AutoTag 'n Search My Photos: Leveraging the Social Graph for Photo Tagging

Shobana Balakrishnan, Surajit Chaudhuri, Vivek Narasayya
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
{shobanab,surajitc,viveknar}@microsoft.com

ABSTRACT

Personal photo collections are large and growing rapidly. Today, it is difficult to search such a photo collection for people who occur in them since it is tedious to manually associate face tags in photos. The key idea is to learn face models for friends and family of the user using tagged photos in a social graph such as Facebook as training examples. These face models are then used to automatically tag photos in the collection, thereby making it more searchable and easier to organize. To illustrate this idea we have developed a Windows app called AutoTag 'n Search My Photos. In this demo paper we describe the architecture, user interaction and controls, and our initial learnings from deploying the app.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Photo tagging; photo search; people tag

1. INTRODUCTION

With the proliferation of digital cameras and phones the number of photos taken daily is staggering. In 2013, the average adult in the US older than 18 years took 500 photos a year [6] and that number is growing steeply since. Today, it is common for people to store large personal photo collections spread across home computers, devices and cloud services such as Flickr, DropBox, Box and OneDrive. Many people also share a subset of these photos on social networks, most notably Facebook. In 2013 Facebook stated that on average 350 million photos are uploaded daily [5].

With the number of photos in personal photo collections growing quickly, the ability to search these photos becomes crucial. One of the important dimensions for searching personal photos is by people who appear in them. *Face tagging*, the process of annotating the photo with the names of people whose faces appear in the photo, can improve searchability of photos. However, manual face tagging of photos is tedious. While some photo management applications [1, 4] make face tagging somewhat easier by automatically grouping faces with similar facial features, considerable manual effort is still required. Therefore, in practice, most users today are still unable to search for people in their personal photo collections.

Our key observation is that there already exists a large collection of photos that users share in their social networks, in particular Facebook, that contain face tags of people who occur in the photo. We use this rich "training data" to automatically learn *face models* of friends and family in the user's social network. Once a face model is learned for a person, we then use it to detect and automatically tag occurrences of that person in photos of the user's personal collection. The methodology of automatic face tagging by bootstrapping learning from the user's social graph is applicable to different social networks. To demonstrate this idea we have built a prototype Windows 8.1 universal application called AutoTag 'n Search My Photos, that learns face models for strong connections of the user from Facebook, and then uses these models to auto-tag photos in Pictures Library.

Our approach is different from other client desktop applications that rely solely on manual tagging to bootstrap, and thereby provides an easier on-ramp using models automatically learned from tagged photos in Facebook. We keep the user in control over the face models learned and automatic tagging and perform this computation locally on the user's Windows device. However more generally, the computational logic may run elsewhere close to the photo store. We utilize the Face SDK from Microsoft Research [2] to perform face detection and recognition. The app leverages manual tagging and tag corrections or confirmations from the user to refine the face models. We support efficient search for people and other metadata over photos stored locally and on Facebook.

The AutoTag 'n Search My Photos app [7] has been released externally via the Microsoft Garage [8], a portal for experimental apps. We demonstrate key features of the app including automated learning of face models and photo tagging, searching for people in photos, and refinement of models based on feedback from the user.

2. FUNCTIONALITY & ARCHITECTURE

For brevity we will refer to the app in the rest of the paper as AutoTag. AutoTag has the following key features:

- Automatically learns face models of strong connections in user's Facebook graph, and uses these models to tag photos in user's Pictures Library (including OneDrive photos).
- Provides a person-centric view of user's photos in Pictures Library and Facebook, and Search over people tags. Photos can be selectively shared on Facebook with tag propagation.
- Clusters unidentified faces for easy manual tagging and allows tags to be associated with people not on Facebook. Learns and improves face models from user interactions.

WWW 2015 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2742843 .

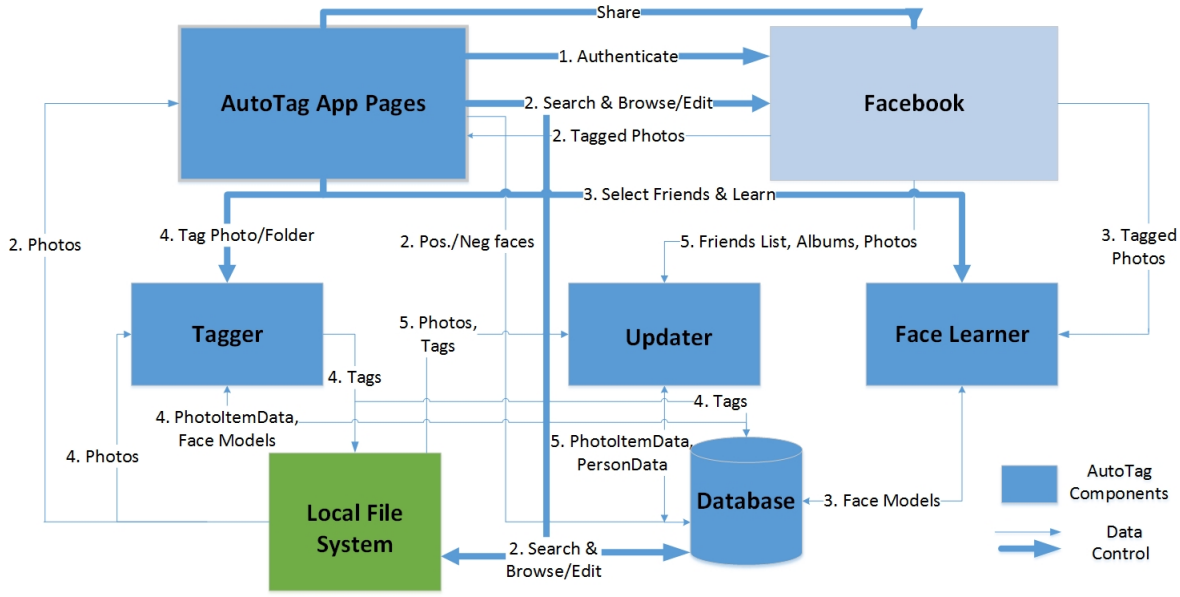

Figure 1: AutoTag 'n Search My Photos Software Components

Figure 1 illustrates the software architecture of AutoTag. It comprises of: (a) A set of app pages for user interaction that we describe in Section 2.1, (b) Asynchronous tasks: Face Learner, Tagger and Updater, which we describe in Section 2.2.

2.1 AutoTag User Experience

First Launch: On first launch, AutoTag authenticates to Facebook as in Figure 1 Flow 1, and requests access (via the Facebook Graph API [9]) to the user's public profile, friend list, custom friends' lists, relationships and photos, the user's friends' photos and the permission to post to Facebook. Since the publishing of this paper, Facebook has mandated a change to v2.x of the Graph API whereby apps do not get access to the friends list, relationships and user's friends' photos unless friends have also installed the app and allowed access to the same. Therefore to learn the face model of a friend the app must learn from the user's tagged photos or the friend must install the app and provide access as well.

Home Screen: The app provides a unified people centric view of all photos scanned based on the people in them as shown in Figure 2. All photos of the person associated with a *Facebook* ID or a *UserCreated* ID (for friends not on Facebook) can be browsed by clicking on the profile picture tile. The tiles with green tags represent *Imported* person tags available on the photo metadata that are scanned by AutoTag, and are not associated with an ID. The user can merge an *Imported* person with a *Facebook* or *UserCreated* ID, thereby using tags already existing on photos generated by other applications [1, 4] to improve face models and maintain unique person identity. The details pane associated with a person view provides an indication of the quality of the face model learned. The home screen indicates the current function being performed by the tasks namely tagging recent photos, learning faces or idle.

Other Views: Other photo grid views of Pictures Library, Facebook Photos, or All Photos are available from the home screen. Photos can be browsed in the photo view and manually tagged by the user by clicking/tapping on a face to invoke an edit box with auto-complete of tag name and suggestions provided by the Tagger as will be explained later. An unidentified photos view as shown in Figure 4(b) clusters all faces that AutoTag could not tag based on their facial features for quick manual tagging.

Search: The Search box supports searching of photos by people in them as well as keyword search on photo metadata. In Figure 1 flow 2, the Search results page as indicated obtains results from Facebook and Pictures Library and presents a unified view.

2.2 Face Learning and Tagging Components

AutoTag uses the Face SDK which provides an API to perform face detection, alignment, extraction and recognition. The face detection interface takes an input image and provides all face bounding boxes associated with the image. The face alignmentor takes an input image and bounding box and outputs a set of compact distinctive representations of the face called features. A person can have both positive features which are "true" features and negative features which are "false". The face model of a person is a collection of both positive and negative features. The Face SDK is used as a black-box library with the interfaces above with flexibility to employ multiple face detection algorithms (e.g. Frontal, multi-view detector) as well as multiple recognition algorithms (e.g., Neural net, Regression). Edits/corrections made to tags by the user update the positive and negative faces of a person (Flow 2 browse and edit) which are stored in the database as will be explained in Section 3.

Database: The app state is stored in a SQLite [3] database as shown in Figure 1. The PersonData table contains metadata of all the people known to AutoTag including the person id, tag name and the face models consisting of features associated with the positive and negative faces stored as a binary blob for consumption by the face recognition API. The PhotoItemData table contains metadata associated with photos known to AutoTag including person tags.

Face Learner: The Face Learner automatically queries Facebook for tagged photos of the person to be learned (flow 3 of Figure 1). For any friend on Facebook who has disallowed apps access to their photos, the Learner can still learn a face model from other friends who may have tagged photos of the given friend and have allowed the app photo access. In order to automate the learning process and have high quality positive features, the learner clusters the per-person tagged photos from Facebook based on facial features

and uses distance-based heuristics to pick distinctive positive and negative exemplars of the person and weed out incorrectly tagged photos in Facebook. This is a key challenge, and the threshold for selecting faces from each cluster has been tuned from experimentation based on feature distance between clusters. Examples of bad tags include photos in Facebook that may be incorrectly tagged as the person which typically will be in smaller clusters with larger feature distance between "positive" and "negative" clusters. Another bad tag example is a Facebook bounding box with a person's tag even though the bounding box does not contain a face. These are automatically discarded if the detector does not detect a face whose centroid is within a threshold of the centroid of the Facebook bounding box. Face models can be improved as more tagged photos accumulate in Facebook by re-learning the face model.

Tagger: The Tagger is responsible for tagging photos within a specified set of folders. The user can specify a subset of people in their friends list to be tagged. In Windows 8.1, folders on OneDrive including the user's Camera roll can be included in Pictures Library thereby allowing the scope of Tagger to include all local photos as well as Photos on OneDrive. The Tagger works with the local version of the photos on OneDrive that are maintained in sync by the OneDrive sync service. For a photo to be tagged, the Tagger calls the face detection API to determine the bounding boxes of faces. For each face it extracts the facial features, and identifies the facial match to any of the face models of people it is attempting to tag. It picks the highest confidence match above a predetermined confidence threshold, which has been tuned from experimentation and set to be conservative to reduce the false positives. The Tagger adds XMP tags that store the "People Tag" and "Bounding Box" information on the photo and does not overwrite any existing tags on the photo (flow 4 in Figure 1). On a per photo basis, it also stores in the database for each person run through the face recognition API, a ranked list of suggested tags if the confidence is within a small range *below* the threshold. These suggested tags are displayed to the user when they browse photos allowing them to accept a suggested tag. The Tagger also clusters the photos by the people autotagged to provide an easy view for editing and confirming tags.

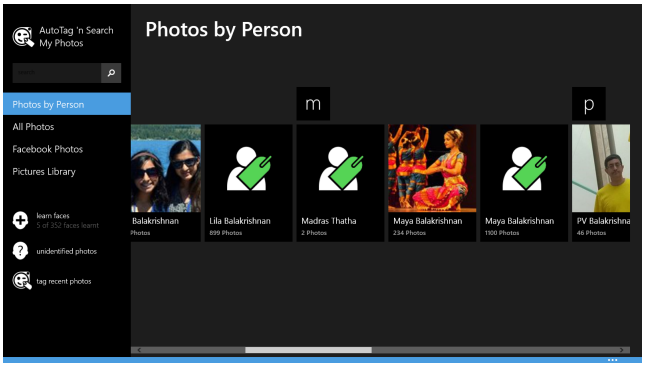

Figure 2: Home Screen

Updater: The Updater (flow 5 in Figure 1) is the component that keeps the photo metadata and person metadata up-to-date with changes in the file system and Facebook. It periodically scans the file system for photo additions/deletions/metadata changes in Pictures Library. Any new tag names scanned are identified as *Imported* tags. The Updater also periodically checks Facebook for updates to the following: friends list, users albums, and metadata of Facebook photos (stored in PhotoItemData table).

3. DEMONSTRATION PLAN

We will demonstrate the following demo scenarios.

- Automatic learning of face models and auto-tagging.
- Tagging of users not on Facebook.
- Improving tagging accuracy using feedback.
- Searching for people in photo collections.

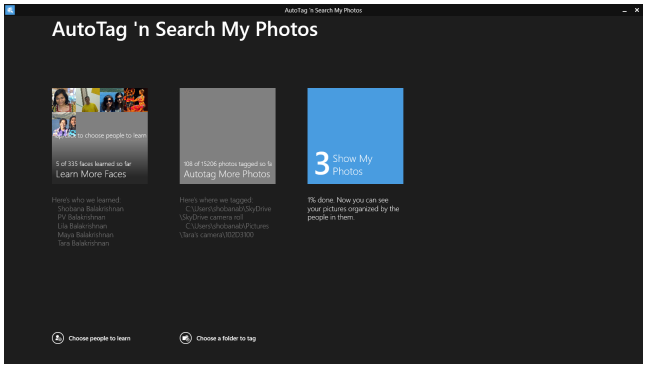

Figure 3: First Launch Experience

AutoTagging: First we demonstrate the AutoTag first launch experience where the app automatically learns face models of the logged in user and their strong connections from Facebook comprising of their spouse/in a relationship with and friends on custom lists. For most users, the strong connections typically corresponds to family and close friends. Figure 3 shows the first launch experience where the app learns 5 out of 20 strong connections and 335 total connections in Facebook, at which point the user stops learning and AutoTag proceeds to tagging. Tagging is performed on the entire Pictures Library scope proceeding with the most recent photos sorted by Date Taken, and Creation Date. Tagging can continue while the user interacts with their photos through the app by proceeding to Step 3 in Figure 3 which launches the home screen shown in Figure 2. The key functionality demonstrated here is automatic learning of face models and auto-tagging.

User-created people: Next, we demonstrate how friends and family who are not on Facebook (e.g. children) can be managed in AutoTag with a *UserCreated* ID. The *Unidentified Faces* view clusters faces that are not associated with tags. In the demo scenario we show how a new *UserCreated* profile can be created with minimal user input by associating a person name with a cluster. AutoTag then learns a face model for that person and subsequently uses the face model for auto-tagging that person in other photos.

Leveraging feedback: In this scenario we demonstrate how the tagging *recall* improves with feedback from the user in the form of accepting tag suggestions and tag confirmations. We start with only a few manually tagged photos of a *UserCreated* person and demonstrate that AutoTag is still able to tag several more photos of her, and suggests her as tags in a few others, and cannot recognize her in some difficult photos (glasses off, eyes closed, side profile). Confirming tags generated by AutoTag as shown in Figure 4 as well as accepting tag suggestions result in the recall improving as demonstrated by autotagging the folder again after these user actions are performed. Modifying the tag of an Autotag generated tag name creates a positive face for the newly tagged person and a negative face for the incorrectly tagged person which improves the tagging *precision* with future tagging. While precision and recall numbers are highly dependent on the photos, we qualitatively demonstrate the effectiveness of user feedback in improving tagging accuracy.

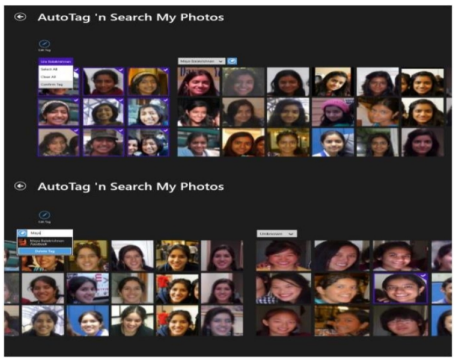

Figure 4: Improving Tagging Accuracy (a) confirming tags of selected faces (b) editing tag of selected face

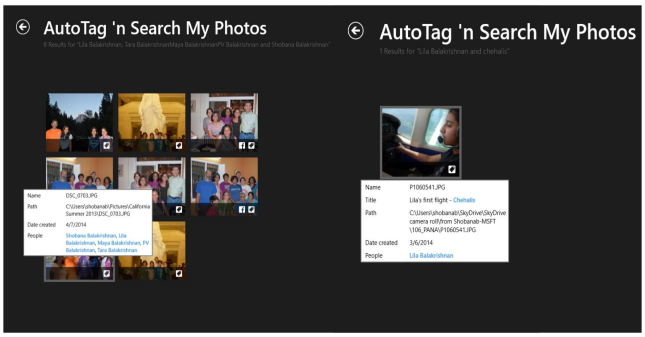

Figure 5: Searching (a) family photo with people tags (b) person tag + keyword

Search: Searching for photos with the people in them can be done by choosing people from the auto-complete box. For example, for a user with around 30,000 photos searching for family photos of all 5 family members can be tedious but once autotagged, search produces the result almost instantaneously as shown in Figure 5(a). Search federates the people query to both sources - local Windows Search and Facebook graph query and returns the combined results. We can combine a people query with other keywords. AutoTag indexes all words in description, title, Facebook comments and location so the query can consist of a combination of keywords and people tags as shown in 5(b). Here, the user searches for photos of a particular person highlighted in blue with "Chehalis" in the metadata and is easily able to find the photo of interest.

4. LEARNINGS

Prior to external release, we deployed the app internally to obtain feedback that helped refine the user experience and tune face recognition parameters. In order to understand the accuracy (precision and recall) of auto-tagging we collected data during the face model learning and auto-tagging steps from 23 app users and performed offline analysis. The dataset contained 107 people learned by the Face learner using 1685 positive faces. The tagging involved 3,279 faces (extracted from photos in the tagging folder) that were candidates for being tagged. We computed the following:

- #TP: actual true positives in the tagging folder
- #Reco. TP: # of True Positives tagged by Recognizer
- #Reco. FP: # of False Positives tagged by Recognizer
- #precision: #Reco. TP/(#Reco. TP + #Reco. FP)
- #Reco. recall: #Reco. TP/#TP

Table 1: Offline analysis of photos tagged of 23 users

#TP	#Reco. TP	#Reco. FP	#precision	#Reco. recall
2159	1468	105	93.32%	67.99%

While the accuracy is dependent on the photos tagged and the quality of the models learned – which in turn depends on the quality of tagged faces in Facebook – they are indicative that we can achieve a reasonably high precision with such an automated face learning approach. From the multiple datasets collected we refined the heuristics for selection of positive and negative faces and evaluated multiple confidence thresholds for auto-tagging. We tuned the confidence threshold biasing precision over recall. To improve the detection recall, we allow users to manually specify (position and size) a bounding box for faces not detected by the Face SDK.

While bootstrapping face models from tagged photos in Facebook is useful for users of Facebook, for non-Facebook users providing an option to learn from other sources such as Flickr and utilizing the Windows People contacts as a Friends list might be an alternative. Although our face model learning approach works for any reliable source that generates tag and bounding box information ([1, 4]) additional analysis may be needed if tags are imprecise or if no bounding box is available.

5. CONCLUDING REMARKS

AutoTag 'n Search My Photos is a Windows universal app for automatically tagging people in a user's personal collection, thereby enabling people search over the collection. AutoTag was released externally via the Windows 8 store at the end of October 2014 and as of January 2015 has over five thousand downloads and between 400-500 monthly active users. We have demonstrated the viability of using tagged photos in Facebook to perform unsupervised training of a face recognizer for a small number (in the tens) of the user's strong connections, and then perform tagging on the entire user's personal collection with minimal user intervention. As of April 30th, Facebook will be deprecating the v1.0 Graph API and introducing restrictions in the 2.x APIs on access to user's friends lists and photos. Therefore, some of the functionality described in this paper will need to change to cope with these new restrictions.

6. REFERENCES

[1] Picasa. http://en.wikipedia.org/wiki/Picasa, 2002.
[2] Microsoft Research Face SDK. http://research.microsoft.com/en-us/projects/facesdk/, 2011.
[3] SQLite. http://sqlite.org/about.html, 2012.
[4] Windows Photo Gallery. http://en.wikipedia.org/w/index.php?title=Windows_Photo_Gallery, 2012.
[5] A Focus on Efficiency. https://fbcdn-dragon-a.akamaihd.net/hphotos-ak-prn1/851575_520797877991079_393255490_n.pdf, 2013.
[6] The Photography Consumer. http://reports.mintel.com/display/637639, 2013.
[7] AutoTag 'n Search My Photos. http://apps.microsoft.com/windows/en-us/app/84cc8f26-6344-4e2f-8707-c1312311fefb/, 2014.
[8] The Garage. http://www.microsoft.com/en-us/garage/#garage-workbench, 2014.
[9] Facebook Graph API. https://developers.facebook.com/docs/graph-api, 2015.