Enabling Continuous Object Recognition in Mobile Augmented Reality

XIANG SU, Norwegian University of Science and Technology, Norway and University of Oulu, Finland

AI JIANG, University of St Andrews, United Kingdom

JACKY CAO, University of Oulu, Finland and Norwegian University of Science and Technology, Norway

WENXIAO ZHANG, The Hong Kong University of Science and Technology, Hong Kong

PAN HUI, The Hong Kong University of Science and Technology, Hong Kong and University of Helsinki, Finland

JUAN YE, University of St Andrews, United Kingdom

Mobile Augmented Reality (MAR) applications enable users to interact with physical environments through overlaying digital information on top of camera views. Detecting and classifying complex objects in the real world presents a critical challenge to enable immersive user experiences in MAR applications. Aiming to provide continuous MAR experiences, we address a key challenge of continuous object recognition, which requires accommodating an increasing number of recognition requests on different types of images in MAR systems and possible new types of images in emerging applications. Inspired by the latest advance in continual learning approaches in computer vision, this paper presents a novel MAR system to enhance its scalability with continual learning in realistic scenarios. Our experiments demonstrate that 1) the system enables efficiently recognising objects without requiring retraining from scratch; and 2) edge computing further reduces latency for continual object recognition.

CCS Concepts: • **Human-centered computing → Mixed / augmented reality**; • **Computing methodologies → Machine learning**.

Additional Key Words and Phrases: mobile augmented reality, edge computing, continual learning

ACM Reference Format:
Xiang Su, Ai Jiang, Jacky Cao, Wenxiao Zhang, Pan Hui, and Juan Ye. 2018. Enabling Continuous Object Recognition in Mobile Augmented Reality. In *IUI '22: 27th Annual Conference on Intelligent User Interfaces, March 22–25, 2022, Helsinki, Finland.* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Augmented reality allows for the interweaving of digital data with physical spaces [2][3]. Mobile Augmented Reality (MAR) applications enable users to interact with physical environments through overlaying digital information on top of camera views of the real world [6]. One critical challenge is to provide unique and continuous experiences in different realistic scenarios whilst relying on mobile personal displays, which require MAR applications to continually accumulate new knowledge based on users' context and environments. Ideally, MAR applications should provide the best possible performance in various scenarios regarding object recognition accuracy, and end-to-end latency [11]. However, most current MAR applications optimise their performance for just certain scenarios [4]. This means that they lack the ability to detect and classify objects in complex real-world situations. In addition, immersive MAR experiences require low end-to-end latency. Therefore, MAR applications need to consider the important trade-off between continuous accurate

Manuscript submitted to ACM

object recognition and latency [7]; that is, not requiring much time to re-train the computer vision model every time when new types of images become available.

This paper novelly adopts a continual learning approach [1] in MAR systems and distribute components in an edge architecture to enhance the computation efficiency. We design and implement continual learning in both centralised and distributed manners; that is, the system allows distribution of each trained model on edge nodes to handle object recognition and retrieval. Each edge node hosts only one type of images instead of the whole set, which leads to more efficient object searching. Therefore, the system can significantly enhance the performance of MAR applications with low bandwidth, latency, and jitter. Our MAR system enables: 1) continuous computation-intensive object recognition with a large-scale database of reference images; 2) automatic selection of suitable models based on relatedness of one task (a task here refers to learning new types of images) to another; and 3) inclusion of new models which build on existing ones without needing to store the datasets of existing models. In addition, our MAR client integrates ARCore supported maker-less object tracking and our MAR server can be deployed on the Cloud or edge, leveraging the computation capabilities of Graphics Processing Units (GPUs). Our experiments demonstrate 1) how continual learning benefits MAR with recognising objects from new real-world objects and 2) how edge computing facilitates MAR in terms of data transfer and processing latency for continuous object recognition.

2 SYSTEM DESIGN AND IMPLEMENTATION

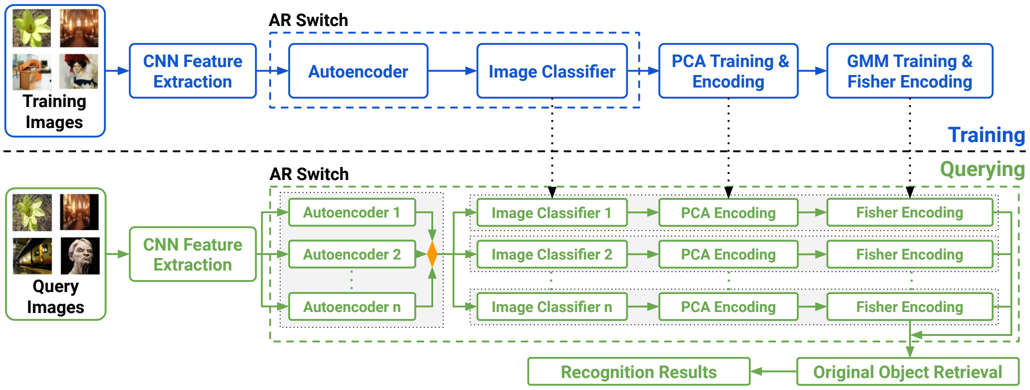


Fig. 1. Workflow of MAR Server.

The MAR client extracts the current camera frame and sends a recognition request to the MAR Server. Figure 1 presents MAR workflow on the server. When MAR server receives an object recognition request, it first extracts features of the request image from a Convolutional Neural Network (CNN). These features are used to automatically select the most related prior model for image classification and object retrieval. When the classifiers are available, MAR server reduces the feature dimension using Principal Component Analysis (PCA) with the offline trained parameters and creates a single Fisher Vector of the image using the trained Gaussian Mixture Model (GMM) model. To find the original image classifier within the dataset, the MAR server performs original object retrieval. With the original image classification and the feature matching result, MAR server calculates the pose of the target object within the camera frame and sends the result to the MAR client. Our MAR server leverages image retrieval to actually recognise objects; whereas other types of MAR applications may classify objects for augmentation. AR Switch is a continual learning system which deals with new tasks by automatically selecting the most related prior model to enable efficient learning of the new model, i.e., handling new models without storing all previous data. AR Switch enables the selection of an associated model based on the data itself. The switch component performs as a model recogniser that provides the relevance of its associated models for a given test sample.

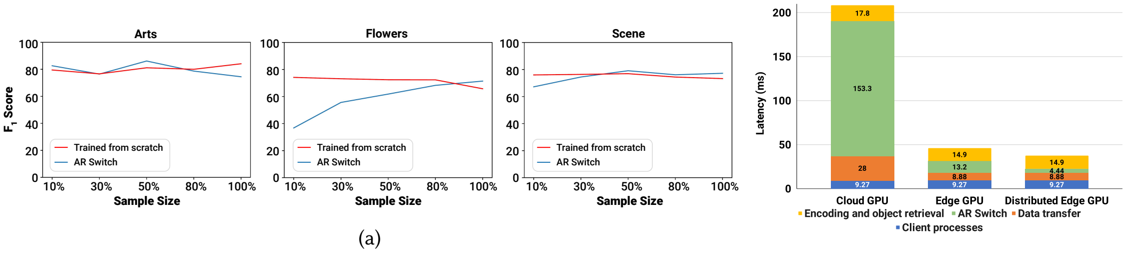
Offline Training: Offline training starts with extracting feature points using CNN from reference images and reducing their dimensions with SIFT [8]. AR Switch creates autoencoders and image classification models by utilising the extracted features from training and validation image sets, and both types of models are trained on one type of image datasets. Afterwards, we train a statistical model with Principal Component Analysis (PCA) to compress feature points. MAR server builds a Gaussian Mixture Model (GMM) using the PCA-transformed feature points and leverages Fisher Vectors to re-encode images into fixed length vectors from extracted SIFT descriptors. Additionally, hash tables of compressed feature points are generated to facilitate online object recognition. As shown in Figure 1, we employ an autoencoder to learn the latent representations for each object type; for example, sculptures, book covers, cafes, and restaurants. Autoencoders are a representation learning technique that learn non-linear correlations between features and reduce the dimensions of the original feature space. When a new dataset becomes available, we will use the autoencoder to find an existing submodel that is closest to the new dataset. Then we will take the selected submodel and re-train it with the new dataset and deploy the final model on a new edge node. To decide whether fine tuning a submodel or creating a new submodal, we measure the relatedness between tasks and set the threshold on the relatedness value [1].

Online Recognition: We use the reconstruction errors of an autoencoder from each dataset to choose the appropriate classifier for an input image frame. For example, AR Switch can decide what type of objects are in an input image frame; e.g., sculpture or painting, and then send them to a classifier to recognise which type of classes the image belongs to; e.g., a flower, architecture or portrait painting. We implement AR Switch in two different architectures, i.e. Cloud and edge. In the centralised Cloud architecture, we deploy all AR Server components as a whole on the Cloud. We implement two alternatives on the edge. The first alternative edge architecture deploys all AR Server components as a whole on one edge server. In the second alternative, we deploy autoencoders and switch on a main edge server and distribute classifiers onto several agent edge servers. The distribution of computation improves system performance in terms of latency. The main edge server communicates with the agent edge servers through UDP. The distribution of the AR Switch components is presented with dotted lined boxes of Figure 1.

3 EXPERIMENTS AND ANALYSIS

We analyse 1) system’s capacity to accommodate a large number of images and to continuously learn new types of images, and 2) break down end-to-end latency for the MAR system on Cloud and edge architectures. We utilise three public datasets, including *Art* [5], *Flower* [9], and indoor *Scene* [10]. *Art* contains 9000 images (with total size 581MB) covering five different types of arts; including drawings and watercolours, works of painting, sculpture, graphic art, and iconography. *Flower* consists of 102 flower categories (with total size 353MB), each of which has between 40 and 258 images. *Scene* contains 67 indoor categories and has a total of 15620 images (with total size 2.4GB). The number of images varies across categories, but there are at least 100 images per category.

Transfer Learning with AR Switch: This experiment demonstrates how AR Switch enables transfer learning to help learn new objects within the same categories with less initial training data. To do that, we train a submodel with 2 datasets first and then learn the remaining dataset as a new task. We iterate each of the three datasets and average the results. We quantify and measure the recognition accuracy using F_1-score. We compare with baseline classifiers that



(a)

(b)

Fig. 2. (a) Comparison of F_1-scores with baseline classifiers that train data from scratch across different sample proportions on three datasets. (b) Latency breakdown for an edge and cloud server, both of which utilise GPU architectures.

train data from scratch and change the fraction of training data from 10% to 100% with 20% increasing intervals. Figure 2a presents that with less training data, AR Switch performs better than the model that is trained initially from scratch when there is only 10% of training data on both Flowers and Scene datasets. Whilst the model trained from scratch functions slightly better than AR Switch on Arts, this is reasonable since the distribution of Arts is more diverse than Flowers and Scene. On the other hand, the benefits of continual learning pertain to much less training time. Training a classifier from complete scratch to produce image classification models requires 24.0 minutes for Arts, 21.3 minutes for Flowers, and 17.5 minutes for Scene, respectively. In comparison, the total processing time for edge-based AR Switch to process a query takes just 32.0 ms, which requires only 1/654 of the training-from-scratch training time. The results verify that our system achieves faster convergence of retraining classifiers on new and unseen data.

Latency Breakdown for MAR in Edge and Cloud Architectures: The second experiment focuses on breakdown latency in the MAR system. Both cloud and edge-deployed MAR servers are provided with pre-trained image classification models. Two edge architectures are used, unified and distributed (see Section 2). Unified architecture contains both autoencoders and classifiers on one server; whereas distributed architecture separates the classifiers to different edge servers, and a main edge contains the autoencoders and switch. Figure 2b presents the latency for cloud, edge, and distributed edge architectures. Results are obtained by initiating 100 recognition requests from a client, measuring the latency, and calculating final median values. These results show that the edge server requires only 18.8% of the cloud end-to-end latency, a consequence of proximity and powerful GPU. The data transfer to the edge is significantly reduced to less than 10 ms. CNN feature extraction generates the largest proportion of latency in the MAR server software which could be further optimised, in addition, reducing PCA, Fisher encoding and original object retrieval latency is possible.

4 DISCUSSION

Future MAR applications should provide continuous, context-aware, and multi-purpose experiences. This research is one of the first efforts towards the development of continuous object recognition in MAR applications. Enabling continuity requires more datasets and computationally-intensive algorithms, which imposes high latency and bandwidth network requirements. We present a continual learning based approach to recognise a increasing number of new visual objects. Our experiment demonstrates its feasibility and advantage in reduced computation cost, low end-to-end latency, and improved accuracy compared to training from scratch. In our future research, we will conduct more experimentation and perform a comprehensive analysis by exploring the accuracy of object recognition in real-world scenarios, such as recognising objects from images of different qualities, like background lighting and 3D rotations.

REFERENCES

[1] R. Aljundi, P. Chakravarty, and T. Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7120–7129, 2017.

[2] R. T. Azuma. A survey of augmented reality. *Presence: Teleoper. Virtual Environ.*, 6(4):355–385, Aug. 1997.

[3] J. Cao, K.-Y. Lam, L.-H. Lee, X. Liu, P. Hui, and X. Su. Mobile augmented reality: User interfaces, frameworks, and intelligence, 2021.

[4] T. Y.-H. Chen, H. Balakrishnan, L. Ravindranath, and P. Bahl. Glimpse: Continuous, real-time object recognition on mobile devices. *GetMobile: Mobile Comp. and Comm.*, 20(1):26–29, July 2016.

[5] Danil. Art images: Drawing/painting/sculptures/engravings. https://www.kaggle.com/thedownhill/art-images-drawings-painting-sculpture-engraving, 2018. Accessed: 2021-12-22.

[6] T. Höllerer and S. Feiner. Mobile augmented reality. *Telegeoinformatics: Location-based computing and services*, 21, 2004.

[7] Q. Liu, S. Huang, J. Opadere, and T. Han. An edge network orchestrator for mobile augmented reality. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 756–764, 2018.

[8] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[9] M.-E. Nilsback and A. Zisserman. 102 category flower dataset. http://www.robots.ox.ac.uk/~vgg/data/flowers/102/index.html, 2008. Accessed: 2021-12-22.

[10] A. Quattoni and A. Torralba. Indoor scene recognition. http://web.mit.edu/torralba/www/indoor.html, 2009. Accessed: 2021-12-22.

[11] X. Su, J. Cao, and P. Hui. *5G Edge Enhanced Mobile Augmented Reality.* Association for Computing Machinery, New York, NY, USA, 2020.