AutoTune: Autonomous Learning for Face Recognition in the Wild

ANONYMOUS AUTHOR(S)

Face recognition is a key enabling service for smart-spaces, allowing building management agents to easily monitor 'who is where', anticipating user needs and tailoring their local environment and experiences. Although facial recognition, especially through the use of deep neural networks, has achieved stellar performance over large datasets, the majority of approaches require supervised learning, that is, to be trained with tens or hundreds of images of users in different poses and lighting conditions. In this paper, we motivate that this enrollment effort is unnecessary if the smart-space has access to a wireless identifier e.g., through a smart-phone's MAC address. By learning and refining the noisy and weak association between a user's smart-phone and facial images, AutoTune can fine-tune a deep neural network to tailor it to the environment, users and conditions of a particular camera or set of cameras. In particular, we introduce a novel soft-association technique which limits the impact of erroneous decisions taken early on in the training process from corrupting the clusters. We also show how this zero-effort enrolment system becomes better over time at recognizing and identifying users by increasing the inter-cluster and minimizing the intra-cluster distance. Through extensive experiments with multiple users on two sites, we demonstrate the ability of AutoTune to design an environment-specific, continually evolving facial recognition system with entirely no user effort. We believe that this framework is a step towards Mark Weiser's vision of the third wave of computing which recedes into the background of our physical spaces.

CCS Concepts: • **Human-centered computing** → Ubiquitous and mobile computing; • **Computing methodologies** → Machine learning; • **Computer system organization** → Embedded and cyber-physical systems;

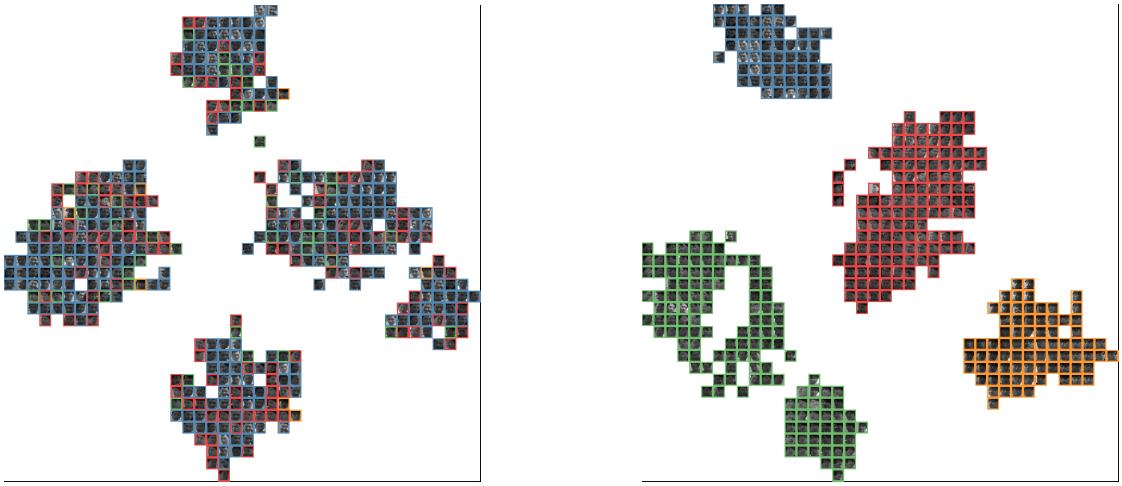
Additional Key Words and Phrases: Adaption of Learning Systems, Face Recognition, Heterogeneous Transfer Learning

ACM Reference Format:
Anonymous Author(s). 2018. AutoTune: Autonomous Learning for Face Recognition in the Wild. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 0, 0, Article 0 (2018), 25 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Facial recognition and verification are key components of smart spaces, e.g., offices and buildings for determining who is where. Knowing this information allows a building management system to tailor ambient conditions to particular users, perform automated security (e.g., opening doors for the correct users without the need for a swipe card), and customize smart services (e.g., coffee dispensing). A vast amount of research over the past decades has gone into designing tailored systems for facial recognition and with the advent of deep learning, progress has accelerated. As an example of a state-of-the-art face recognizer, FaceNet achieves extremely high accuracies (e.g., 99.5%) on very challenging datasets through the use of a low dimensional embedding, allowing similar faces to be clustered through their Euclidean distance. Although FaceNet and similar approaches can operate at near-perfect levels of performance, they suffer from two overarching issues in indoor (wild) environments. Firstly, viewing angles and image sizes from smart cameras show even higher levels of variability than experienced in the FaceNet and VGG datasets. For example, Fig. 1 shows an example where FaceNet trained on the VGG database performs poorly in a new environment. Secondly, and more critically, FaceNet is a *supervised* training method which uses large numbers of images (e.g., 300) of the same person in different scenes and angles to

2474-9567/2018/0-ART0 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn



(a) Pre-trained FaceNet on VGG datasets

(b) Fine-tuned Model via AutoTune

Fig. 1. Similarity comparison of the face images of four subjects in a new environment based on their features extracted by VGG-pre-trained FaceNet (left) and fine-tuned model by AutoTune (right) respectively. Features are projected to 2d plane via t-SNE [26]. Images belonging to the same subject identities are framed with the *same* colors. It can be seen that face features provided by the pre-trained model are much less discriminative than the fine-tuned model of AutoTune, with significant color overlap. (For interpretation of this figure, the reader is referred to the color version of this article)

train an accurate clustering and embedding. To build such a database without access to a user's social profile or additional labelled data would require a significant amount of effort in terms of subject enrollment. As an example of one-shot enrollment, if FaceNet is used on a single image e.g., from a user's webpage or ID card, recognition accuracy is in the order of 10-15%, clearly not suitable for the proposed task of knowing who is where.

In this work, we propose a novel way to automatically achieve high levels of recognition with *zero* user enrollment effort. To achieve this, we exploit the fact that users are typically colocated with their mobile devices e.g., smartphones and fitness monitors. To provide ubiquitous connectivity, these devices have some form of wireless interface e.g., BLE, WiFi, cellular. These provide a unique identifier, ranging from the hardware level (e.g., IMEI or MAC addresses) to the network authentication level (e.g., usernames). Our aim is to use these identifiers to crowdsource a set of faces and a set of identifiers to refine a pre-trained classifier with the goal of improving its performance over time. However, the binding between an image and a wireless identifier is not reliable, as multiple users could be present in the same area, a user may not be carrying their device or they may have changed their device. In this work, we present AutoTune, a system which can be used to gradually improve the performance of facial recognition systems in the wild, with zero user effort, tailoring them to the visual dynamics of a particular smart space. In particular, our contributions are:

- We observe that wireless signals of users' devices provide valuable, albeit noisy, clues for face recognition. Namely, wireless signals can serve as a weak label.
- We create AutoTune, a novel pipeline to simultaneously label face images in the wild and adapt the pre-trained deep neural network to recognize the faces of users in new environments. The key idea is to repeat the face-identity association and network update in tandem.
- To cope with noises in observations and mis-labeling, we propose a novel probabilistic framework in AutoTune and design a new fuzzy center loss to enhance the robustness of network fine-tuning.

- We deployed AutoTune in two real-world environments and experimental results demonstrate that AutoTune is able to achieve > 0.85 F_1 score of image labeling in both environments, outperforming the best competing approach by $> 25\%$. Compared to the best competing approach, using the features extracted from the fine-tuned model and training a classifier based on the cross-modality labeled images can give a $\sim 19\%$ performance gain for online face recognition.
- We investigate the sensitivity and robustness of AutoTune under various conditions. Results indicate that AutoTune is a stable pipeline that works with comparable performance with different parameter settings. Moreover, we examine AutoTune in an adversarial simulation environment and results show its robustness against noises in visual and WiFi observations.

The rest of this paper is organized as follows. §2 introduces the background of this work. Preliminaries are given in §3. We describe the AutoTune solution in §4, and provide the system implementation details in §5. The proposed approach is evaluated and compared with state of the art methods in §6. Finally, we discuss the results in §7 and conclude in §8.

2 RELATED WORK

Deep face recognition: Face recognition is arguably one of the most active research areas in the past few years, with a vast corpus of face verification and recognition work [31, 39, 48]. With the advent of deep learning, progress has accelerated significantly. Here we briefly overview state-of-the art work in Deep Face Recognition (DFR). Taigman et al. pioneered this research area and proposed DeepFace [38]. It uses CNNs supervised by softmax loss, which essentially solves a multi-class classification problem. When introduced, DeepFace achieved the best performance on the labeled face in the wild (LFW) [15] benchmark. Since then, many DFR systems have been proposed. In a series of papers [35, 36] Sun et al. extended on DeepFace incrementally but steadily increased the recognition performance. A critical point in DFR happened in 2015, when researchers from Google [34] used a massive dataset of 200 million face identities and 800 million image face pairs to train a CNN called Facenet, which largely outperformed prior art on the LFW benchmark when introduced. A point of difference is in their use of a "triplet-based" loss [7], that guides the network to learn both inter-class dispersion and inner-class compactness. The architecture of Facenet has since become the mainstream architecture in DFR, and research has incrementally improved the recognition performance by introducing different training losses, e.g., center loss [44], Large-margin softmax loss [21], angular softmax [20] and so forth. Although the above methods have proven remarkably effective in face recognition, the training needs a vast amount of labeled images to train the supervised DFR network. A large amount of labeled data is not always achievable in a particular domain, and typically using a small amount of training data will incur poor generalization ability in the wild.

Cross-modality Matching: Cross-modal matching has received considerable attention in different research areas. Methods have been developed to establish mappings from images [11, 17, 42] and videos [41] to textual descriptions (e.g., captioning), developing image representation from sounds [27, 28], recognizing speaker identities from Google calendar information [25], and generating visual models from text [49]. In cross-modality matching between images and radio signals, however, work is very limited and all dedicated to trajectory tracking of human [2, 30, 40]. The field of face representation learning via wireless signals is a blank space.

Transfer Learning: At a high level, AutoTune falls into the category of transfer learning in the way it updates the face representation. Transfer learning is the ability to extend what has been learned from one domain (source domain) to another nonidentical but similar domain (target domain) that shares common features. Most transfer learning approaches aims to reduce the discrepancy in feature distribution from source and target domain [29]. It is become more appealing recently in deep learning where pre-trained models in source domain are used as the starting point on computer vision tasks given the vast compute and time resources required to develop neural network models on these problems [1, 6, 9, 23]. The process of updating the pre-trained deep neural networks

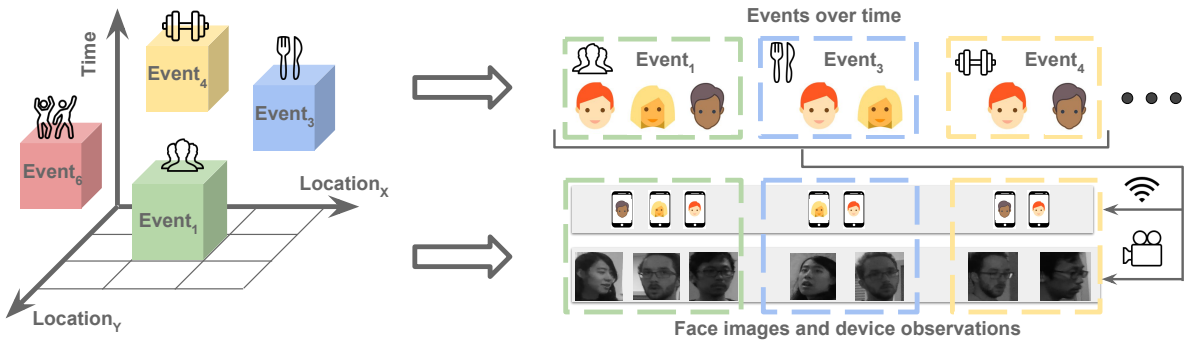


Fig. 2. An illustrative example of *events* and showing their linked face and device *observations*. An *event* is uniquely determined by spatial-temporal attributes and its participants. An event is also linked with two types of observations. The face observation are images cropped from surveillance videos and the device observation are sniffed MAC address of participants' devices.

to the target domain is referred as fine-tuning [46]. However, our problem of DFR adaption is distinct in that the labels in the source domain and target domain are different, i.e., different subjects. Moreover, as the source face images for pre-training is sensitive that are not sharable with target domain users. Both of them make the conventional distribution alignment methods in transfer learning non-applicable in this domain adaption problem [12].

3 PRELIMINARIES

3.1 Key Terms

In this section, we introduce some key terms and background that pave the way for §4.

3.1.1 Event. As we will soon discuss in the next section, a key concept in AutoTune is that of an *event*. An event $e \in E$ is the setting in which people interact with each other in a specific part of the environment for a given time interval. It is uniquely identified by three attributes: effective timeslot, location, and participants. Fig. 2 demonstrates a few examples of events. As we can see, an instance of an event might be a meeting in a conference room, exercise in a gym, or a lunch in the canteen. The spatio-temporal attributes of an event, i.e., timeslot and location, can either be obtained by side channel information (e.g. calendar), or for simplicity, they can be assumed to be fixed size cubes. However, the participants of an event, especially if it happens in a public space, are difficult to directly obtain. Details about how we create events in real-word experiments will be introduced in §5.3.3.

3.1.2 Observations. A smart camera and a WiFi sniffer are assumed to sense the identities in an environment. With the rapid development of camera systems and increasing deployment of WiFi access points, our assumption is practical. With these two sensor modalities. Two types of observations can be attached by an event: i) cropped face images detected by the cameras, and ii) device observations (MAC addresses) which give weak user attendance or presence information through WiFi sniffing. Fig. 2 illustrates the relationship between identity observations and events through examples.

3.1.3 Pre-trained Model. A pre-trained model is used to transform the face images into discriminative features or representations, thereby aiding accurate clustering or classification. In this work, a deep neural network (DNN) for discriminative face representation is trained by a model provider, e.g., Facebook or Google, who are able to access a large number of annotated face images. This model needs to be fine-tuned by smart-cameras in a particular space or environment, e.g., images acquired from an office or home. Note that our formulation does not

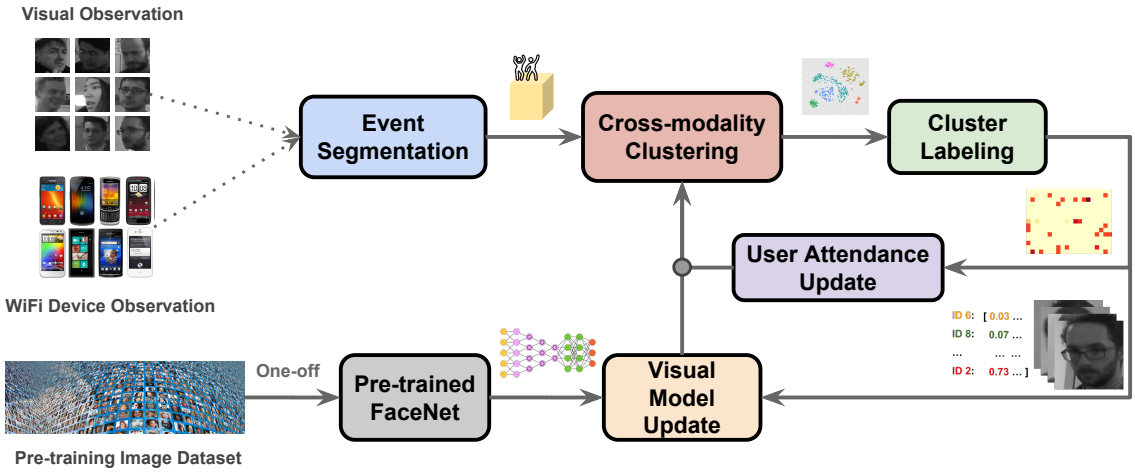


Fig. 3. Workflow of AutoTune. AutoTune consists of 5 steps i) Event segmentation ii) Cross-Modality clustering iii) Cluster Labeling iv) Visual Model Update v) User Attendance Update. AutoTune sequentially repeats the above five steps until the changes in the user attendance model are negligible.

require access to the images used for pre-training, just the weights and architecture of the pre-trained network model.

3.2 Challenges and Objectives

AutoTune is motived by the fact that, neither of the above sensor observations can *directly* provide *accurate* identity information. Face images are strong biometrics, but they cannot directly convey identity information without external labels i.e. a recognition model is needed to transfer biometrics to actual identity. The pre-trained model is no exception here. Given the mapping between MAC address and user identity, device observations are able to provide unique and direct identity information. In practice, the mapping between a MAC address to a user is easy to get for cameras in smart spaces. In order to authenticate legitimate WiFi users, say Eduroam [1], device MAC addresses and user account information are bound together. However, device observations are usually noisy because a detected MAC address does not necessarily indicate that the device owner is co-located. Therefore, the objectives of AutoTune are two-fold: i) automatically label face images that have noisy observations; and ii) reliably adapt the pre-trained model to recognize the faces of users in new environments.

4 AUTOTUNE SOLUTION

We are now in a position to introduce the solution, *AutoTune*, which assigns IDs to images from noisy observations of images and WiFi MAC addresses, and uses such learned ID-image associations to tune the pre-trained deep face representation model automatically. Our approach is based on two key observations: i) although collected by different modalities, both face images and device MAC addresses are linked with the identities of users who attend in certain events; and ii) the tasks of model tuning and face-identity association should not be dealt with separately, but rather progress in tandem.

Based on the above insights, AutoTune works as follows (see Fig 3). i) *Event Segmentation:* Given the face images and sniffed WiFi MAC addresses, AutoTune first segments them into events based on the time and location

[1] https://en.wikipedia.org/wiki/Eduroam

they were captured. ii) *Cross-Modality Clustering:* Then the face images are clustered based on their appearance similarity computed by the pre-trained face representation model, and also taking into account information on device attendance in events. iii) *Cluster Labeling:* Each image cluster should broadly correspond to a user, and the cluster's images are drawn from a set of events. AutoTune assigns each cluster to the user whose device has been detected in as similar as possible set of events. iv) *Visual Model Update:* Once images are labeled with user identity labels, AutoTune then fine-tunes the pre-trained face representation model. v) *User Attendance Update:* We further use the cluster labels to update our belief on which device (MAC address) has participated in each event. With the two models updated, we are in a position to iteratively repeat the steps of clustering, labeling, and model updates, until the changes in the user attendance model become negligible.

4.1 Event Segmentation

As defined in §3.1.1, an event is denoted as e. We assume a collection of n images $\{x_i \in X\}_{i=1}^{n}$ are collected by one or more smart cameras, which contain the cropped face images of users. These will have wide variability, e.g., different views, poses, scales and illumination. Let $\{l_j \in L\}_{j=1}^{m}$ be a set of m WiFi MAC addresses, i.e., m people of interest (POI) to recognize.

Assuming after data collection, we have images $\{x_i \in X\}_{i=1}^{n}$ spanning multiple events, e.g., over the period of several days. AutoTune segments images into $|E|$ subsets, namely the number of events. An image subset $\{x_{e_k,i} \in X_{e_k}\}$ in event e_k is determined by the captured time and camera location. For instance, any images captured during the timeslot of event e_k, e.g, 14:00-15:00 on the 1st of May, by cameras in a particular room, are put into the image subset X_{e_k}. All the images in the same subset share the same event index. It is worth noting that one subject could have multiple face images in $\{x_{e_k,i} \in X_{e_k}\}$, e.g., captured at different lighting conditions or viewing angles. On the other hand, by inspecting the sniffed MAC addresses, we can also maintain an m-dimensional attendance observation vector $I_{e_k}^{\tau}$ for each event. $I_{e_k}^{\tau}[j]$ which is set to 1 only if the user j's MAC address $l_j \in L$ is detected in the event e_k. We refer readers to Fig. 2 to see the examples of such event attachments. Due to the uncertain nature of WiFi sensing, the initial attendance observation $I_{e_k}^{\tau}$ is noisy and should be iteratively updated. These event attachments build *coarse-grained* correlations between a set of face images and a set of device IDs, through their shared event e_k. However, at this stage the fine-grained mapping between an image $x_{e_k,i}$ and a MAC address $l_{e_k,j}$ is unknown. Note that, in the following sections, we use the superscript τ to denote the τ-th iteration of those variables who will be updated in AutoTune.

4.2 Cross-modality Clustering

Before assigning fine-grained ID labels, AutoTune needs to cluster images across events. Unlike conventional clustering that merely depends on the feature similarity, AutoTune merges face images *across events* into a cluster (potentially belonging to the same subject) by incorporating attendance information as well. Recall that the attendance observation vector $I_{e_k}^{\tau}$ already reveals the identities of subjects (in the form of MAC addresses) in the event e_k, and the captured images X_{e_k} may contain the faces of these subjects. Despite the noise in observations, the overlapped subjects in different events can be employed as a prior that guides the image clustering. For example, if there are no shared MAC addresses sniffed in two events, then it is very likely that the face images captured in these two events should lie in different clusters.

Formally, the likelihood that two cross-event face images $x_{e_k,i}$ and $x_{e_p,j}$ belong to the same subject is conditioned on two factors: i) the similarity of their feature representation between $z_{e_k,i}^{\tau}$ and $z_{e_p,j}^{\tau}$; and ii) the overlap ratio between the attendance observations in their corresponding events. The resulted joint similarity is a log likelihood $log(\mathbf{Pr}(y_{e_k,i}^{\tau} = y_{e_p,j}^{\tau}))$ defined as follows:

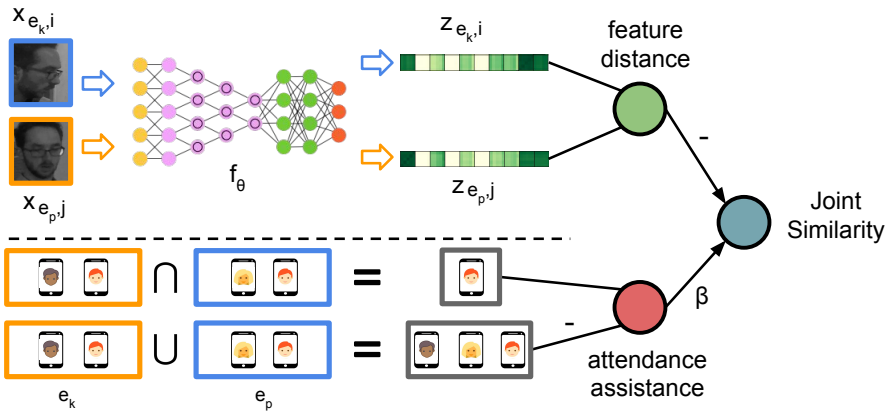


Fig. 4. Cross-modality clustering. The final similarity is jointly determined by attendance assistance derived from event logs and similarity of features transformed via f_θ, see Equ. 1. Detailed explanations can be found in §4.2.

$$\begin{aligned} \mathbf{Pr}(y^\tau_{e_k,i} = y^\tau_{e_p,j}) &\propto \frac{exp(\beta * |I^\tau_{e_k} \otimes I^\tau_{e_p}|)}{exp(\beta * |I^\tau_{e_k} \oplus I^\tau_{e_p}|)} * exp(-D(z^\tau_{e_k,i}, z^\tau_{e_p,j})) \\ log(\mathbf{Pr}(y^\tau_{e_k,i} = y^\tau_{e_p,j})) &\propto \beta * \underbrace{(|I^\tau_{e_k} \otimes I^\tau_{e_p}| - |I^\tau_{e_k} \oplus I^\tau_{e_p}|)}_{\text{attendance assistance}} - \underbrace{D(z^\tau_{e_k,i}, z^\tau_{e_p,j})}_{\text{feature distance}} \end{aligned} \tag{1}$$

Here $\otimes$ and $\oplus$ are element-wise AND and OR, and $|\cdot|$ here is the L^1-norm. z is the features transformed by the face representation model and D is a distance measure between face features. β is a hyper-parameter that controls the contributions of the attendance assistance and feature similarity. Fig. 4 demonstrates how the joint similarity between two cross-event images is determined. The above derivation is inspired by the Jaccard coefficients, with the difference lying in the log function. The rationale behind the term $|I_{e_k} \oplus I_{e_p}|$ is that the more different subjects attending events, the more uncertain that any two images drawn across these events will point to the same subject. In contrast, when the intersection $|I_{e_k} \otimes I_{e_p}|$ is significant enough, the chance that these two images point to the same subject will become higher. Based on the joint similarity, images across events are grouped into g clusters. We will soon discuss how to determine g based on the complete set of MAC addresses $\{l_j \in L\}_{j=1}^m$ in §4.3.2.

4.3 Cluster Labeling

4.3.1 Fine-grained ID Association. After clustering, an image cluster is linked with multiple events that are associated with its member images. We introduce an event vector $V^\tau_{G_i}$ for an image cluster G_i, with $V^\tau_{G_i}[k]$ that is set to 1 only if it contains images attached with the event e_k. The length of the event vector is determined by the number of events $|E|$. Fig. 5 shows an example of how an event vector is developed. Similarly, for a device ID l_j, its corresponding event vector $V^\tau_{l_j}$ can be determined by inspecting its occurrences in all user attendance observations. $V^\tau_{l_j}[k]$ is set to 1 only if the device ID (MAC address) l_j is detected in the event e_k. AutoTune then assigns cluster G_i with device IDs based on the matching level of their event vectors. Formally, a matching

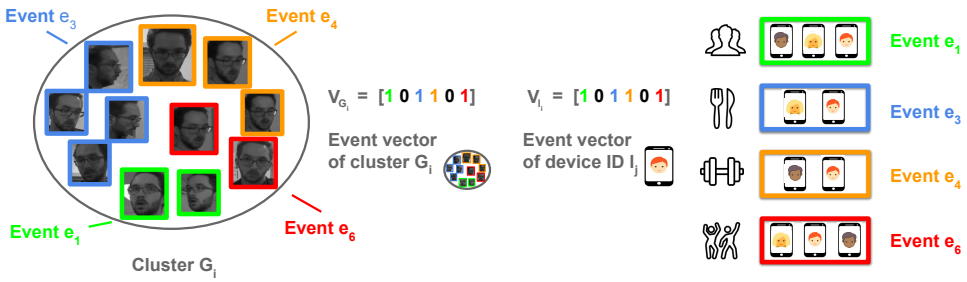


Fig. 5. Event vectors of an image cluster and device ID. The kth element of the vector $V_{G_i}[k]$ is set to 1 only it contains images attached with the event e_k. The event vector V_{l_j} of the device ID l_j is similarly developed. See §4.3 for more details.

problem of bipartite graph can be formulated as follows:

$$\mathcal{L}_A = \sum_{ij} a_{ij}^{\tau}(V_{G_i}^{\tau} - V_{l_j}^{\tau})^2$$
$$\sum_{1 \le j \le m} a_{ij}^{\tau} = 1, \quad \forall i \in \{1, \ldots, g\} \tag{2}$$

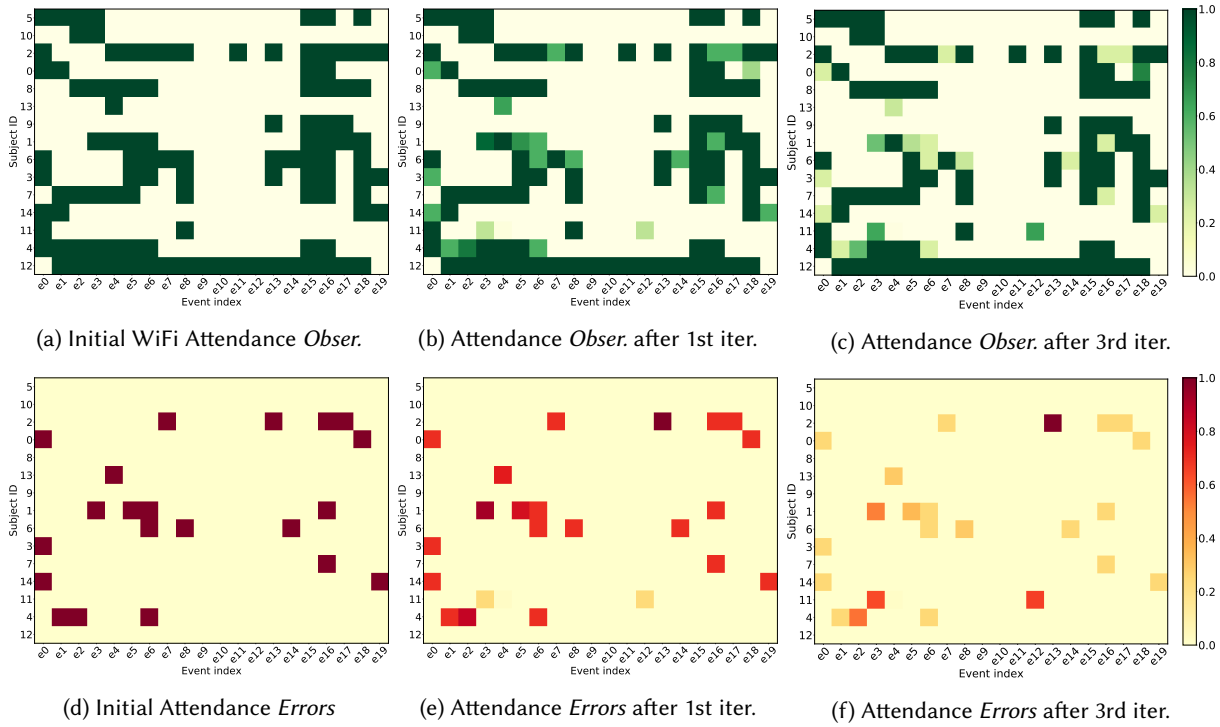
where the solution of the binary variable a_{ij}^{τ} assigns a device ID to a cluster, and the images in a cluster are labeled by the same assigned ID. We note that when $m < g$, AutoTune adds dummy nodes to create the complete bipartite graph. Then the complete bipartite graph problem can be solved by the Hungarian algorithm [16].

4.3.2 Voting for Non-POI. We now obtain the instance-level ID association. However, in practice, the number of clusters g can vary due to the captured face images outside the POI, e.g., face images of short-term visitors. The choice of g has significant impact on the performance of clustering [10] which could further affect the following fine-tuning performance. To include the non-POI, the number of clusters g should be greater than the number of POI m, but the exact number of g is impossible to know beforehand. To bypass this issue, we set the cluster number g, from m to $3 * m$ and run the clustering (§4.2) and association (§4.3.1) for multiple rounds. AutoTune is robust to extra clusters of non-POI, when g is greater than m. Although there are some unused clusters of non-POI after clustering, the association step will sieve them and only choose the most consistent m clusters with the event logs of m device IDs. Lastly, for every image, its assigned ID is finalized by decision voting on the individual association results computed with different g, ranging from m to $3 * m$.

Typically, a naive voting procedure would give a hard ID label for an image and use these <Image, ID> pairs as training data to update the face representation model. However, due to the noise in device presence observations and errors in clustering, there will be mis-labeling in the training data that may confuse the model update. To account for the uncertainty of ID assignment, we adopt soft labels for voting. More specifically, instead of voting for the most likely ID y_i^{τ} for an image x_i, we introduce a normalized vector m-dimension $\overrightarrow{y}_i^{\tau}$, which is a valid probability distribution. For example, $\overrightarrow{y}_i^{\tau}[j] = 0.4$ means that there are 40% associations assigning the ID j to the image i. Moreover, the soft labeled data $< x_i, \overrightarrow{y}_i^{\tau} >$ can be compatible with the computation of cross-entropy. This voting procedure is essentially a smoother for assignments of multiple hypotheses. It copes with the uncertain number of captured subjects by averaging multiple association proposals.

4.4 Visual Model Update

We are now in a position to introduce the model update in AutoTune. By taking the labeled data as inputs, the face representation model f_{θ}^{τ} updates its weights and biases θ to $f_{\theta}^{\tau+1}$.



(a) Initial WiFi Attendance *Obser.* (b) Attendance *Obser.* after 1st iter. (c) Attendance *Obser.* after 3rd iter.

(d) Initial Attendance *Errors* (e) Attendance *Errors* after 1st iter. (f) Attendance *Errors* after 3rd iter.

Fig. 6. An example of user attendance update (§4.5) in AutoTune, a 15-subject subset of 20 events. **Top**: Attendance observations in events; **Bottom**: (absolute) errors of attendance observations in events. It can be seen that the IDs in events are gradually corrected as the model updates in AutoTune.

4.4.1 Discriminative Face Representation. Face representation learning optimizes a representation loss $\mathcal{L}_R$ to force the learnt features as discriminative as possible. Strong discrimination bears two properties: inter-class dispersion and intra-class compactness. Inter-class dispersion pushes face images of different subjects away from one another and the intra-class compactness pulls the face images of the same subject together. Both of the properties are critical to face recognition. At iteration τ, given the current labels y_i^τ for the ith face image x_i and the transformed features $z_i^\tau = f_\theta^\tau(x_i)$, the representation loss $\mathcal{L}_R$ is determined by a composition of softmax loss and center loss:

$$\begin{aligned}\mathcal{L}_R &= \mathcal{L}_{softmax} + \mathcal{L}_{center} \\ &= \underbrace{\sum_i -log(\frac{e^{W_{y_i^\tau}^T z_i^\tau + b_{y_i^\tau}}}{\sum_{j=1}^m e^{W_{l_j}^T z_i^\tau + b_{l_j}}})}_{\text{softmax loss}} + \underbrace{\sum_i \frac{\lambda}{2}||z_i^\tau - c_{y_i^\tau}||^2}_{\text{center loss}}\end{aligned} \tag{3}$$

where W and b are the weights and bias parameters in the last fully connected layer of the pre-trained model. The center loss $\mathcal{L}_{center}$ explicitly enhances the intra-class compactness while the inter-class dispersion is implicitly strengthened by the softmax loss $\mathcal{L}_{softmax}$ [44]. λ is a hyper-parameter that balances the above sub-losses.

4.4.2 Fuzzy Center Loss. The center loss $\mathcal{L}_{center}$ in Equ. 3 is shown to be helpful to enhance the intra-class compactness [44]. However, we cannot directly adopt it for fine-tuning as computing the centers requires explicit labels (see Equ. 3) of images, but the association steps above only provide probabilistic ones through soft labels. To solve this, we propose a new loss called *fuzzy center loss* $\mathcal{L}_{fuzzy}$ to replace the center loss. Similar to the core idea of fuzzy clustering [45], we allow each face image to belong to more than one subject. In fact, the membership grades indicating the degree to which an image belongs to each subject can be directly retrieved from the soft labels and the fuzzy center c_k^τ is given as:

$$c_k^\tau = \frac{\sum_i^n z_i^\tau * (\overrightarrow{y}_i^\tau[k])^\eta}{\sum_i^n (\overrightarrow{y}_i^\tau[k])^\eta} \tag{4}$$

The fuzzifier η is a hyper-parameter that determines the level of cluster fuzziness. A large η results in smaller membership values and hence, fuzzier clusters. This gives the fuzzy center loss as follows:

$$\mathcal{L}_{fuzzy} = \sum_i \sum_k \overrightarrow{y}_i^\tau[k] * ||z_i^\tau - c_k^\tau||^2 \tag{5}$$

4.4.3 Fine-tuning. We leave the softmax loss $\mathcal{L}_{softmax}$ the same as in Equ. 3, because the soft labels are compatible with the computation of cross-entropy. The new representation loss to minimize is:

$$\mathcal{L}_R = \mathcal{L}_{softmax} + \mathcal{L}_{fuzzy} \tag{6}$$

AutoTune updates the model parameters θ^τ to $\theta^{\tau+1}$ based on the gradients of $\nabla_\theta \mathcal{L}_R$, which are calculated via back prorogation of errors. Compared with the dataset used for pre-training, which is usually in the order of millions [5], the data used for fine-tuning is much smaller (several thousands). The mis-match between the small training data and the complex model architecture could result in overfitting. To prevent overfitting, we use the dropout mechanism in training, which is widely adopted to avoid overfitting [19]. Meanwhile, as observed in [14, 33], the soft label itself can play the role of a regularizer, and make the trained model robust to noise and reduce overfitting.

4.5 User Attendance Update

The device presence observations by WiFi sniffing are noisy because the WiFi signal of a device is opportunistic and people do not carry/use theirs devices all the time. Based on the results of the cluster labelling step introduce in §4.3, we have the opportunity to update our belief on which users attended each event.

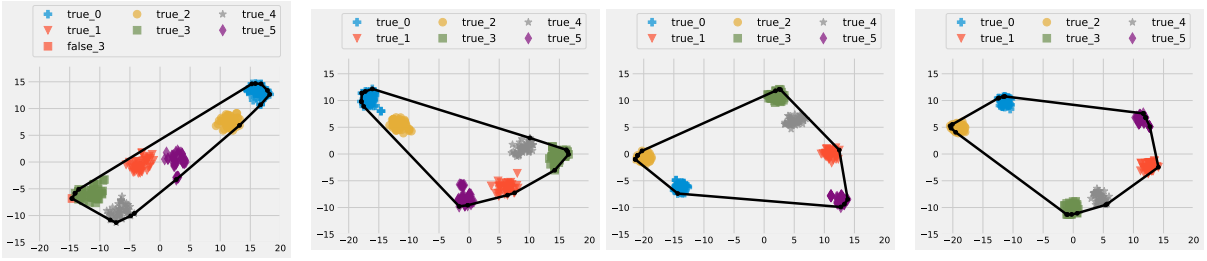
The update mechanism is as follows: Each image is associated with a user probability vector, whose elements denote the probability that the image correspondes to a particular user. By averaging the user probability vectors of all images that have been drawn from the same event e_k, and normalizing the result, we can estimate the user attendance of this event. The elements of the resulting user attendance vector $\hat{I}_{e_k}$ denote the probabilities of different users attending event e_k.

We can now use $\hat{I}_{e_k}$ as a correction to update our previous wifi attendance vector $I_{e_k}^\tau$ as follows:

$$I_{e_k}^{\tau+1} = I_{e_k}^\tau - \beta \cdot (I_{e_k}^\tau - \hat{I}_{e_k}) \tag{7}$$

where β is a pre-defined parameter that controls the ID update rate. In principle, a large update rate will speed up the convergence rate, at the risk of missing the optima.

AutoTune sequentially repeats the above steps of clustering, labelling and model updates, until the changes in the user attendance model are negligible. Algorithm. 1 summarizes the workflow.



(a) Pre-trained Model (b) 1st iteration (c) 2nd iteration (d) 3rd iteration

Fig. 7. An example of visual model update (§4.4) in AutoTune with a six subject subset. Note that the pre-trained model in Fig. 7a has a narrow convex hull (black lines), clusters are dispersed and there is misclassification. After the first iteration, clusters become more distinct and span more of the embedding space. Over time, clusters become more compact and move further away from each other, leading to purer clustering results.

ALGORITHM 1: AutoTune

Input: pre-trained model f_θ^0, WiFi sniffed attendance observations I^0, images X, number of POI m, threshold ξ
Output: adapted model f_θ^*, corrected attendance observations I^*, soft image labels Y^*
Initialize: Segment attendance observations I^0 and images X into subsets $I_{e_k}^0$ and X_{e_k} based on events E ▷ §4.1
$\tau = 0$
while $\sqrt{\frac{1}{|E|}\sum_{k=1}^{|E|} ||I_{e_k}^{(\tau)} - I_{e_k}^{(\tau-1)}||^2} > \xi$ **do**
  $Z^\tau = f_\theta^\tau(X)$ ▷ feature transformation
  for $g \leftarrow m$ *to* $3 * m$ **do**
    $G^\tau \leftarrow$ `cross_modality_clustering`$(Z^\tau, I^\tau, \mathrm{g})$ ▷ §4.2;
    $A_g^\tau \leftarrow$ `ID_association`(G^τ, I^τ) ▷ §4.3.1;
  end
  $Y^\tau \leftarrow$ `voting`(A^τ) ▷ §4.3.2;
  $c^\tau \leftarrow$ `fuzzy_center`(Y^τ, Z^τ) ▷ §4.4.2;
  $f_\theta^{\tau+1} \leftarrow$ `visual_model_update`$(Y^\tau, c^\tau, f_\theta^\tau, \mathrm{X})$ ▷ §4.4.3 ;
  $I^{\tau+1} \leftarrow$ `user_attendance_update`(Y^τ, I^τ) ▷ §4.5 ;
  $\tau \leftarrow \tau + 1$
end

5 IMPLEMENTATION

In this section, we introduce the implementation details of AutoTune. The code reproducing our system is publicly available at https://github.com/[no.name.for.bind.review].

5.1 Data Collection Tools

AutoTune automatically labels face images cropped from surveillance cameras by leveraging sniffed WiFi signals. Therefore there are two modules in our data collection system, a face extraction module and a WiFi sniffing module [2].

[2]The study has received ethical approval XXX [no name for blind review]

5.1.1 Face Extraction Module. This module consists of a front-end remote camera and a back-end computation server. Specifically, the remote camera in our experiment is a *GoPro Hero 4* [3]. The camera is able to communicate and transfer data to the back-end through a wireless network. To avoid capturing excess data without people in it, we implement a motion detection module in our system with a circular buffer. The system works by continuously taking low-resolution images, and comparing them to one another for changes caused by something moving in the camera's field of view. When a change is detected, the camera takes a higher-resolution video for 5 seconds video and goes back to look for changes. All the collected videos are sent back to the backend at midnights. On the backend, a cascaded convolutional network based face detection module [47] is used to remove videos if no faces are detected in their frames. The cropped faces from the remaining videos are supplied AutoTune. Note that this implementation could easily be replaced with an off-the-shelf IP-based smart camera.

5.1.2 WiFi Sniffing. This module is realized on a WiFi-enabled laptop running Ubuntu 14.04. Our sniffer uses Aircrack-ng [4] and tshark [5] to opportunistically capture the WiFi packets in its surrounding. The captured packet has unencrypted information such as transmission time, source MAC address and the Received Signal Strengths (RSS). As AutoTune aims to label face images for POI, our WiFi sniffer only records the packets containing POI's device MAC addresses and discards them otherwise, so as to not harvest addresses from people who have not given consent. A channel hop mechanism is used in the sniffing module to cope with cases where the POI's device(s) may connect to different WiFi networks, namely, on different wireless channels. The channel hop mechanism forces the sniffing channel to change by every second and monitor the active channels periodically (1 second) in the environment. The RSS value in the packet implies how far away the sniffed device is from the sniffer [13]. By putting the sniffer near the camera, we can use a threshold to filter out those devices with small RSS values, e.g., less than -55 dBm in this work, as they are empirically unlikely to be within the camera's field of view.

5.2 Face Recognition Model

The face representation model used in AutoTune is FaceNet [34], which is a state-of-the-art face recognizer. We now introduce how to pre-train and fine-tune FaceNet in this work.

5.2.1 Pre-training. FaceNet adopts the Inception-ResNet-v1 [37] as its backbone and its weights are pre-trained on the VGGFace2 dataset [5]. This dataset contains 3.31 million images of 9131 subjects, with an average of 362.6 images for each subject. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession. Pre-training is supervised by the triplet loss [7] and the training protocols, e.g, parameter settings, can be found in [34]. We found that the learnt face representation by FaceNet is generalizable and it is able to achieve an accuracy of 99.65% on the LFW face verification task [6]. As claimed in [34], FaceNet not only learns a powerful recognition model but also gives a very discriminative feature representation that can be used for image clustering. In fact, though not in the same form, the triplet loss used for FaceNet pre-training has a similar effect as the center loss (see §4.4.1) that they both enhance the intra-class compactness and diminish inter-class dispersion.

5.2.2 Fine-tuning. AutoTune fine-tunes FaceNet in every model update iteration. The fine-tuning process is detailed in §4.4.3, and we specify the fine-tuning protocol. After each round of label association (§4.3), the labeled data is split into a training set and validation set, with a ratio of $8:2$ respectively. The pre-trained FaceNet is then fine-tuned on the training set, and the model that achieves the best performance on the validation set is

[3] https://shop.gopro.com/cameras
[4] https://www.aircrack-ng.org/
[5] https://www.wireshark.org/docs/man-pages/tshark.html
[6] http://vis-www.cs.umass.edu/lfw/

saved. Note that the fine-tuning process in AutoTune does not involve the test set. As we discuss in the §6.3, the online testing is performed on a held-out set that is collected on different days. To enhance the generalization ability, we use dropout training for regularization [43]. The dropout ratio is set to 0.2. We set the batch size to 50 and one fine-tuning takes 100 epochs.

5.3 Parameter Configuration

5.3.1 Face Detection. As discussed in §5.1, we use a cascaded convolution network to detect faces in videos. It is cascaded by three sub-networks, a proposal network, a refine network and an output network. Each of them can output bounding boxes of potential faces and the corresponding detection probabilities, i.e., confidences. Face detection with small confidence will be early discarded and not sent to the next sub-network. In this work, the confidence threshold is set to 0.7, 0.7 and 0.9 for three sub-networks respectively. Following the original setting in [47], we set the minimal face size in detection to 40×40 pixels.

5.3.2 Clustering Algorithm. In §4.2, we use a clustering algorithm to merge the images across events. Specifically, the clustering algorithm used in AutoTune is the agglomerative clustering [3], which is a method that recursively merges the pair of clusters that minimally increases a given linkage distance. The similarity metric adopted in our agglomerative clustering algorithm the is Euclidean distance. A linkage criterion determines which distance to use between sets of data points. In AutoTune, this linkage criterion is set to the average of the distances of each observation of the two sets.

5.3.3 Event Duration. Depending on the context, the duration of events can be variable. However, for simplicity we use fix-duration events in this work. Specifically, we split a day into 12 intervals, each of which is 2 hours. We then discard those events that has neither face images nor the sniffed MAC addresses.

5.3.4 Threshold of Convergence. The convergence of AutoTune depends on when the residual of ID predictions in two iterations are below a certain threshold ξ (see Algorithm. 1). Apparently, ξ affects the convergence speed, however, we fix it to 0.01 to control the variables in the following evaluations. In practice, we observe that AutoTune can work comparably for a relatively wide range of ξ.

6 EVALUATION

In this section, we evaluate the proposed AutoTune extensively on datasets collected from both real-world experiments and simulation. We've deployed two testbeds, one in the UK and the other in China, and collected datasets as described in the previous section. The simulation dataset is developed based on a public dataset of face images.

6.1 Competing Approaches and Evaluation Metrics

6.1.1 Competing Approaches. We compare the performance of AutoTune with the 3 competing approaches:

- **Template Matching (TM)** [4] employs a template matching method to assign ID labels to clusters of face images. This is the most straightforward method that is used when one or more profile photos of POI are available, e.g., crawled from their personal homepage or Facebook.
- **One-off Association (OA)** [24] uses one-off associations to directly label the image clusters without fine-tuning of the face representation model itself. This approach is equivalent to adopting the results of AutoTune after the first half of an iteration.
- **Deterministic AutoTune (D-AutoTune)** is the deterministic version of AutoTune. In D-AutoTune, the association and update procedures are the same as in AutoTune, but it adopts hard labels rather than soft labels and use the simple center loss instead of the proposed fuzzy center loss (see §4.3.2).

6.1.2 Evaluation Metrics. AutoTune contains two main components, offline label assignment and online inference. For the offline label assignment, we evaluate its performance with the following metrics:

$$\begin{aligned} Precision &= \frac{TP}{TP+FP} \\ Recall &= \frac{TP}{TP+FN} \\ F_1 &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \\ Accuracy &= \frac{TP+FN}{TP+TN+FP+FN} \end{aligned} \tag{8}$$

where TP, TN, FP, FN are true positive, true negative, false positive and false negative respectively. Each metric captures different aspects of classification [32]. Online face recognition has two kinds of tests, face identification and verification. Identification is a multi-class classification task and aims to find an unknown person in a dataset to answer the question "who is the person". Verification is a binary classification task and aims to check if this person is the correct one. We follow [20, 22] to use the Cumulative Match Characteristic (CMC) for identification evaluation and the Receiver Operating Characteristic (ROC) for verification evaluation [8].

6.2 Offline Cross-modality Face Labeling

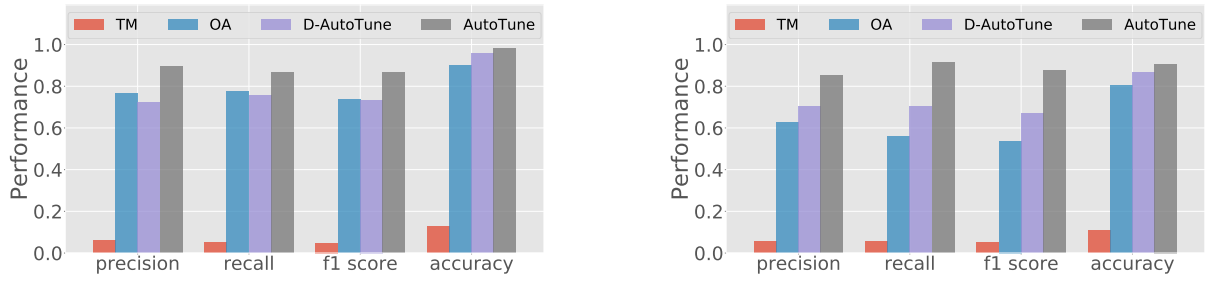
AutoTune automatically labels images captured in the wild by exploiting their correlations with the device presences. The quality of image labeling is crucial for the follow-up face recognition in the online stage. In this section, we investigate the image labeling performance of AutoTune.

6.2.1 Data Collection. We have deployed the data collection system as discussed in §5.1 in the following two testbeds:

- **Office:** This first dataset is collected in an office building in the UK, where 20 long-term occupants work inside an office. These occupants are naturally chosen as people of interest (POI). Face images are captured with a surveillance camera that faces the office entrance. The presence logs of occupants' WiFi MAC addresses are collected by a sniffer that is situated in the center of the office. In total there are 13, 707 face images collected from 8am to 11pm over two weeks. These images are mixed with the faces of 20 occupants and 11 short-terms visitors who came to this office during the experiments. Along with the face images, the presence logs of occupants WiFi MAC addresses are also collected for the same time period.
- **CommonRoom:** We collect another dataset in a common room of a university in China. There are no long-term occupants in this site and all registered undergraduates can enter. Of the 37 people that appeared during the three week experiment period, 12 subjects are selected as the POI, and their WiFi MAC address presence is continuously recorded by our sniffing system. The other settings remain the same as the *office* dataset. The challenge in this dataset lies in that the captured face images, both for POI and non-POI, are all of Asian people, while the initial face representation model is trained on the database of primarily Caucasian people.

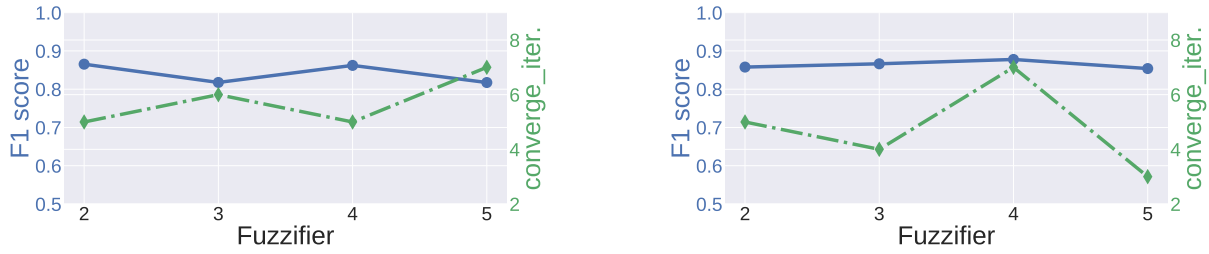
6.2.2 Label Assignment Performance. Fig. 8 shows the performance of label assignment, i.e., matching an identifier to a face image. For the office dataset, AutoTune outperforms the best competing approach (OA), by 0.13 in F_1 score and 7% in accuracy. The advantage of AutoTune is more obvious in the CommonRoom experiment where it beats the best competing approach (OA) by 0.34 in F_1 score and 10% in accuracy.

As the only method that uses the website images (one-shot learning) to label images rather than the device ID information, TM struggles in both experiments and is 9-fold worse than AutoTune. We observe that the website face images are dramatically different from the captured images in real-world, due to different shooting conditions



(a) Office (b) CommonRoom

Fig. 8. Label assignment performance of the various techniques compared with AutoTune across the two different datasets.



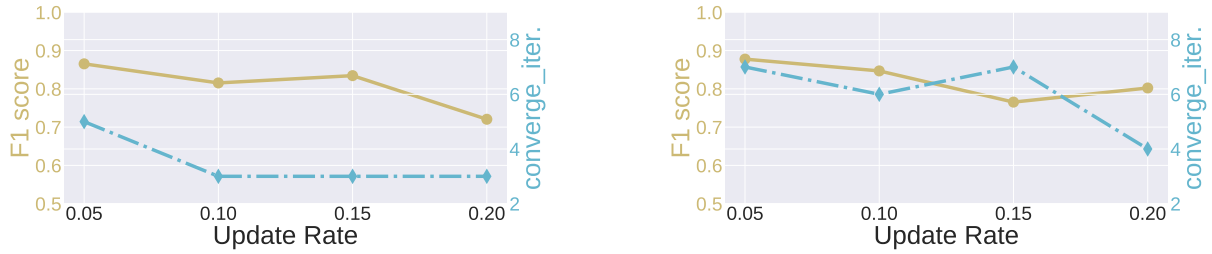
(a) Office (b) CommonRoom

Fig. 9. Impact of fuzzifier η (introduced in §4.4.2). It can be seen that the F_1 score of AutoTune is not very sensitive to the fuzzifier. The convergence is affected, but none of cases exceeds 7 iterations on both datasets.

and image quality. These results implies that, although the device observations via WiFi sniffing are noisy, when the amount of the them is enough, they can be more informative than face images crowd-sourced on the web.

Meanwhile, we note that adopting soft labels (see §4.3.2) can further improve the labeling performance. In terms of F_1 score, the full-suite AutoTune is around 15% and 22% better than D-AutoTune in two experiments respectively. Similar improvements are witnessed in terms of the accuracy. The reason of larger performance gap in the CommonRoom dataset is that there are more images of non-POI being captured in this experiment. In addition, the pre-trained facenet model does not generalize very well to Asian faces, which are the primary ethnicity in the CommonRoom environment. In fact, all the method perform worse in the CommonRoom experiment except for the full-suite AutoTune.

6.2.3 Impact of fuzzifier η. In §4.4.2, we introduced the fuzzifier η for the fuzzy center loss. η is a hyper-parameter that handles uncertainties in labels when updating the centers. The larger η is, the more uncertain the labels are. This section investigates AutoTune's sensitivity to this hyper-parameter. We freeze the update rate β (will soon discuss in §6.2.4) to 0.05 and vary the fuzzifier in Equation. 4 from 2 to 5 at a step length of 1. For the office experiment, AutoTune achieves the best F_1 score of 0.86 when fuzzifier is set to 2, though the standard deviation on all examined fuzzifier values is only 0.023 (see Fig. 9a). We see a smaller standard deviation ($\sim$ 0.009) on the CommonRoom dataset, and the best F_1 score of 0.88 is achieved when η is set to 4 (see Fig. 9b). Overall, AutoTune is not very sensitive to the fuzzifier and its performance remains very stable for a relatively wide range of η. This good property enables users set η relatively freely for AutoTune without much domain knowledge of



(a) Office (b) CommonRoom

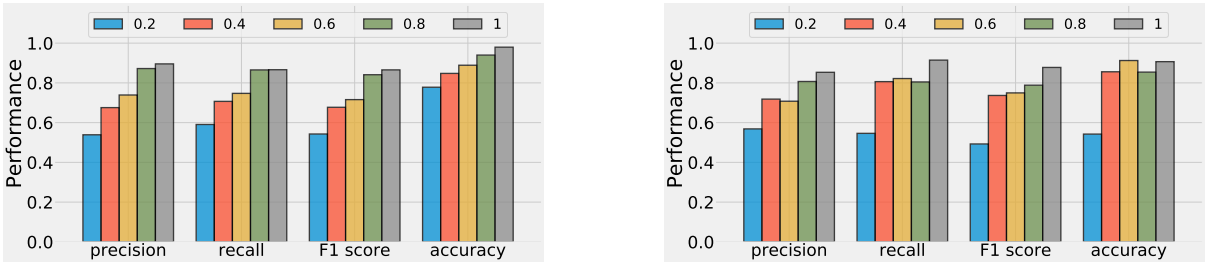
Fig. 10. Impact of ID update rate β (introduced in §4.5). It can be seen that a high update rate causes the network to converge rapidly to potentially incorrect assignments.

Table 1. Performance of AutoTune of various sensitivity tests in both real-world environments.

	Office								**CommonRoom**							
	Fuzzifier η				**Update Rate β**				**Fuzzifier η**				**Update Rate β**			
Control Variable	**2**	**3**	**4**	**5**	**0.05**	**0.1**	**0.15**	**0.2**	**2**	**3**	**4**	**5**	**0.05**	**0.1**	**0.15**	**0.2**
Precision	0.90	0.81	0.87	0.82	0.90	0.85	0.87	0.73	0.83	0.83	0.85	0.82	0.85	0.81	0.75	0.77
Recall	0.87	0.82	0.86	0.83	0.87	0.86	0.86	0.75	0.90	0.92	0.91	0.91	0.91	0.90	0.84	0.86
F1 Score	0.87	0.82	0.86	0.82	0.87	0.82	0.83	0.72	0.86	0.87	0.88	0.85	0.88	0.85	0.77	0.80
Accuracy	0.98	0.98	0.98	0.95	0.98	0.91	0.92	0.90	0.89	0.89	0.91	0.89	0.91	0.89	0.89	0.88
Converge. Iteration	5	6	5	7	5	3	3	3	5	4	7	3	7	6	7	4

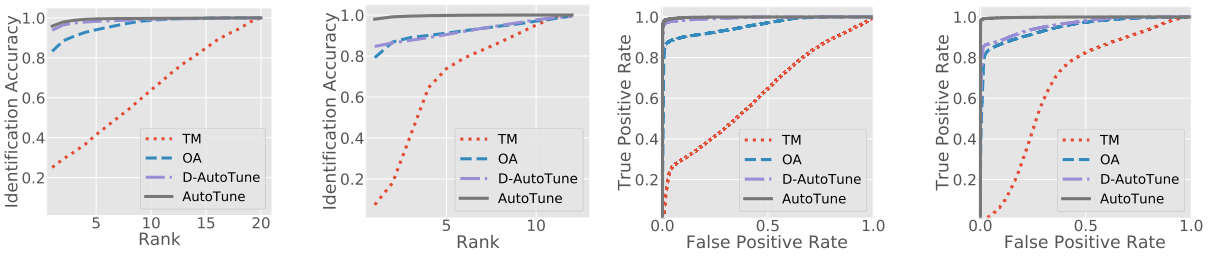
fuzzy clustering or the need for parameter tuning. Tab. 1 summarizes the other performance metrics of AutoTune under different fuzzifier values.

6.2.4 Impact of ID update rate β. This section investigates the impact of ID update rate β introduced in §4.5. The ID update uses the fine-tuned face recognition model to update the device ID observations. A large ID update rate forces the device ID observations to quickly become consistent with the deep face model predictions. However, a large update rate also runs the risk of missing the optima. We freeze the fuzzifier to 2 and 4 on Office and CommonRoom environments respectively and only vary the update rate β from 0.05 to 0.20 at a step length of 0.05. The choice of the fuzzifier value is based on the best results that discussed in §6.2.3. Fig. 10a demonstrates that AutoTune achieves the best performance on the office dataset when the update rate is set to 0.05. The performance declines by 10% when the rate rises to 0.2. This is because the updated ID observations quickly become the same as the model predictions and the model predictions are not quite correct yet. When it comes to CommonRoom dataset (see Fig. 10b), similar trend of F_1 score change can be seen. Although overall, the convergence becomes faster when the update increases, we observe that there is a fluctuation point at the update rate of 0.15, where AutoTune takes 7 iterations to converge. By inspecting the optimization process, we found that, under this parameter setting, AutoTune oscillated because the large update step makes it jump around the vicinity of optima but unable to approach to it furthermore. In practice, we suggest users of AutoTune to select their update rate from a relatively safe region between 0.05 to 0.1. Tab. 1 summarizes the other performance metrics of AutoTune under different update rates.



(a) Office (b) CommonRoom

Fig. 11. Impact of the number of events. It can be seen that as the number of events increases, the accuracy naturally increases as well, as there are more unique events to provide disambiguation.



(a) Office (b) CommonRoom

Fig. 12. Identification Performance

(a) Office (b) CommonRoom

Fig. 13. Verification Performance

6.2.5 Impact of number of events. AutoTune exploits co-located device and face presence in different events to establish the cross-modality correlations. In this section, we investigate the impact of the number of events on the performance of labeling. We vary the proportion of involved events in AutoTune by uniformly selecting them from 20% to 80%, and compare them with full set of events on the two datasets, respectively. As shown in Fig. 11, AutoTune performs better with increasing numbers of events on both datasets. The gap of F_1 score between the case with all events (100%) and case with the least amount of events (20%) can be as large as > 0.32 on both datasets. As discussed in §4.3.1, the ID association needs sufficiently diverse events to create discriminative enough event vectors. Otherwise, there will be faces or devices with the same event vectors that hinders AutoTune's ability to disambiguate their mutual association. However, we also observe that when 80% events are used, the performance of AutoTune remains stable on both datasets.

6.3 Online Face Recognition

After the cross-modality label association, an image database of POI is developed and an online face recognition system can be trained based on it. As in [34], we use the face representation model to extract face features and then use a classifier, say linear SVM in this work, to recognize faces. Particularly, as AutoTune and D-AutoTune are able to update the face representation model in their labeling process, the feature extractor for them are the updated model by themselves. For competing approaches that do not have an evolving model updates, we use the original Facenet model (in §5.2) as to extract face features and train the same classifier on the developed image-identity database by themselves. Without any overlapping with the images used for cross-modality labeling, the test

sets are collected in the same places, i.e., *office* and *CommonRoom*, but in different days. The ratio between the training set and test set of the SVM classifier is 7 : 3.

6.3.1 Face Identification. Fig. 12 compares the face identification results of both AutoTune and the competing approaches. Face identification is a multi-class classification task and aims to find an unknown person in a dataset of POI. As we can see, by using the face database developed by AutoTune, the identification accuracy quickly saturates and is able to have no errors within three guesses (rank-3) on both datasets. For the office dataset where there are 20 POI, the rank-1 accuracy of AutoTune can be as high as 95.8% and outperforms the best competing approach (OA) by 12.5%. The advantage of AutoTune is more significant on the CommonRoom dataset, and it leads the best competing approach OA by ∼ 19% (98.0% vs 79.1%). In addition, although D-AutoTune's performance is inferior to AutoTune, it is still more accurate than competing approaches, especially on the office dataset. We note that these results are consistent with face labeling results in §6.2.2. Overall, the face identification systems built by AutoTune is highly accurate, considering that AutoTune is only supervised by the weak and noisy device presence information. Face identification error analysis at subject level is discussed in Appendix A.1.

6.3.2 Face Verification. We further investigate another online recognition task, face verification, which is a binary classification task aiming to check if the observed person is the correct one. Like the identification case, we use the face database built by different approaches and and adopt the adapted face representation modelto extract face features where are applicable. We then develop an one-vs-all classifier for each subject. Fig. 13 shows that AutoTune is able to give a very reliable verification performance. For the office dataset, the Area Under Curve (AUC) of AutoTune is 99.8%, with an equal error rate (EER) of 1.5%. Comparable performance can be found on the CommonRoom dataset as well, where the AUC of AutoTune is 99.8% and the EER is 1.1%. Again, AutoTune and D-AutoTune completely outperform the other approaches, where D-AutoTune is less effective than AutoTune. These results imply that the face verification system built by AutoTune is sufficiently reliable and can serve for many challenging application scenarios.

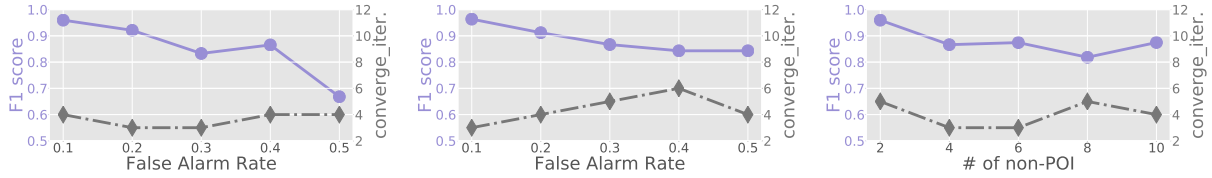
6.4 Robust Analysis in Simulation Environment

In order to further examine the label-association performance of AutoTune when faced with different conditions, we've also conducted simulation studies. The benefit of simulation is that we have the full control over the data, and we can alter various settings to validate AutoTune under various adversarial conditions. From the experimental results on the above two real-world datasets, we discover the following three types of noise that could impact the performance of AutoTune.

- **False-alarm Faces** is the case that in an event, the number of distinct detected faces of POI is *more than* the number of distinct sniffed MAC addresses.
- **False-alarm Devices** is the case that in an event, the number of detected faces of POI is *less than* the number of sniffed MAC addresses.
- **Non-POI Disturbance** is the case that in an event, there are detected faces that belong to a non-POI, whose MAC address are not in our database i.e. they should be discarded as an outlier.

6.4.1 Data Synthesis. Although there are a number of public databases of face images, unfortunately, none of them come with device presence observations, i.e., sniffed MAC addresses. As an alternative, we develop a synthetic dataset on top of a widely used database, the PIE face database[7]. In total 40 subjects are randomly selected, in which 30 of them are POI and the rest are mixed in to make the dataset realistic. We then assign to these "events" noisy identity observations that simulate device presence information. Based on different types of noise, different synthetic datasets are generated on which AutoTune is examined. One average, each subject in

[7] http://www.cs.cmu.edu/afs/cs/project/PIE/MultiPie/Multi-Pie/Home.html



(a) False-alarm Faces (b) False-alarm Devices (c) Non-POI Disturbance

Fig. 14. Results on simulation data showing the impact of sources of noise

our simulation has 168 images after data augmentation [18]. As there is no device presence information in this dataset, we simulate a dataset by first randomly placing the face images into different "events", in which multiple subjects "attend". Considering the fact the number of images of subjects might be skewed in events, we adopt the F_1 score as the metric to evaluate the AutoTune's performance. Tab. 2 summarizes the performance of AutoTune under all types of noisy observations in the simulation environment.

6.4.2 False-alarm Faces. WiFi sniffing is an effective approach to detect the presence of users' mobile devices, however, such detection is not guaranteed to perfectly match the user's presence. For instance, a device could be forgotten at home, be out of battery or simply WiFi function might be turn off. We observe that *false-alarm faces* happened in $\sim$ 10% of the events in our real-world datasets. In simulation, we vary the error rate of such false alarm faces from 0.1 to 0.5. Error rate at 0.1 means that on average, 10% of the detected faces in each event are false detections. Fig. 14a shows the F_1 score and convergence iterations of AutoTune under different levels of such noises. As we can see AutoTune tolerates false-alarm faces well and is able to keep the F_1 score above 0.83 when the false alarm rate is below 0.4. Though it degrades to 0.67 when the rate rises up to 0.5. However, we found this scenario, i.e., on average half of WiFi MAC addresses are missed in all meetings, is rare in the real-world. Finally, we found that false-alarm faces do not not affect the convergence much. AutoTune quickly converges within 4 iterations in all the cases.

6.4.3 False-alarm Devices. Though surveillance cameras are becoming increasingly ubiquitous, there are cases where the subjects are out of the camera field of view (FOV). For instance, occlusions are common when several subjects are present in the camera FOV at the same time. In this case, missed faces of a subject can be an instance of device false alarm, if her device MAC address is sniffed. We found that *false-alarm devices* happened in 6% of the events in our real-world datasets. In simulation, we vary the rate of such false alarm devices from 0.1 to 0.5. An error rate of 0.1 means that on average, 10% of the detected devices in each event are false detections. Fig. 14b shows that although the F_1 score of AutoTune decreases, it degrades slowly and stops at 0.84 after the false-alarm rate becomes 0.5. Moreover, as the injected noise becomes stronger, AutoTune needs more iterations to converge. However, the largest convergence iteration is still below 6 (rate at 0.4), which is acceptable. Overall, AutoTune is very robust to false-alarm devices.

6.4.4 Non-POI Disturbance. Non-POI disturbance happens when subjects are captured by the camera, but we do not possess their MAC addresses in the database. We found such noise dominates all the three types of errors. Non-POI disturbance happened in $\sim$ 16% of the events in the office dataset and in 45% in the CommonRoom dataset. In simulation, we vary the number of non-POI from 2 to 10 and the probability of each non-POI's presence in an event is set to 0.1. Fig. 14c shows that AutoTune does not suffer when the disturbance is mild (2 non-POI). Although the F_1 score drops to 0.87 when the number of non-POI reaches 4, the performance does not further degrade much as the disturbance becomes larger. In addition, the convergence of AutoTune is not linearly related to the disturbance levels. AutoTune quickly converges within 5 iterations in all cases.

Table 2. Performance of AutoTune under various types of noisy observations in the simulation environment

	False-alarm Face Rate					False-alarm Device Rate					# of Non-POI				
Control Variable	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**2**	**4**	**6**	**8**	**10**
Precision	0.98	0.94	0.83	0.88	0.67	0.97	0.93	0.93	0.89	0.90	0.97	0.87	0.89	0.85	0.90
Recall	0.97	0.93	0.88	0.86	0.69	0.97	0.92	0.88	0.87	0.87	0.96	0.88	0.90	0.86	0.90
F1 Score	0.96	0.92	0.83	0.87	0.67	0.96	0.91	0.87	0.84	0.84	0.96	0.87	0.87	0.82	0.87
Accuracy	0.97	0.96	0.98	0.99	0.98	0.97	0.93	0.89	0.87	0.87	0.97	0.92	0.90	0.84	0.88
Converge. Iteration	4	3	3	4	4	3	4	5	6	4	5	3	3	5	4

7 DISCUSSION

This section discusses some important issues related to AutoTune.

Overheads: Compared with conventional face recognition methods, AutoTune incurs overheads due to FaceNet fine-tuning. In our experiment, fine-tuning is realized on one NVIDIA K80 GPU. One fine-tuning action takes around 1.2 hours and 0.5 hours for the office dataset and common room dataset respectively. As we see in §6.2, AutoTune is able to converge within 5 iterations for most of the time, depending on the hyper-parameter setting. Therefore, fine-tuning overheads can be controlled in 7 hours for the office dataset and 2.5 hours for the common room dataset. Compared to the FaceNet pre-training that takes days and requires more GPUs, the fine-tuning costs of AutoTune is much cheaper. In future work, we will look into an online version of AutoTune, which can incrementally fine-tune the network on the fly when the data is streaming in.

Privacy: In practice, AutoTune requires face images and device ID of users to operate, which may have certain impact on user privacy. For example, a user may be able to be identified without explicit consent in a new environment, if the owner has the access to the face image of this user. In this work, we do not explicitly study the attack model in this context, we note potential privacy concerns worth exploring in future work.

Limitation: The diversity across events plays a fundamental role in AutoTune. The intuition behind is simple: AutoTune selects out clusters of images and name them based on the unique patterns of event participation. In §6.2.5, we implicitly show that higher diversity (through more events) across events helps make the participation pattern more unique and favours the cross-modality labeling of AutoTune. We believe that the uniqueness of participation pattern is reasonably assumed as long as there are a large number of events.

8 CONCLUSION

In this work, we described AutoTune, a novel pipeline to simultaneously label face images in the wild and adapt a pre-trained deep neural network to recognize the faces of users in new environments. A key insight that motivates is that enrollment effort of face labeling is unnecessary if a building owner has access to a wireless identifier, e.g., through a smart-phone's MAC address. By learning and refining the noisy and weak association between a user's smart-phone and facial images, AutoTune can fine-tune a deep neural network to tailor it to the environment, users, and conditions of a particular camera or set of cameras. Particularly, a novel soft-association technique is proposed to limit the impact of erroneous decisions taken early on in the training process from corrupting the clusters. Extensive experiments with multiple users in two real-world environments and various adversarial simulation environments are conducted. Results demonstrate the ability of AutoTune to design an environment-specific, continually evolving facial recognition system with entirely no user effort even if the face and WiFi observations are very noisy.

REFERENCES

[1] Amr Ahmed, Kai Yu, Wei Xu, Yihong Gong, and Eric Xing. 2008. Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks. In *European Conference on Computer Vision*. Springer, 69–82.

[2] Alexandre Alahi, Albert Haque, and Li Fei-Fei. 2015. RGB-W: When vision meets wireless. In *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 3289–3297.

[3] Doug Beeferman and Adam Berger. 2000. Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 407–416.

[4] Lacey Best-Rowden, Hu Han, Charles Otto, Brendan F Klare, and Anil K Jain. 2014. Unconstrained face recognition: Identifying a person of interest from a media collection. *IEEE Transactions on Information Forensics and Security* 9, 12 (2014), 2144–2157.

[5] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. 2018. VGGFace2: A dataset for recognising faces across pose and age. In *IEEE Conference on Automatic Face and Gesture Recognition*.

[6] Xudong Cao, David Wipf, Fang Wen, Genquan Duan, and Jian Sun. 2013. A practical transfer learning algorithm for face verification. In *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 3208–3215.

[7] De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. 2016. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1335–1344.

[8] Brian DeCann and Arun Ross. 2013. Relating roc and cmc curves via the biometric menagerie. In *IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. IEEE.

[9] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*. 647–655.

[10] Chris Fraley and Adrian E Raftery. 1998. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The computer journal* 41, 8 (1998), 578–588.

[11] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*. 2121–2129.

[12] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. 513–520.

[13] Suining He and S-H Gary Chan. 2016. Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons. *IEEE Communications Surveys & Tutorials* 18, 1 (2016), 466–490.

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[15] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*. Technical Report. Technical Report 07-49, University of Massachusetts, Amherst.

[16] Roy Jonker and Ton Volgenant. 1986. Improving the Hungarian assignment algorithm. *Operations Research Letters* 5, 4 (1986), 171–175.

[17] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3128–3137.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[19] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[20] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. Sphereface: Deep hypersphere embedding for face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[21] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. 2016. Large-Margin Softmax Loss for Convolutional Neural Networks.. In *ICML*. 507–516.

[22] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. 2017. Deep Hyperspherical Learning. In *Advances in Neural Information Processing Systems*. 3953–3963.

[23] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. 2015. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791* (2015).

[24] Chris Xiaoxuan Lu, Xuan Kan, Stefano Rosa, Bowen Du, Hongkai Wen, Andrew Markham, and Niki Trigoni. 2017. Towards Self-supervised Face Labeling via Cross-modality Association. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*.

[25] Chris Xiaoxuan Lu, Hongkai Wen, Sen Wang, Andrew Markham, and Niki Trigoni. 2017. SCAN: learning speaker identity from noisy sensor data. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 67–78.

[26] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[27] Arsha Nagrani, Samuel Albanie, and Andrew Zisserman. 2018. Seeing Voices and Hearing Faces: Cross-modal biometric matching. *arXiv preprint arXiv:1804.00326* (2018).

[28] Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. 2016. Ambient sound provides supervision for visual learning. In *European Conference on Computer Vision*. Springer, 801–816.
[29] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.
[30] Savvas Papaioannou, Hongkai Wen, Zhuoling Xiao, Andrew Markham, and Niki Trigoni. 2015. Accurate positioning via cross-modality training. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM, 239–251.
[31] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. 2015. Deep Face Recognition.. In *BMVC*, Vol. 1. 6.
[32] David Martin Powers. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. (2011).
[33] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. 2014. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596* (2014).
[34] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 815–823.
[35] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. 2014. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*. 1988–1996.
[36] Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2014. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1891–1898.
[37] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning.. In *AAAI*.
[38] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1701–1708.
[39] Xiaoyang Tan, Songcan Chen, Zhi-Hua Zhou, and Fuyan Zhang. 2006. Face recognition from a single image per person: A survey. *Pattern recognition* 39, 9 (2006), 1725–1745.
[40] Jin Teng, Boying Zhang, Junda Zhu, Xinfeng Li, Dong Xuan, and Yuan F Zheng. 2014. EV-Loc: integrating electronic and visual signals for accurate localization. *IEEE/ACM Transactions on Networking (TON)* 22, 4 (2014), 1285–1296.
[41] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. 2014. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729* (2014).
[42] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 3156–3164.
[43] Stefan Wager, Sida Wang, and Percy S Liang. 2013. Dropout training as adaptive regularization. In *Advances in neural information processing systems*. 351–359.
[44] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. 2016. A discriminative feature learning approach for deep face recognition. In *ECCV*.
[45] Xuanli Lisa Xie and Gerardo Beni. 1991. A validity measure for fuzzy clustering. *IEEE Transactions on pattern analysis and machine intelligence* 13, 8 (1991), 841–847.
[46] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *Advances in neural information processing systems*. 3320–3328.
[47] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. 2016. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters* 23, 10 (2016), 1499–1503.
[48] Wenyi Zhao, Rama Chellappa, P Jonathon Phillips, and Azriel Rosenfeld. 2003. Face recognition: A literature survey. *ACM computing surveys (CSUR)* 35, 4 (2003), 399–458.
[49] C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. 2013. Learning the visual interpretation of sentences. In *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 1681–1688.

A APPENDIX

A.1 Further Face Identification Results

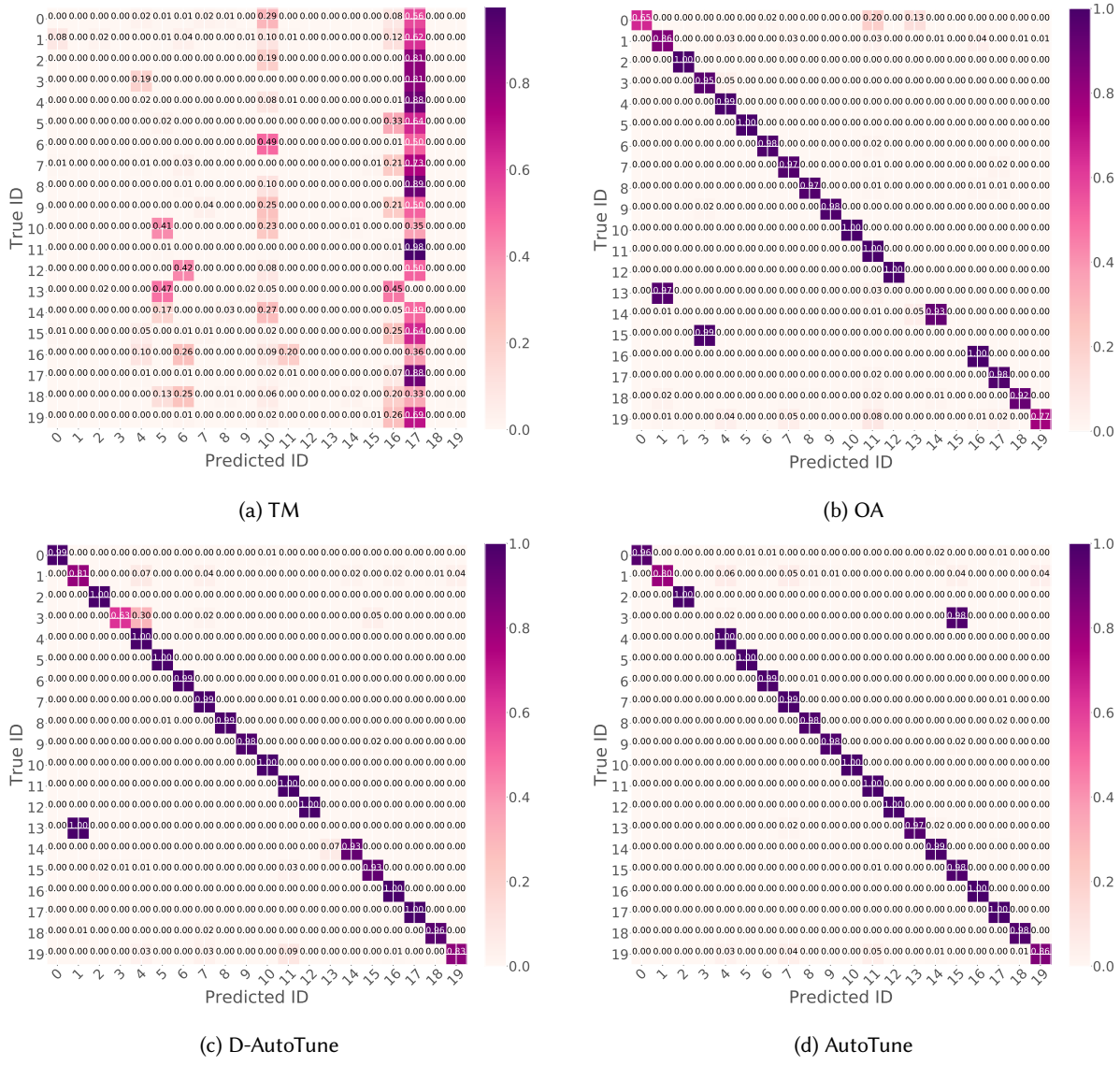
A.1.1 Results on Office Dataset. We provide the confusion matrix of the rank-1 prediction of AutoTune and competing approaches on two real-world datasets. Details of this experiment can found in §6.3.1. We note that rank-1 predictions of subject #3 on the office dataset is very inaccurate with 0% accuracy. By inspecting the source data, we observed that the low accuracy is because the errors of false-alarm devices of this subject are significant. As a result, the true image cluster of subject #3 is not very compatible with her device based on the event vectors. It turns out that her device ID is more similar to another subject's (#15) image clusters, which possess those "missed" event attendance information. Moreover, we also observed that both subject #3 and #5

are female with the same ethnicity, and the pre-trained model is confused in differentiating their faces as well. Therefore, AutoTune fails to label her images due to both insufficient face discrimination given by the pre-trained model (i.e., bad initialization) and non-informative device observations.

Note that, as discussed in §4.3.2, we always set a greater number of clusters than the number of POI in clustering, to help deal with the uncertain presences of non-POI. Therefore, although one image cluster of subject #15 is mis-associated with the label #3, it does not affect the recognition performance of subject #15, as there is another image cluster of subject #15, which matches her device's presence pattern more, and is labeled correctly. We here highlight that this robustness is brought by the voting capability.

Although AutoTune fails to predict subject #3, it still has the best overall accuracy compared to other competing approaches. AutoTune's Cumulative Match Characteristic of identification is also shown to be the best among all the methods (see §6.3.1).

A.1.2 Results on CommonRoom Dataset. The results on the CommonRoom dataset is much neater and the rank-1 prediction of AutoTune is superior to the competing approaches. However, we note that there are $\sim$ 29% predictions of the subject #7 that are mis-classified to subject #4. By looking into the source data, we found that the number of his face images is too limited in the training data, due to lack of event participation. This limited dataset does not contain many variations that can enable his learnt face features.



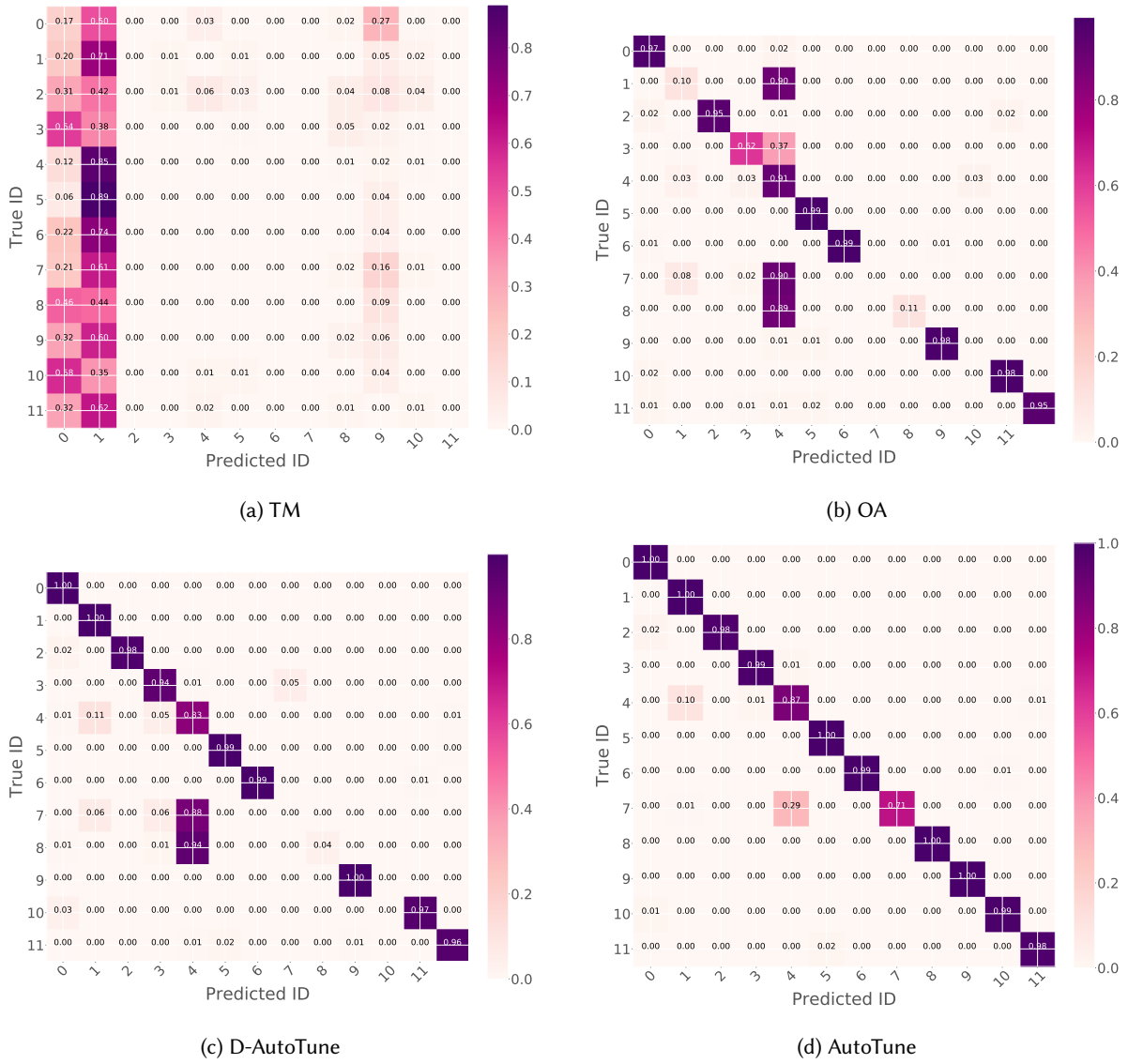
(a) TM

(b) OA

(c) D-AutoTune

(d) AutoTune

Fig. 15. Confusion matrix of face identification with 20 people using different methods on the *office* dataset.



(a) TM

(b) OA

(c) D-AutoTune

(d) AutoTune

Fig. 16. Confusion matrix of face identification with 12 people using different methods on the *CommonRoom* dataset.