Leveraging archival video for building face datasets

Deva Ramanan[1,3] Simon Baker[2,3] Sham Kakade[1]
[1]Toyota Technological Institute at Chicago
Chicago, IL 60637
{ramanan,sham}@tti-c.org
[2]Microsoft Research
Redmond, WA 98052
sbaker@microsoft.com

Abstract

We introduce a semi-supervised method for building large, labeled *datasets of faces by leveraging archival video. Specifically, we have implemented a system for labeling 11 years worth of archival footage from a television show. We have compiled a dataset of 611,770 faces, orders of magnitude larger than existing collections. It includes variation in appearance due to age, weight gain, changes in hairstyles, and other factors difficult to observe in smaller-scale collections.*

Face recognition in an uncontrolled setting can be difficult. We argue (and demonstrate) that there is much structure at varying timescales in the video data that make recognition much easier. At local time scales, one can use motion and tracking to group face images together - we may not know the identity, but we know a single label applies to all faces in a track. At medium time scales (say, within a scene), one can use appearance features such as hair and clothing to group tracks across shot boundaries. However, at longer timescales (say, across episodes), one can no longer use clothing as a cue. This suggests that one needs to carefully encode representations of appearance, depending on the timescale at which one intends to match.

We assemble our final dataset by classifying groups of tracks in a nearest-neighbors framework. We use a face library obtained by labeling track clusters in a reference episode. We show that this classification is significantly easier when exploiting the hierarchical structure naturally present in the video sequences.

From a data-collection point of view, tracking is vital because it adds non-frontal poses to our face collection. This is important because we know of no other method for collecting images of non-frontal faces "in the wild".

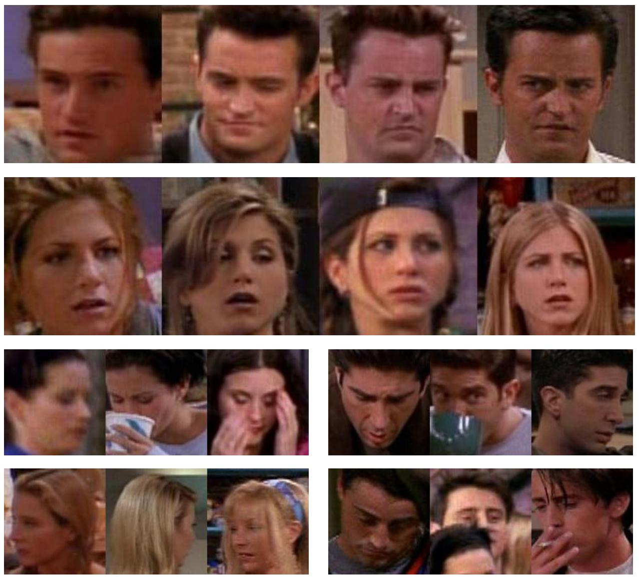

Figure 1. We describe a method for labeling extremely large face datasets from archival video collections. We demonstrate our approach on footage spanning 11 years from the television show Friends. The final face clusters contain appearance variation due to aging, weight gain/loss (**top row**), and changes in hairstyles (**middle row**) among other factors. By tracking neighboring frames around frontal-face detections, we also include face images that are not restricted to detector responses (**bottom two rows**). This is important because current methods for collecting images of non-frontal poses require a controlled lab environment. Our system builds a face database by clustering faces tracked in video footage. From this perspective, our system also provides a framework for automatically tracking and identifying people in video - see Figure 2.

1. Introduction

We introduce a system for building extremely large face datasets from archival video. Such footage provides a rich structured dataset (since there is much continuity in the footage) and an immense quantity of such data is available. In this paper, we exploit the temporal coherence in these video streams with a semi-supervised algorithm to provide a huge labeled dataset. We have implemented a system for processing 11 years worth of archival footage from the television show Friends. Our dataset of 611,770 faces is orders of magnitude larger than existing collections. It is notewor-

[3]Work done while at Carnegie Mellon University.

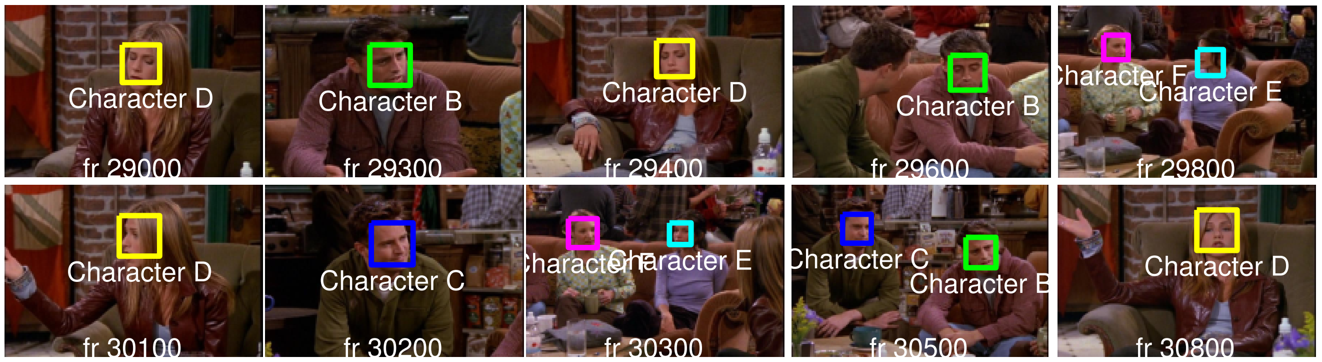

Figure 2. Our system can also be viewed as one that tracks and identifies people in video. Here, we show results for a long scene involving many characters and frequent shot changes (discontinuities in camera motion). Frame numbers are overlaid on images. Our system works by (1) detecting frontal faces, (2) building a torso, face, and hair model for each detection, and then (3) tracking using the body models. Our system then clusters the body models from across a video to link up tracks from different shots. The process of tracking and clustering is fully automatic.

thy in that it contains variation due to extreme pose changes, age, weight gain, changes in hairstyles, and other factors difficult to observe in smaller-scale collections.

While there is abundant literature on face recognition (see [10, 18, 6] for review), our work is inspired by two recent approaches for automatically annotating face images. In 'Names and Faces' [2], Berg et al. detail a method for automatically building face datasets by leveraging news photos tagged with captions. They convincingly argue that databases built from "the wild" are important for training and evaluating real-word recognition systems. One limitation is that the resulting images are biased to be responses of a frontal-face detector. This is particularly limiting because pose variation is one reason *why* recognition is hard [10]. One ideally wants to collect large datasets with pose variation to better train/evaluate real world systems. We demonstrate one can do this by tracking around frontal face detections found in a video.

Our second inspiration comes from work on person-spotting in video. Sivic et al. [14] introduces work on shot-based face-tracking and identification by retrieval. While their focus is on producing a ranked list of shots given a query, our goal is a large, multi-pose, labeled dataset of face clusters. Everingham et al. [5] describe the automatic 'Buffy' system for naming characters in TV footage by exploiting additional cues such as transcribed audio and text (which was obtained from the web). We show that these auxillary cues may not be needed if one is willing to perform a modest amount of manual labeling (20 minutes). Hence our approach could be extended to unconstrained footage lacking transcriptions such as home videos.

In this paper, we argue that constraints, at *multiple* timescales, in the video stream itself can be used to group faces. Grouped faces are then required to share the same identity label. Finding methods to group instances such that all instances share the same label is a popular theme in the semi-supervised learning literature — taking it's roots from the work of Yarowsky [17] and Blum and Mitchell [3].

We do this using a hierarchical procedure exploiting the structure at multiple time scales. At the shortest timescale, features based on clothing and hair are good cues as to which faces share the same label. At the next level in our hierarchy, dynamics (through spatial contiguity) provide a means to group faces, and, importantly, add non-frontal faces into the group. At even longer timescale, we can drop the dynamic constraints and group tracks with similar appearance. Most previous work on video-based face recognition [14, 1, 13] exploits tracks of faces during recognition. We show that by adding another layer of structure that groups similar-looking tracks, one can significantly boost recognition performance.

Another distinction from previous work is our use of hair and body models for face tracking and recognition. We find such appearance models to be more stable than facial appearance during those frames where the face is partially occluded or turned away from the camera. In our case, we want to add these "interesting" cases to our face collection. Previous work has shown such models aid in video-recognition [5, 7], but we demonstrate that they only apply at local timescales - characters tend to wear the same clothes within a scene but not across episodes. Hence, a hierarchical grouping approach naturally applies.

After we performed this grouping procedure on the 11-year collection of video (which is completely done in an unsupervised manner), we show that with a minimal amount of labeling (taking 20 minutes to hand-label), one can build a quite accurate large labeled dataset (consisting of about 600,000 faces over 22 episodes). As in co-training [3], the complexity of our supervised learning problem has been reduced due to our grouping. We give results showing signif-

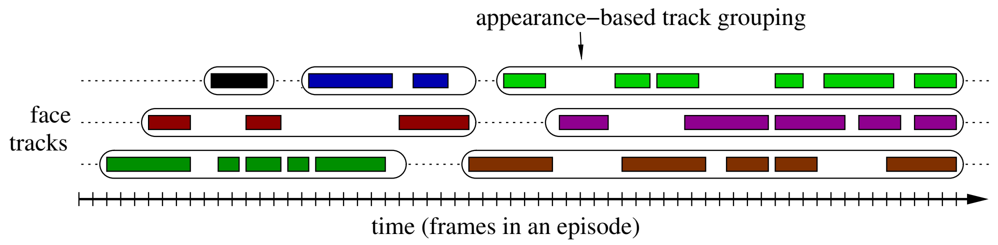

Figure 3. A system-level overview of our semi-supervised labeling process. First, we obtain low-level face tracks by detecting frontal faces and tracking in neighboring frames around the detections. To track, we build a color histogram model of the face, hair, and torso (Sec. 3). We crudely visualize the color models for each track with the **colored rectangles**. These tracks tends to be fairly short since shot changes are common. We then merge tracks by clustering the body models (the **rounded ovals**). This matching is intended to merge tracks at shorter time scales, during which we expect a character to stay in consistent costume (Sec. 4). For each cluster, we can now build a large exemplar-based model of facial appearance using the associated face images. We use this exemplar model to match to a reference set library of faces. This matching provides a character label for each cluster.(Sec. 5).

icant performance improvements in comparison to not having done this grouping.

2. Overview of approach

We describe a semi-supervised system for labeling characters in large archival video collections. Our hierarchical procedure works in stages. At the lowest level (Sec 3), we have a procedure which first groups frontal faces together (such that we are certain they have the same label). To do this, we build a color histogram model of the face, hair, and torso. Then it tracks in neighboring frames around these grouped detections — adding dynamic constraints in addition to using the aforementioned features. These tracks are quite reliable since body appearance is stable over short time scales. However, shot changes are common, and so these tracks can be small (on the order of a few hundred frames). At the next level (Sec 4), we group tracks by an agglomerative clustering procedure, where we exploit the fact that characters tend to wear consistent clothing during a scene — body appearance can be a strong cue for matching across shot changes. Hence, temporally disjoint tracks are merged into groups with the same identity.

Finally, we construct a labeled dataset by handlabelling a small number of groups — this can be done quickly because one is labeling clusters rather than individual tracks. The labeled library is acquired by labeling the track clusters from a single episode. Our supervised learning algorithm is now just nearest neighbors (Sec.5). However, here it is unreasonable to use any features that have to do with clothing and hairstyles, i.e. we can no longer use the some of the features that were only relevant at the shorter timescale.

3. Finding and tracking faces

We use the ViolaJones frontal-face detector [16] to find faces in each frame of a video. We construct a crude shot detector by thresholding the change in a global color histogram computed at each frame. Given a pool of candidates found within a shot, we would like to (1) group them into individual people and (2) enlarge each group to capture non-frontal faces by tracking.

Grouping detected faces: We would like to find a set of face detections within a shot that look like one another. To do this, we build a color histogram for the hair, face, and torso of each putative detection using rectangles flanking the original face detection (Fig.4). We represent the appearance of each detection with a concatenated vector of the normalized color histograms. We cluster these vectors to obtain groups of similar looking detections using agglomerative clustering. We used a conservative distance threshold to ensure no cluster contains multiple people (we found this restriction easy to meet). We finally disregard small clusters that contained less than 20 faces. We find empirically that this procedure removes the fast majority of false positive face detections. Missed detections (and face clusters that are mistakenly disregarded) can be recovered during the following tracking stage.

Tracking: We would like to track around the detections from each cluster to find "interesting" faces. Because we want to track through difficult cases (i.e, the extreme pose changes from Fig 1), the appearance of the face may change. This makes tracking difficult. We exploit the fact that *body appearance*, such as clothing and hair, tend to be consistent even during these difficult situations. For each cluster, we now have a color histogram representation of the face, hair, and torso. We also obtain a background color histogram by binning those pixels outside the part rectangles. We use these part-specific color models to track in frames around the original detections. One could imagine using a histogram-based tracker such as meanshift [4], but we want to enforce spatial relationships between part-specific models - the face, hair, and torso are often distinctly colored and distinctly arranged. We do this using part-based color tracking [11]. We score the match-cost of a part by summing

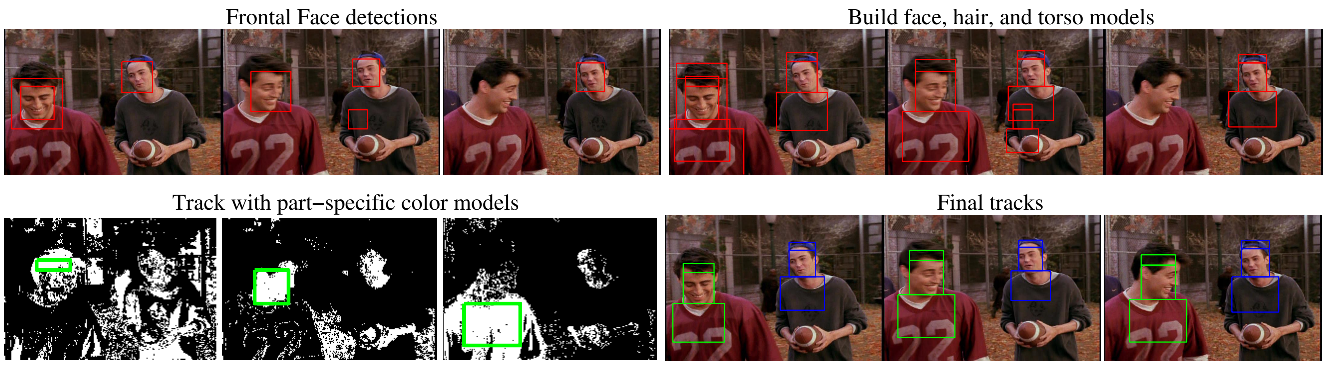
Frontal Face detections
Build face, hair, and torso models
Track with part−specific color models
Final tracks

Figure 4. Low-level tracking. We first run a frontal-face detector on all frames in the video (**upper left**). For each detection, we build a color histogram model for the face, hair, and torso (**upper right**). We want to find good detections with many similar body models in neighboring frames - we do this by clustering the body models within a shot, and throwing away small clusters. We then track through frames neighboring the good clusters to find additional non-frontal poses. We track using part-specific color models built from the clusters. In the **bottom left**, we show thresholded likelihood ratio maps for the hair, face, and torso color models (run on the third frame in the sequence). We finally track by enforcing a rigid spatial alignment of the hair, face and torso parts (**bottom right**).

up the number of correctly labeled pixels from a thresholded likelihood ratio map (of background versus part). We found a simple *rigid* part model to suffice, where the head is forced to lie directly above the torso (and the hair lies directly above the head). Doing so makes tracking quite fast - we can densely evaluate the score of the person being at every image location with three box filters (one for each part). Using a matlab integral image [16] implementation, our tracking takes less than half a second per frame. We search using 3 different scales (.8X, 1X, and 1.2X), where the base scale is set to the average scale of the detections within a cluster.

4. Clustering tracks using appearance

Given a set of collection of tracks, we now wish to merge those tracks that contain similar looking people. We assume that at short time scales, people look similar. We informally call this "within-scene" clustering, since people tend to wear consistent clothes within a scene. We build an appearance representation for each track. It consists of color histograms for the hair, face, and torso, and a set of exemplar gray-scale patches of the face. We adopt the eigenface framework for representing face patches [15]. Each face image is resized to a canonical size of 50×50 grayscale patch. As is standard practice [15], we subtract out the mean and scale the patch to have unit variance. Each patch is represented as a rasterized vector of 2500 values. We compute a 50 dimensional eigenspace of face images from all the detected patches from a reference episode with Principle Components Analysis [15]. We use this basis for all our experiments. We henceforth represent each face image by the 50 coefficients specifying the projection onto this linear subspace.

Rectification? Typically images are rectified before applying linear subspace methods. However, rectification becomes difficult to define during extreme pose changes (such as the ones shown in Fig 1) - reference points such as corners of the eyes or mouth may not be visible. We took a data-driven approach, and hoped to avoid explicit preprocessing steps by assuming that our face library *spans* the set of encountered miss-registrations.

We represent each track with a set of $K = 10$ exemplars computed with k-means. Each track is also represented with three normalized color histogram vectors capturing the color of the face, hair and torso. We write the i^{th} track as $T_i = \{e^i_{1:K}, f_i, h_i, t_i\}$. We would now like to merge tracks that are similar.

To do so, we define a distance function between tracks similar to Everingham et al [5].

$$d(T_i, T_j) = \alpha_e d_e(t_i, t_j) + \alpha_d d_f(f_i, f_j) + \alpha_h d_h(h_i, h_j) + \alpha_t d_t(t_i, t_j) \tag{1}$$

where the exemplar distance is computed from the best matching exemplars

$$d_e(t_i, t_j) = \min_k \min_l ||e^i_k - e^j_l||^2 \tag{2}$$

and the color histogram distances are computed with the chi-squared distance [8]

$$d_f(f_i, f_j) = \sum_k \frac{(f_i(k) - f_j(k))^2}{(f_i(k) + f_j(k))/2} \tag{3}$$

The above distance function has parameters α. Given labeled ground-truth tracks from a reference episode, we can learn the α's in a straightforward manner. We consider all pairs of tracks, and compute their color and exemplar distance features. Pairs of tracks corresponding to the same

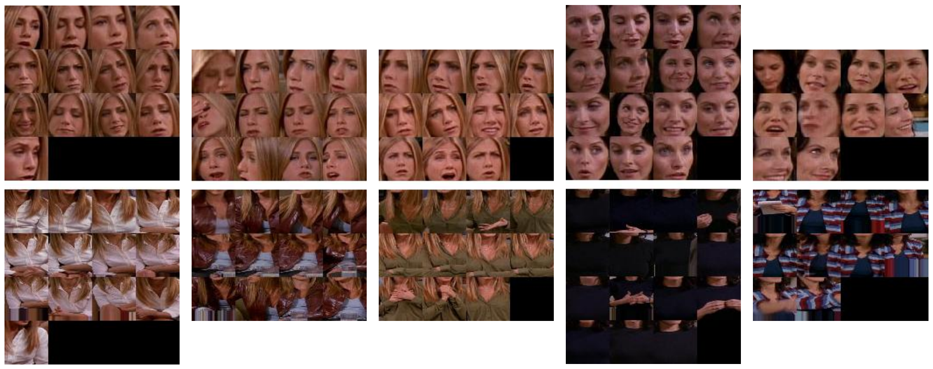

Figure 5. We show the largest clusters resulting from our track-grouping algorithm (Sec.4) applied to a given episode. Here, low-level face tracks are clustered using a distance function strongly weighting body appearance. We visualize a track with the face and torso patch from the median frame (face patches on **top** and torso patches on **bottom**). One can easily verify that the clustered tracks come from different *shots*, implying one cannot merge tracks by "better" tracking - see Figure 2. Instead, our clustering procedure merges tracks together by exploiting consistency in clothing appearance. This allows our system to build rich exemplar-based appearance models of each individual. We then use the exemplars to label the clusters by nearest-neighbor matching to a reference library.

person wearing the same clothes are counted as positives, while pairs corresponding to different people are negatives. We learn a linear weighting of distance features that predicts the positives using logistic regression. The learned weighting for clustering tracks is

$$\alpha_{track} = \begin{bmatrix}\alpha_e & \alpha_f & \alpha_h & \alpha_t\end{bmatrix} \tag{4}$$

$$= \begin{bmatrix}.0015 & 1.59 & 3.22 & 12.02\end{bmatrix} \tag{5}$$

It is difficult to compare color and exemplar features since they are scaled differently. Comparing the color features amongst themselves, we see that the torso appearance is by far the strongest cue, followed by hair appearance, and then facial color.

We feed the distance function into an agglomerative clustering algorithm, which merges clusters in a greedy fashion (starting with the closest). An important addition is that we *recompute* the exemplar sets from the enlarged cluster as we merge. We increase the number of exemplars additively; two merged tracks will be represented with 20 exemplar patches. This means that as our system groups tracks together, it gradually learns a richer model of facial appearance (that makes the clustering process progressively more accurate). Example results are shown in Fig. 5.

5. Labeling clusters

We now have a collection of tracks clustered according to appearance. We find this process to be fairly reliable, since body appearance is quite a strong cue at a local temporal scale. We now want to identify the characters in each cluster. We do this with a nearest neighbor framework. We assume we have a collection of reference clusters with ground truth character labels. We obtain this by manually labeling the clusters from a single reference episode. We use the "half-way" episode during the fifth season as the reference. We typically encounter 80 scene-clusters from an episode, and so labeling each cluster with a name is fairly easy (about 10-15 minutes). To avoid over-merging of tracks when clustering the reference episode, we use a conservative distance threshold to prematurely stop the agglomerative grouping.

When identifying characters across episodes, clothing will likely not be consistent. Note that even within a scene, clothing can change if a character removes an overcoat. Hence, at some temporal scale, we expect a different representation of appearance to be useful. We wish to learn how to match characters across episodes. We take the same learning approach. Given two reference episodes, we compute the distance between clusters from one episode to the other. Distance features computed from clusters corresponding to identical characters are labeled as positives. We learn a linear predictor of positives using logistic regression. The learned between-episode metric is

$$\alpha_{ep} = \begin{bmatrix}.0017 & 0 & 4.10 & 0\end{bmatrix} \tag{6}$$

As one might suspect, torso color and facial color are not predictive of identity at longer time scales. We obtain the above weighting by training a predictor using all subsets of features, with the above combination performing the best on validation data. Note that hair color is still quite useful, as we show in the Results section.

6. Results

We have run our tracking and clustering algorithm on recorded footage from the television show 'Friends', processing 22 episodes (2 per year for 11 years). This corresponds to roughly a million frames. Our final dataset consists of 611,770 face images. There are two natural quantities useful for evaluation - one is *precision*; what fraction of the images labeled 'Joey' really are of Joey? Another is *recall*; what fraction of all the Joey images in the video are in the dataset? Since our dataset is constructed by matching to a reference library, we can trade-off these quantities by thresholding the match cost. As in typical retrieval applications [9], we summarize a precision-recall (PR) curve with the area underneath - this is simply the average precision (AP). We use these measures to evaluate both the dataset and the importance of various cues used during the semi-supervised labeling process.

Ground-truth: As a proxy for labeling all the millions of images individually, we label all *tracks* found. We separately score the accuracy of tracking in terms of precision/recall. We typically encounter 300 tracks in an episode - labeling them takes roughly 20 minutes per episode. For 22 episode, this takes a few hours. We argue that this labeling procedure is noteworthy in of itself, since we generate ground truth for a million frames!

Tracking: The above "ground-truth" labels is predicated on individual tracks being accurate. We score the precision of our tracks by randomly selecting 50 tracks and scoring the fraction of tracked frames that lie on the original detected person. We find that our tracks are quite precise; 92% of time they do not wander off a person. We score the recall of our tracks by randomly selecting 300 frames from across the episodes in our dataset and scoring what fraction of characters were found. We find that 50% percent of the time, people are tracked. This is impressive given the unconstrained nature of the footage.

Accuracy vs supervision: We examine the accuracy of our dataset versus the amount of supervision in Fig. 8. By labeling two episodes, we can obtain 80% precision for a recall of 50%. This means our clusters are 80% accurate if we look at best half of our dataset. With three labeled episodes, we reach the fidelity of [5]. This is noteworthy because our dataset is significantly more varied due to age and pose variation (Fig. 9).

Representations of appearance: We use our large dataset to examine the effect of appearance representation and temporal scale on matching accuracy (Fig. 7). When matching tracks at a local temporal scale (within a scene), clothing and hair models provide a powerful cue, increasing average precision from .392 to .779. When matching tracks between episodes, clothing is much less helpful. But hair still provides a sharp improvement of 10% over an exemplar-based eigenface approach.

Hierarchical grouping: We use our large dataset to examine the effect of track grouping on matching performance (Fig. 7). We see that labeling the groups of tracks is easier than labeling individual tracks - performance improves by a significant 10%. This is because we can extract rich exemplar-based models of appearance from larger collections of data, making face recognition more reliable.

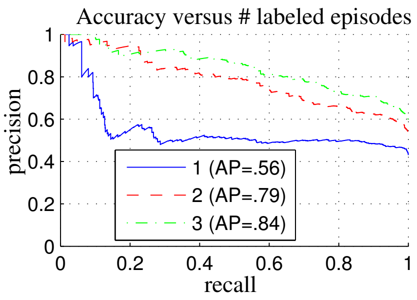

Figure 6. We score the accuracy of our face collection versus the number of hand-labeled episodes. With two labeled episodes, we can operate at 70% precision and 70% recall. With three labeled episodes, we are comparable to the performance of [5]. This is impressive because our dataset contains considerable more variation (see Fig. 9).

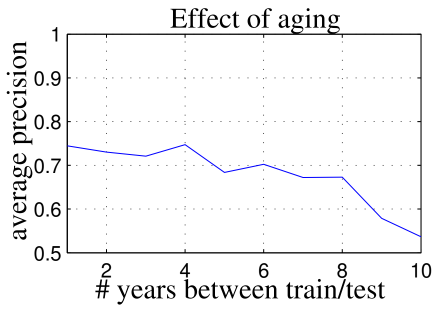

Figure 8. Our large dataset allows us to examine factors such as aging. We look at average precision versus the disparity in years between the labeled query episode and the test episode, averaged over all episodes in our collection. Performance drops dramatically from .745 AP to .536 AP as we go from 1 to 10 years.

Aging: Our dataset allows us to analyze the effect of aging on recognition [12]. Most datasets looking at aging rely on a few tens of images per individual. This is because a dedicated collection process necessarily takes tens of years, and so one typically amasses smaller datasets by hand. We obtain tens of thousands of images of our main characters spanning 11 years. We can now look at face matching performance versus age. Interestingly, we see little drop in performance for age differences within 5 years. As we look to longer time spans of 10 years between the reference and query episode, performance dramatically drops from .745 to .536 AP.

Properties of the dataset: It is useful to compare our dataset to that of Berg et al [2]. Ours is an order of magnitude larger - 611,770 face images. Both have a small-tailed distribution of names. Popular political figures appearing overwhelmingly often in their dataset, while the main characters appear overwhelmingly often in our video data. Note that secondary characters such as family members and co-workers do appear, but much less frequently. However, in

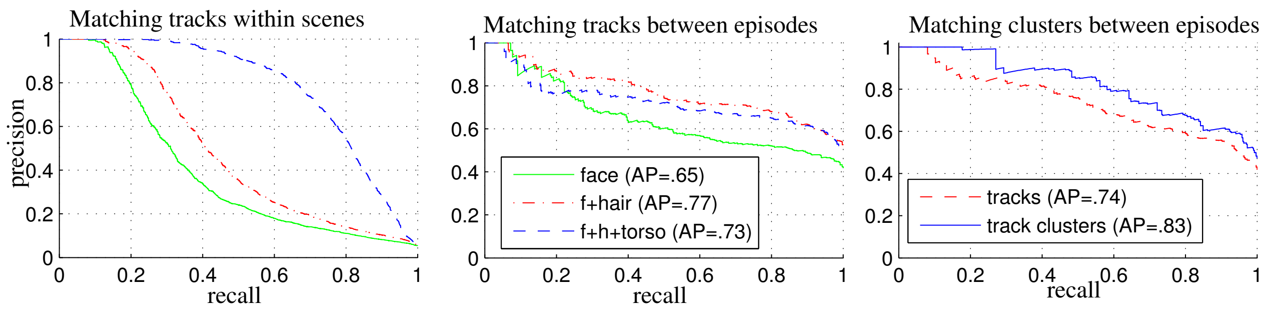

Figure 7. On the **left** and **middle** plots, we look at the effect of appearance representation and temporal scale on matching accuracy (the two plots use the same labeling convention). When matching tracks within a shot, clothing and hair models provide a powerful cue (increasing AP to .779 versus .392 for standard eigenface-based matching). When matching tracks between episodes, clothing is much less helpful. But hair still provides a sharp improvement of 10% over eigenface-based matching. On the **middle** and **right** plots, we look at effect of our track grouping on matching performance. Matching groups of tracks provides a large improvement of 10% over matching individual tracks.

order for their labels to appear in our dataset, they must exist in the reference episodes. We show "interesting" images of the main characters spanning 11 years in Fig.9. These are images for which no frontal face detection was found in the vicinity. We see large pose changes, illumination effects, and expression changes that cause detection failures. Given our criteria, we can verify that 17.8% of the faces in our collection are "interesting". These images explicitly display the benefits of using video for collecting datasets.

Tracking and identifying people: An alternative view of our system is one that can find, track, and identify people over short and long timescales. Consider the task of identifying and locating all main characters whenever they are on-screen. Since our P/R for tracking an individual character is 92/50 (Sec. 6: Tracking) and we correctly label a track with an accuracy of 70/80 (Fig. 6), our overall accuracy for marking an on-screen character is 64/40. To a fair degree, we do in fact know *who* is *where* over a period of 11 years.

Future directions: We are currently in the process of obtaining copyright permission to make this dataset freely available - we hope it will become a useful tool for researchers in the area. We feel that such a large dataset opens up several research directions. We are currently pursuing methods for automated construction of 3D models and large-scale data-driven face recognition, in addition to further studies of aging on recognition performance.

References

[1] O. Arandjelovi and R. Cipolla. Face recognition from video using the generic shape-illumination manifold. In *ECCV*, 2006.

[2] T. Berg, A. Berg, J. Edwards, M. Maire, R. White, Y. Teh, E. Learned-Miller, and D. Forsyth. Faces and names in the news. In *CVPR*, 2004.

[3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT' 98: Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, New York, NY, USA, 1998. ACM Press.

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE T. Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[5] M. Everingham, J. Sivic, and A. Zisserman. Hello! my name is... buffy – automatic naming of characters in tv video. In *BMVC*, 2006.

[6] R. Gross, S. Baker, I. Matthews, and T. Kanade. Face recognition across pose and illumination. In *Handbook of Face Recognition*. Springer-Verlag, 2004.

[7] G. Jaffre and P. Joly. Costume: A new feature for automatic video content indexing. In *Proceedings of RIAO*, 2004.

[8] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Computer Vision*, 43(1):29–44, 2001.

[9] NIST. Trec video retrieval evaluation. www.nlp-ir.nist.gov/projects/trecvid.

[10] J. C. Ralph Gross, Jianbo Shi. Quo vadis face recognition?

[11] D. Ramanan, D. Forsyth, and A. Zisserman. Strike a pose: Tracking people by finding stylized poses. In *CVPR*, June 2005.

[12] N. Ramanathan and R. Chellapa. Modeling age progression in young faces. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2006.

[13] G. Shakhnarovich and B. Moghaddam. Face recognition in subspaces. In *Handbook of Face Recognition*. Springer-Verlag, 2004.

[14] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: video shot retrieval for face sets. In *International Conference on Image and Video Retrieval (CIVR)*, 2005.

[15] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[16] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001.

[17] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196, Morristown, NJ, USA, 1995.

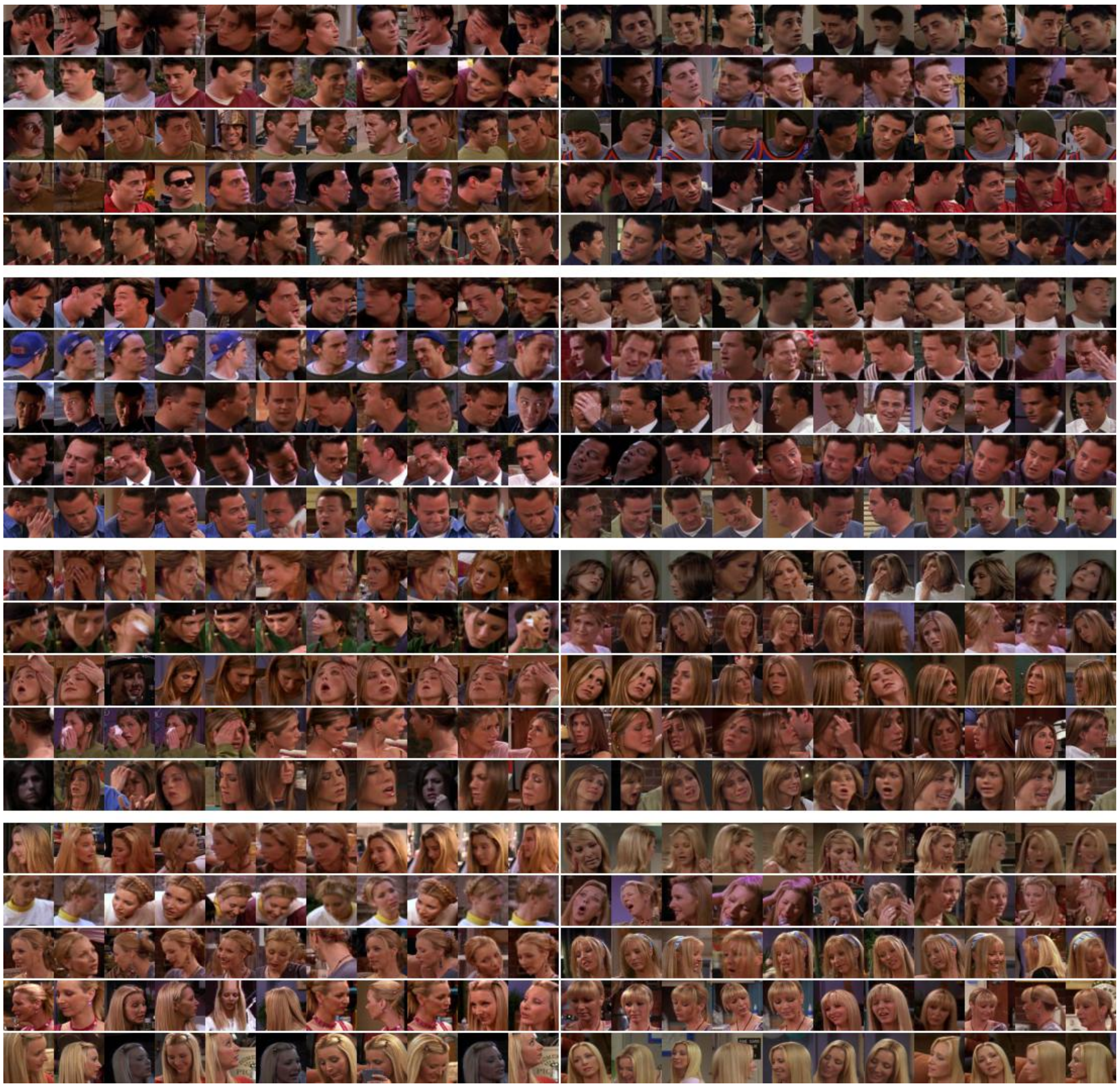

Figure 9. We show 11 years worth of "interesting" images of 4 of the main characters. We define an interesting face image to be one that was not found near any original frontal-face detection. These images are added to our collection by *tracking* around frames adjacent to actual detections. They espouse the benefit of using video to build face collections - one can track to find such interesting cases. We observe large pose changes, illumination effects, and expression changes that are likely to make both face detection and recognition difficult. Such variety makes this dataset attractive for training/evaluating recognition systems. For each character, we show 10 random images from this "interesting" set from each year of the show. For each of the 4 character clusters, the first year (1994) is the **top-left**, the send year (1995) is **top-right**, and the final year (2004) is the **bottom-right**.

[18] W. Zhao, R. Chellappa, A. Rosenfeld, and P. Phillips. Face recognition: A literature survey. In *ACM Computing Surveys*. 2003.