size In	Size Out	Filter
conv1	$224 \times 224 \times 3$	$111 \times 111 \times 32$	3×3, 2
conv2	$111 \times 111 \times 32$	$109 \times 109 \times 16$	3×3, 1
pool1	$109 \times 109 \times 16$	$54 \times 54 \times 16$	2×2, 2
conv3	$54 \times 54 \times 16$	$54 \times 54 \times 1$	1×1, 1
pool2	$54 \times 54 \times 1$	$27 \times 27 \times 1$	2×2, 2
flatten	$27 \times 27 \times 1$	729	
fc	729	4	

accuracy on pristine images, they are still susceptible to image distortions [9, 42]. Recent studies have shown more than 40% accuracy drop on ResNet [42] and VGGNet [9] when the images contain distortion. In contrast, CollabAR ensures better robustness against distortions while maintaining a lower computation latency. We initialize all the CNN layers with the values trained on the full ImageNet dataset. Then, to fine-tune the experts on a target dataset, we add a fully connected layer as the last layer with the number of units corresponding to the number of classes in the target dataset.

In the fine-tuning process, we first train the pristine expert, $Expert_P$, using pristine images in the target dataset. Then, the three distortion experts, i.e., motion blur expert $Expert_{MBL}$, Gaussian blur expert $Expert_{GBL}$, and Gaussian noise expert $Expert_{GN}$, are initialized with the weights of $Expert_P$, and fine-tuned using the corresponding distortion images. During the fine-tuning, half of the images in the mini-batch are pristine and the other half are distorted with a random distortion level. This ensures better robustness against variations in the distortion level (i.e., it helps in minimizing the effect of domain-induced changes in feature distribution [15]) and helps the CNNs to learn features from both pristine and distorted images [43]. For $Expert_{MBL}$, the distortion level is configured by varying the blur kernel length l and the motion angle a. Their values are randomly selected from the uniform distributions where $l \sim U(0, 40)$ and $a \sim U(0, 180)$. Similarly, for $Expert_{GBL}$, the distortion level is controlled by the aperture length $k \sim U(0, 41)$. Lastly, for $Expert_{GN}$, the distortion level is decided by the variance of the additive Gaussian noise $\sigma^2_{GN} \sim U(0, 0.04)$.

6 COLLABORATIVE MULTI-VIEW IMAGE RECOGNITION

Mobile users that are in close proximity usually exhibit a strong correlation in both device usage and spatial-temporal contexts [19, 25]. This is especially true in mobile AR scenario where the requested image recognition tasks are usually correlated temporally (e.g., consecutive image frames from the same user [5, 6, 44]) and spatially (e.g., different users aim to recognize the same object [10, 19]). For instance, in a museum, visitors that are in *close proximity* may use the AR application to recognize and augment information of the *same artwork*. Although images are taken from different positions or at different times, they contain the same object [19]. CollabAR exploits the spatial-temporal correlation among the images to enable collaborative multi-view image recognition. Note that CollabAR is not limited to the multi-user scenario. It can also aggregate the consecutive image frames captured by a single user as long as they are spatially and temporally correlated.

6.1 Correlated Image Lookup

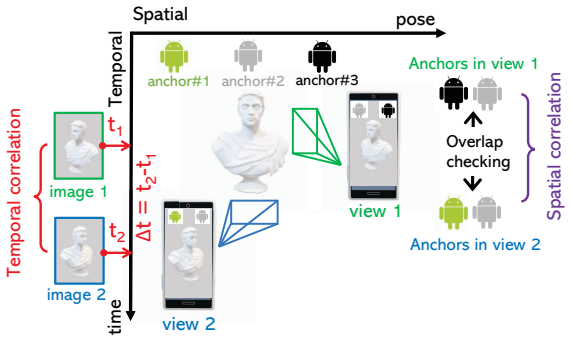


Figure 6: An example of correlated image lookup. Images of the same artwork are taken from different poses, $view1$ and $view2$, at different times, t_1 and t_2. The anchors in the image views are used to check the spatial correlation, and the timestamps are used to check the temporal correlation.

6.1.1 Anchor-based Pose Estimation. To identify the spatially and temporally correlated images, CollabAR exploits the Google ARCore [22] to estimate the pose (the position and orientation) of the device when the image was taken. When the user is moving in space, ARCore leverages a process called concurrent odometry and mapping (COM) which aggregates the image frames taken by the camera and the inertial data captured by the device's motion sensors to simultaneously estimate the orientation and location of the mobile device relative to the world.

Specifically, CollabAR leverages the Cloud anchors [45] provided by ARCore as the references for pose estimation. An anchor is a virtual identifier which describes a fixed pose in the physical space. The numerical pose estimation of the anchor is updated by the ARCore over time, which makes the anchor a stable pose reference against cumulative tracking errors [2, 46]. In our design, anchors are created and managed by the administrator in advance. When running the application, anchors in the space can be resolved by the users to establish a common spatial reference among them. As a toy example shown in Figure 6, three cloud anchors have been placed in space to identify three different poses. When the user is taking an image of the artwork from $view1$ at time t_1, the anchors in the camera view will be sent to the edge and associated with the timestamp t_1. As shown previously in Figure 3, the anchors and the timestamps are stored in the *anchor-time cache* in the format of $<$ imageID, {anchorIDs}, timestamps $>$. When a new image, $image2$, is taken at time t_2 from $view2$ either by the same or by a different user, the *spatial-temporal image lookup module* searches the anchor-time cache to find previous images that are spatially and temporally correlated with the new one.

6.1.2 Spatial-temporal Image Lookup. For any image in the cache, $image_{Cached}$, it is considered as spatially correlated with the new image, $image_{New}$, if the images satisfy:

$$\{\text{anchorIDs}\}_{\text{New}} \cap \{\text{anchorIDs}\}_{\text{Cached}} \neq \emptyset, \tag{1}$$

where $\{\text{anchorIDs}\}_{\text{New}}$ and $\{\text{anchorIDs}\}_{\text{Cached}}$ are the sets of anchors that appeared in views of $image_{New}$ and $image_{Cached}$, respectively. If two images contain the same anchor in their views, they are spatially correlated. For instance, in Figure 6, $image1$ contains the same anchor, $anchor\#2$, that also appears in the view of

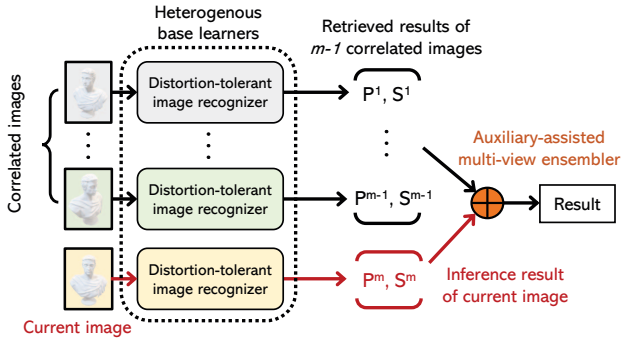


Figure 7: The architecture of the Auxiliary-assisted Multi-view Ensemble Learning (AMEL).

*image*2. Thus, they are spatially correlated. Similarly, the cached image is considered as temporally correlated to the new image if the images satisfy:

$$\Delta t = t_{New} - t_{Cached} \quad \text{and} \quad \Delta t \leqq \mathsf{T}_{\text{fresh}}, \tag{2}$$

where t_{New} and t_{Cached} are the timestamps of the new and cached images, respectively. Δt is the freshness of $image_{Cached}$ with respect to $image_{New}$. If the freshness Δt is within the freshness threshold, $\mathsf{T}_{\text{fresh}}$, $image_{Cached}$ is considered as temporally correlated with $image_{New}$. Note that the setting of the freshness threshold $\mathsf{T}_{\text{fresh}}$ varies with the application scenarios. In cases where the object at a given position changes frequently, e.g., animals in the zoo, we should use a lower freshness threshold. Differently, in scenarios where the positions of the objects do not change very often, e.g., exhibited items in a museum or large appliances in a building, a higher freshness threshold can be tolerated. An image in the anchor-time cache is considered as spatially and temporally correlated to the new image if they satisfy Equations 1 and 2 at the same time. The IDs of the identified correlated images are forwarded to the multi-view ensembler for aggregation.

6.2 Auxiliary-assisted Multi-view Ensemble Learning

In image recognition, ensembling the results of multiple base classifiers is a widely used approach to improve the overall recognition accuracy [27, 47]. Following this trend, CollabAR aggregates the recognition results of the spatially and temporally correlated images to improve the recognition accuracy of the current image.

As shown previously in Figure 3, using the image IDs identified by the correlated image lookup module as the key, the multi-view ensembler retrieves the results of the correlated images from the results cache. This is done by retrieving any stored records in the result cache, i.e., < imageID, inferenceResult >, with imageID matches to the identified images. As shown in Figure 7, assuming that $m-1$ correlated images are identified, the inference result of the current image (the probability vector $\mathcal{P}^m$) is aggregated with that of the $m-1$ correlated images ($\mathcal{P}^1, ..., \mathcal{P}^{m-1}$) by the ensembler.

However, given the heterogeneity of the m images (i.e., images are captured from different angles, suffer from different distortions with different distortion levels), the images lead to unequal recognition performance. To quantify their performance and help the ensembler in aggregating the results dynamically, auxiliary features [28, 48] can be used. We propose to use the **normalized entropy** as the auxiliary feature. The normalized entropy $\mathcal{S}^k$ measures the recognition quality and the confidence of the distortion-tolerant image recognizer on the recognition of the kth image instance. $\mathcal{S}^k$ can be calculated from the probability vector as follows:

$$\mathcal{S}^k(\mathcal{P}^k) = -\sum_{i=1}^{|C|} \frac{p_i \log p_i}{\log |C|}, \tag{3}$$

where $\mathcal{P}^k = \{p_1, ..., p_{|C|}\}$ is the probability vector of the kth base learner on the image instance, and $|C|$ is the number of object categories in the dataset. The value of normalized entropy is between 0 and 1. A value close to 0 indicates that the distortion-tolerant image recognizer is confident about its recognition on the image instance, whereas a value close to 1 means that it is not confident.

In AMEL, the set of normalized entropy, $(\mathcal{S}^1, ..., \mathcal{S}^m)$, is used as the weight in averaging heterogeneous outputs of the spatial-temporal correlated images. Specifically, given the current image and the $m-1$ correlated images, the final aggregated recognition output $\mathbf{P}$ can be expressed as $\mathbf{P} = \sum_{i=1}^{m}(1-\mathcal{S}^i)\mathcal{P}_i$, where $\mathcal{P}_i$ is the probability vector of the ith image and $(1-\mathcal{S}^i)$ is the associated weight. This allows AMEL to dynamically adjust the weights in aggregating the m images during the classification. As the images result in unequal recognition accuracy, AMEL is more robust against this variance than standard averaging methods which would assign equal weights to multi-view images during the aggregation [26]. We evaluate AMEL in Section 8.3.3.

Table 3: Summary of the MVMDD dataset. It has 32,400 images in total, including 1,296 pristine and 31,104 distorted images that are generated using data augmentation.

Pristine image set	Object categories	6
	Number of views	6
	Background complexity	2
	Size of object in image	3
	Number of instances	6
Total pristine images	6×6×2×3×6=1,296	
Augmented image set	Types of distortion	3
	Distortion levels	8
Total augmented images	1,296×3×8=31,104	
Total images	32,400	

7 MULTI-VIEW MULTI-DISTORTION IMAGE DATASET

We create a Multi-View Multi-Distortion image Dataset (MVMDD) to study the impact of image distortion on multi-view AR. Our dataset includes a pristine image set and an augmented distortion image set. The details are summarized in Table 3.

Pristine image set: we collect pristine images (i.e., images without distortion) using a commodity Nokia 7.1 smartphone. Six categories of everyday objects are considered, *cup*, *phone*, *bottle*, *book*, *bag*, and *pen*. Each category has *six instances*. For each instance, images are taken from *six different views* (six different angles with a 60° angle difference between any two adjacent views), *two different background complexity levels* (a clear white table background and a noisy background containing other non-target objects), and *three distances*. We adjust the distance between the camera and

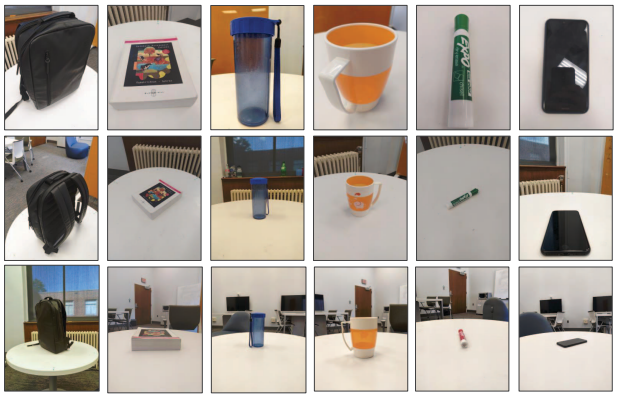


Figure 8: Examples of the pristine images that are collected in our MVMDD dataset.

the object such that the sizes of the object in the images are different. For the three distances, the object occupies approximately the whole, half, and one-tenth of the total area of the image. The resolution of the images is 3024×4032. As summarized in Table 3, we collect 6×6×6×3×2 = 1,296 pristine images in total. Figure 8 provides examples of the collected images.

Data augmentation for image distortion: to ensure robustness against image distortion, a large volume of distorted images is required to train the DNNs. However, in practice, it is difficult to collect a large number of images with different types of distortions. To address this challenge, we apply the three distortion models introduced in Section 3.1 to synthesize images with different distortions. We consider eight severity levels for each distortion type. Specifically, for motion blur, we adjust the motion blur kernel length l from 5 to 40, with a step size of 5, to create eight levels of blur. Similarly, we configure the aperture size of the 2D circularly symmetric centralized Gaussian function from 6 to 41, with a step size of 5, for the Gaussian blur, and change the variance of the zero-mean Gaussian distribution from 0.005 to 0.04, with a step size of 0.005, for the Gaussian noise. As summarized in Table 3, we generate 1,296×3×8=31,104 distorted images in total. The dataset is publicly available at https://github.com/CollabAR-Source/MVMDD.

8 EVALUATION

8.1 Experimental Setup

8.1.1 Implementation. The client of CollabAR is implemented on Android smartphones. We leverage the Google ARCore SDK to realize the anchor-based pose estimation module introduced in Section 6.1.1. The edge server is a desktop with an Intel i7-8700k CPU and a Nvidia GTX 1080 GPU. We realize the server of CollabAR using the Python Flask framework. The distortion-tolerant image recognizer and the multi-view ensembler are implemented using Keras 2.3 on top of the TensorFlow 2.0 framework. The client and the server communicate through the HTTP protocol.

8.1.2 Benchmark dataset. Four datasets are considered in the evaluation: Caltech-256 [12], MIRO [49], MobileDistortion (collected in Section 3.3), and MVMDD (collected in Section 7). The **Caltech-256** dataset has 257 categories of objects with more than 80 instances per category. The **MIRO** dataset has 12 categories of objects with ten instances per category. For each instance, it contains multi-view images that are captured from ten elevation angles and 16 azimuth angles. In the evaluation, we randomly select six distinct angles to represent six different views. Our own **MVMDD** dataset has six categories of objects with six instances per category. Each object instance is captured from six views and three different distances, with two different backgrounds. In addition to the original pristine images, for both Caltech-256 and MIRO, we apply the data augmentation methods (Section 3.1) to generate new sets of distorted images. Lastly, we also prepare 300 image instances for each distortion type from the **MobileDistortion** image set we collected in Section 3.3. We use it to examine the distortion classifier on distorted images in real-world mobile AR scenarios.

8.2 Distortion Classifier Performance

First, we evaluate the performance of the image distortion classifier using all four benchmark datasets. We perform 3-fold cross validation on each of the four datasets.

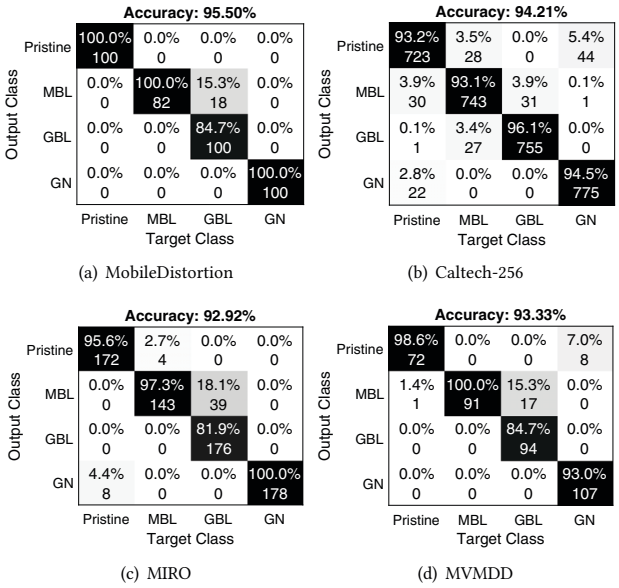
The confusion matrix and the accuracy of the image distortion classifier are shown in Figure 9. The classifier achieves 95.5%, 94.2%, 92.9%, and 93.3% accuracy on the MobileDistortion, Caltech-256, MIRO, and MVMDD datasets, respectively. Moreover, the confusion matrix indicates that most of the classification errors happen in differentiating Motion blur (MBL) and Gaussian blur (GBL). As shown in Figure 9, for the four different datasets, 15.3%, 3.9%, 18.1%, and 15.3% of the GBL distorted images are misclassified as MBL. In our experiments we observe that MBL and GBL can be misclassified as each other regardless of the distortion level. This is due to the similarity between these two distortions. Specifically, MBL and GBL have similar impact on the image spectrum and result in misclassification when the direction of the motion blur is parallel or perpendicular to the central horizontal line of the image. In addition, we also notice that when the distortion level is low, i.e., l<10 for MBL and σ^2_{GN}<0.01 for GN, the distorted images are sometimes misclassified as pristine images. This is intuitive, as the impact of the image distortion on the Fourier spectrum is limited when the distortion level is low, and thus, results in similar spectrum feature as pristine images. Overall, on all four image datasets, the distortion classifier achieves 94% accuracy on average.

8.3 Image Recognition Accuracy

Below, we evaluate the image recognition accuracy of CollabAR. As Caltech-256 does not contain multi-view images, we use MIRO and MVMDD as the datasets. We first examine the performance of the distortion-tolerant image recognizer, followed by the evaluation of CollabAR in multi-view and multiple distortions scenarios. We perform 3-fold cross validation in this experiment.

8.3.1 Performance of Distortion-tolerant Image Recognizer. We examine the recognition accuracy of both MobileNet-based and AlexNet-based implementations. We consider the single-distortion scenario and gradually increase the distortion level to investigate its impact on the recognition accuracy.

The results are shown in Figure 10. First, regardless of the distortion type, the MobileNet-based implementation outperforms the AlexNet-based one by 15% and 40% on MVMDD and MIRO,



(a) MobileDistortion (b) Caltech-256

(c) MIRO (d) MVMDD

Figure 9: Confusion matrix and accuracy of the image distortion classifier on different datasets: (a) MobileDistortion, (b) Caltech-256, (c) MIRO, and (d) MVMDD. On average, the distortion classifier achieves 94% accuracy on both real-world and synthesized distorted images.

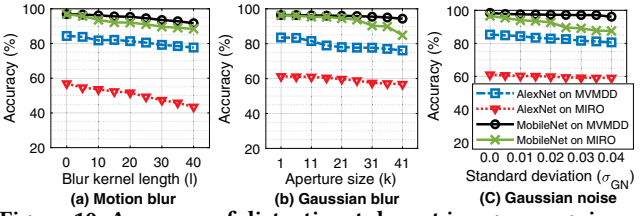
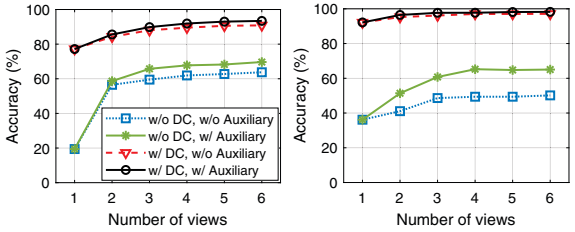


Figure 10: Accuracy of distortion-tolerant image recognizer when image contains (a) Motion blur, (b) Gaussian blur, and (c) Gaussian noise, respectively. MobileNet-based implementation outperforms the AlexNet-based one by 15% and 40% on MVMDD and MIRO datasets, respectively.

respectively. The improvement can be attributed to the use of the depthwise separable convolution as an efficient building block for image feature extraction, and the use of a linear bottleneck for better feature transformation [4]. Second, MobileNet-based recognition experts are more robust against image distortions. With the highest examined distortion level, i.e., $l = 40$ for MBL, $k = 41$ for GBL, and $\sigma^2_{GN} = 0.04$ for GN, we achieve 94.4% and 86.9% accuracy on average across the three distortions on MVMDD and MIRO, respectively. However, we still experience a modest accuracy drop when the distortion level is high. With the highest examined distortion level, CollabAR suffers 5% and 9% drop in accuracy on average on the two datasets, respectively. The accuracy drop will become much more severe when the image contains multiple distortions, but it can be resolved with the help of multi-view collaboration.

8.3.2 Performance of Multi-view Collaboration. Below, we evaluate the performance of CollabAR in the multi-view scenario where spatially and temporally correlated images are aggregated to improve



(a) MIRO (b) MVMDD

Figure 11: Accuracy of CollabAR in the multi-view single-distortion scenario on (a) MIRO and (b) MVMDD datasets.

the single-view accuracy. Moreover, we investigate how multi-view collaboration can help in multiple distortions cases.

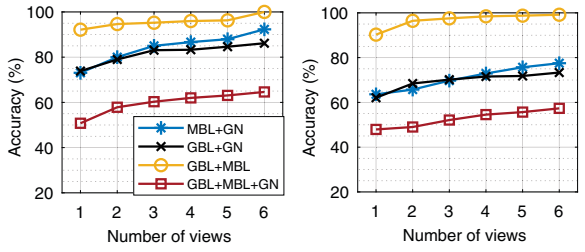
Setup: we apply MobileNet-based implementation for the recognition experts, as it demonstrates a higher accuracy than the AlexNet-based one. To microbenchmark the end-to-end system design, we evaluate CollabAR with two design options: (1) with and without the image distortion classifier, and (2) with and without the auxiliary feature for the multi-view ensembler. We consider up to six views to be aggregated for collaborative recognition.

Multi-view Single-distortion: in the single-distortion case, the image of each view contains one of the three distortions (i.e., MBL, GBL, or GN). We set the image distortion to a high level (i.e., $30 < l < 40$ for MBL, $31 < k < 41$ for GBL, and $0.03 < \sigma^2_{GN} < 0.04$ for GN) to evaluate CollabAR with the most severe distortions.

The performance is shown in Figure 11. First, with the image distortion classifier, CollabAR can correctly select the recognition expert that is dedicated to the examined distortion image, and can significantly improve the recognition accuracy. Given different number of views aggregated, CollabAR with the distortion classifier can improve the overall accuracy by 20% to 60% when compared to the case without a distortion classifier. Second, with the auxiliary feature, CollabAR can dynamically adjust the weights in multi-view aggregation, and can improve the overall accuracy up to 5% and 14% on MIRO and MVMDD, respectively. Lastly, the accuracy increases with the number of aggregated views. In case where CollabAR is incorporated with both the distortion classifier and the auxiliary feature, the overall accuracy can be improved by 15% and 7% with six views aggregated, on MIRO and MVMDD, respectively.

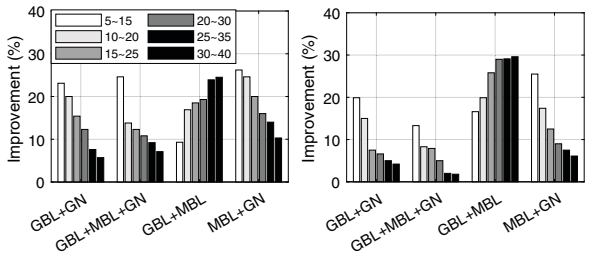
Multi-view Multi-distortion: below, we examine CollabAR, with both the distortion classifier and the auxiliary feature, in the multiple distortions scenario. We consider four multiple distortion combinations: MBL+GN, GBL+GN, GBL+MBL, and GBL+MBL+GN.

Figure 12 shows the performance of CollabAR given different distortion combinations in the testing image. The distortion is set to a modest level (i.e., $5 < l < 15$ for MBL, $6 < k < 16$ for GBL, and $0.005 < \sigma^2_{GN} < 0.015$ for GN). First, we observe a significant drop in the accuracy compared to that in the single-distortion scenario. This is expected: since the dedicated recognition experts are fine-tuned on images with specific types of distortions, multiple distortions will cause a mismatch in the feature distribution between the corrupted testing images and the dedicated recognition experts, and will thus result in a performance drop. Fortunately, the accuracy improves with the number of views aggregated. For different combinations



(a) MVMDD (b) MIRO

Figure 12: Accuracy in multi-view multiple distortions scenario. With six views aggregated, the accuracy improves by up to 16.5% and 20.3% on MIRO and MVMDD, respectively.



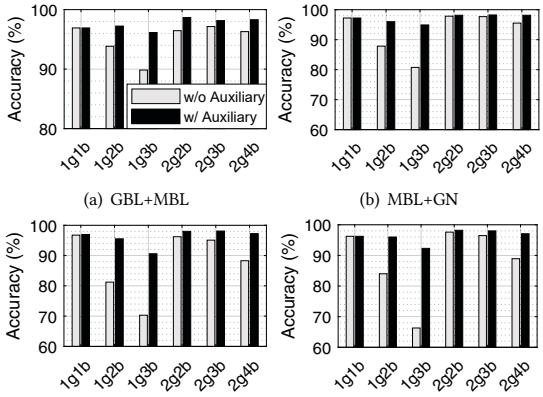
(a) MVMDD (b) MIRO

Figure 13: Accuracy improvement of CollabAR in multi-view multi-distortion scenario given different distortion combinations and distortion levels (i.e., by configuring l and k from 5 to 40 with a step size of 10). With six views aggregated, the overall accuracy is improved by 9.5% on average across different scenarios.

of image distortions, the accuracy can improve by up to 16.5% and 20.3% on MIRO and MVMDD, respectively.

Figure 13 shows the accuracy improvements of the multi-view aggregation given different distortion combinations and distortion levels. Overall, with six views aggregated, the overall recognition accuracy can be improved by 9.5% on average. Specifically, the improvements become modest when the distortion level is high. This is reasonable: if all the aggregated images suffer from severe distortions, simply aggregating the multi-view results would not improve the overall performance by much. One exception in our experiment is the 'GBL+MBL'. This is because our distortion-tolerant image recognizer is robust against the 'GBL+MBL' when the distortion level is low (as shown in Figure 12), and thus we achieve a higher improvement when the distortion level of 'GBL+MBL' is higher. In practical mobile AR scenarios, we believe that the probability of all the correlated images suffering severe multiple distortions is fairly low. Thus, as it will be shown in the following section, CollabAR achieves over 96% accuracy as long as there is one low-distortion image in the multi-view collaboration.

8.3.3 Advantages of AMEL. Below, we investigate the performance of CollabAR in heterogeneous multi-view multiple distortions scenarios, where images from different views are diverse in distortion type and distortion level. We define a 'good view' if the image is pristine and we define a 'bad view' if the image contains high distortion levels (i.e., $30 < l < 40$ for MBL, $31 < k < 41$ for GBL, and



(a) GBL+MBL (b) MBL+GN

(c) GBL+GN (d) GBL+MBL+GN

Figure 14: CollabAR accuracy on the MVMDD dataset with and without the auxiliary feature. The images contain multiple distortions and are diverse in the distortion level.

$0.03 < \sigma^2_{GN} < 0.04$ for GN). We also demonstrate the advantages of the proposed auxiliary-assisted multi-view ensemble over the conventional ensemble method without the auxiliary feature.

Figure 14 compares the performance of CollabAR on the MVMDD dataset with and without the auxiliary feature. CollabAR can always achieve a higher accuracy with the auxiliary feature. The improvement is most pronounced when there are more 'bad views' than 'good views' in the multi-view aggregation (e.g., '1g2b', '1g3b', '2g3b', and '2g4b', where '1g2b' refers to 'one good view and two bad views'). For instance, as shown in Figure 14(d), in a scenario where the image contains three types of distortion, the auxiliary feature can improve the performance by 8% on average and up to 26% in a more diverse case ('1g3b'). Overall, CollabAR achieves 96% and 88% accuracy on average, with and without the auxiliary feature, respectively. We achieve similar results on the MIRO dataset.

8.4 System Profiling

Below, we present a comprehensive profiling of CollabAR in terms of computation and communication latency. We consider two deployments of CollabAR: (1) the whole system is deployed on the mobile client, and (2) the edge-assisted design in which the computation-intensive recognition pipeline is deployed on the server. We implement CollabAR on four different platforms. In addition to the edge server, we also consider three different commodity smartphones, Nokia 7.1, Google Pixel 2 XL, and Xiaomi 9, as the mobile clients. The three smartphones are heterogeneous in both computation and communication performance. We leverage the TensorFlow Lite [50] as the framework when implementing CollabAR on the smartphones.

8.4.1 Computation Latency. To examine the computation latency of CollabAR, we tear down its image recognition pipeline into two processing components: (1) the ***image distortion classifier*** and (2) the ***AMEL*** image recognition. Two possible implementations of the AMEL, i.e., AlexNet-based and MobileNetV2-based, are considered. We run 30 trials of the end-to-end image recognition pipeline and report the average time consumed by each of the components. Table 4 shows the computation latency (in ms) when we deploy the

Table 4: The computation latency (in ms) of CollabAR when deployed on different hardware platforms. The edge-assisted design achieves the lowest overall latency of 8.1ms.

Processing Component	Processing Unit	Hardware Platform			
		Edge server	Nokia 7.1	Pixel 2 XL	Xiaomi 9
AMEL (AlexNet)	CPU	48.0	255.4	189.9	124.2
	GPU	6.8	331.0	127.5	71.2
AMEL (MobileNetV2)	CPU	28.1	108.4	81.9	33.7
	GPU	4.7	46.9	26.0	33.1
Distortion classifier	CPU	4.4	29.3	22.33	13.1
	GPU	3.4	20.3	8.7	8.4
Lowest overall latency	CPU	32.5	137.7	104.2	46.8
	GPU	8.1	67.2	34.7	41.5

system on different hardware platforms. First, regarding the latency of AMEL, MobileNet-based design can always achieve lower latency than AlexNet-based one, for the same hardware platform and processing unit used. This is because MobileNetV2 is more lightweight and is well-optimized for running on resource-limited devices [51]. Second, the edge server achieves the lowest overall computation latency of 8.1ms when using its GPU for the computation. Our edge-assisted design achieves more than four times speedup over the best examined mobile platform. Although the latency for individual mobile platforms can be reduced when more efficient DNN model and powerful mobile GPU are used, the edge-assisted design ensures low computation latency when mobile devices are heterogeneous in computation power.

8.4.2 Communication Latency. Communication latency is a factor of delay when we deploy the recognition process on the edge. There are three phases involved in a single end-to-end image recognition:

- Network delay: client transmits either the resized image or the original image to the edge server, and the server sends the recognition result back to the client.
- Image resizing: resizing process is needed when the client decides to transmit the resized image. The resizing converts the original image from 3,024×4,032×3 to the size of 224×224×3.
- Image encoding: the encoding compresses the raw image using the JPEG lossless compression.

In our experiment, the client and the server are connected through a single-hop Wi-Fi network in our lab. We measure the latency when using both 2.4GHz and 5GHz wireless channels. Table 5 shows the latency of the mobile clients when communicating with the edge server. First, because of the higher data rate, the 5GHz channel achieves a lower latency than the 2.4GHz. Second, comparing to resizing and sending the resized image, sending the original image directly to the server increases the latencies in both network delay and image encoding. Overall, with more advanced wireless chips, Pixel 2 XL and Xiaomi 9 achieve similar round-trip communication latencies of 13.4ms and 9.7ms, respectively. Nokia 7.1 has the highest latency of 24.6ms. The communication latency is largely affected by the wireless network condition. Our current measurement is conducted under modest background traffic load. To address the high communication latency in congested networks, we can compress the images with a higher ratio. For instance, one can apply a higher degree of compression to parts of the image that are less likely to contain important features [5].

8.4.3 End-to-end Latency. Putting all together, when executing on the edge, the end-to-end system latency is 32.7ms, 21.5ms, and 17.8ms, for Nokia 7.1, Pixel 2 XL, and Xiaomi 9, respectively. The results indicate that our edge-assisted design can provide accurate image recognition without sacrificing the end-to-end system latency. Overall, for all the three commodity smartphones that we have examined, the edge-assisted design allows us to achieve 30fps continuous image recognition.

Table 5: Communication latency (in ms) for different clients. The results are averaged over 30 trials. For each recognition request, the clients take 10ms to 25ms in communication.

Content		Hardware Platform		
		Nokia 7.1	Pixel 2 XL	Xiaomi 9
Image resizing		4.1	2.5	2.3
Image encoding	Resized image	10.3	4.6	2.4
	Original image	722.5	348.1	182.5
Network delay (Resized image)	2.4GHz	14.3	6.6	5.0
	5GHz	10.2	6.3	5.0
Network delay (Original image)	2.4GHz	120.7	60.3	32.0
	5GHz	91.2	29.1	30.4
Lowest round-trip latency		24.6	13.4	9.7

9 DISCUSSION

In this section, we discuss our limitations and outline potential solutions for future work.

Real-world image distortions. In the current presentation, CollabAR is evaluated with distortion images that are synthesized using simple distortion models (namely, the zero-mean additive Gaussian noise model for Gaussian noise [32], the non-uniform motion blur kernel for Motion blur [16], and the two-dimensional Gaussian kernel for Gaussian blur [17]). However, there is a non-negligible gap between the artificial and the real-world distortions [42, 52]. Inevitably, the gap may result in poor recognition performance when the system is applied to 'in-the-wild' AR scenarios. To alleviate this problem, instead of modeling the image distortions using simple noise models, one can leverage the Generative Adversarial Network (GAN) to generate more realistic image distortions using real-world distortion images as input [52–54]. The distortion images generated from GAN can be further used to train CollabAR and make it more robust to real-world distortions.

Mobile device heterogeneity. By aggregating multi-view images, CollabAR has shown great improvement in recognition accuracy. However, the current evaluation is performed on multi-view images taken from a single type of device (i.e., Nokia 7.1). In practice, owing to the heterogeneity in the underlying hardware and signal processing pipelines, different mobile devices are showing variations in their image outputs on the same physical object/scene [55]. For image recognition system, this device heterogeneity may lead to feature mismatch among images captured by different devices. For the future work, a potential solution is leveraging the cycle-consistent adversarial networks [53, 56] to learn the translation function among different mobile devices, and use it to reduce the domain shift caused the device heterogeneity.

Impact of image offloading on recognition. During image offloading, the information loss due to image resizing and compression process (e.g., the JPEG compression) may also lead to accuracy degradation [42, 54]. However, its impact on CollabAR is small. In our design, since all images are processed with the same offloading procedure, they suffer the same information loss during resizing

and compression. The domain shift and feature mismatch between the training and testing images are largely eliminated, and thus, the accuracy loss due to image offloading is negligible.

10 CONCLUSION

In this paper, we presented CollabAR, an edge-assisted collaborative image recognition system that enables distortion-tolerant image recognition with imperceptible system latency for mobile augmented reality. To achieve this goal, we propose the distortion-tolerant image recognition to improve robustness against real-world image distortions, and the collaborative multi-view image recognition to improve the overall recognition accuracy. We implement the end-to-end system on four different commodity devices, and evaluate its performance on two multi-view image datasets. Our evaluation demonstrates that CollabAR achieves over 96% recognition accuracy for images with severe distortions, while reducing the end-to-end system latency to as low as 17.8ms.

ACKNOWLEDGMENTS

We would like to thank our anonymous reviewers and shepherd for their insightful comments and guidance on making the best possible representation of our work. This work was supported in part by the Lord Foundation of North Carolina and by NSF awards CSR-1903136 and CNS-1908051.

REFERENCES

[1] P. Jain, J. Manweiler, and R. Roy Choudhury, "OverLay: Practical mobile augmented reality," in *ACM MobiSys*, 2015.
[2] K. Chen, T. Li, H.-S. Kim, D. E. Culler, and R. H. Katz, "Marvel: Enabling mobile augmented reality with low energy and low latency," in *ACM SenSys*, 2018.
[3] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012.
[4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *IEEE CVPR*, 2018.
[5] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *ACM MobiCom*, 2019.
[6] M. Xu, M. Zhu, Y. Liu, X. Lin, and X. Liu, "DeepCache: Principled cache for mobile deep vision," in *ACM MobiCom*, 2018.
[7] H. Ji and C. Liu, "Motion blur identification from image gradients," in *IEEE CVPR*, 2008.
[8] "i-MARECULTURE," https://imareculture.eu.
[9] S. F. Dodge and L. J. Karam, "Quality robust mixtures of deep neural networks," *IEEE Transactions on Image Processing*, 2018.
[10] W. Zhang, B. Han, P. Hui, V. Gopalakrishnan, E. Zavesky, and F. Qian, "CARS: Collaborative augmented reality for socialization," in *ACM HotMobile*, 2018.
[11] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A large-scale hierarchical image database," in *IEEE CVPR*, 2009.
[12] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," 2007.
[13] S. Ghosh, R. Shet, P. Amon, A. Hutter, and A. Kaup, "Robustness of deep convolutional neural networks for image degradations," in *IEEE ICASSP*, 2018.
[14] T. S. Borkar and L. J. Karam, "DeepCorrect: Correcting DNN models against image distortions," *IEEE Transactions on Image Processing*, 2019.
[15] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Springer ECCV*, 2010.
[16] J. Sun, W. Cao, Z. Xu, and J. Ponce, "Learning a convolutional neural network for non-uniform motion blur removal," in *IEEE CVPR*, 2015.
[17] J. Flusser, S. Farokhi, C. Höschl, T. Suk, B. Zitová, and M. Pedone, "Recognition of images degraded by gaussian blur," *IEEE Transactions on Image Processing*, 2015.
[18] W. Zhang, B. Han, and P. Hui, "Jaguar: Low latency mobile augmented reality with flexible tracking," in *ACM MM*, 2018.
[19] P. Guo, B. Hu, R. Li, and W. Hu, "FoggyCache: Cross-device approximate computation reuse," in *ACM MobiCom*, 2018.
[20] "HoloLens," https://www.microsoft.com/en-us/hololens.
[21] "Magic leap," https://www.magicleap.com/.
[22] "Google ARCore," https://developers.google.com/ar/.
[23] "Apple ARKit," https://developer.apple.com/documentation/arkit.
[24] X. Ran, C. Slocum, M. Gorlatova, and J. Chen, "ShareAR: Communication-efficient multi-user mobile augmented reality," in *ACM HotNets*, 2019.
[25] H. Verkasalo, "Contextual patterns in mobile service usage," *Personal and Ubiquitous Computing*, 2009.
[26] Z. Zhou, *Ensemble methods: Foundations and algorithms*. Chapman and Hall/CRC, 2012.
[27] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *IEEE ICDCS*, 2017.
[28] N. F. Rajani and R. J. Mooney, "Stacking with auxiliary features," in *AAAI IJCAI*, 2017.
[29] M. Labbe and F. Michaud, "Appearance-based loop closure detection for online large-scale and long-term operation," *IEEE Transactions on Robotics*, 2013.
[30] X. Zeng, K. Cao, and M. Zhang, "MobileDeepPill: A small-footprint mobile deep learning system for recognizing unconstrained pill images," in *ACM MobiSys*, 2017.
[31] H. Qiu, X. Liu, S. Rallapalli, A. J. Bency, K. Chan, R. Urgaonkar, B. Manjunath, and R. Govindan, "Kestrel: Video analytics for augmented multi-camera vehicle tracking," in *IEEE IoTDI*, 2018.
[32] W. Liu and W. Lin, "Additive white gaussian noise level estimation in SVD domain for images," *IEEE Transactions on Image Processing*, 2012.
[33] Y. Zhou, S. Song, and N. Cheung, "On classification of distorted images with deep convolutional neural networks," in *IEEE ICASSP*, 2017.
[34] S. Dodge and L. Karam, "Human and DNN classification performance on images with quality distortions: A comparative study," *ACM Transactions on Applied Perception*, 2019.
[35] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, 2012.
[36] R. Liu, Z. Li, and J. Jia, "Image partial blur detection and classification," in *IEEE CVPR*, 2008.
[37] X. Yi and M. Eramian, "LBP-based segmentation of defocus blur," *IEEE Transactions on Image Processing*, 2016.
[38] A. Jain, *Fundamentals of digital image processing*. Prentice Hall, 1989.
[39] C. Poynton, *Digital video and HD: Algorithms and Interfaces*. Elsevier, 2012.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016.
[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *IEEE ICLR*, 2015.
[42] Z. Sun, M. Ozay, Y. Zhang, X. Liu, and T. Okatani, "Feature quantization for defending against distortion of images," in *IEEE CVPR*, 2018.
[43] X. Peng, J. Hoffman, Y. Stella, and K. Saenko, "Fine-to-coarse knowledge transfer for low-res image classification," in *IEEE ICIP*, 2016.
[44] Y. Li and W. Gao, "DeltaVR: Achieving high-performance mobile VR dynamics through pixel reuse," in *ACM/IEEE IPSN*, 2019.
[45] "Google cloud anchors," https://developers.google.com/ar/develop/developer-guides/anchors.
[46] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," in *ACM UbiComp*, 2012.
[47] Y. Lin, T. Liu, and C. Fuh, "Local ensemble kernel learning for object category recognition," in *IEEE CVPR*, 2007.
[48] M. M. Derakhshani, S. Masoudnia, A. H. Shaker, O. Mersa, M. A. Sadeghi, M. Rastegari, and B. N. Araabi, "Assisted excitation of activations: A learning technique to improve object detectors," in *IEEE CVPR*, 2019.
[49] A. Kanezaki, Y. Matsushita, and Y. Nishida, "RotationNet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," in *IEEE CVPR*, 2018.
[50] "Tensorflow lite," https://www.tensorflow.org/lite.
[51] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
[52] J. Chen, J. Chen, H. Chao, and M. Yang, "Image blind denoising with generative adversarial network based noise modeling," in *IEEE CVPR*, 2018.
[53] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE ICCV*, 2017.
[54] A. Bulat, J. Yang, and G. Tzimiropoulos, "To learn image super-resolution, use a GAN to learn how to do image degradation first," in *Springer ECCV*, 2018.
[55] A. Ignatov, N. Kobyshev, R. Timofte, K. Vanhoey, and L. Van Gool, "DSLR-quality photos on mobile devices with deep convolutional networks," in *IEEE ICCV*, 2017.
[56] A. Mathur, A. Isopoussu, F. Kawsar, N. Berthouze, and N. D. Lane, "Mic2Mic: Using cycle-consistent generative adversarial networks to overcome microphone variability in speech systems," in *ACM/IEEE IPSN*, 2019.