IMPROVING FACE CLUSTERING IN VIDEOS

A Dissertation Presented

by

SOUYOUNG JIN

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2020

College of Information and Computer Sciences

© Copyright by SouYoung Jin 2020

All Rights Reserved

IMPROVING FACE CLUSTERING IN VIDEOS

A Dissertation Presented

by

SOUYOUNG JIN

Approved as to style and content by:

Erik Learned-Miller, Chair

Subhransu Maji, Member

Liangliang Cao, Member

David Huber, Member

James Allan, Chair of the Faculty
College of Information and Computer Sciences

DEDICATION

To my parents,

ChangWhan Jin and Kyung Me Song,

Whom I love the most.

ACKNOWLEDGMENTS

Right before I started writing this Acknowledgment page, I wrote an email to my advisor, Prof. Erik Learned-Miller, to give him my progress on this thesis writing. The email, which might be my last official research progress report to my advisor, made me look back my previous 5.5 years of Ph.D. student life. I first met Erik when I was visiting Amherst as a visiting student for prospective Ph.D. study. I was not sure about my future, but I was deeply impressed by Erik who was very tall and a very nice professor, and we had our first picture together on that day. During the long journey of my Ph.D. life, Erik always has played a very important role. Whenever I was not sure about myself, he pushed me to work harder, think deeper, and encouraged me to continue my study. There were many difficult moments. Two months before my first submission in 2016, I was struggling to produce the first baseline result. Erik came to my cubicle and said that "You will be in a big trouble if you don't make this project working. What are the things required? Do the things right away!!!". It was pretty strict and harsh for me, but pushed me forward and helped me evolve me to the next level. It was a very difficult moment to me, but he helped me to overcome the difficulty. This work was eventually published in ICCV 2017, and it was also my first paper with Erik during my Ph.D. study. Without Erik, I would not have improved this much. He is the best advisor and a very passionate researcher. I wish I could be as successful as Erik, not just in doing research, but also being a good mentor who can also guide junior researchers such as me to achieve their goals.

I am also very appreciate to all of my lab members, my thesis committee members, and all of my collaborators. On my Ph.D. defense day, the school campus was closed due to the snow storm, but all of committee members (Prof. Subhransu Maji, Prof. Liangliang Cao, Prof. David Huber, Prof. Erik Learned-Miller) were willing to attend my talk even in the bad weather condition. They also gave me a lot of valuable feedback and helped me improve my thesis. I especially want to express my gratitude to Prof. Maji, who is co-leading the Vision Lab. His energy and passion in research has opened my eyes to broader world to research.

I would also like to thank my dear collaborators: Dr. Lei Zhang in Microsoft Redmond, Dr. Chris Stauffer in Facebook and VSR, and Prof. Rosemary Cowell in the department of Psycological and Brain Sciences in UMass Amherst.

This long journey might not be possible if my fellow friends were not next to me. I appreciate all my friend relations here in Amherst, and cherish all my memories in my life. First, I would like to thank Eunjung Jee, Kyeongmin Rim, Shinyoung Cho, and Andrew ByungWook Chung, who are always next to me as a part of my family in Amherst. Thanks to God, I could join the wonderful Vision Lab and had opportunities to meet many valuable friends in my life. I would like to say thank you to all of my lab mates, including Pia, Aruni, Jong-Chyi, Tsung-Yu, Ashish, Chenyun, Matheus, Gopal, Zezhou, Mikayla, Hang and Huaizu. Especially, I would like to mention two of these names: Ashish Singh and Tsung-Yu Lin. Ashish always tells me that I am the best scholar in the world, even though he witnessed many of my struggling moments. Those words indeed made me believe in myself and keep working hard. Lastly, I am very blessed that I could find my best friend here in UMass, Tsung-Yu, who co-entered/graduated UMass together.

Finally, I give props to my lovely family and fiancé who are always there to support me and give me more than enough love. I am sure my dad in heaven would be very proud of his younger daughter ♡

ABSTRACT

IMPROVING FACE CLUSTERING IN VIDEOS

FEBRUARY 2020

SOUYOUNG JIN
B.Sc., DONGGUK UNIVERSITY
M.Sc., KOREA ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY
Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Erik Learned-Miller

Human faces represent not only a challenging recognition problem for computer vision, but are also an important source of information about identity, intent, and state of mind. These properties make the analysis of faces important not just as algorithmic challenges, but as a gateway to developing computer vision methods that can better follow the intent and goals of human beings. In this thesis, we are interested in *face clustering in videos*. Given a raw video, with no caption or annotation, we want to group all detected faces by their identity. We address three problems in the area of face clustering and propose approaches to tackle them.

The existing link-based face-clustering system is sensitive to a false connection between two different people. We introduce a new similarity measure that helps the verification system to provide very few false connections at moderate recall. Further, we also introduce a novel clustering method called Erdős and Rényi clustering, which is based on the observations from a random graph model theory, that large clusters can be fully connected by joining just a small fraction of their node pairs. Our results present state-of-the-art results on multiple video data sets and also on standard face databases.

What happens if faces are not sufficiently clear for direct recognition, due to the small scale, occlusion, or extreme pose? We observe that, when humans are uncertain about the identity of two faces, we use clothes or other contextual cues, e.g. specific objects or textures, to infer identity. With this observation, we propose the Face-Background Network (FB-Net), which takes as input not only the faces but also the entire scene to enhance the performance of face clustering. In order for the network to learn background features that are informative about the identity, we introduce a new dataset that contains face identities in the context of consistent scenes. We show that FB-Net outperforms the state-of-the-art method which uses face-level features only for the task of video face clustering.

The performance of face clustering depends on a good face detector. However, improving the performance of a face detector requires expensive labeling of faces. In this work, we propose an approach to reduce mistakes of the existing face detector by using many hours of freely available unlabeled videos on the web. Specifically, with the observation that false positives/negatives are often isolated in time, we demonstrate a method to mine hard examples automatically using temporal continuity in videos. In particular, we analyze the output of a trained detector on video sequences and mine detections that are isolated in time, which is likely to be hard examples. Our experiments show that re-training detectors on these automatically obtained examples often significantly improves performance. We present experiments on multiple architectures and multiple data sets, including face detection, pedestrian detection, and other object categories.

TABLE OF CONTENTS

Page

CHAPTER

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

Videos are good sources to get to know about something we have not experienced before. We eventually want an AI system to watch a video and learn human life through it, and the first step to make this possible is to help the machine understand videos, *i.e.* video understanding. Building a video understanding system requires many essential components, such as object detection, tracking, recognition, and sentiment analysis. In this thesis, we are interested in the topic of *face-clustering in videos* – a problem of grouping faces in a video so that each group contains a unique individual [32, 19, 55, 125]. Specifically, given a raw video, with no caption or annotation, we want to group all detected faces by their identity. The topic of face-clustering is one of the important topics in video understanding, not only because the topic has many applications but also because of the difficulty that machines have in solving this problem. On the other hand, humans are good at identifying the faces of the same person even under severe poses and occlusions without paying too much attention to it.

For face clustering, the link-based clustering algorithm [102] is one of popularly used algorithms. In this chapter, we first give an overview of how the link-based clustering algorithm works on faces (Chapter 1.1). Then, we address three important problems in the area of face-clustering (Chapter 1.2).

1.1 Link-Based Clustering Algorithm for Face Clustering

Suppose we have a face-detector and run the detector on every frame in a video. Each of the detected faces is considered as a node in a graph. Our goal is to connect nodes from the same identity while separate the nodes from two different identities.

To do that, we first compute a pairwise distance matrix between all nodes with a distance (or similarity) measure, e.g. $\ell 2$-distance, which indicates how different (or similar) two faces are. Specifically, given a face recognition network that is trained to identify multiple people, we compute the embedding for each node, by using a pre-classification layer of features from the network. Then, we create a link between nodes only if a distance score is below a certain threshold. Finally, connected nodes are formed as a cluster.

1.2 Problems and Contributions

In this thesis, three important problems in the area of face-clustering are addressed. We also propose novel approaches to tackle the problems. Firstly, we propose a new link-based clustering algorithm with a new similarity measure between faces for better face-clustering. Secondly, we introduce a novel network architecture that takes as input both the target face and the entire corresponding video frame in order to incorporate additional information outside faces for person-identity clustering in videos. Finally, a new method is proposed for pseudo-labeling that uses temporal consistency cues from unlabeled videos to mine large amounts of hard examples without human annotation.

1.2.1 Improving Face-Clustering Using Erdős-Rényi Clustering

Existing link-based face-clustering system creates a link between two faces based on a face verification system. However, in the link-based clustering system, just a single incorrect connection between two different people can lead to poor clustering results. Thus, in Chapter 3, We introduce a novel verification method, *rank-1 counts verification*, that provides very few false connections at moderate recall. We then introduce a novel clustering method, motivated by the classic graph theory results of Erdős and Rényi [30], which is based on the observations that large clusters can be fully connected by joining just a small fraction of their point pairs. Finally, the rank-1 counts verification is used in the link-based clustering scheme.

We make three contributions in this work:

- A new approach to combining high-quality face detection [54] and generic tracking [104] to improve both precision and recall of our video face detection.
- A new method, *Erdős-Rényi clustering*, for large-scale clustering of images and video tracklets. We argue that effective large-scale face-clustering requires face verification with fewer false positives, and we introduce *rank-1 counts verification*, showing that it indeed achieves better true positive rates in low false positive regimes. Rank-1 counts verification, used with simple link-based clustering, achieves high quality clustering results on three separate video data sets.
- A principled evaluation for the end-to-end problem of face detection and clustering in videos; until now there has been no clear way to evaluate the quality of such an end-to-end system, but only to evaluate its individual parts (detection and clustering).

1.2.2 Improving Face-Clustering Using Face-Background Network (FB-Net)

What happens if faces are not sufficiently clear for direct recognition, due to distance, occlusion, or other factors? When faces are not clearly visible, humans may use clothes or other contextual cues to infer identity. Inspired by this, in Chapter 4, we propose the Face-Background Network (FB-Net), which takes as input not just faces but also the entire scene to enhance face-clustering. In order for the network to learn background features that are informative about identity, we introduce a new dataset that contains not just face identities but also faces in the context of consistent scenes. These images contain views of the same characters from different shots within the same scene, allowing the network to learn how consistent identities are correlated with consistent scene elements, especially the same scene elements from different points of view. Thus, the dataset can help the network learn not just face level features, but also parts of the background that can improve face-clustering. Our FB-Net uses a transformer module [128] in a novel way to learn useful scene level features that improve face verification and hence face-clustering. Our results show that FB-Net outperforms the state-of-the-art method, which exploits face-level features only, in video face-clustering.

This work contains the following contributions:

- We introduce the FB-Net that takes as input not just faces but also the entire scene to enhance face-clustering.
- The FB-Net is trained with a transformer module in a novel way to learn useful scene level features with face/background processors. The learned embeddings are evaluated on the test videos in which the network has never seen the actors before. FB-Net outperforms the state-of-the-art method which exploits face-level features only.
- To make the network to learn background features that are informative about identity, we provide a new dataset that allows the network to learn consistent scene elements from different points of view.

1.2.3 Improving Face-Detection by Training with Hard Example Mining

The performance of face-clustering depends on a good face-detector, while an existing face-detector also makes mistakes. However, improving the performance of a face-detector requires expensive labeling of faces. In Chapter 5, we propose an approach to reduce mistakes of the existing face-detector by using many hours of freely available unlabeled videos on the web by using *temporal continuity*. Specifically, important gains have recently been obtained in object detection by using training objectives that focus on *hard negative* examples, i.e., negative examples that are currently rated as positive or ambiguous by the detector. These examples can strongly influence parameters when the network is trained to correct them. Unfortunately, they are often sparse in the training data and are expensive to obtain. In this work, we show how large numbers of hard negatives can be obtained *automatically* by analyzing the output of a trained detector on video sequences. In particular, detections that are *isolated in time*, i.e., that have no associated preceding or following detections, are likely to be hard negatives. We describe simple procedures for mining large numbers of such hard negatives (and also *hard positives*) from unlabeled video data. Our experiments show that retraining detectors on these automatically obtained examples often significantly improves performance. We present experiments on multiple architectures and multiple data sets, including face detection, pedestrian detection, and other object categories.

In this work, we have three contributions as follows:

- We use temporal consistency cues from unlabeled videos to mine large amounts of hard examples without human annotation.

- We show improvements using standard architectures on well-known Pedestrian and Face detection benchmarks.

- Our hard-example mining can be easily extended to other categories, utilizing the abundance of unlabeled videos on YouTube for almost any object category.

CHAPTER 2

RELATED WORK

In this chapter, we first discuss face tracking and then the problem of clustering faces in videos (Chapter 2.1). Then, we review work related to the recognition of characters in videos and movies (Chapter 2.2). We also review previous work related to using contextual cues in recognition of a person of interest (Chapter 2.3). Finally, we discuss object detection (Chapter 2.4).

2.1 Face Tracking and Clustering in Videos

We can divide the face clustering work into two categories: fully unsupervised and with some supervision. We then discuss prior work using reference images.

Recent work on *robust face tracking* [124, 98, 85] has gradually expanded the length of face tracklets, starting from face detection results. Ozerov *et al.* [85] merge results from different detectors by clustering based on spatio-temporal similarity. Clusters are then merged, interpolated, and smoothed for face tracklet creation. Similarly, Roth *et al.* [98] generate low-level tracklets by merging detection results, form high-level tracklets by linking low-level tracklets, and apply the Hungarian algorithm to form even longer tracklets. Tapaswi *et al.* [124] improve on this [98] by removing false positive tracklets.

With the development of multi-face tracking techniques, *the problem of clustering TV characters* has also been widely studied [123, 46, 32, 10, 139, 138, 126]. Given precomputed face tracklets, the goal is to assign a name or an ID to a group of face tracklets with the same identity. Wu *et al.* [139, 138] iteratively cluster face tracklets and link clusters into longer tracks in a bootstrapping manner. Tapaswi *et al.* [126] train classifiers to find thresholds for joining tracklets in two stages: within a scene and across scenes. Similarly, in Chapter 3, we aim to generate face clusters in a fully unsupervised manner.

Though solving this problem may yield a better result for face tracking, some forms of supervision specific to the video or characters in the test data can improve performance. Tapaswi *et al.* [123] perform face recognition, clothing clustering and speaker identification, where face models and speaker models are first trained on other videos containing the same main characters as in the test set. In [32, 10], subtitles and transcripts are used to obtain weak labels for face tracks. More recently, Haurilet *et al.* [46] solve the problem without transcripts by resolving name references only in subtitles. Our approaches in Chapter 3 and Chapter 4 are more broadly applicable because it does not use subtitles, transcripts, or any other supervision related to the identities in the test data, unlike these other works [123, 46, 32, 10].

A standard procedure for face clustering is to leverage constraints in a video by learning cast-specific metrics [19, 138, 139, 140] by considering face images within tracks as similar. The constraints can be further used to jointly fine-tune face representations [155]. Recent methods have focused on using temporal consistency to identify false positive and missed detections and improve clustering performance [55] and using inductive biases in the representation space [125].

As in the proposed verification system in Chapter 3, some existing work [21, 45] uses reference images. For example, index code methods [45] map every single image to a code based upon a set of reference images, and then compare these codes. On the other hand, our Erdős-Rényi Clustering algorithm compares the relative distance of two images with the distance of one of the images to the reference set, which is different. In addition, we use the newly defined rank-1 counts, rather than traditional Euclidean or Mahalanobis distance measures to compare images [21, 45] for similarity measures.

2.1.1 Person Re-Identification

The person re-identification task [89, 156, 24, 63, 70, 157] is defined to match pedestrians from different non-overlapping cameras. While related, person re-identification is significantly different from video face clustering in terms of camera setting and lack of diverse context. Specifically, in the person re-identification task, the camera is assumed to be stationary all the time. As a result, multiple people can have the same background, which is often not associative with their identity. Unlike the person re-identification task, the face clustering task aims to group characters in a video/movie using faces as the primary supervision. In addition, video frames do not always show the whole person body. It is not guaranteed that a movie character does not change clothes across different scenes. Furthermore, due to constantly changing camera views for the same scene and person, person re-identification methods cannot be directly applied to the video face clustering problem.

2.2 Face Recognition in Videos

Previous works have addressed the character identification task in videos in a variety of ways. Due to the availability of multi-modal information from videos, early efforts focused on using the supervision from transcripts (speaker names and dialogues) [10, 32, 90, 113, 20]. Recent works have further focused on using other forms of multi-modal context like speech [82] and temporal consistency using face tracks [86, 144]. Apart from the in-domain context, recent work has also used supervision from web data [2, 83, 129] to improve performance. Automatic identification methods using only visual data primarily make use of constraints from different modalities of local context like clothing[123] and hairstyle [83].

2.3 Context-Based Video Understanding

Contextual information has been widely studied for human and computer vision prediction tasks. Visual context comes in various forms. [25] provides a taxonomy of sources of contextual information, and how they can benefit different stages of visual recognition. Previous works in different sub-areas of computer vision (object detection [11, 107], scene understanding [72]) have reported significant performance improvements by using contextual information.

In Chapter 2.3.1, we first review the studies on how contexts are used for face detection and recognition. Then, in Chapter 2.3.2, we discuss attention mechanism to study the contextual information.

2.3.1 Context for Face Detection/Recognition

Significant improvements have been observed by simultaneously modeling context representation for the face detection task [122, 50]. With respect to face or person recognition in the wild, previous works have demonstrated the utility of additional information from outside of the face region. This could be attributed to factors like occlusion, pose and illumination variation that make this problem challenging. Early works incorporated multiple forms of additional cues like clothing, timestamps, scenes, etc [4, 38, 114, 67].

With the advent of deep learning, recent work has focused on obtaining more robust features by integrating different forms of contextual cues. By combining additional information like full-body, pose [151] and weighted full body cues [56] with face-level features, these methods achieve better performance than only using faces. Further recent works consider social context [60, 64, 52] along with jointly learning representations for multiple regions of interest (face, head, upper body, whole-body) in identifying the person. In contrast, our work focuses on adaptively learning background features (local and global scene context) that are most informative for matching identities. Unlike the previous methods, in the FB-Net (Chapter 4), no extra ground-truth annotation is used to learn contextual cues.

2.3.2 Attention in Neural Networks

Recently, a large amount of work has been proposed that utilizes attention, primarily in the language-related tasks [128, 142]. In videos, attention has been primarily incorporated in tasks like action recognition and video classification. Attention in these tasks have been formulated in various different ways, including self-attention [132], second-order pooling or gating [78, 41, 141, 73], human pose [8] and graph-based architectures[133]. In our FB-Net, we utilize self-attention to learn contextual representation conditioned on the face region in the image. Unlike [40], we do not use self-attention for the test-time objective, *i.e.* classification task. Instead, we utilize the self-attention to learn better face embeddings.

2.4 Object Detection

Convolutional neural networks (CNNs) have recently been applied to achieve state-of-the-art results in object detection [43, 42, 47, 95, 92, 71, 14, 68]. Many of these object detectors have been re-purposed for other tasks such as face detection [91, 61, 145, 33, 65, 155, 148, 54, 135, 50, 153] or pedestrian detection [150, 29, 14], [15, 49, 62, 152], achieving impressive results [53, 146, 27]. In this section, we first talk about the approaches that focus on harder examples to improve performance. We also review the semi-supervised work for object detection.

2.4.1 Training with Hard Examples

Massive class imbalance is an issue with sliding-window-style object detectors — being densely applied over an image. Such models see far more "easy" negative samples from background regions than positive samples from regions containing an object. Some form of hard negative mining is used by most successful object detectors to account for this imbalance [22, 26, 34, 43, 42, 47, 108, 150, 69, 131, 118]. Early approaches include *bootstrapping* [119] for training SVM-based object detectors [22, 34], where false positive detections were added to the set of background training samples in an incremental fashion. Other methods [99, 26] apply a pre-trained detector on a larger dataset to mine false positives and then re-train.

Hard negative mining has also improved the performance of deep learning based models [110, 74, 42, 108, 150, 131, 69]. Shrivastava *et al.* [108] proposed an *Online Hard Example Mining* (OHEM) procedure,training using only high-loss region proposals. This technique, originally applied to the Fast R-CNN detector [42], yielded significant gains on the PASCAL and MS-COCO benchmarks. Lin *et al.* [69] propose the *focal loss* to down-weight the contribution of easy examples and train a single-stage, multi-scale network [68]. The A-Fast-RCNN [134] does adversarial generation of hard examples using occlusions and deformations. While similar to our work, our model is trained with hard examples from *real* images and variations are not limited to occlusion and spatial deformations. Zhang *et al.* [150] show that effective bootstrapping of hard negatives, using a boosted decision forest [37, 5], significantly improves over a Faster R-CNN baseline for *pedestrian detection*. Recent *face detection* methods, such as Wan *et al.* [131] and Sun *et al.* [118], have also used the bootstrapping of hard negatives to improve the performance of CNN-based detectors — a pre-trained Faster R-CNN is used to mine hard negatives; then the model is re-trained. However, these methods require a human-annotated dataset of suitable size. Our unsupervised approach in Chapter 5 does not rely upon bounding-box annotations and thus can be trained upon potentially unlimited data.

2.4.2 Semi-Supervised Learning

Using mixtures of labeled and unlabeled data is known as *semi-supervised learning* [12, 18, 136]. Rosenberg *et al.* [97] ran a trained object detector on unlabeled data and then trained on a subset of this noisy labeled data in an incremental re-training procedure. In Kalal *et al.* [57], constraints based on video object trajectories are used to correct patch labels of a random forest classifier; these corrected samples are used for re-training. Tang *et al.* [121] adapt still-image object detectors to video by selecting training samples from unlabeled videos, based on the consistency between detections and tracklets, and then follow an iterative procedure that selects the easy examples from videos and hard examples from images to re-train the detector. Rather than adapting to the video domain, we seek to improve detector performance on the source domain by selecting hard examples from videos. Singh *et al*. [112] gather discriminative regions from weakly-labeled images and then refine their bounding-boxes by incorporating tracking information from weakly-labeled videos.

CHAPTER 3

END-TO-END FACE DETECTION AND CAST GROUPING IN MOVIES USING ERDŐS-RÉNYI CLUSTERING

The problem of identifying face images in video and clustering them together by identity is a natural precursor to high impact applications such as video understanding and analysis. This general problem area was popularized in the paper "Hello! My name is...Buffy" [32], which used text captions and face analysis to name people in each frame of a full-length video. In this work, we use only raw video (with no captions), and group faces by identity rather than naming the characters. In addition, unlike face clustering methods that start with detected faces, we include detection as part of the problem. This means we must deal with false positives and false negatives, both algorithmically, and in our evaluation method. We make three contributions:

- A new approach to combining high-quality face detection [54] and generic tracking [104] to improve both precision and recall of our video face detection.

- A new method, *Erdős-Rényi clustering*, for large-scale clustering of images and video tracklets. We argue that effective large-scale face clustering requires face verification with fewer false positives, and we introduce *rank-1 counts verification*, showing that it indeed achieves better true positive rates in low false positive regimes. Rank-1 counts verification, used with simple link-based clustering, achieves high quality clustering results on three separate video data sets.

- A principled evaluation for the end-to-end problem of face detection and clustering in videos; until now there has been no clear way to evaluate the quality of such an end-to-end system, but only to evaluate its individual parts (detection and clustering).

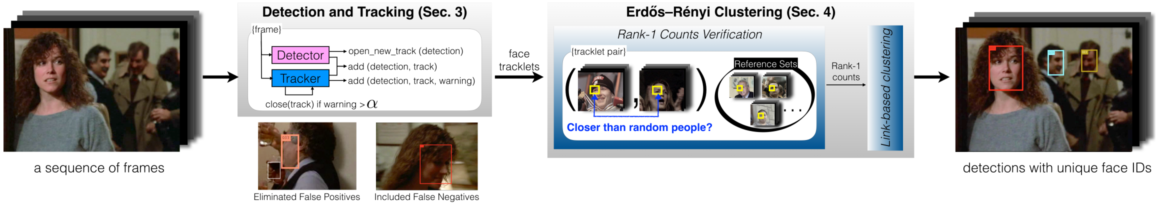

a sequence of frames
Detection and Tracking (Sec. 3)
{frame}
Detector
Tracker
open_new_track (detection)
add (detection, track)
add (detection, track, warning)
close(track) if warning > α
Eliminated False Positives
Included False Negatives
face tracklets
Erdős–Rényi Clustering (Sec. 4)
Rank-1 Counts Verification
{tracklet pair}
Reference Sets
Closer than random people?
Rank-1 counts
Link-based clustering
detections with unique face IDs

Figure 3.1: Overview of approach. Given a movie, our approach generates tracklets (Chapter 3.1) and then does Erdős-Rényi Clustering and FAD verification between all tracklet pairs. (Chapter 3.2) Our final output is detections with unique character Ids.

3.1 Detection and tracking

Our goal is to take raw videos, with no captions or annotations, and to detect all faces and cluster them by identity. We start by describing our method for generating *face tracklets*, or continuous sequences of the same face across video frames. We wish to generate clean face tracklets that contain face detections from just a single identity. Ideally, exactly one tracklet should be generated for an identity from the moment his/her face appears in a shot until the moment it disappears or is completely occluded.

To achieve this, we first detect faces in each video frame using the *Faster R-CNN* object detector [93], but retrained on the WIDER face data set [147], as described by Jiang et al. [54]. Even with this advanced detector, face detection sometimes fails under challenging illumination or pose. In videos, those faces can be detected before or after the challenging circumstances by using a tracker that tracks both forward and backward in time. We use the *distribution field tracker* [104], a general object tracker that is not trained specifically for faces. Unlike face detectors, the tracker's goal is to find in the next frame the object most similar to the target in the current frame. The extra faces found by the tracker compensate for missed detections (Fig. 3.1, bottom of block 2). Tracking helps not only to catch false negatives, but also to link faces of equivalent identity in different frames.

One simple approach to combining a detector and tracker is to run a tracker forward and backward in time from *every single face detection* for some fixed number of frames, producing a large number of "mini-tracks". A Viterbi-style algorithm [35, 23] can then be used to combine these mini-tracks into longer sequences. This approach is computationally expensive since the tracker is run many times on overlapping subsequences, producing heavily redundant mini-tracks. To improve performance, we developed the following novel method for combining a detector and tracker. Happily, it also improves precision and recall, since it takes advantage of the tracker's ability to form long face tracks of a single identity.

The method starts by running the face detector in each frame. When a face is first detected, a tracker is initialized with that face. In subsequent frames, faces are again detected. In addition, we examine each current tracklet to see where it might be extended by the tracking algorithm in the current frame. We then check the agreement between detection and tracking results. We use the intersection over union (IoU) between detections and tracking results with threshold 0.3, and apply the Hungarian algorithm[59] to establish correspondences among multiple matches. If a detection matches a tracking result, the detection is stored in the current face sequence such that the tracker can search in the next frame given the detection result. For the detections that have no matched tracking result, a new tracklet is initiated. If there are tracking results that have no associated detections, it means that either **a)** the tracker could not find an appropriate area on the current frame, or **b)** the tracking result is correct while the detector failed to find the face. The algorithm postpones its decision about the tracked region for the next α consecutive frames ($\alpha = 10$). If the face sequence has any matches with detections within α frames, the algorithm will keep the tracking results. Otherwise, it will remove the tracking-only results. The second block of Fig. 3.1 summarizes our proposed face tracklet generation algorithm and shows examples corrected by our joint detection-tracking strategy. Next, we describe our approach to clustering based on low false positive verification.

3.2 Erdős-Rényi Clustering and Rank-1 Counts Verification

In this section, we describe our approach to clustering face images, or, in the case of videos, face tracklets. We adopt the basic paradigm of *linkage clustering*, in which each

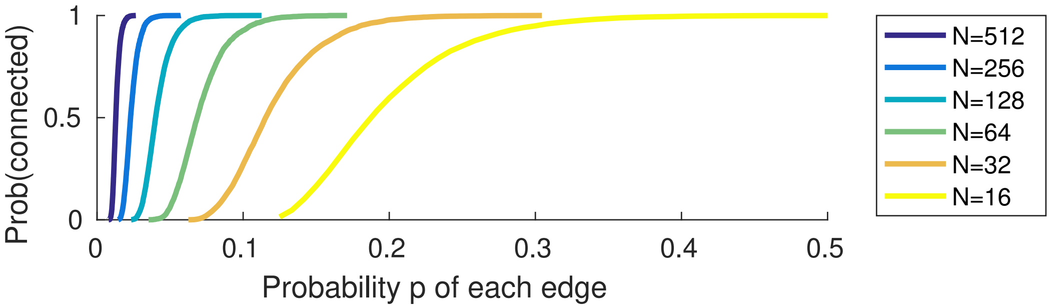


Figure 3.2: Simulation of cluster connectedness as a function of cluster size, N, and the probability p of connecting point pairs. The figure shows that for various N (different colored lines), the probability that the cluster is fully connected (on the y-axis) goes up as more pairs are connected. For larger graphs, a small probability of connected pairs still leads to high probability that the graph will be fully connected.

pair of points (either images or tracklets) is evaluated for linking, and then clusters are formed among all points connected by linked face pairs. We name our general approach to clustering *Erdős-Rényi clustering* since it is inspired by classic results in graph theory due to Erdős and Rényi [30], as described next.

Consider a graph G with n vertices and probability p of each possible edge being present. This is the Erdős-Rényi random graph model [30]. The expected number of edges is $\binom{n}{2}p$. One of the central results of this work is that, for $\epsilon > 0$ and n sufficiently large, if

$$p > \frac{(1+\epsilon)\ln n}{n}, \tag{3.1}$$

then the graph will almost surely be connected (there exists a path from each vertex to every other vertex). Fig. 3.2 shows this effect on different graph sizes, obtained through simulation.

Consider a clustering system in which links are made between tracklets by a *verifier* (a face verification system), whose job is to say whether a pair of tracklets is the "same" person or two "different" people. While graphs obtained in clustering problems are not uniformly random graphs, the results of Erdős and Rényi suggest that this verifier can have a fairly low recall (percentage of same links that are connected) and still do a good job connecting large clusters. In addition, false matches may connect large clusters of different identities, dramatically hurting clustering performance. This motivates us to build a verifier that focuses on low false positives rather than high recall. In the next section, we present our approach to building a verifier that is designed to have good recall at low false positive rates, and hence is appropriate for clustering problems with large clusters, like grouping cast members in movies.

3.2.1 Rank-1 Counts for Fewer False Positives

Our method compares images by comparing their multidimensional feature vectors. More specifically, we count the number of feature dimensions in which the two images are closer in value than the first image is to any of a set of reference images. We call this number the *rank-1 count* similarity. Intuitively, two images whose feature values are "very close" for many different dimensions are more likely to be the same person. Here, an image is considered "very close" to a second image in one dimension if it is closer to the second image in that dimension than to any of the reference images.

More formally, to compare two images I_A and I_B, our first step is to obtain feature vectors A and B for these images. We extract 4096-D feature vectors from the *fc7* layer of a standard pre-trained face recognition CNN [87]. In addition to these two images, we use a fixed reference set with G images (we typically set $G = 50$), and compute CNN feature vectors for each of these reference images.[1] Let the CNN feature vectors for the reference images be $R^1, R^2, ..., R^G$. We sample reference images from the *TV Human Interactions Dataset* [88], since these are likely to have a similar distribution to the images we want to cluster.

For each feature dimension i (of the 4096), we ask whether

[1] The reference images may overlap in identity with the clustering set, but we choose reference images so that there is no more than one occurrence of each person in the reference set.

$$|A_i - B_i| < \min_j |A_i - R_i^j|.$$

That is, is the value in dimension i closer between A and B than between A and all the reference images? If so, then we say that the ith feature dimension is *rank-1* between A and B. The cumulative *rank-1 counts* feature $\mathbf{R}$ is simply the number of rank-1 counts across all 4096 features:

$$\mathbf{R} = \sum_{i=1}^{4096} I\left[|A_i - B_i| < \min_j |A_i - R_i^j|\right],$$

where $I[\cdot]$ is an indicator function which is 1 if the expression is true and 0 otherwise.

Taking inspiration from Barlow's notion that the brain takes special note of "suspicious coincidences" [9], each rank-1 feature dimension can be considered a suspicious coincidence. It provides some weak evidence that A and B may be two images of the same person. On the other hand, in comparing all 4096 feature dimensions, we expect to obtain quite a large number of rank-1 feature dimensions even if A and B are *not* the same person.

When two images and the reference set are selected randomly from a large distribution of faces (in this case they are usually different people), the probability that A is closer to B in a particular feature dimension than to any of the reference images is just

$$\frac{1}{G+1}.$$

Repeating this process 4096 times means that the expected number of rank-1 counts is simply

$$E[\mathbf{R}] = \frac{4096}{G+1},$$

since expectations are linear (even in the presence of statistical dependencies among the feature dimensions). Note that this calculation is a fairly tight *upper bound* on the expected number of rank-1 features *conditioned on the images being of different identities*, since most pairs of images in large clustering problems are different, and conditioning on ”different” will tend reduce the expected rank-1 count. Now if two images I_A and I_B have a large rank-1 count, it is likely they represent the same person. The key question is how to set the threshold on these counts to obtain the best verification performance.

Recall that our goal, as guided by the Erdős-Rényi random graph model, is to find a threshold on the rank-1 counts $\mathbf{R}$ so that we obtain very few false positives (declaring two different faces to be “same”) while still achieving good recall (a large number of same faces declared to be “same”). Fig. 3.3 shows distributions of rank-1 counts for various subsets of image pairs from Labeled Faces in the Wild (LFW) [51]. The red curve shows the distribution of rank-1 counts for *mismatched* pairs from all possible mismatched pairs in the entire data set (not just the test sets). Notice that the mean is exactly where we would expect with a gallery size of 50, at $\frac{4096}{51} \approx 80$. The green curve shows the distribution of rank-1 counts for the matched pairs, which is clearly much higher. The challenge for clustering, of course, is that we don’t have access to these distributions since we don’t know which pairs are matched and which are not. The yellow curve shows the rank-1 counts for *all* pairs of images in LFW, which is nearly identical to the distribution of mismatched rank-1 counts, *since the vast majority of possible pairs in all of LFW are mismatched*. This is the distribution to which the clustering algorithm has access.

If the 4,096 CNN features were statistically independent (but not identically distributed), then the distribution of rank-1 counts would be a binomial distribution (blue curve). In this case, it would be easy to set a threshold on the rank-1 counts to guarantee a small number of false positives, by simply setting the threshold to be near the right end of the mismatched (red) distribution. However, the dependencies among the CNN features prevent the mismatched rank-1 counts distribution from being binomial, and so this approach is not possible.

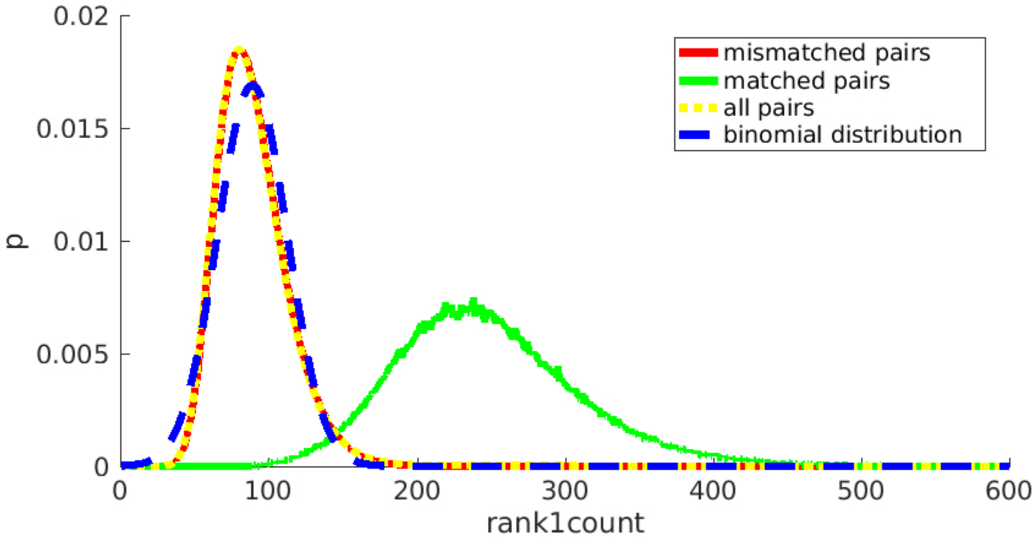


Figure 3.3: LFW distribution of rank-1 counts. Each distribution is normalized to sum to 1.

3.2.2 Automatic Determination of Rank-1 Count Threshold

Ideally, if we could obtain the rank-1 count distribution of mismatched pairs of a test set, we could set the threshold such that the number of false positives becomes very low. However, it is not clear how to get the actual distribution of rank-1 counts for mismatched pairs at test time.

Instead, we can estimate the shape of the mismatched pair rank-1 count distribution using one distribution (LFW), and use it to estimate the distribution of mismatched rank-1 counts for the test distribution. We do this by fitting the *left half* of the LFW distribution to the *left half* of the clustering distribution using scale and location parameters. The reason we use the left half to fit the distribution is that this part of the rank-1 counts distribution is almost exclusively influence by *mismatched pairs*. The *right side* of this matched distribution then gives us an approximate way to threshold the test distribution to obtain a certain false positive rate. It is this method that we use to report the results in the leftmost column of Table 3.2.

Table 3.1: Verification performance comparisons on all possible LFW pairs. The proposed rank-1 counts gets much higher recall at fixed FPRs.

FPR	**Rank1count**	L2	Template Adaptation [21]	Rank-Order Distance [158]
1E-9	**0.0252**	0.0068	0.0016	0.0086
1E-8	**0.0342**	0.0094	0.0017	0.0086
1E-7	**0.0614**	0.0330	0.0034	0.0086
1E-6	**0.1872**	0.1279	0.0175	0.0086
1E-5	**0.3800**	0.3154	0.0767	0.0427
1E-4	**0.6096**	0.5600	0.2388	0.2589
1E-3	0.8222	0.7952	0.5215	**0.8719**
1E-2	0.9490	0.9396	0.8204	**0.9656**
1E-1	**0.9939**	0.9915	0.9776	0.9861

A key property of our rank-1 counts verifier is that it has good recall across a wide range of the low false positive regime (FPR). Thus, our method is relatively robust to the setting of the rank-1 counts threshold. In order to show that our rank-1 counts feature has good performance for the types of verification problems used in clustering, we construct a verification problem using *all possible pairs* of the LFW database [51]. In this case, the number of mismatched pairs (quadratic in N) is much greater than the number of matched pairs. As shown in Table 3.1, we observe that our verifier has higher recall than three competing methods (all of which use the same base CNN representation) at low false positive rates.

Using rank-1 counts verification for tracklet clustering. In our face clustering application, we consider every pair (I, J) of tracklets, calculate a value akin to the rank-1 count R, and join the tracklets if the threshold is exceeded. In order to calculate an R value for tracklets, we sample a random subset of 10 face images from each tracklet, compute a rank-1 count R for each pair of images, and take the maximum of the resulting R values.

3.2.3 Averaging over Gallery Sets

While our basic algorithm uses a fixed (but randomly selected) reference gallery, the method is susceptible to the case in which one of the gallery images happens to be similar in appearance to a person with a large cluster, resulting in a large number of false negatives. To mitigate this effect, we implicitly average the rank-1 counts over an exponential number of random galleries, as follows.

The idea is to sample random galleries of size g from a larger *super-gallery* with G images; we used $g = 50, G = 1000$. We are interested rank-1 counts, in which image A's feature is closer to B than to any of the gallery of size g. Suppose we know that among the 1000 super-gallery images, there are K (e.g., $K = 3$) that are closer to A than B is. The probability that a random selection (with replacement) of g images from the super-gallery would contain none of the K closer images (and hence represent a rank-1 count) is

$$r(A, B) = \left(1.0 - \frac{K}{G}\right)^g .$$

That is, $r(A, B)$ is the *probability* of having a rank-1 count with a random gallery, and using $r(A, B)$ as the count is equivalent to averaging over all possible random galleries. In our final algorithm, we sum these probabilities rather than the deterministic rank-1 counts.

3.2.4 Efficient Implementation

For simplicity, we discuss the computational complexity of our fixed gallery algorithm; the complexity of the average gallery algorithm is similar. With F, G, and N indicating the feature dimensionality, number of gallery images, and number of face tracklets to be clustered, the time complexity of the naive rank-1 count algorithm is $\mathcal{O}(F * G * N^2)$.

However, for each feature dimension, we can sort N test image feature values and G gallery image feature values in time $\mathcal{O}((N + G)\log(N + G))$. Then, for each value in test image A, we find the closest gallery value, and increment the rank-1 count for the test images that are closer to A. Let Y be the average number of steps to find the closest gallery value. This is typically much smaller than N. The time complexity is then $\mathcal{O}(F * [(N + G)\log(N + G) + N * Y])$.

3.2.5 Clustering with Do-Not-Link Constraints

It is common in clustering applications to incorporate constraints such as *do-not-link* or *must-link*, which specify that certain pairs should be in separate clusters or the same cluster, respectively [130, 105, 76, 66, 80]. They are also often seen in the face clustering literature [19, 138, 139, 85, 126, 155]. These constraints can be either rigid, implying they must be enforced [130, 105, 80, 85], or soft, meaning that violations cause an increase in the loss function, but those violations may be tolerated if other considerations are more important in reducing the loss [76, 66, 138, 139, 155].

In this work, we assume that if two faces appear in the same frame, they must be from different people, and hence their face images obey a do-not-link constraint. Furthermore, we extend this hard constraint to the tracklets that contain faces. If two tracklets have any overlap in time, then the entire tracklets represent a do-not-link constraint.

We enforce these constraints on our clustering procedure. Note that connecting all pairs below a certain dissimilarity threshold followed by transitive closure is equivalent to single-linkage agglomerative clustering with a joining threshold. In agglomerative clustering, a pair of closest clusters is found and joined at each iteration until there is a single cluster left or a threshold met. A naïve implementation will simply search and update the dissimilarity matrix at each iteration, making the whole process $\mathcal{O}(n^3)$ in time. There are faster algorithms giving the optimal time complexity $\mathcal{O}(n^2)$ for single-linkage clustering [109, 81]. Many of these algorithms incur a dissimilarity update at each iteration, i.e. update $d(i,k) = \min(d(i,k), d(j,k))$ after combining cluster i and j (and using i as the cluster id of the resulting cluster). If the pairs with do-not-link constraints are initialized with $+\infty$ dissimilarity, the aforementioned update rule can be modified to incorporate the constraints without affecting the time and space complexity:

$$d(i,k) = \begin{cases} \min(d(i,k), d(j,k)) & d(i,k) \neq +\infty \\ & \text{AND } d(j,k) \neq +\infty \\ +\infty & \text{otherwise} \end{cases}$$

3.3 Experiments

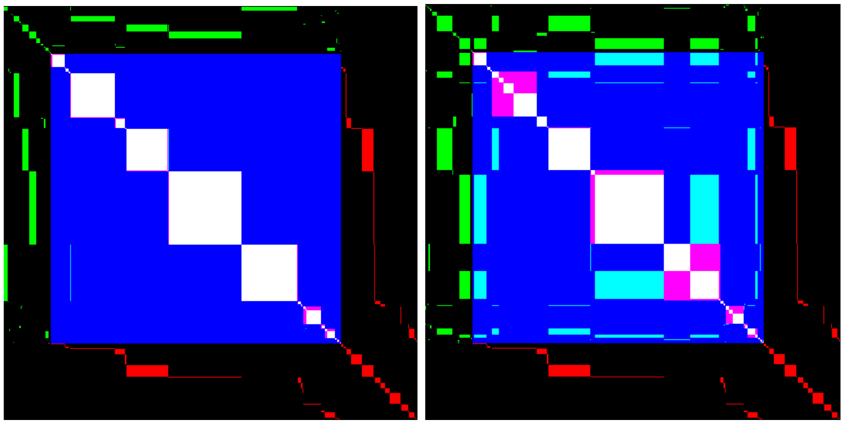
We evaluate our proposed approach on three video data sets: *the Big Bang Theory* (BBT) Season 1 (s01), Episodes 1-6 (e01-e06) [10], *Buffy the Vampire Slayer* (Buffy) Season 5 (s05), Episodes 1-6 (e01-e06) [10], and *Hannah and Her Sisters* (Hannah) [85]. Each episode of the BBT and Buffy data set contains 5-8 and 11-17 characters respectively, while Hannah has annotations for 235 characters.[2] Buffy and Hannah have many occlusions which make the face clustering problem more challenging. In addition to the video data sets, we also evaluate our clustering algorithm on LFW [51] which contains 5730 subjects.[3]

An end-to-end evaluation metric. There are many evaluation metrics used to independently evaluate detection, tracking, and clustering. Previously, it has been difficult to evaluate the relative performance of two end-to-end systems because of the complex trade-offs between detection, tracking, and clustering performance. Some researchers have attempted to overcome this problem by providing a reference set of detections with suggested metrics [79], but this approach precludes optimizing complete system performance. To support evaluation of the full video-to-identity pipeline, in which false positives, false negatives, and clustering errors are handled in a common framework, we introduce *unified pairwise precision* (UPP) and *unified pairwise recall* (UPR) as follows.

Given a set of annotations, $\{a_1, a_2, ..., a_A\}$ and detections, $\{d_1, d_2, ..., d_D\}$, we consider the union of three sets of tuples: false positives resulting from unannotated face detections $\{d_i, \emptyset\}$; valid face detections $\{d_i, a_j\}$; and false negatives resulting from unmatched anno-

[2] We removed garbage classes such as 'unknown' or 'false_positive'.

[3] All known ground truth errors are removed.



(a) Rank-1 Count (b) Rank-Order Distance [158]

Figure 3.4: Visualization of the combined detection and clustering metric for the first few minutes of the Hannah set.

tations $\{\emptyset, a_j\}$. Fig. 3.4 visualizes every possible pair of tuples ordered by false positives, valid detections, and false negatives for the first few minutes of the Hannah data set. Further, groups of tuples have been ordered by identity to show blocks of identity to aid our understanding of the visualization, although the order is inconsequential for the numerical analysis.

In Fig 3.4, the large blue region (and the regions it contains) represents all pairs of annotated detections, where we have valid detections corresponding to their best annotation. In this region, white pairs are correctly clustered, magenta pairs are the same individual but not clustered, cyan pairs are clustered but not the same individual, and blue pairs are not clustered pairs from different individuals. The upper left portion of the matrix represents false positives with no corresponding annotation. The green pairs in this region correspond to any false positive matching with any valid detection. The lower right portion of the matrix corresponds to the false negatives. The red pairs in this region correspond to any missed clustered pairs resulting from these missed detections. The ideal result would contain blue and white pairs, with no green, red, cyan, or magenta.

The unified pairwise precision (UPP) is the fraction of pairs, $\{d_i, a_j\}$ within all clusters with matching identities, *i.e.*, the number of white pairs divided by the number of white, cyan, and green pairs. UPP decreases if: two matched detections in a cluster do not correspond to the same individual; if a matched detection is clustered with a false positive; for each false positive regardless of its clustering; and for false positives clustered with valid detections. Similarly, the unified pairwise recall (UPR) is the fraction of pairs within all identities that have been properly clustered, *i.e.*, the number of white pairs divided by number of white, magenta, and red pairs. UPR decreases if: two matched detections of the same identity are not clustered; a matched detection should be matched but there is no corresponding detection; for each false negative; and for false negative pairs that should be detected and clustered. The only way to achieve perfect UPP and UPR is to detect every face with no false positives and cluster all faces correctly. At a glance, our visualization in Fig. 3.4 shows that our detection produces few false negatives, many more false positives, and is less aggressive in clustering. Using this unified metric, others can tune their own detection, tracking, and clustering algorithms to optimize the unified performance metrics. Note that for image matching without any detection failures, the UPP and UPR reduce to standard pairwise precision and pairwise recall.

The UPP and UPR can be summarized with a single F-measure (the weighted harmonic mean) providing a single, unified performance measure for the entire process. It can be α-weighted to alter the relative value of precision and recall performance:

$$F_\alpha = \frac{1}{\frac{\alpha}{UPP} + \frac{1-\alpha}{UPR}} \tag{3.2}$$

where $\alpha \in [0, 1]$. $\alpha = 0.5$ denotes a balanced F-measure.

3.3.1 Threshold for Rank-1 Counts

The leftmost column in Table 3.2 shows our clustering results when the threshold is set automatically by the validation set. We used LFW as a validation set for BBT, Buffy and

Table 3.2: Clustering performance comparisons on various data sets. The leftmost shows our **rank1count** by setting a threshold automatically. For the rest of the columns, we show f-scores using optimal (oracle-supplied) thresholds. (**1st place**,**2nd place**,**3rd place**).

			Verification system + Link-based clustering algorithm					Other clustering algorithms					
Test set			**Rank-1 Count (automatic threshold)**	**Rank-1 Count**	L2	Template Adaptation [21]	Rank-Order Distance [158]	Rank-Order Distance based Clustering [158]	Affinity Propagation [36]	DBSCAN [31]	Spectral Clustering [106]	Birch [154]	MiniBatch KMeans [103]
Video	BBT s01 [10]	e01	.7145	.7225	**.7386**	.7170	**.8064**	**.7278**	.1707	.4137	.6884	.3776	.2166
		e02	.7414	**.7671**	**.7561**	**.7520**	.7154	.6537	.1593	.3216	.6147	.2337	.2018
		e03	**.8428**	**.8552**	**.8329**	.8192	.6660	.6367	.2130	.2985	.6578	.2366	.2131
		e04	**.7602**	**.7690**	.7151	**.7687**	.6364	.7001	.2118	.2886	.6520	.2156	.1847
		e05	**.8217**	**.8250**	.7420	**.7858**	.6330	.7035	.2335	.2444	.5980	.1812	.2120
		e06	**.7563**	**.7578**	.6342	**.7247**	.5577	.5588	.1615	.1948	.5806	.1511	.1387
		Average	**.7728**	**.7828**	.7365	**.7612**	.6692	.6634	.1916	.2936	.6319	.2326	.1945
	Buffy s05 [10]	e01	**.6634**	**.6938**	.4950	**.6902**	.3819	.5935	.1711	.1755	.5762	.1439	.1285
		e02	.5582	**.6645**	.3315	.5452	.2800	**.5837**	.1705	.1185	**.5892**	.1151	.1087
		e03	**.5378**	**.5479**	.3735	**.5569**	.2390	.4595	.1346	.1322	.4566	.1077	.1063
		e04	.4203	**.4859**	.3523	.4549	.3049	**.5171**	.1643	.1445	**.5273**	.1187	.1179
		e05	**.6235**	**.6952**	.5064	**.6739**	.3073	.5640	.1435	.1740	.5540	.1390	.1251
		e06	**.5932**	**.6923**	.3001	**.5856**	.2807	.5455	.1765	.1009	.5071	.1041	.0995
		Average	**.5661**	**.6299**	.3931	**.5845**	.2990	.5439	.1601	.1409	.5351	.1214	.1143
	Hannah [85]		**.6436**	**.6813**	.2581	.3620	**.4123**	.3955	.1886	.1230	.3344	.1240	.1052
Image	LFW [51]		**.8532**	**.8943**	**.8498**	.3735	.5989	.5812	.3197	.0117	.2538	.4520	.3133

Hannah while Hannah was used for LFW. Note that the proposed method is very competitive even when the threshold is automatically set.

3.3.2 Comparisons

In this work, we have introduced a new similarity measure, rank-1 counts, which is applied to a link-based clustering algorithm. We can divide other clustering algorithms into two broad categories–(i) link-based clustering algorithms (like ours) that use a different similarity/distance measure and (ii) clustering algorithms that are not link-based (such as spectral clustering [106]).

The first part of Table 3.2 shows the comparisons to various similarity/distance functions [21, 84, 158] with the link-based clustering algorithm. L2 shows competitive performance in LFW while the performance drops dramatically when a test set has large pose variations. We also compare against a recent so-called “template adaptation” method [21] which also requires a reference set. It takes 2nd and 3rd place on Buffy and BBT. In addition, we compare to the Rank-Order distance [158], which is motivated by the observation that top neighbors of the faces of the same identity are usually shared[4].

Further, we also compare against several generic clustering algorithms (Affinity Propagation [36], DBSCAN [31], Spectral Clustering [106], Birch [154], KMeans [103]), where L2 distance is used as pairwise metric. For algorithms that can take as input the similarity matrix (Affinity Propagation, DBSCAN, Spectral Clustering), do-not-link constraints are applied by setting the distance between the corresponding pairs to ∞. Note that this is just an approximation, and in general does not guarantee the constraints in the final clustering result (*e.g.* for single-linkage agglomerative clustering, a modified update rule is also needed in Section 3.2.5).

Note that all other settings (feature encoding, tracklet generation) are common for all methods. In Table 3.2, except for the leftmost column, we report the best $F_{0.5}$ scores using optimal (oracle-supplied) thresholds for (number of clusters, distance). The link-based clustering algorithm with our rank-1 counts outperforms the state-of-the-art on all four data sets in $F_{0.5}$ score.

One reason that our rank-1 count outperforms is that the proposed similarity considers “very similar” features only. When two face embeddings are compared, some features are not activated as they are not relevant to the current faces (*e.g.* profile face does not show one part of a face). Those inactivate features are likely to be very similar to each other, even if they are not very similar. Our rank-1 count similarity measure uses a reference set to detect active features, and it computes if the feature values from two faces are actually very similar or not. Meanwhile, other approaches, such as L2 and Rank-Order Distance, take consideration of all features.

[4] In rank-order method, since the top-N closest neighbors are considered, using a large collection of reference faces (as in our method) will not enhance clustering performance.

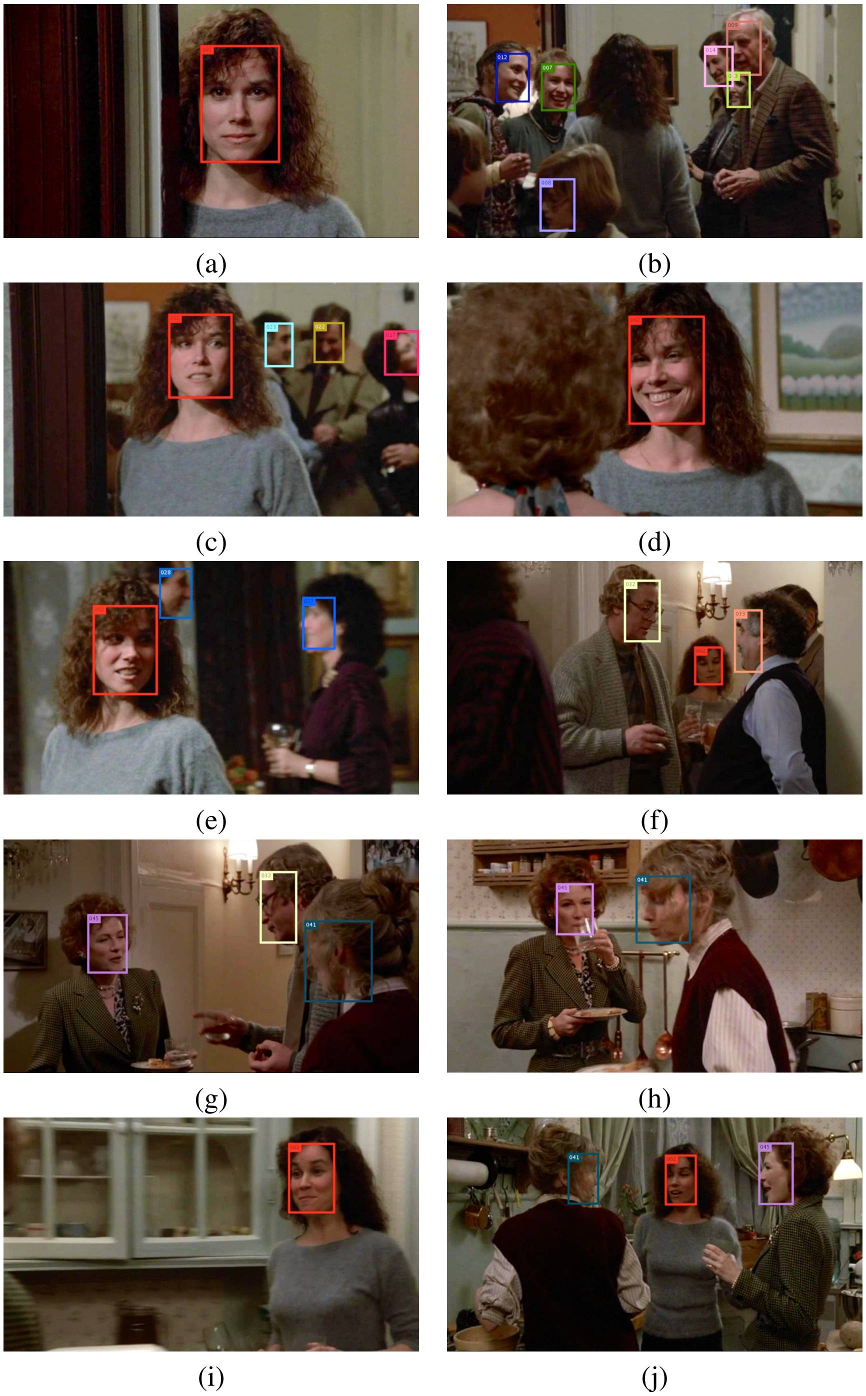
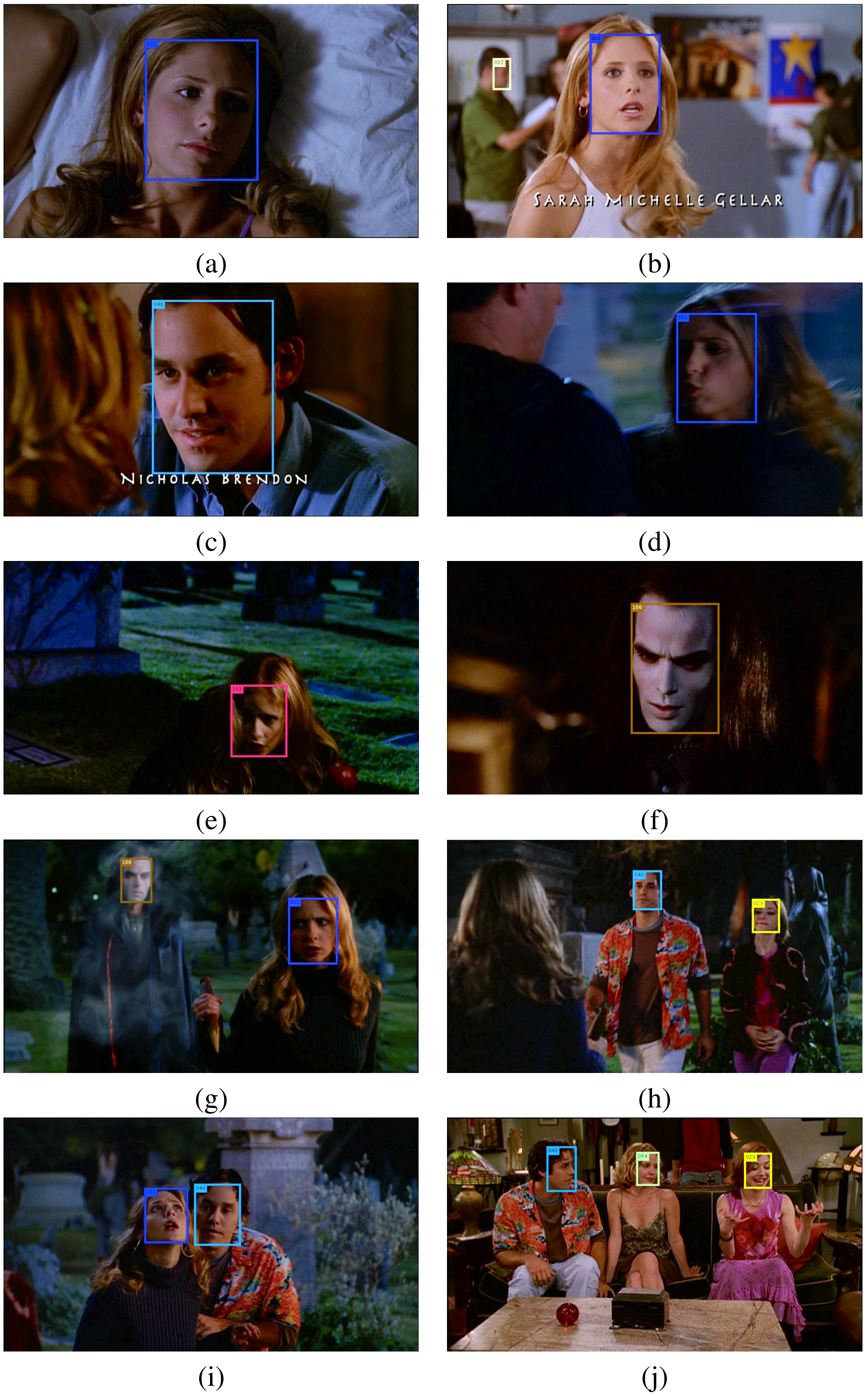

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 3.5: **Clustering results from *Hannah and Her Sisters*.** Each unique color shows a particular cluster. It can be seen that most individuals appear with a consistent color, indicating successful clustering.

Figures 3.5, 3.6 and 3.7 show some clustering results on Hannah, Buffy and BBT.



(a) (b)

(c) (d)

(e) (f)

(g) (h)

(i) (j)

Figure 3.6: **Clustering results from *Buffy the Vampire Slayer*.** A failure example can be seen in frame (e), in which the main character Buffy (otherwise in a purple box) in shown in a pink box.

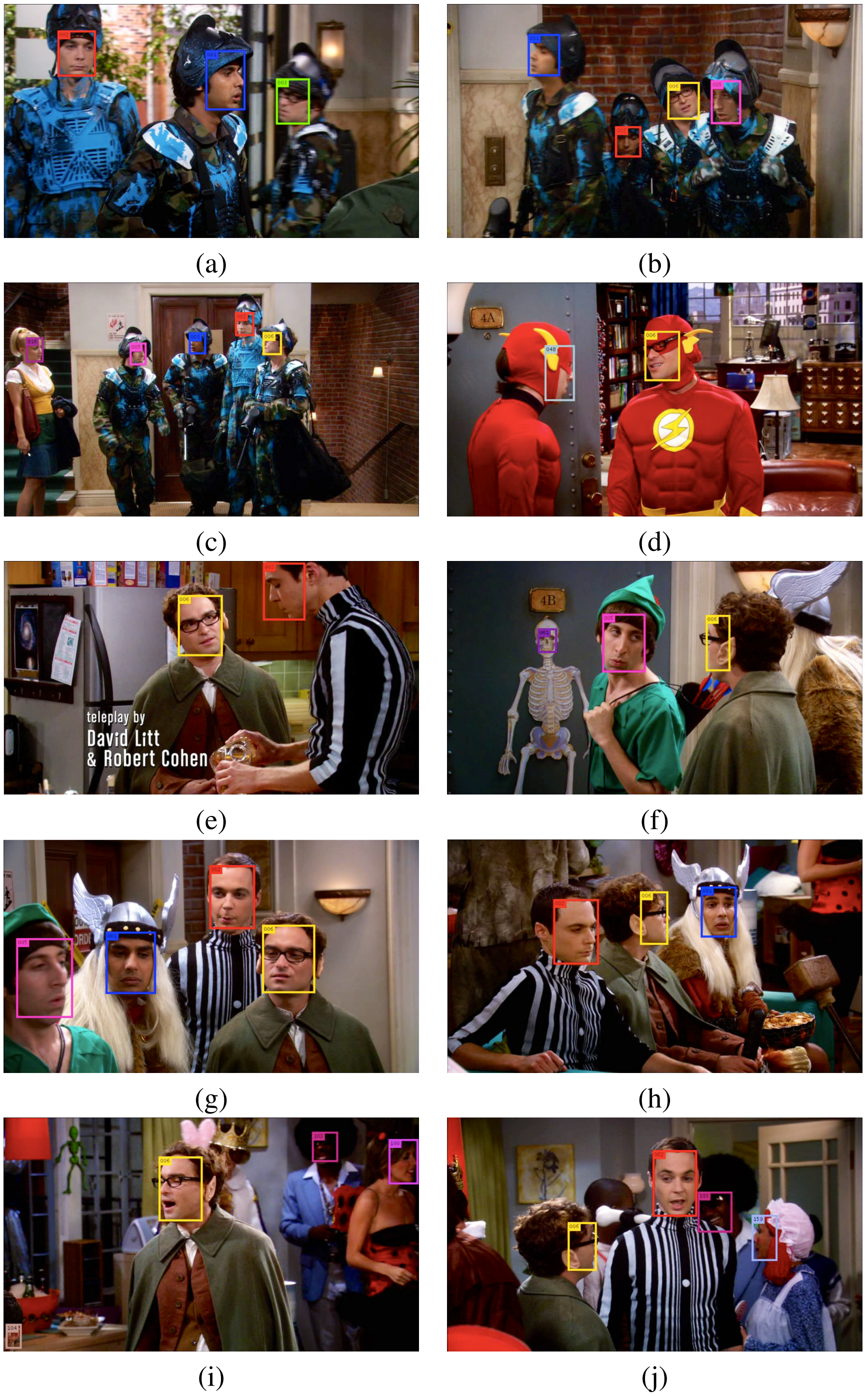


Figure 3.7: **Clustering results from *the Big Bang Theory*.** A failure example can be seen in frame (d), in which the main character Howard (otherwise in a magenta box) in shown in a gray box.

3.4 Discussion

We have presented a system for doing end-to-end clustering in full length videos and movies. In addition to a careful combination of detection and tracking, and a new end-to-end evaluation metric, we have introduced a novel approach to link-based clustering that we call Erdős-Rényi clustering. We demonstrated a method for automatically estimating a good decision threshold for a verification method based on rank-1 counts by estimating the underlying portion of the rank-1 counts distribution due to mismatched pairs.

This decision threshold was shown to result in good recall at a low false-positive operating point. Such operating points are critical for large clustering problems, since the vast majority of pairs are from different clusters, and false positive links that incorrectly join clusters can have a large negative effect on clustering performance.

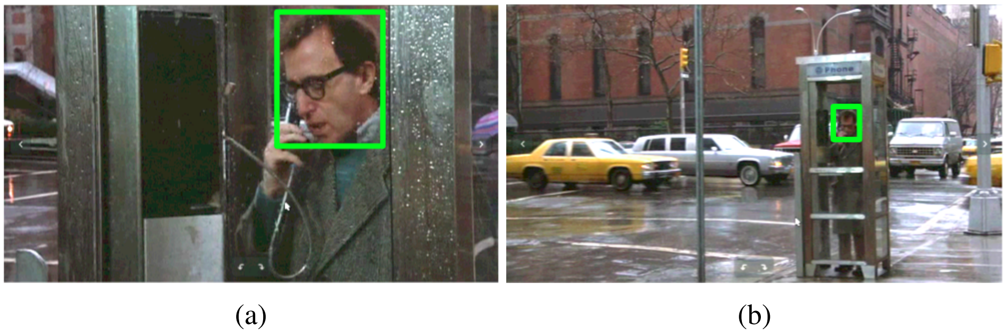
There are several things that could disrupt our algorithm: a) if a high percentage of different pairs are highly similar (e.g. family members), b) if only a small percentage of pairs are different (e.g., one cluster contains 90% of the images), and if same pairs lack lots of matching features (e.g., every cluster is a pair of images of the same person under extremely different conditions). Nevertheless, we showed excellent results on 3 popular video data sets. Not only do we dominate other methods when thresholds are optimized for clustering, but we outperform other methods even when our thresholds are picked automatically.

CHAPTER 4

CONTEXT-BASED VIDEO FACE CLUSTERING VIA FACE-BACKGROUND NETWORK (FB-NET)

A movie scene is often composed of multiple shots where each shot is taken by a different camera to capture different scene perspectives (e.g. different fields of view, camera placements and angles) [137]. These different viewpoints guide human audiences and give them a more vivid understanding of each scene. However, such camera shot switching can easily cause failures in face verification or clustering. This is not only because there is a large change in appearance of faces across different camera shots, but also because some target faces may not be clearly visible (e.g. too small faces, occlusion, or extreme poses).

How do humans track identities across multiple shots of the same scene? We observe that when faces are not clearly visible, humans often use contextual information beyond the target face by looking at surrounding areas, such as specific objects or textures on or around the person to gather additional identity cues. Fig. 4.1 shows two frames from the same scene but from different shots. Unlike (a) where the target face is easy to see, the face of the man in (b) is too small to be clearly visible. By looking at surrounding areas, e.g., the *wet metallic photo booth*, humans conclude that these two frames represent the same scene and hence that the two marked faces are likely to be the same person. In addition, the “door study” from Simons and Levin [111] also gives good evidence that humans use contexts rather than just relying on faces. In the experiment, a participant is asked for providing directions by an experimenter, while the experimenter is replaced by someone else to purposely mislead the participant. The experiment shows around half of the people did not notice the replacement, which indicates that humans use contexts as well as faces to identify the person.



(a) (b)

Figure 4.1: Unlike (a) which clearly shows who the movie character is, it is hard to recognize the person in (b). Still, humans can seek for meaningful information from the entire scenes, such as *wet metallic photo booth*, in order to verify whether the two green marked people are actually the same person or not.

To capture these intuitions, we propose the **Face-Background Network (FB-Net)** for face recognition, which takes as input both the target face and the entire corresponding video frame. To train this system, we introduce a new dataset: the **Scene-based Face Dataset (SFD)**. This dataset is collected from publicly available on-line movie clips which contain a single scene but often with multiple shots. That is, each clip contains a variety of camera angles or viewpoints within the same physical scene, and typically with a consistent set of characters.

Thus, the dataset contains strong correlations between scenes and persons but remains challenging due to the aforementioned difficulties of face recognition across different camera shots. In addition, the dataset contains many examples that are from the same scene but with different people. This diversity of people within the same scene keeps the network from overfitting to 'scene similarity', i.e., to conclude that a similar scene always implies the same people. Thus, to perform well on this dataset, it is necessary for the network to learn both face and background features.

In total, we have collected 317 movie clips of 78 movies for 55 actors. With the dataset, our network is trained to seek additional clues from the entire input frame that can provide supporting details to represent the target face[1] in the frame.

FB-Net contains two branches (see Fig. 4.2). The first branch focuses on learning face features by using only face regions as input, and the second branch focuses on learning background features for scene understanding. In our background feature extractor, we incorporate a transformer network to extract useful features from a scene. By training the FB-Net with the **Scene-based Face Dataset (SFD)**, we show that our network performs better than using face features alone for classification. This implies that our network is effective in learning both face and background features together, and they both contribute to the accuracy of our network. In addition, by investigating the attention map learnt from the transformer module, we found that our learned features localize distinctive static objects in the background. This is different from conventional saliency detection since the latter would focus on the most distinctive regions, i.e. faces, while ours do not. In summary, our paper contains the following contributions:

1. We introduce the FB-Net that takes as input not just faces but also the entire scene to enhance face clustering.
2. The FB-Net is trained with a transformer module in a novel way to learn useful scene level features with face/background processors. The learned embeddings are evaluated on the test videos in which the network has never seen the actors before. FB-Net outperforms the state-of-the-art method which exploits face-level features only.

[1] Distinguishing doppelgangers or identical twins is out of our scope.

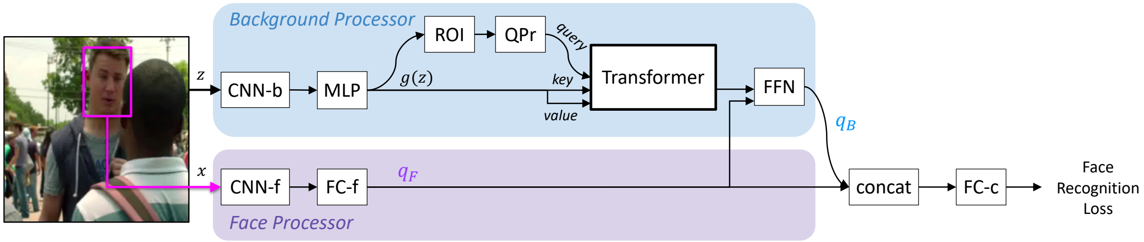


Figure 4.2: The FB-Net takes as input a video frame and the coordinates of a target face (pink box). The input face is processed by the **Face Processor** to obtain face identity features. In the **Background Processor**, our model obtains additional cues from the entire frame to improve the classification accuracy of the target person. We use the transformer network to learn distinctive background features from the areas outside of the face. The outputs from the Background and Face Processors are concatenated and used to compute a total face embedding.

3. To make the network to learn background features that are informative about identity, we provide a new dataset that allows the network to learn consistent scene elements from different points of view.

4.1 Face-Background Network (FB-Net)

The ultimate goal of this work is to cluster faces in *novel* movies and videos. To do this, we use the following standard sequence of steps:

1. Train a face *classifier* using labeled faces, using a standard classification loss (cross entropy).
2. Use a pre-classification layer of features from this learned classifier as an *embedding* for each face.
3. For a new movie, do a forward pass to compute the embedding for each face.
4. Using the full set of embeddings, use an off-the-shelf clustering algorithm to cluster the faces in the new movie.

Note that we do not expect to see any of the same people or characters at test time that we have seen at training time. That is, the set of test identities does not overlap with the set of training identities. Rather, we have learned to embed faces in a feature space so that we can assess their similarity even if we have never seen those faces before.

The key contribution of this work is to have the background itself have a major influence on the embedding. This is challenging since we also have not seen these particular backgrounds before at test time. Thus, the goal of the network is to learn *what types of features in the background* are likely to be useful to help establish identity in the context of movies. In particular, this type of information will be particularly important in cases where the face itself is not a good source of information because it is either too small, partially occluded, etc. (see Fig. 4.1).

In this section, we introduce the Face-Background Network (Chapter 4.1.1) that takes as input the entire image z as well as a face bounding box x as in Fig. 4.2. FB-Net is composed of both a face processor and a background processor, one for each different type of input.

To train this network, we need a dataset that contains entire video frames as well as target face coordinates, where each of the target faces is labeled by *face identity* and *the bounding box coordinates of the target face*. In Chapter 4.1.4, we will talk about how the dataset is constructed.

4.1.1 Face/Background Processors

FB-Net is composed of two modules: the Face Processor and the Background Processor. In the **Face Processor**, an input face x is passed into a CNN and a fully-connected (FC) layer, where the output of this processor, an embedding q_F, is expected to encode the face well enough to classify x into the correct face category.

In the **Background Processor**, the goal is to seek additional cues from the background image z for the classification of x. We specifically want our network to have a look at something specific or unique with respect to the target person instead of paying attention to what is universal to any person. For example, detecting a bottle might not provide much information about a person as bottles are quite common, and it can appear with any person. On the other hand, if a person holds *a green ceramic cup with special flower patterns*, humans can easily guess the identity of the person holding the cup even if the person's face is not clearly visible. In order to capture this attention, we use the Transformer Network [128]. In Chapter 4.1.2, we give an overview of the Transformer Network, which is followed by Chapter 4.1.3 to describe how we apply the Transformer in our FB-Net.

4.1.2 Overview of the Transformer Network

For machine translation (seq2seq), Ashish *et al*. has introduced the Transformer Network [128], which includes multiple self-attention layers. Given a sentence, a self-attention layer encodes each word in a sentence with the contextual information from all words in the sentence. For example, in a sentence, *"The Law will never be perfect, but its application should be just."*, knowing that *its* indicates *Law* can improve the translation results.

In particular, a self-attention layer takes as input query/key/value vectors. In the previous example, each word is considered as a **query**, and all other words in the sentence are used as **keys** or **values**. A query vector is first compared to each of the key vectors to figure out which values need more attention. The query vector is finally encoded with the corresponding value vectors.

More formally, a self-attention layer computes the dot products of the query vector (Q) with each of the key vectors (K), which is followed by a softmax function to get an attention map to re-scale weights in value vectors (V) as

$$\text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V. \tag{4.1}$$

Finally, the attention-weighted value vectors are added to the query vector.

In [40], Girdhar *et al*. has applied the Transformer Network to the video action classification task to represent the target person (query) better by adding context from other people and objects in the nearby video frames (key, value) for better action classification.

4.1.3 Transformer in FB-Net

In the background processor of our FB-Net, we want to find additional clues from the entire video frame z (key,value) that will provide supporting evidence to classify the identity of the target face x (query). Thus, attention layers in the transformer learn where to look at in addition to the target face region. A key difference in our work is that we are using a Transformer to learn an *embedding* rather than to directly classify faces. Thus, we are the first to use Transformers in the context of clustering new entities.

In particular, the background processor forwards z into a CNN and multi-layer perceptron (MLP) to get a H'xW'xD spatial convolutional features, $g(z)$, which are used for *key/value* vectors. From $g(z)$, a query vector is computed by first ROI pooling the region of the target face. The pooled region is then passed into the query processor so that we can obtain a D-dim *query* vector. The transformer module strengthens the query vector by adding the supporting details from $g(z)$. We adopt the architecture of the 2-head 3-layer Action Transformer [40] with some minor modifications. More formally, each *self-attention* head takes as input query/key/value vectors and maps them to Q, K, V using linear projections, where each of the projected vectors are $\frac{D}{2}$-dim. Then, a scaled dot-product attention, a, is computed by comparing Q feature to K features and background features are then updated by weight-averaging V with a as

$$a_{xy} = \frac{QK_{xy}^T}{\sqrt{D}}; A = \sum_{x,y} \left[\text{Softmax}(a)\right]_{xy} V_{xy}. \tag{4.2}$$

The head finally outputs Q'' by adding A to Q with LayerNorm [7] and Dropout [116] operations and a 2-layer FFN as

$$Q' = \text{LayerNorm}(Q + \text{Dropout}(A)) \tag{4.3}$$

$$Q'' = \text{LayerNorm}(Q' + \text{Dropout}(\text{FFN}_a(Q'))). \tag{4.4}$$

The output of each head, Q'', is concatenated and passed to each of the head in the next layer as Q. Finally, the output of the transformer and q_F are forwarded to a FFN, which outputs a D-dim background feature, q_B. Finally, we concatenate q_B and q_F, and compute the D-dim feature. This feature is then passed to FC-c to get the face recognition output.

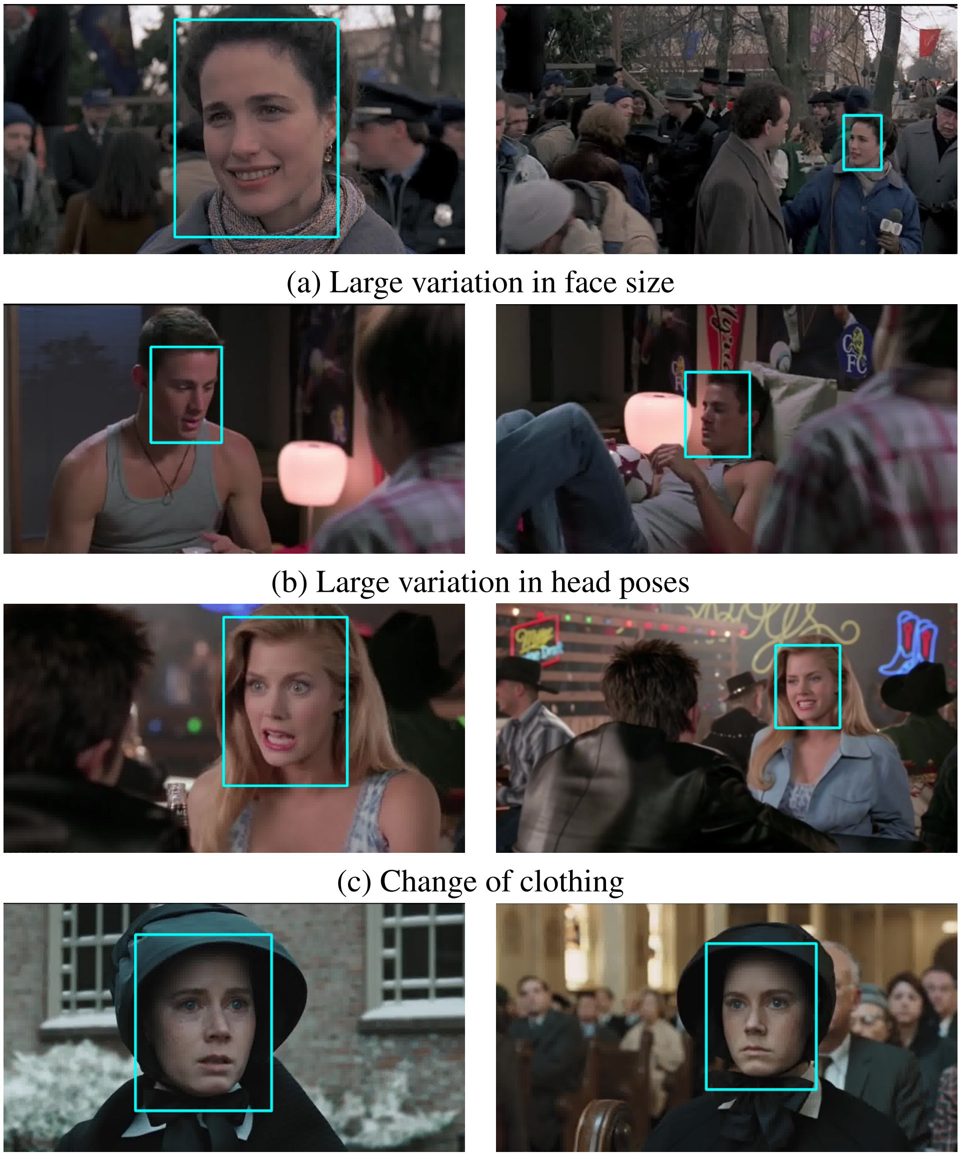
4.1.4 Scene-Based Face Dataset (SFD)

We want our model to be able to recognize a person even when the face region does not provide enough information by detecting supporting evidence from the background. Unlike face recognition in static images, a video shows multiple camera view of the same scene. When we look at the two frames in Fig. 4.3 (b), we may notice these frames share many common items such as the bright lamp and the gray undershirt, which can help to recognize that the two persons in the frames are actually the same person.

To help our network learn to detect important clues, we collect a new dataset called the Scene-Based Face Dataset (SFD), which contains video frames collected from online movie clips (e.g. in YouTube). With an observation that movie clips are often cut/edited for a particular scene, we densely collect video frames from each movie clip. Each of the video frames[2] is annotated with (i) the coordinates of a target face and the ground-truth identity of the actor.

Data Collection. We start with a face detection model (**D**) and a face recognition model (**F**) trained on faces of movie actors. We then run the face detector and the actor recognizer on each frame of video clips. Since an actor can have different hair styles or cosmetics on different movies, there is no guarantee that every face of the actor would be retrieved. However, since the face recognition model learns the unique features of an actor, the model could successfully recognize a few faces of the actor with high confidence even if his/her hair style changes a lot. Since face recognition results could still include false positives,

[2] A frame can contain more than one face, thus it can have multiple sets of annotations.



(a) Large variation in face size

(b) Large variation in head poses

(c) Change of clothing

(d) Same clothing styles across different scenes

Figure 4.3: Examples of our Scene-Based Face Dataset. (a-c) show the same actor in the same scene, but in two different shots. (d) shows the same actor in two different scenes.

we check if the recognized person actually performed in the movie by using a list of lead actor[3].

That is, for each m in *movie title*, we collect data as follows.

[3] As we collect movie clips, we can easily obtain a list of lead actors for each movie.

1. Get a list (l) of lead actors

2. Search publicly available videos with the keyword patterns such as

$$< m > + \text{“movie clips”}.$$

3. For each frame in each video clip, c,

    (a) Run $\mathbf{D}$ and $\mathbf{F}$ on the detections.

    (b) If a face is classified as a with confidence score higher than $\theta = 0.9$ and a is in the list l, we label the face with a.

Specifically, we use the MTCNN face detection model [149] for $\mathbf{D}$ and a ResNet-50 model that is trained on VGGFace2 dataset [16] for $\mathbf{F}$.

Although we assume that human-edited movie clips are scene-level distinction, a movie clip could be composed of multiple short scenes. Since the objective of constructing this dataset is to provide many video frames from various camera views/angles of a scene, having a short scene that may contain one or two shots is not actually helpful to learn scene-based face understanding. Thus, we manually checked if each of the collected clips are from a scene. In particular, we check if a movie clip contains an event that is happening in the same *location* (e.g. in a room) and in chronological order (*time*).

Fig. 4.3 shows some examples of video frames in SFD. The dataset contains large variation in the size of faces and head poses. In addition, since we collect several movie clips from a movie, it is not guaranteed that an actor always wear the same clothes. For example, in 4.3 (c), an actor can take off her jacket in the same scene as well[4]. Furthermore, sometimes an actor wear the same outfits in the entire movie as in 4.3.

[4] This violates one of the big assumptions in person re-identification work that a person always appear with the same clothes.

Dataset Statistics. In total, we collect 55 actors from 317 movie clips of 78 movies. Each of the valid video clips is split into 30-second chunks, and we use the last chunk for validation. The rest of video frames are used for training set. For training, we randomly choose at maximum 20 images from every 30-sec. For validation, we randomly choose 10 images from each chunk. We eventually obtain 59726/2979 images for train/val set respectively.

4.1.5 Implementation Details

Pre-Processing. FB-net requires of two inputs: a face image, x and the entire image z that includes the face. For CNN-f and CNN-b in Face/Background processors, we both use the network architecture of ResNet50 [48], which takes a 224x224x3 image. For x, we apply the same pre-processing tricks as in typical face classification models. That is, x is first resized into 256x256, which is followed by a random crop of 224x224.

For z, we have to think more carefully since we want a random crop always include the target face. We first resize z such that the smallest side to be 256. Suppose the resized width and height to be w and h. We also resize the coordinate (x_{min}, y_{min}, x_{max}, x_{max})of the target face accordingly. Now, we want to randomly sample a 224x224 crop, i.e. (zx_{min}, zy_{min}), (zx_{min} + 224, zx_{min} + 224), such that the crop could include the target face while still within the range of the given frame. More formally, we want to randomly sample zx_{min} and zy_{min} from $[s_1, s_2]$ and $[t_1, t_2]$ respectively where s_1, s_2, t_1, t_2 are defined as

$$\begin{aligned} s_1 &= \max(0, x_{\text{max}} - 224); s_2 = \min(x_{\text{max}}, w - 224) \\ t_1 &= \max(0, y_{\text{max}} - 224); t_2 = \min(y_{\text{max}}, h - 224). \end{aligned} \tag{4.5}$$

We also horizontally flip training images randomly. This random horizontal flip is applied consistently to x and z. During training, we use center crop for both x and z. If the center crop does not include the target face, we simply shift the crop coordinate to include it.

Face Processor. For CNN-f, we adopt the entire ResNet-50 [48] architecture except the classification layer. On top of the CNN-f, we add an FC layer a feed-forward network (FFN), which is followed by a classification layer.

Background Processor. The architecutre of CNN-b is also adopted from the first two layers of ResNet-50. As in [128, 40], we add spatial location information to the end of he outputs of the second layers (512x25x25). Then, the 514x25x25 convolutional features are forwarded into an MLP which is composed of two 3x3 Conv layers, a ReLU, a Dropout, and a LayerNorm. From the output of MLP (256x25x25), the coordinates of the target face are ROI-pooled [94], which is followed by the query processor. We use HighRes query processor described in [40]. The output of this query processor is used as query for the transformer.

For FFN and concat, we modify the architecture of FFN in the original Transformer paper [128] while ours is composed of two FCs, a ReLU, a Dropout, and LayerNorm. The input to the modules are a 512-dim feature vector (after concatenating two 256-dim vectors), and the output is 256-dim vector. For FFN-a in self-attention layer, we modify so that the network takes a 128-dim vector.

4.1.6 Training Details

Pre-Training. We use pre-trained models of ResNet-50 [48] for CNN-b and CNN-f, and do not update parameters in two models during training. For CNN-f, We use a model which is pre-trained on MS-Celeb-1M [44] and fine-tuned on VGGFace2 [16] to recognize face images of 8631 people. Unlike CNN-f that is supposed to understand human faces, we hope for CNN-b to understand general objects. Thus, we use an ImageNet [100] pre-trained model.

SGD Parameters. We train the FB-Net with a fixed learning rate of 0.001 for 35k iterations using the SGD optimizer with 32 batch size, where the momentum is 0.9. We use dropout in self-attention layers with the 0.3 probability while 0.1 is used for the rest of network. We use our validation set to pick the best model.

4.2 Experiments

4.2.1 Baselines

To show the superiority of the proposed FB-Net, we compare our FB-Net to the following baselines.

- `Face(VGGFace2)`. We first use the ResNet-50 [48] model pre-trained on MS-Celeb-1M [44] and fine-tuned on VGGFace2 [16]. This model was trained to recognize face images of 8631 people. We only use face regions as inputs to the network.

- `Background(ImageNet)`. This baseline uses the ResNet-50 [48] model pre-trained on ImageNet [100] with the entire video frame as input.

- `Face+Background(entire)`.To show that simply using both face and background does not actually improve the performance, we provide another baseline without including the transformer architecture. Specifically, given both a target face and the corresponding *entire* video frame, we use `Face(VGGFace2)` and `Background(ImageNet)` to compute the embedding for the target face and the entire scene. We then concatenate the extracted features from `Face(VGGFace2)` and `Background(ImageNet)`.

  Further, we also provide two stronger baselines by cropping the entire frame around the target face so that the network can focus on the context around the target face. In particular, `Face+Background(2x)` takes as background input a twice larger region than the target face box. Similarly, `Face+Background(4x)` takes four times larger region as background input.

- `Face(SFD)`. To show that training on our dataset does not hurt the model performance, we prepare another baseline that is trained on SFD. This baseline could be also used to address the effectiveness of using background information. For the network architecture, we use the FB-Net without the entire background processor. The model is trained with the cross-entropy loss to predict the identities in SFD.

A related problem to *video face clustering*is *person re-identification* [89, 156, 24, 63, 70, 157] in which the goal is to tell whether a person of interest seen in one camera has been observed by another camera. Unlike *video face. clustering* which focuses on faces but over a longer period of time, the *person re-identification* methods typically use the whole body on short time scales. Thus, person re-identification methods cannot be directly applied to the video face clustering problem.

4.2.2 Video Face Clustering

To show how FB-Net can be generalized over the identities that the model has not be seen before, we evaluate our model on various video face clustering benchmarks: the Big Bang Theory (BBT) Season 1 (s01), Episodes 1-6 (e01-e06), Buffy the Vampire Slayer (Buffy) Season 5 (s05), Episodes 1-6 (e01-e06) [123, 125][5].

We first run the MTCNN face detector [149] on every frame in videos to get target faces. For evaluation, we use the target faces that have corresponding matches with the ground-truth detections (IoU>0.2). We compute a pairwise distance matrix between all valid faces (with corresponding frames) with L2 distance. Then, we apply the linked based clustering [102] on the distance matrix.

To evaluate clustering outputs, we use BCubed clustering evaluation metric [3], and show f-scores using optimal (oracle-supplied) thresholds. In Table 4.1, we compare our model to the four baselines for video face clustering. On average, our FB-Net trained on SFD outperforms `Face(VGGFace2)`, which exploits only face-level features, by 3.70% and 4.28% on BBT_s01 and Buffy_s05 respectively. We expected that using both face and

[5] Recently, Tapaswi *et al.* [125] has extended the BBT and Buffy dataset [123] by adding annotations for background characters. We evaluate our FB-Net on the updated datasets.

Table 4.1: We compare our FB-Net to various baselines. We show f-scores using optimal (oracle-supplied) thresholds. FB-Net outperforms Face(VGGFace2), which exploits face-level features only, 3.70% and 4.28% on the Big Bang Theory Season 1 and Buffy the Vampire Slayer Season 5 respectively. Further, we observe that simply concatenating face and background features does not guarantee to enhance face clustering performance.

		Face (VGGFace2)	Background (ImageNet)	Face + Background (entire)	Face + Background (4x)	Face + Background (2x)	Face (SFD)	**FB-Net (ours)**
train on SFD?							✓	✓
face?		✓		✓	✓	✓	✓	✓
background?			✓	✓	✓	✓		✓
BBT s01 [125]	e01	0.9016	0.4652	0.8650	0.9043	0.9043	0.9210	**0.9450**
	e02	0.8555	0.4738	0.8221	0.8441	0.8596	0.9395	**0.9405**
	e03	0.9063	0.4630	**0.9286**	0.9264	0.9278	0.9038	0.9202
	e04	0.9009	0.4356	0.9217	0.9243	0.9185	0.8628	**0.9262**
	e05	0.9036	0.4042	0.9013	**0.9316**	0.9152	0.8816	0.8325
	e06	0.7674	0.3950	0.7611	0.7716	0.7878	0.8486	**0.8930**
	Average	0.8726	0.4395	0.8666	0.8837	0.8855	0.8929	**0.9096**
Buffy s05 [125]	e01	0.5895	0.4549	0.5818	0.5709	0.5871	0.6523	**0.7263**
	e02	0.5580	0.3843	0.5549	0.5483	0.5465	0.5354	**0.6052**
	e03	0.5570	0.4164	0.5526	0.5513	0.5580	**0.5697**	0.5616
	e04	0.6271	0.4373	0.6166	0.6177	0.6195	0.6398	**0.6413**
	e05	0.7018	0.4652	0.7158	0.7231	0.7333	**0.7563**	0.6886
	e06	0.5656	0.4136	0.5546	0.5581	0.5612	0.5792	**0.6325**
	Average	0.5998	0.4286	0.5961	0.5949	0.6009	0.6221	**0.6426**

background features should improve over face-only features. However, we notice that in most of the episodes, `Face+Background` shows even worse performance than using the face-only features. This implies that simply adding the pre-trained face/background features does not guarantee to enhance the clustering performance. One possible reason that simply using background features reduces performance is because the background includes noisy information that might not be helpful to recognize the target person.

t-SNE Visualization with q_B. To demonstrate that the background processor in FB-Net actually captures meaningful features, we extract q_B, the output of the background processor, and compute two-dimensional t-SNE[6] [127] features. Fig. 4.4 and Fig. 4.5 show the visualization of these t-SNE features[7] on the fourth episode of the Big Bang Theory

[6] t-Distributed Stochastic Neighbor Embedding (t-SNE) is a way of visualizing high-dimensional datasets by reducing dimensionality reduction.

[7] We take every 50th frame.

Season 1 and the first episode of Buffy the Vampire Slayer Season 5. Note that q_B features are discriminative to distinguish different actors. In addition, the visualization shows that similar backgrounds tend to be grouped closely.

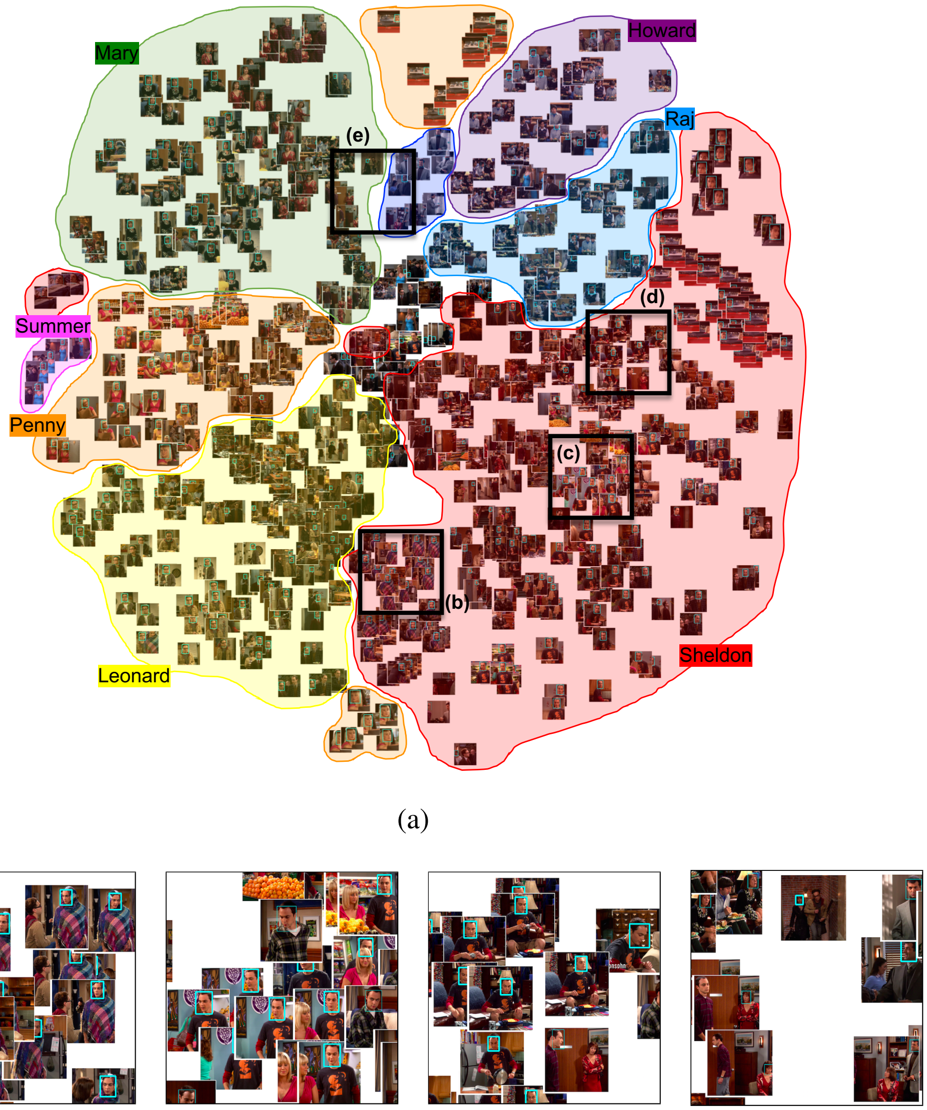
We want to highlight that within each of the *actor* clusters, frames are closer to each other when they share background features. For example in Fig. 4.4 (b) and (c), the giant cluster of *Sheldon* also shows sub-clusters depending on where he is located, while (b) and (c) are quite close within the *Sheldon* cluster as the actor is wearing the same purple T-shirt with orange patterns. We can also check this phenomenon across different actors. Fig. 4.4 (e) shows the boundary of *Mary* and *Gablehauser* clusters. At the bottom of (e), we can see the frames from the same scene are adjacent while these frames are also close to the corresponding identities.

4.2.3 Visualization of Attention Maps

In the background processor of FB-Net, we have a transformer module that is composed of 2-head 3-layer self-attention layers. To understand what the network learns, we visualize the two attentions from the last layer. To show in RGB space, we re-scale the attention values to be [0, 255].

In Fig. 4.6, we show attention maps on several examples of the held-out set of SFD dataset. Both Attention-A and Attention-B draw attentions to parts of background that are informative about identity. We also observe that Attention-A mostly looks for clues on the target person, while Attention-B is more likely to seek for other clues. For example in the first column of Fig. 4.6, Attention-A detects her *necklace* while Attention-B focuses on the *golf clubs* behind her. The FB-Net learns the attention without any explicit supervisions. It also captures the other people than the target person to understand scene better.

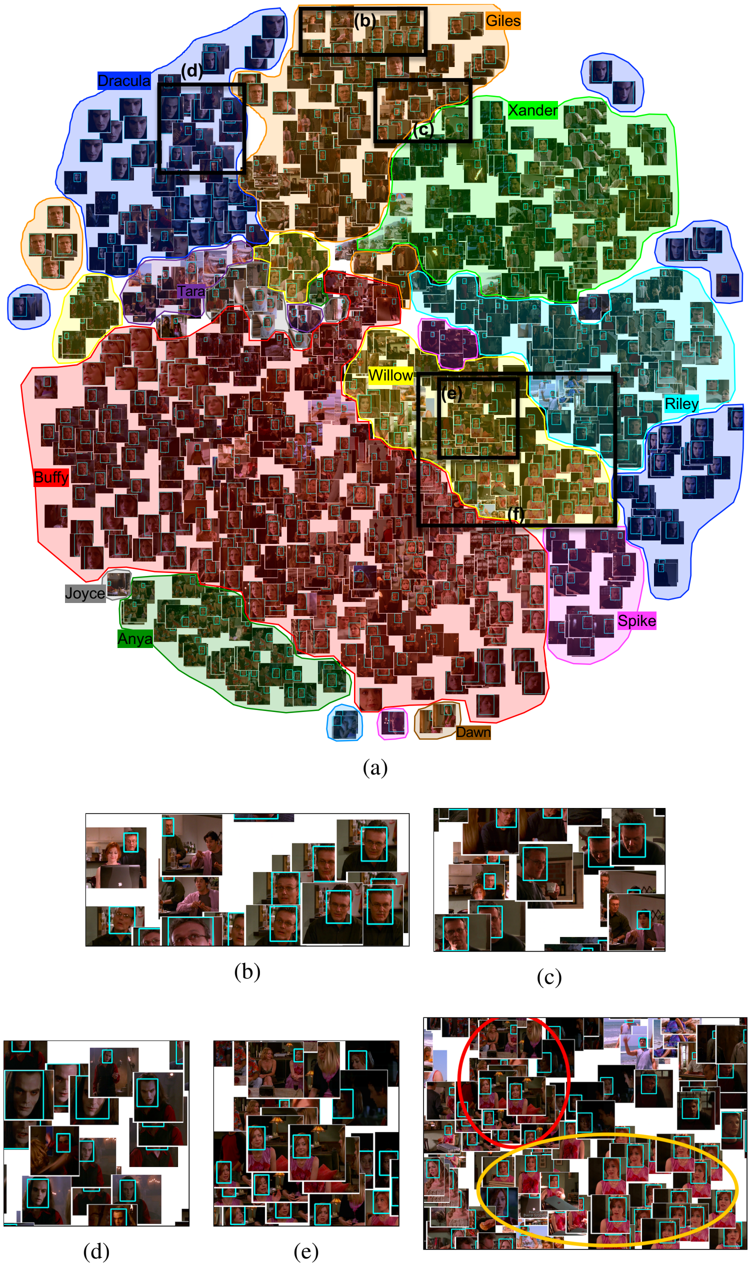
We also show the attention maps on the fourth episode of the Big Bang Theory Season 1. in Fig. 4.7. The transformer captures not just the clothing patterns of *Sheldon* (Attention-A), but also shows the phenomenon to track informative elements behind him.



(a)

(b) (c) (d) (e)

Figure 4.4: t-SNE [127] visualization of background features (q_B) on the fourth episode of the Big Bang Theory Season 1. The frames are already clustered well by actors, while it is observed as in (b-d) that each of the actor clusters forms sub-clusters for different scenes and/or outfits. (e) This scene-level grouping is also observed near the boundary of identity clusters .



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.5: (a) t-SNE [127] visualization of background features (q_B) on the first episode of Buffy the Vampire Slayer Season 5. (b-f) zoom in to the areas marked by the corresponding black rectangles. The figures (b) and (c) both visualize the frames in the kitchen scene. As shown in (c) we note that the features at the shared boundary between *Giles* and *Xander* are representing the kitchen scene. We also observe in (d) that when a face is occluded, the background feature is still able to recognize the same scene. In (f) we visualize that features q_B capture both clothes and backgrounds. The character *Willows* marked in red and orange ellipses are in pink clothes but different background. We zoom in the red circle in (e).

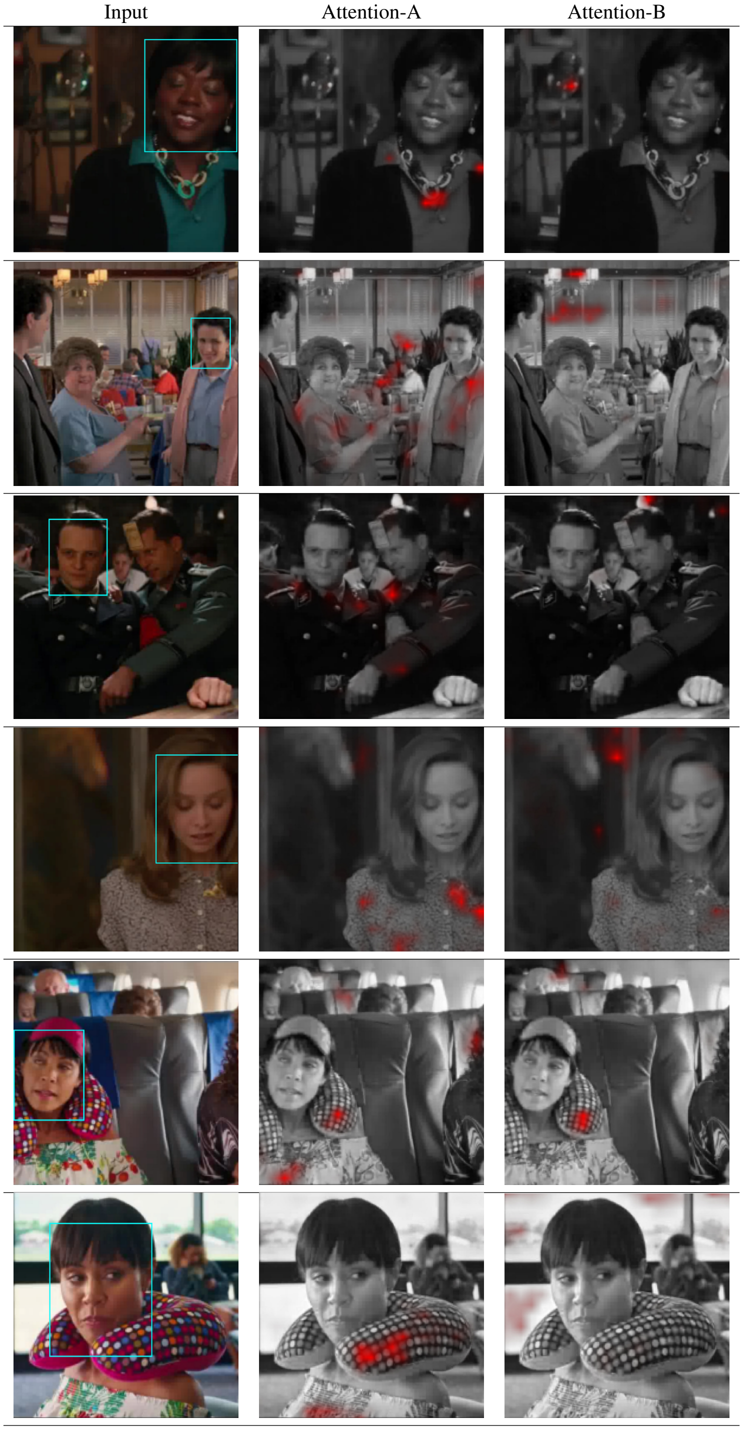


Figure 4.6: Visualization of the attention maps on SFD. The *first* row shows the original input images where the target face is marked with a box. The *second* and *third* rows show two attention maps from the last layer of the transformer module. We observe that Attention-A tries to look for clues on the target person, while Attention-B is more likely to check the actual scene information.

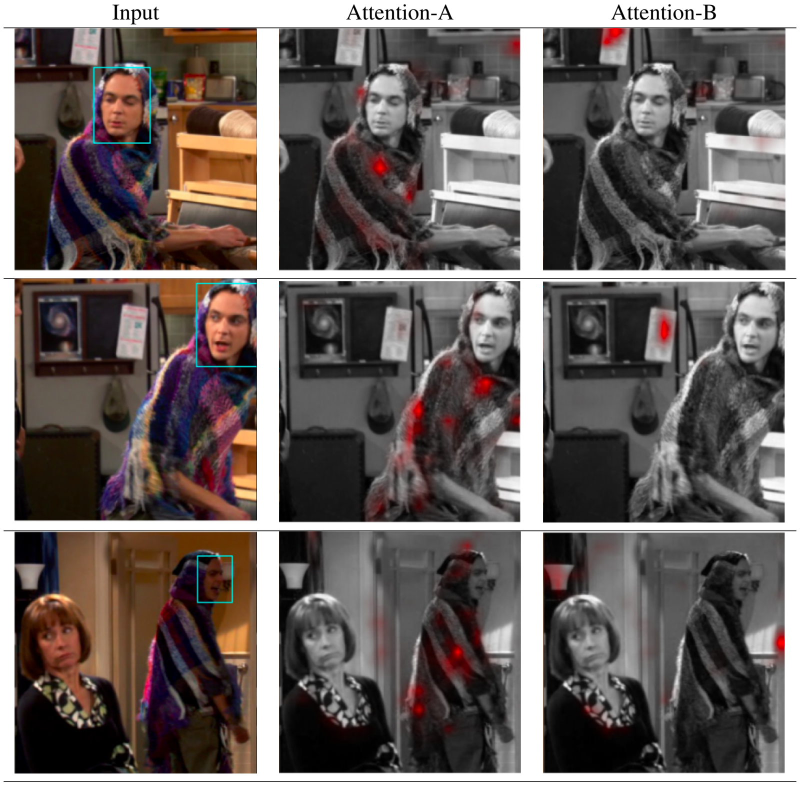


Figure 4.7: Visualization of the attention maps on the fourth episode of the Big Bang Theory Season 1.

4.3 Discussion

We have presented a novel framework to improve face clustering in videos by exploiting scene level information to address the challenges of face clustering across different camera shots. A new network architecture, FB-Net, is proposed for face recognition that leverages contextual information from the entire scene to learn robust identity embeddings. Furthermore, we introduced a new dataset that contains face identities in the context of consistent scenes. We conducted experiments on standard video face clustering benchmarks, and our experimental results demonstrate significant boost in performance by utilizing scene-contextualized face embeddings, and it shows improved performance over the state-of-the art that utilizes face-level features only. Through both qualitative and quantitative results, we observe that by explicitly learning how consistent scene elements are correlated across different camera shots, we can learn better identity representations especially when face regions are ambiguous.

CHAPTER 5

UNSUPERVISED HARD EXAMPLE MINING FROM VIDEOS FOR IMPROVED OBJECT DETECTION

Detection is a core computer vision problem that has seen major advances in the last few years due to larger training sets, improved architectures, end-to-end training, and improved loss functions [96, 95, 28, 159]. In this work, we consider another direction for improving detectors – by dramatically expanding the number of hard examples available to the learner. We apply the method to several different detection problems (including face and pedestrian), a variety of architectures, and multiple data sets, showing significant gains in a variety of settings.

Many discriminative methods are more influenced by challenging examples near the boundary of a classifier than easy examples that have low loss. Some classifiers, such as support vector machines, are completely determined by examples near the classifier boundary (the "support vectors") [101]. More recent techniques that emphasize examples near the boundary include general methods such as *active bias* [17], which re-weights examples according to the variance of their posteriors during training. In the context of class imbalance in training object detectors, on-line hard example mining (OHEM) [108] and the *focal loss* [69] were designed to emphasize hard examples.

In this paper, we introduce simple methods for automatically mining both hard negatives and hard positives from videos using a previously trained detector. To illustrate, Figure 5.1 shows a sequence of consecutive video frames from two videos containing a face and a pedestrian respectively. The results of the Faster R-CNN detector (trained for each class) run on each frame are marked as rectangles, with true positives as yellow boxes

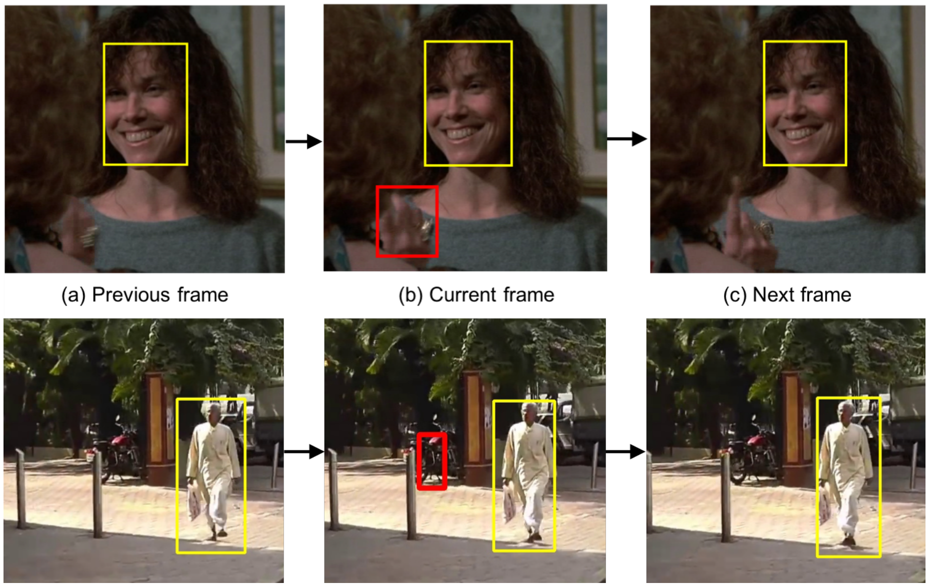


Figure 5.1: **Detector flicker in videos.** Three consecutive frames from a video are shown for face and pedestrian detection. On the top row, the boxes show face detections from the Faster R-CNN [96] (trained on WIDER face) [146, 54]. On the bottom row are detections from the same detector trained on the Caltech pedestrian dataset [27]. Yellow boxes show true positives and red boxes show false positives. For the true positives, the same object is detected in all three frames whereas for the false positives, the detection is *isolated* – it occurs neither in the previous nor the subsequent frame. These detections that are "isolated in time" frequently turn out to be false positives, and hence provide important sources of hard negative training data for detectors.

and false positives as red boxes. Notice that false positives are neither preceded nor followed by a detection. We refer to such isolated-in-time detections as **detector flickers** and postulate that these are usually caused by false positives rather than true positives.[1] This hypothesis stems from the idea that a false positive, caused by something that usually does not look like a face (or other target object), such as a hand, only momentarily causes a detector network to respond positively, but that small deviations from these hard negatives will likely not register as positives. Similar observations can be found in the literature on adversarial examples, where many adversarial examples have been shown to be "unstable" with respect to minute perturbations of the image [75, 77, 6]. In addition, leveraging the

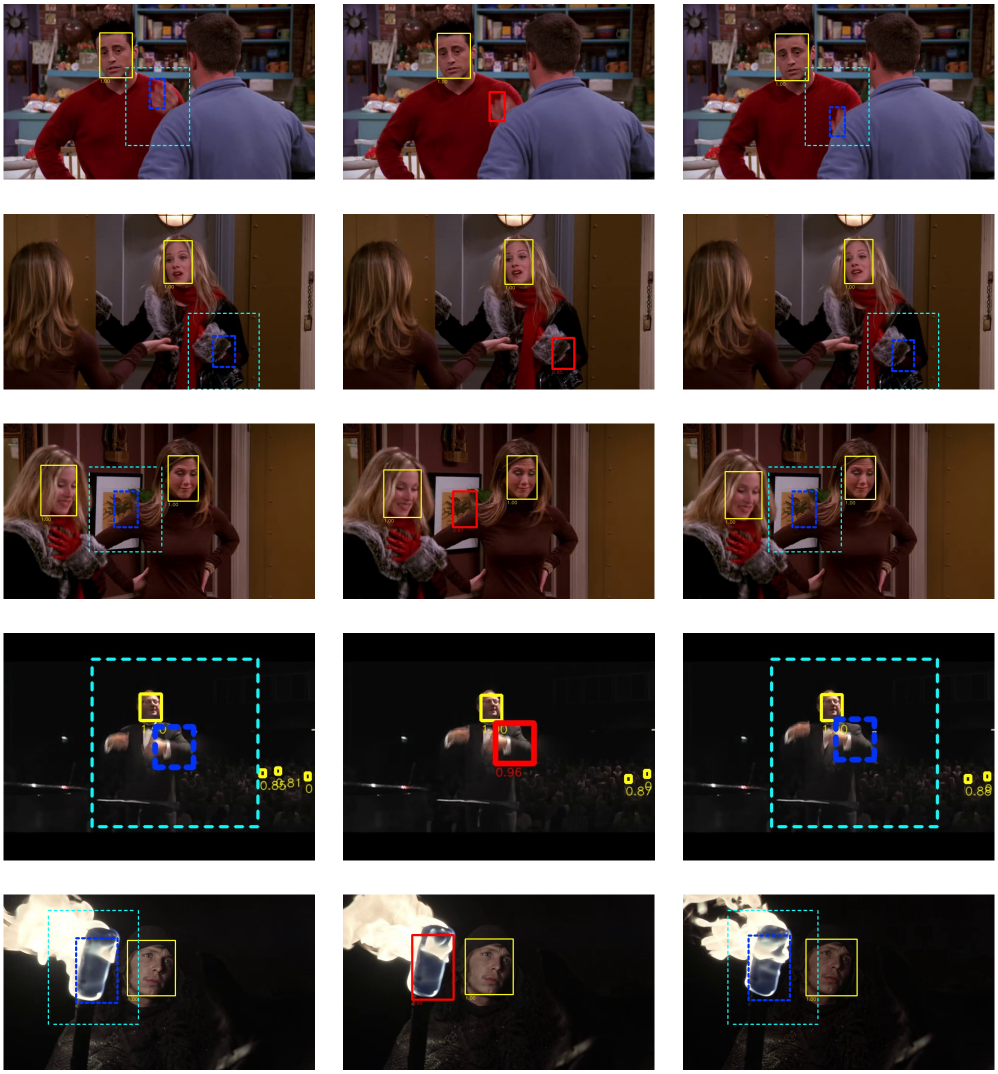
[1] Note we are *not* claiming that most false positives will be isolated, but only that flickers are likely to be false positives, a very different statement.

continuity of labelling across space and time has a long history in computer vision. Spatial label dependencies are widely modeled by Markov random fields [39] and conditional random fields [120], while the smoothness of labels across time is a staple of tracking methods and other video processing algorithms [117, 58, 143].

As our experiments show, a large percentage of detector flickers are indeed false positives, and more importantly, they are hard negatives, since they were identified incorrectly as positives by the detector. Such an *automatically generated training set* of hard negatives can be used to fine-tune a detector, often leading to improved performance. Similar benefits are gained from fine-tuning with *hard positives*, which are obtained in an analogous fashion from cases where a consistently detected object "flickers off" in an isolated frame. While these flickers are relatively rare, it is inexpensive to run a modern detector on many hours of unlabeled video, generating essentially unlimited numbers of hard examples. Being an unsupervised process, training sets gathered automatically in this fashion do include some noise. Nevertheless, our experiments show that significant improvements can be gleaned by retraining detectors using these noisy hard examples. An alternative to gathering such hard examples automatically is, of course, to obtain them manually. However, the rarity of false positives for modern detectors makes this process extremely expensive. Doing this manually requires that every positive detection be examined for validity. With typical false positive rates around one per 1000 images, this process requires the examination of 1000 images per false positive, making it prohibitively expensive.

5.1 Mining Hard Examples from Videos

This section discusses methods for automatically mining hard examples from videos, including data collection (Chapter 5.1.1), our hard negative mining algorithm (Chapter 5.1.2), statistics of recovered hard negatives (Chapter 5.1.3) and extension to hard positives (Chapter 5.1.4). Details of re-training the detector on these new samples are in the Experiments section (Chapter 5.2.1).



frame f-1 frame f frame f+1

Figure 5.2: **Mining hard negatives from detector-flicker.** The solid boxes denote detections, and the dashed boxes are associated with the tracking algorithm. Given all of the high-confidence **face detections** in a video (**yellow** boxes), the proposed algorithm generates a **tracklet** (**blue** *dashed* boxes) for the **current detection** (**red** box in frame f) by applying template matching within the **search regions** of the adjacent frames (**cyan** *dashed* boxes). As there are no matching detections in adjacent frames for the current detection (i.e. no yellow box matches the blue dashed boxes in frames f-1 or f+1), it is correctly considered to be an "isolated detection" and added to the set of ***hard negatives***. The remaining detections in frame f, which are temporally consistent, are added to the set of ***pseudo-positives***.

5.1.1 Video Collection

To mine hard examples for face detection, we used 101 videos from sitcoms, each with a duration of 21-25 minutes and a full-length movie of 1 hour 47 minutes, *"Hannah and her sisters"* [85]. Further, we performed YouTube searches with keywords based on: *public address*, *debate society*, *orchestra performance*, *choir practice* and *courtroom*, downloading 89 videos of durations ranging from 10 to 25 minutes. We obtained videos that were expected to feature a large number of human faces in various scenes, reflecting the everyday settings of our face benchmarks. Similarly, for pedestrian detection, we collected videos from YouTube by searching with the two key phrases: *driving cam videos* and *walking videos*. We obtained 40 videos with an average duration of about 30 minutes.

5.1.2 Hard Negative Mining

Running a pre-trained face detector on every frame of a video gives us a large set of detections with noisy labels. We crucially differ here from recent bootstrapping approaches [131, 118] by (a) using large amounts of *unlabeled* data available on the web instead of relying only on the limited fully-supervised training data from WIDER Face [146] or Caltech Pedestrians [27], and (b) having a novel filtering criterion on the noisy labels obtained from the detector that retains the hard negative examples and minimizes noise in the obtained labels.

The raw detections from a video were thresholded at a relatively high confidence score of 0.8, based on visual inspection of a small subset of the data. For every detection in a frame, we formed a short tracklet by performing template matching in adjacent frames, within a window of ± 5 frames — the bounding box of the current detection was enlarged by 100 pixels and this region was searched in adjacent frames for the best match using normalized cross correlation (NCC). To account for occlusions, we put a threshold on the NCC similarity score (set as 0.5) to reject cases where there was a lot of appearance-change between frames. Now in each frame, if the maximum intersection-over-union (IoU) between the tracklet prediction and detections in the adjacent frames was below 0.2, we considered it to be an isolated detection resulting from **detector flicker**. These isolated detections were taken as ***hard negatives***. The detections that *were* found to be consistent with adjacent frames were considered to have a high probability of being true predictions and were termed ***pseudo-positives***. For the purpose of creating the re-training set, we kept only those frames that had at least one pseudo-positive detection in addition to one or more hard negatives. Illustrative examples of this procedure are shown in Figure 5.2, where we visualize only the previous and next frames for simplicity.

5.1.3 Results of Automatic Hard Negative Mining

Our initial mining experiments were performed using a standard Faster R-CNN detector trained on WIDER Face [146] for faces and Caltech [27] for pedestrians. We collected 13,888 video frames for faces, where each frame contains at least one pseudo-positive and one hard negative (detector flicker). To verify the quality of our automatically mined hard negatives[2], we randomly sampled 511 hard negatives for inspection. 453 of them are true negatives, while 16 samples are true positives, and 42 samples are categorized as *ambiguous*, which correspond extreme head pose or severe occlusions. The precision for true negatives is 88.65% and precision for true negatives plus *ambiguous* is 96.87%.

For pedestrians, we collected 14,967 video frames. We manually checked 328 automatically mined hard negatives, where 244 of them are true negatives and 21 belong to *ambiguous*. The precision for true negatives is 74.48% and precision for true negatives plus *ambiguous* is 82.18%.

To further validate our method on an existing fully-annotated video dataset, we used the Hannah dataset [85], which has every frame annotated with face bounding boxes. Here, out of 234 mined hard negatives, 187 were true negatives, resulting in a precision of 79.91%. We note that the annotations on the Hannah movie are not always consistent and involve

[2] This verification was based on the picture viewed in isolation, separate from the video.

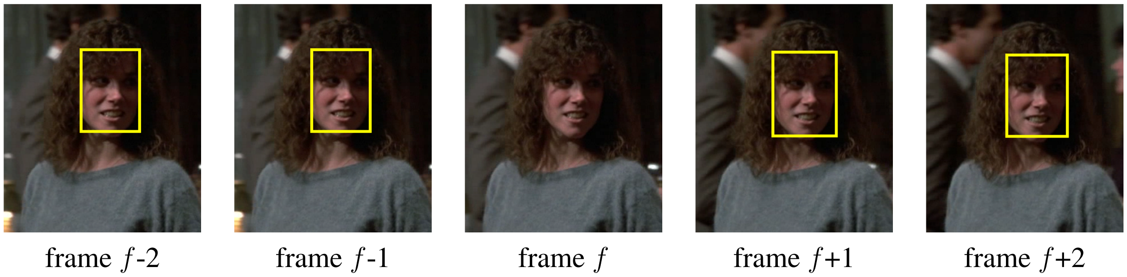


Figure 5.3: **Hard positive samples.** Given a sequence of video frames, we notice that the face of the actor is consistently detected, except at frame f. Such isolated "off-flickers" can be harvested in an unsupervised fashion to form a set of *hard positives*.

a significant domain shift from WIDER. Considering the fact no human supervision is provided, the mined face hard negatives are consistently of high quality across various domains.

5.1.4 Extension to Hard Positive Mining

In principle, the same concept for using detector flickers can be directly applied to obtaining ***hard positives***. The idea is to look for "off-flickers" of a detector in a video tracklet – given a series of detections of an object in a video, such as a face, we can search for single frames that have no detections but are surrounded by detections on either side. Of course, these could be caused by short-duration occlusions, for example, but a large percentages of these "off-flickers" are hard positives, as in Fig. 5.3. We generate tracklets using the method from [55] and show results incorporating hard positives on pedestrian and face detection in the experiments section. The manually calculated purity over 300 randomly sampled frames was 94.46% for faces and 83.13% for pedestrians.

5.2 Experiments

We evaluate our method on face and pedestrian detection and perform ablation studies analyzing the effect of the hard examples.For pedestrians, we show results on the Caltech dataset [27], while for face detection, we show results on the WIDER Face [146] dataset.

The Caltech Pedestrian Dataset [27] consists of videos taken from a vehicle driving through urban traffic, with about 350k annotated bounding-boxes from 250k video frames.

The WIDER dataset consists of 32,203 images having 393,703 labeled faces in challenging situations of scale, pose and occlusion. The evaluation set of WIDER is divided into *easy*, *medium*, and *hard* sets according to the detection scores of object proposals from EdgeBox [159]. From easy to hard, the faces get smaller and more crowded. We show results on all three sets of WIDER.

5.2.1 Retraining Detectors with Mined Hard Examples

We experimented with two ways to leverage our mined *hard negative* samples. In our initial experiments, a single mini-batch is formed by including one image from the original labeled training dataset and another image sampled from our automatically-mined hard negative video frames. In this way, positive region proposals are sampled from the original training dataset image, based on manual annotation, while negative region proposals are sampled from both the original dataset image and the mined hard negative video frame. Thus, we can *explicitly* force the network to focus on the hard negatives from the mined video frame. However, this method did not produce better results in our initial experiments. An alternate approach was found to be more effective – we simply provided the *pseudo-positives* in the mined video frames as true object annotations during training and *implicitly* allowed the network to pick the hard-negatives. The inclusion of video frames with *hard positives* is more straightforward – we can simply treat them as additional images with object annotations at training time. The models were fine-tuned with and without OHEM, and we consistently chose the setting that gave the best validation results. While OHEM would increase the likelihood of hard negatives being selected in a mini-batch, it would also place extra emphasis on any mislabels in the hard examples. This would magnify the effect of a small amount of label noise and can in some cases decrease the overall performance.

5.2.2 Ablation Settings

In addition to the comparisons to the baseline Faster R-CNN detectors, we conduct various ablation studies on the Caltech Pedestrian and WIDER Face datasets to address the effectiveness of hard example mining.

Effect of training iterations. To account for the possible situation where simply training the baseline model longer may result in a gain in performance, we create another baseline by fine-tuning the original model for additional iterations with a lower learning rate, matching the number of training iterations used in our hard example trained models. We refer to this model as "`w/ more iterations`".

Effect of additional video frames. Unlike the baseline detector, our fine-tuned models use additional video frames for training. Although this additional data is unlabeled, it is possible that just using the high-confidence detection results on unlabeled video frames as *pseudo-groundtruths* during training is sufficient to boost performance, without correcting the wrong detections (hard negatives) using our detector flicker approach. Therefore we train another detector, "`Flickers as Positives`", starting from the baseline model, that takes exactly the same training set as our hard negative model, but where *all* the high-confidence detections on the video frames are used as positive labels.

Effect of automatically mined hard examples. We include the results from our proposed method of considering detector flickers as hard negatives and hard positives separately – "`Flickers as HN`" and "`Flickers as HP`". Finally, we report results from fine-tuning the detector on the union of both types of hard examples (`Flickers as HN + HP`).

5.2.3 Pedestrian Detection

For our `baseline` model, we train the VGG16-based ***Faster R-CNN*** object detector [96] with OHEM [108] for 150K iterations on the **Caltech Pedestrian** training dataset [27]. We used *all* the frames from set00-set05 (which constitute the training set), irrespective of whether they are flagged as "reasonable" or not by the Caltech meta-data. Following Zhang *et al.* [150], we set the IoU ratio for RPN training to 0.5, while all the other experimental settings are identical to [96]. The number of labeled Caltech images is 128,419 and our mining provides 14,967 hard negative and 42,914 hard positive frames. We fine-tune the baseline model with hard examples and the annotated examples from the Caltech Pedestrian *training* dataset, with a fixed learning rate of 0.0001 for 60K iterations, using OHEM. We evaluate our model on the Caltech Pedestrian testing dataset under the *reasonable* condition.

The ROC curves of various settings of our models are shown in Fig. 5.4(a). Fine-tuning the existing detector for more iterations gives a modest reduction in log average miss rate, from 23.83% to 22.4%. Using all detections without correcting the hard negatives (`Flickers as Pos`) also gives a small improvement – the extra training data, although noisy, still has some positive contribution during fine-tuning. Our proposed model, fine-tuned with the mined hard negatives (`Flickers as HN`), has a log average miss rate of **18.78%**, which outperforms the `baseline model` by **5.05%**. Fine-tuning with hard positives (`Flickers as HP`) also shows an improvement of **4.39%** over the baseline. Combining both hard positives and hard negatives results in the best performance of **18.72%** log average miss rate.

In Figure 5.4(b) we report results using the state-of-the-art ***SDS-RCNN*** [13] pedestrian detector [3]. Every 3rd frame is sampled from the Caltech dataset for training the original detector [13], and we keep this setting in our experiments. For SDS-RCNN, there are 42,782 labeled training images while the mining gives us 42,782 hard negative and 177,562 hard positive frames. The inclusion of hard negatives in training (`Flickers as HN`) improves the performance of SDS-RCNN in the low False Positives regime compared to the baseline – the detector learns to eliminate a number of false detections, thereby increasing

[3] Running the authors' released code from `https://github.com/garrickbrazil/SDS-RCNN`

precision, but it also ends up hurting the recall. Including mined hard positives (`Flickers as HP`) we get the best performance of **8.71%** log average miss rate, outperforming the model using both the mined hard negative and positive samples (`Flickers as HP + HN`), which gets 9.12%.

5.2.4 Face Detection

We adopt the Faster R-CNN framework, using VGG16 as the backbone network. We first train a baseline detector starting from an ImageNet pre-trained model, with a fixed learning rate of 0.001 for 80K iterations using the SGD optimizer, where the momentum is 0.9 and weight decay is 0.0005. For hard negatives, the model is fine-tuned for 50k iterations with learning rate 0.0001. For hard positives, and the combination of both types of hard examples, we train longer for 150k iterations. Following the **WIDER Face** protocol, we report Average Precision (AP) values in Table 5.1 on the three splits – 'Easy', 'Medium' and 'Hard'. OHEM is not used as it was empirically observed to decrease performance.

Fine-tuning the baseline model for more iterations improves performance slightly on the Easy and Medium splits. Naively considering all the high confidence detections as true positives (`Flickers as Positives`) degrades performance substantially across all splits. Hard negative mining, `Flickers as HN`, slightly outperforms the baseline Faster R-CNN detector (`w/ more iterations`) on the Medium and Hard splits, retaining the same performance of 0.907 AP on the Easy split. Using the mined hard positives, `Flickers as HP`, we observe a significant gain in performance on all three splits. Using both hard positives and hard negatives jointly (`Flickers as HP + HN`) improves over using hard negatives and the baseline, but the improvement is lesser than the gains from `Flickers as HP`.

For faces, we additionally experimented with the recent RetinaNet [69] detector as a second high-performance baseline model. Unfortunately, inclusion of the unlabeled data hurt performance slightly using this model, despite the reasonably high purity of the mined examples. Further details on this experiment and possible explanations are discussed in Chapter 5.3.4.

Table 5.1: Average precision (AP) on the validation set of the **WIDER Face** [146] benchmark. Including hard examples improves performance over the baseline, with `HP` and `HP+HN` giving the best results.

		Easy	Medium	Hard
Faster R-CNN	Baseline	0.907	0.850	0.492
	w/ more iterations	0.910	0.852	0.493
	Flickers as Positives	0.829	0.790	0.434
	Ours: Flickers as HN	0.909	0.853	0.494
	Ours: Flickers as HP	**0.921**	**0.864**	0.492
	Ours: Flickers as HP + HN	**0.921**	**0.864**	**0.497**

5.3 Discussion

In this section, we discuss some further applications and extensions to our proposed hard example mining method.

5.3.1 On the Entropy of the False Positive Distribution

In mining thousands of hard negatives from unlabeled video, we noticed a striking pattern in the hard negatives of face detectors. A large percentage of false positives were generated by a few types of objects. Specifically, a large percentage of hard negatives in face detectors seem to stem from human hands, ears, and the torso/chest area. Since it appears that a large percentage of the false positives in face detection are the result of a relatively small number phenomena, this could explain the significant gains realized by modeling hard negatives. In particular, characterizing the distribution of hard negatives, and learning to avoid them, may involve a relatively small set of hard negatives.

5.3.2 Effect of Domain Shift on FDDB

The FDDB dataset [53] is comprised of 5,171 annotated faces in a set of 2,845 images taken from a subset of the Face in the Wild dataset. The images and the annotation style of FDDB have a significant *domain shift* from WIDER Face, which are discussed in Jamal et al. [1]. Fig. 5.7 compares our method with the Faster R-CNN baseline on FDDB, using the trained models from our experiments on WIDER Face (Chapter 5.2.4). Although hard negatives reduce false positives (Fig. 5.7(b)) and hard positives increase recall (Fig. 5.7(c)), the performance does not consistently improve over the baseline on FDDB. We hypothesize that the advantages from our unsupervised hard examples are counteracted by the effects of domain shift – the large amounts of new training data result in shifting the original detector further away from the target FDDB domain, leading to an overall loss in performance. This may not have hurt our performance as much on WIDER Face because the domain shift between the relatively unconstrained WIDER images and our videos downloaded from YouTube was not severe enough to subsume the advantages from the hard examples.

5.3.3 Extension to Other Classes

The simplicity of our approach makes it easily extensible to other categories in a one-versus-rest setting. YouTube is a promising source of videos for various MS-COCO or PASCAL categories; mining hard negatives after that is fully automatic. To demonstrate this, we selected categories from MS-COCO and ran experiments to check if inclusion of hard negatives improves the baseline performance of a Faster R-CNN detector. We used the training method deployed by Sonntag et al.[115], which allows for a convenient fine-tuning of the VGG16-based Faster R-CNN model on specific object classes of the MS-COCO dataset. The method was used to train a Faster R-CNN detector for a specific class vs background, starting from a multi-class VGG16 classifier pre-trained on Image-Net categories. This baseline detector was then used to mine hard negatives from downloaded

YouTube videos of that category and then re-trained on the union of the new data and the original labeled training data. We show results for two categories: *dogs* and *trains*. A held out subset of the MS-COCO validation set was used for validating training hyper-parameters and the remainder of the validation data was used for evaluation.

For the *dog* category, the labeled data was divided into train/val/test splits of 3041/ 177/ 1521 images. We manually selected and downloaded about 22 hours of dog videos from YouTube. The videos were primarily logs of dog racing and agility championships with about 95% of the frames containing dogs. We used the baseline dog detector to obtain detections on about 15 hours (1,296,000 frames at 24 fps) of dog videos. The hard negative mining algorithm was then run at a detector confidence threshold of 0.8. This yielded 2611 frames with at least one hard negative and one positive detection. The baseline model was then fine-tuned for 30k iterations on the union of the labeled MS-COCO data and the hard negatives. The hyper-parameters and best model were selected using a validation set. Similar experiments with *trains* were performed, with train/val/test splits of 2464/157/1281 images. The results are summarized in the Table 5.2, where inclusion of hard negatives is observed to improve the baseline detector in both cases.

Table 5.2: Results on augmenting Faster R-CNN detectors with hard negatives for '*dog*' and '*train*' categories on MS-COCO.

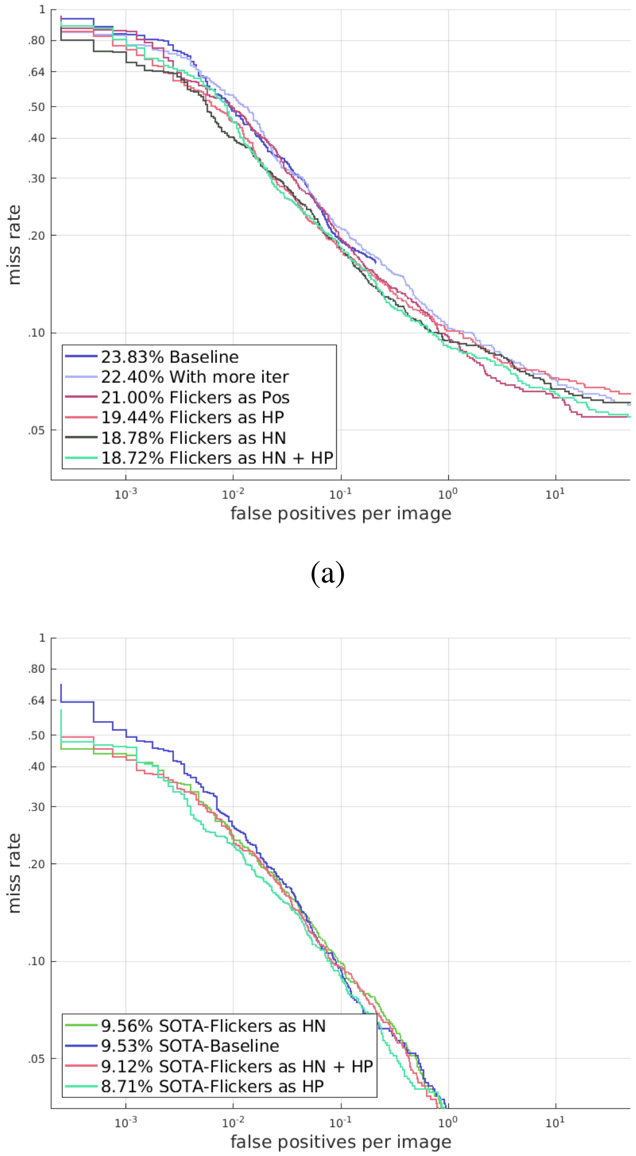
Category	**Model**	**Training iterations**	**Training hyperparams**	**Validation set AP**	**Test set AP**
Dog	Baseline	29000	LR : 1e-3 for 10k, 1e-4 for 10k-20k, 1e-5 for 20k-29k	26.9	25.3
	Flickers as HN	22000	LR : 1e-4 for 15k, 1e-5 for 15k-22k	28.1	26.4
Train	Baseline	26000	LR : 1e-3, stepsize: 10k, lr-decay: 0.1	33.9	33.2
	Flickers as HN	24000	LR : 1e-3, stepsize: 10k, lr-decay: 0.1	35.4	33.7

5.3.4 Experiments on RetinaNet

In addition to the multiple versions of Faster R-CNN, we also tried retraining a RetinaNet [69] detector for improved face detection using our mined hard negatives. For this single-stage architecture, we were unable to achieve any reliable improvements, despite the majority of our hard negatives being mined using the RetinaNet detector. Since this detector still has significant numbers of false positives, and we were able to mine these successfully with our procedure, it was puzzling that we could not achieve better results on this architecture. One possible explanation for this is as follows: the focal loss used in the RetinaNet architecture puts a heavy weight on incorrect examples. While the purity of our mined examples is high, it is not perfect, and a non-negligible percentage of our mined hard negatives are actually true positives. Since these samples would inherit the wrong label, they would be strongly emphasized by the focal loss. Thus, it is possible that while RetinaNet outperforms the Faster R-CNN on standard benchmarks, it may be more susceptible to label noise and thus not a good candidate for our method. In the future, we will investigate different values of the focal loss parameter to see whether this can mitigate the effects of label noise.

5.3.5 Additional Applications

Our method is particularly suited to detection problems since they are well-known for having vast numbers of easy negative examples, which provide little benefit to training. The introduction of large numbers of hard negatives intuitively will help. However, there is no reason the same ideas cannot be applied to the generation of extra training data for regular recognition problems. We intend to investigate this direction, along with more applications of hard positives and negatives, in future work.



(a)

(b)

Figure 5.4: Results on the **Caltech Pedestrian** dataset [27] in *reasonable* condition. (a) Faster R-CNN results: using hard negative samples (`Flickers as HN`) and hard positive samples (`Flickers as HP`) improve the performance over the baseline in; using a combination of both gives the best performance. (b) State-of-the-art SDS-RCNN results: `Flickers as HN` improves the original SDS-RCNN results only in the low false positive regime, while `Flickers as HP` gives the best results.

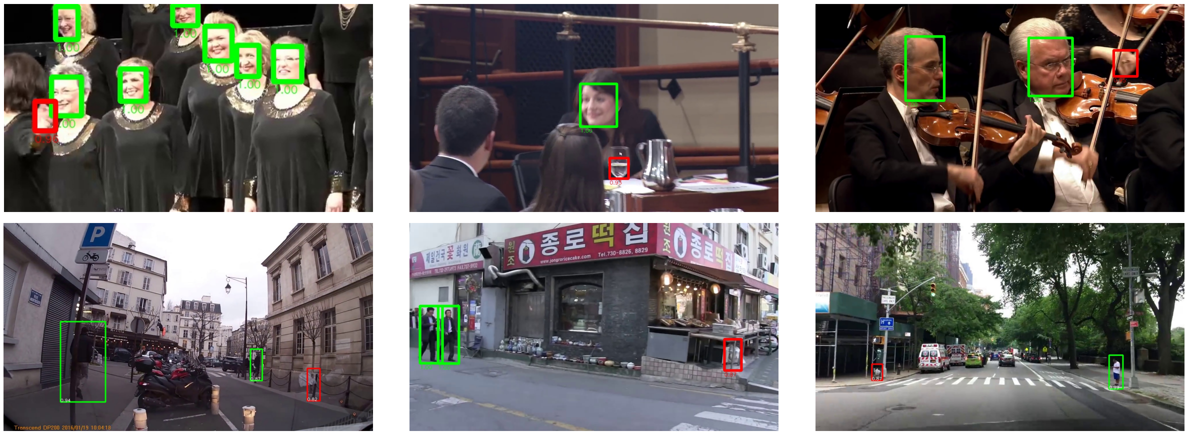


Figure 5.5: **Examples of hard negatives.** Visualization of automatically mined hard negatives for faces (*top row*) and pedestrians (*bottom row*). Red boxes denote the “detection-flicker cases” among the high confidence detections (green boxes).

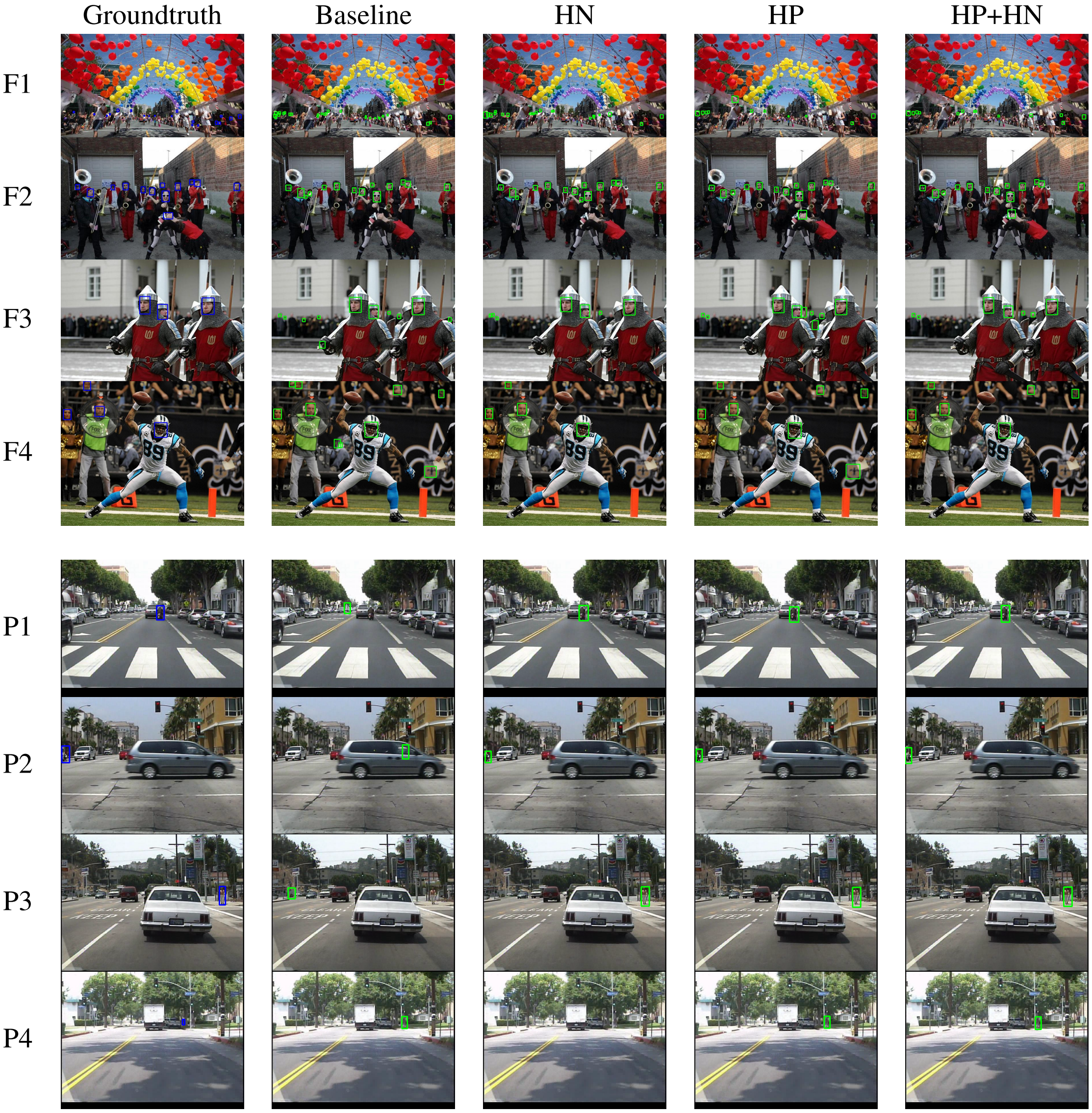
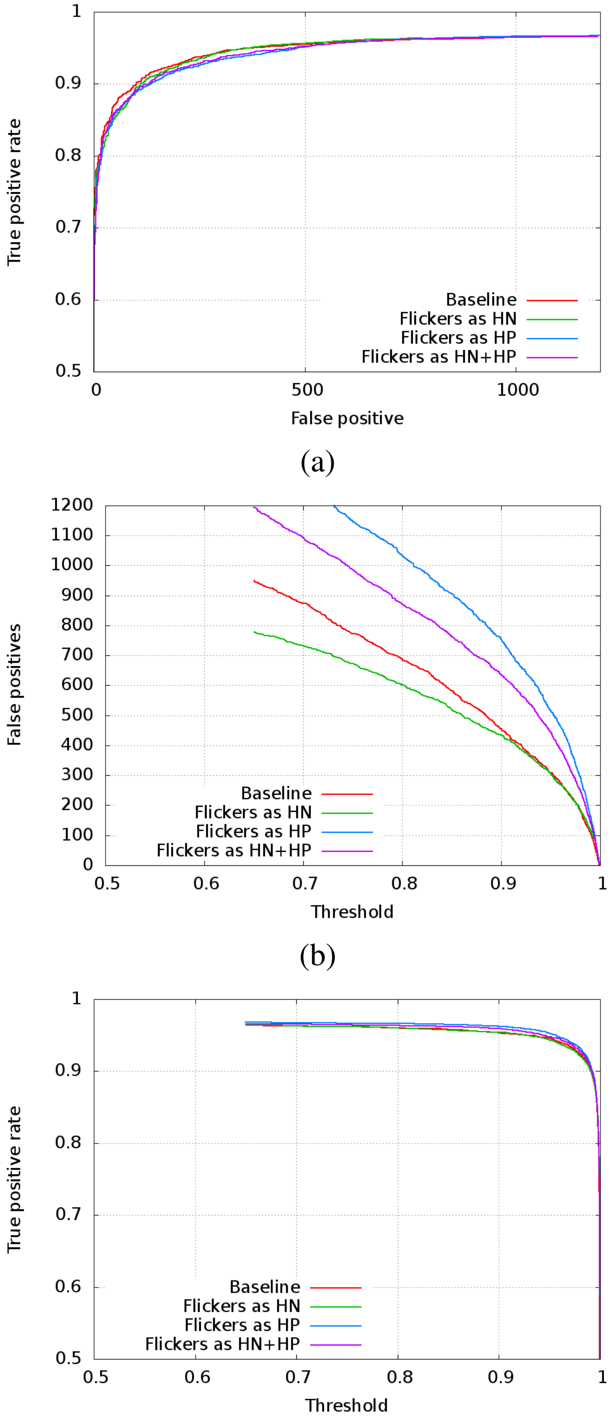


Figure 5.6: **Qualitative comparison.** Faster R-CNN detections for faces (F1-4) and pedestrians (P1-4).The detector fine-tuned with hard negatives (HN) reduces false positives compared to the Baseline (F-1,3,4; P-1,2,3), but can sometimes lower the recall (P4). Hard positives (HP) increases recall (F2, P4) but can also introduce false positives (F4). Using both (HP+HN) the detector is usually able to achieve a good balance.



(a)

(b)

(c)

Figure 5.7: Results on **FDDB**. (a) ROC curves comparing our hard example methods with the baseline Faster R-CNN detector; (b-c) separate plots showing False Positives and True Positive Rate with varying thresholds on detector confidence score (best seen in color and with zoom).

CHAPTER 6

CONCLUSION

Face-clustering in videos is a problem of grouping faces in a video so that each group contains a unique individual [32, 19, 55, 125]. In this dissertation, three important problems in the area of face-clustering have been studied.

First, we have observed that one false connection in a link-based clustering algorithm [102] can result in poor clustering performance. To improve the clustering performance, in Chapter 3, we have presented a system for doing end-to-end clustering in full-length videos and movies. In addition to a careful combination of detection and tracking, and a new end-to-end evaluation metric, we have introduced a novel approach to link-based clustering that we call Erdős-Rényi clustering. We demonstrated a method for automatically estimating a good decision threshold for a verification method based on rank-1 counts by estimating the underlying portion of the rank-1 counts distribution due to mismatched pairs.

Faces might not be sufficiently clear for direct recognition. In Chapter 4, we have presented a novel framework to improve face-clustering in videos by exploiting scene level information to address the challenges of face-clustering across different camera shots. A new network architecture, FB-Net, is proposed for face recognition that leverages contextual information from the entire scene to learn robust identity embeddings. Furthermore, we introduced a new dataset that contains face identities in the context of consistent scenes. We conducted experiments on standard video face-clustering benchmarks, and our experimental results demonstrate a significant boost in performance by utilizing scene-contextualized face embeddings. It shows improved performance over the state-of-the-art that utilizes face-level features only. Through both qualitative and quantitative results, we observe that by explicitly learning how consistent scene elements are correlated across different camera shots, we can learn better identity representations, especially when face regions are ambiguous.

Finally, in Chapter 5, we leverage an existing phenomenon – detector flicker in videos – to mine hard negatives and hard positives at scale in an unsupervised manner. The usefulness of this method for improving an object detector is demonstrated on standard benchmarks for two well-known tasks – face and pedestrian detection, supported by several ablation studies. The simplicity of our hard example mining approach makes it widely applicable to a variety of practical scenarios

This thesis has addressed three challenging problems in the area of face-clustering in videos, and proposed novel approaches to tackle the problems. Without doubts, identifying the faces of the same person in a video would be an essential part for an AI system to understand a video/movie. With a complete video understanding system, we believe that the AI system would learn human life through videos.

As my future work, I want to explore various components that are required to build a video understanding system. Especially, I want my AI system to learn empathy, which is the ability to understand and share the feelings of another. Humans have a strong ability to predict how people in movies/videos feel/think not only by one's facial expression changes but also based on their experiences so far. For example, there is a short video clip that shows a 6-year-old girl crying after hearing the news from her parents that she will go to Disneyland as a part of her birthday gifts[1]. If humans watch the video clip, we can see that she is *extremely happy* even if every piece of her body gestures and facial expressions are supporting the evidence that she is *very sad*. In my future research, I will make my machine indirectly get those experiences by watching a lot of videos on the web and eventually make my AI system to have empathy as well as a smart brain.

[1] A girl who is crying for the Disneyland surprise (https://youtu.be/OOpOhlGiRTM?t=115)

BIBLIOGRAPHY

[1] Abdullah Jamal, Muhammad, Li, Haoxiang, and Gong, Boqing. Deep face detector adaptation without negative transfer or catastrophic forgetting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2018).

[2] Aljundi, Rahaf, Chakravarty, Punarjay, and Tuytelaars, Tinne. Whos that actor? automatic labelling of actors in tv series starting from imdb images. In *ACCV* (2016), Springer, pp. 467–483.

[3] Amigó, Enrique, Gonzalo, Julio, Artiles, Javier, and Verdejo, Felisa. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval 12*, 4 (2009), 461–486.

[4] Anguelov, Dragomir, Lee, Kuang-chih, Gokturk, Salih Burak, and Sumengen, Baris. Contextual identity recognition in personal photo albums. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (2007), IEEE, pp. 1–7.

[5] Appel, Ron, Fuchs, Thomas, Dollár, Piotr, and Perona, Pietro. Quickly boosting decision trees–pruning underachieving features early. In *International Conference on Machine Learning* (2013), pp. 594–602.

[6] Athalye, Anish, and Sutskever, Ilya. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397* (2017).

[7] Ba, Jimmy Lei, Kiros, Jamie Ryan, and Hinton, Geoffrey E. Layer normalization. In *arXiv:1607.06450* (2016).

[8] Baradel, Fabien, Wolf, Christian, and Mille, Julien. Human activity recognition with pose-driven attention to rgb. In *British Machine Vision Conference (BMVC)* (2018), pp. 1–14.

[9] Barlow, Horace. Cerebral cortex as model builder. In *Matters of Intelligence*. Springer, 1987, pp. 395–406.

[10] Bauml, Martin, Tapaswi, Makarand, and Stiefelhagen, Rainer. Semi-supervised learning with constraints for person identification in multimedia data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2013), pp. 3602–3609.

[11] Bell, Sean, Lawrence Zitnick, C., Bala, Kavita, and Girshick, Ross. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016).

[12] Blum, Avrim, and Mitchell, Tom. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory* (1998), ACM, pp. 92–100.

[13] Brazil, Garrick, Yin, Xi, and Liu, Xiaoming. Illuminating pedestrians via simultaneous detection & segmentation. *arXiv preprint arXiv:1706.08564* (2017).

[14] Cai, Zhaowei, Fan, Quanfu, Feris, Rogerio S, and Vasconcelos, Nuno. A unified multi-scale deep convolutional neural network for fast object detection. In *European Conference on Computer Vision* (2016), Springer, pp. 354–370.

[15] Cai, Zhaowei, Saberian, Mohammad, and Vasconcelos, Nuno. Learning complexity-aware cascades for deep pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 3361–3369.

[16] Cao, Qiong, Shen, Li, Xie, Weidi, Parkhi, Omkar M., and Zisserman, Andrew. Vggface2: A dataset for recognising faces across pose and age. In *IEEE Conference on Automatic Face and Gesture Recognition (FG)* (2018).

[17] Chang, Haw-Shiuan, Learned-Miller, Erik, and McCallum, Andrew. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems* (2017), pp. 1003–1013.

[18] Chapelle, Olivier, Scholkopf, Bernhard, and Zien, Alexander. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks 20*, 3 (2009), 542–542.

[19] Cinbis, Ramazan Gokberk, Verbeek, Jakob, and Schmid, Cordelia. Unsupervised metric learning for face identification in tv video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2011), pp. 1559–1566.

[20] Cour, Timothee, Sapp, Benjamin, Nagle, Akash, and Taskar, Ben. Talking pictures: Temporal grouping and dialog-supervised person recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2010), IEEE, pp. 1014–1021.

[21] Crosswhite, Nate, Byrne, Jeffrey, Stauffer, Chris, Parkhi, Omkar M., Cao, Qiong, and Zisserman, Andrew. Template adaptation for face verification and identification. In *IEEE Conference on Automatic Face and Gesture Recognition (FG)* (2017).

[22] Dalal, Navneet, and Triggs, Bill. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2005), pp. 886–893.

[23] Davey, Samuel J, Rutten, Mark G, and Cheung, Brian. A comparison of detection performance for several track-before-detect algorithms. *EURASIP Journal on Advances in Signal Processing 2008* (2008), 41.

[24] DeCann, Brian, and Ross, Arun. Modeling errors in a biometric re-identification system. *IET Biometrics 4*, 4 (2015), 209–219.

[25] Divvala, Santosh K, Hoiem, Derek, Hays, James H, Efros, Alexei A, and Hebert, Martial. An empirical study of context in object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009), IEEE, pp. 1271–1278.

[26] Dollár, Piotr, Tu, Zhuowen, Perona, Pietro, and Belongie, Serge. Integral channel features. *British Machine Vision Conference (BMVC)* (2009).

[27] Dollár, Piotr, Wojek, Christian, Schiele, Bernt, and Perona, Pietro. Pedestrian detection: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009), IEEE, pp. 304–311.

[28] Dollár, Piotr, and Zitnick, C. Lawrence. Fast edge detection using structured forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 37*, 8 (2015), 1558–1570.

[29] Du, Xianzhi, El-Khamy, Mostafa, Lee, Jungwon, and Davis, Larry. Fused dnn: A deep neural network fusion approach to fast and robust pedestrian detection. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on* (2017), IEEE, pp. 953–961.

[30] Erdős, P., and Rényi, A. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences 5* (1960), 17–61.

[31] Ester, Martin, Kriegel, Hans-Peter, Sander, Jörg, and Xu, Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD 96*, 34 (1996), 226–231.

[32] Everingham, Mark, Sivic, Josef, and Zisserman, Andrew. ”Hello! My name is... Buffy” Automatic naming of characters in TV video. In *British Machine Vision Conference (BMVC)* (2006).

[33] Farfade, Sachin Sudhakar, Saberian, Mohammad J., and Li, Li-Jia. Multi-view face detection using deep convolutional neural networks. In *ICMR* (2015), pp. 643–650.

[34] Felzenszwalb, Pedro F, Girshick, Ross B, McAllester, David, and Ramanan, Deva. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence 32*, 9 (2010), 1627–1645.

[35] Forney, G David. The Viterbi algorithm. *Proceedings of the IEEE 61*, 3 (1973), 268–278.

[36] Frey, Brendan J, and Dueck, Delbert. Clustering by passing messages between data points. *Science 315*, 5814 (2007), 972–976.

[37] Friedman, Jerome, Hastie, Trevor, Tibshirani, Robert, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics 28*, 2 (2000), 337–407.

[38] Gallagher, Andrew C, and Chen, Tsuhan. Clothing cosegmentation for recognizing people. In *2008 IEEE Conference on Computer Vision and Pattern Recognition* (2008), IEEE, pp. 1–8.

[39] Geman, Stuart, and Graffigne, Christine. Markov random field image models and their applications to computer vision. In *Proceedings of the international congress of mathematicians* (1986), vol. 1, p. 2.

[40] Girdhar, Rohit, Carreira, João, Doersch, Carl, and Zisserman, Andrew. Video Action Transformer Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

[41] Girdhar, Rohit, and Ramanan, Deva. Attentional pooling for action recognition. In *Advances in Neural Information Processing Systems* (2017), pp. 34–45.

[42] Girshick, Ross B. Fast R-CNN. In *IEEE International Conference on Computer Vision (ICCV)* (2015), pp. 1440–1448.

[43] Girshick, Ross B., Donahue, Jeff, Darrell, Trevor, and Malik, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 580–587.

[44] Guo, Yandong, Zhang, Lei, Hu, Yuxiao, He, Xiaodong, and Gao, Jianfeng. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision (ECCV)* (2016).

[45] Gyaourova, Aglika, and Ross, Arun. Index codes for multibiometric pattern retrieval. *IEEE Transactions on Information Forensics and Security (TIFS) 7*, 2 (April 2012), 518–529.

[46] Haurilet, Monica-Laura, Tapaswi, Makarand, Al-Halah, Ziad, and Stiefelhagen, Rainer. Naming tv characters by watching and analyzing dialogs. In *WACV* (2016).

[47] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision (ECCV)* (2014), pp. 346–361.

[48] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016).

[49] Hosang, Jan, Omran, Mohamed, Benenson, Rodrigo, and Schiele, Bernt. Taking a deeper look at pedestrians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 4073–4082.

[50] Hu, Peiyun, and Ramanan, Deva. Finding tiny faces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), IEEE, pp. 1522–1530.

[51] Huang, Gary B., Mattar, Marwan, Berg, Tamara, and Learned-Miller, Erik. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *The Workshop on Faces in Real-Life Images at ECCV* (2008).

[52] Huang, Qingqiu, Xiong, Yu, and Lin, Dahua. Unifying identification and context learning for person recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2018).

[53] Jain, Vidit, and Learned-Miller, Erik. FDDB: A benchmark for face detection in unconstrained settings. Tech. Rep. UM-CS-2010-009, University of Massachusetts, Amherst, 2010.

[54] Jiang, Huaizu, and Learned-Miller, Erik. Face detection with the faster r-cnn. In *IEEE Conference on Automatic Face and Gesture Recognition (FG)* (2017), IEEE, pp. 650–657.

[55] Jin, SouYoung, Su, Hang, Stauffer, Chris, and Learned-Miller, Erik. End-to-end face detection and cast grouping in movies using erdos-renyi clustering. In *IEEE International Conference on Computer Vision (ICCV)* (2017).

[56] Joon Oh, Seong, Benenson, Rodrigo, Fritz, Mario, and Schiele, Bernt. Person recognition in personal photo collections. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 3862–3870.

[57] Kalal, Zdenek, Matas, Jiri, and Mikolajczyk, Krystian. Pn learning: Bootstrapping binary classifiers by structural constraints. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2010), IEEE, pp. 49–56.

[58] Kläser, Alexander, Marszałek, Marcin, Schmid, Cordelia, and Zisserman, Andrew. Human focused action localization in video. In *European Conference on Computer Vision* (2010), Springer, pp. 219–233.

[59] Kuhn, Harold W. The hungarian method for the assignment problem. *Naval research logistics quarterly 2*, 1-2 (1955), 83–97.

[60] Li, Haoxiang, Brandt, Jonathan, Lin, Zhe, Shen, Xiaohui, and Hua, Gang. A multilevel contextual model for person recognition in photo albums. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 1297–1305.

[61] Li, Haoxiang, Lin, Zhe, Shen, Xiaohui, Brandt, Jonathan, and Hua, Gang. A convolutional neural network cascade for face detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 5325–5334.

[62] Li, Jianan, Liang, Xiaodan, Shen, ShengMei, Xu, Tingfa, Feng, Jiashi, and Yan, Shuicheng. Scale-aware fast r-cnn for pedestrian detection. *IEEE Transactions on Multimedia* (2017).

[63] Li, Wei, Zhao, Rui, Xiao, Tong, and Wang, Xiaogang. Deepreid: Deep filter pairing neural network for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 152–159.

[64] Li, Yao, Lin, Guosheng, Zhuang, Bohan, Liu, Lingqiao, Shen, Chunhua, and van den Hengel, Anton. Sequential person recognition in photo albums with a recurrent network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 1338–1346.

[65] Li, Yunzhu, Sun, Benyuan, Wu, Tianfu, Wang, Yizhou, and Gao, Wen. Face detection with end-to-end integration of a convnet and a 3d model. *European Conference on Computer Vision (ECCV) abs/1606.00850* (2016).

[66] Li, Zhenguo, Liu, Jianzhuang, and Tang, Xiaoou. Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In *International Conference on Machine Learning* (2008).

[67] Lin, Dahua, Kapoor, Ashish, Hua, Gang, and Baker, Simon. Joint people, event, and location recognition in personal photo collections using cross-domain context. In *European Conference on Computer Vision* (2010), Springer, pp. 243–256.

[68] Lin, Tsung-Yi, Dollár, Piotr, Girshick, Ross, He, Kaiming, Hariharan, Bharath, and Belongie, Serge. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[69] Lin, Tsung-Yi, Goyal, Priya, Girshick, Ross, He, Kaiming, and Dollár, Piotr. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002* (2017).

[70] Lisanti, Giuseppe, Masi, Iacopo, Bagdanov, Andrew D., and Bimbo, Alberto Del. Person re-identification by iterative re-weighted sparse ranking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 37*, 8 (August 2015), 1629–1642.

[71] Liu, Wei, Anguelov, Dragomir, Erhan, Dumitru, Szegedy, Christian, Reed, Scott, Fu, Cheng-Yang, and Berg, Alexander C. Ssd: Single shot multibox detector. In *European conference on computer vision* (2016), Springer, pp. 21–37.

[72] Liu, Yong, Wang, Ruiping, Shan, Shiguang, and Chen, Xilin. Structure inference net: object detection using scene-level context and instance-level relationships. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 6985–6994.

[73] Long, Xiang, Gan, Chuang, De Melo, Gerard, Wu, Jiajun, Liu, Xiao, and Wen, Shilei. Attention clusters: Purely attention based local feature integration for video classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 7834–7843.

[74] Loshchilov, Ilya, and Hutter, Frank. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343* (2015).

[75] Lu, Jiajun, Sibai, Hussein, Fabry, Evan, and Forsyth, David. No need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv preprint arXiv:1707.03501* (2017).

[76] Lu, Zhengdong, and Leen, Todd K. Penalized probabilistic clustering. *Neural Computation 19*, 6 (2007), 1528–1567.

[77] Luo, Yan, Boix, Xavier, Roig, Gemma, Poggio, Tomaso, and Zhao, Qi. Foveation-based mechanisms alleviate adversarial examples. *arXiv preprint arXiv:1511.06292* (2015).

[78] Miech, Antoine, Laptev, Ivan, and Sivic, Josef. Learnable pooling with context gating for video classification. *arXiv preprint arXiv:1706.06905* (2017).

[79] Milan, A., Leal-Taixé, L., Reid, I., Roth, S., and Schindler, K. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]* (Mar. 2016). arXiv: 1603.00831.

[80] Miyamoto, Sadaaki, and Terami, Akihisa. Semi-supervised agglomerative hierarchical clustering algorithms with pairwise constraints. In *Fuzzy Systems (FUZZ)* (2010), IEEE, pp. 1–6.

[81] Murtagh, Fionn, and Contreras, Pedro. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2*, 1 (2012), 86–97.

[82] Nagrani, Arsha, Albanie, Samuel, and Zisserman, Andrew. Learnable pins: Cross-modal embeddings for person identity. In *European Conference on Computer Vision (ECCV)* (2018), pp. 71–88.

[83] Nagrani, Arsha, and Zisserman, Andrew. From benedict cumberbatch to sherlock holmes: Character identification in tv series without a script. *arXiv preprint arXiv:1801.10442* (2018).

[84] Otto, Charles, Wang, Dayong, and Jain, Anil K. Clustering millions of faces by identity. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (Mar. 2017).

[85] Ozerov, Alexey, Vigouroux, Jean-Ronan, Chevallier, Louis, and Pérez, Patrick. On evaluating face tracks in movies. In *ICIP* (2013), IEEE, pp. 3003–3007.

[86] Parkhi, Omkar M, Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. A compact and discriminative face track descriptor. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 1693–1700.

[87] Parkhi, Omkar M., Vedaldi, Andrea, and Zisserman, Andrew. Deep face recognition. In *British Machine Vision Conference (BMVC)* (2015).

[88] Patron-Perez, A., Marszaek, M., Zisserman, A., and Reid, I. D. High five: Recognising human interactions in tv shows. In *British Machine Vision Conference (BMVC)* (2010).

[89] Prosser, Bryan James, Zheng, Wei-Shi, Gong, Shaogang, and Xiang, Tao. Person re-identification by support vector ranking. In *British Machine Vision Conference (BMVC)* (2010).

[90] Ramanathan, Vignesh, Joulin, Armand, Liang, Percy, and Fei-Fei, Li. Linking people in videos with their names using coreference resolution. In *European conference on computer vision* (2014), Springer, pp. 95–110.

[91] Ranjan, Rajeev, Patel, Vishal M., and Chellappa, Rama. A deep pyramid deformable part model for face detection. In *BTAS* (2015), IEEE, pp. 1–8.

[92] Redmon, Joseph, Divvala, Santosh, Girshick, Ross, and Farhadi, Ali. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 779–788.

[93] Ren, S., He, K., Girshick, R. B., and Sun, J. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems* (2015).

[94] Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 91–99.

[95] Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* (2016).

[96] Ren, Shaoqing, He, Kaiming, Girshick, Ross B., and Sun, Jian. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems* (2015), pp. 91–99.

[97] Rosenberg, Chuck, Hebert, Martial, and Schneiderman, Henry. Semi-supervised self-training of object detection models. *IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05)* (2005).

[98] Roth, M., Bauml, M., Nevatia, R., and Stiefelhagen, R. Robust multi-pose face tracking by multi-stage tracklet association. In *ICPR* (2012).

[99] Rowley, Henry A, Baluja, Shumeet, and Kanade, Takeo. Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence 20*, 1 (1998), 23–38.

[100] Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, Berg, Alexander C., and Fei-Fei, Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV) 115*, 3 (2015), 211–252.

[101] Schölkopf, Bernhard, and Smola, Alexander J. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

[102] Schütze, Hinrich. Introduction to information retrieval.

[103] Sculley, David. Web-scale k-means clustering. In *WWW* (2010), ACM, pp. 1177–1178.

[104] Sevilla-Lara, L., and Learned-Miller, E. Distribution fields for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2012).

[105] Shental, Noam, Bar-Hillel, Aharon, Hertz, Tomer, and Weinshall, Daphna. Computing Gaussian mixture models with EM using equivalence constraints. In *Advances in Neural Information Processing Systems* (2004).

[106] Shi, Jianbo, and Malik, Jitendra. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 22*, 8 (2000), 888–905.

[107] Shrivastava, Abhinav, and Gupta, Abhinav. Contextual priming and feedback for faster r-cnn. In *European Conference on Computer Vision* (2016), Springer, pp. 330–348.

[108] Shrivastava, Abhinav, Gupta, Abhinav, and Girshick, Ross. Training region-based object detectors with online hard example mining. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 761–769.

[109] Sibson, Robin. SLINK: an optimally efficient algorithm for the single-link cluster method. *The computer journal 16*, 1 (1973), 30–34.

[110] Simo-Serra, Edgar, Trulls, Eduard, Ferraz, Luis, Kokkinos, Iasonas, and Moreno-Noguer, Francesc. Fracking deep convolutional image descriptors. *CoRR, abs/1412.6537 2* (2014).

[111] Simons, Daniel J., and Levin, Daniel T. Failure to detect changes to people during a real-world interaction. *Psychonomic Bulletin & Review 5*, 4 (1998), 644–649.

[112] Singh, Krishna Kumar, Xiao, Fanyi, and Lee, Yong Jae. Track and transfer: Watching videos to simulate strong human supervision for weakly-supervised object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), vol. 1.

[113] Sivic, Josef, Everingham, Mark, and Zisserman, Andrew. “who are you?”-learning person specific classifiers from video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009), IEEE, pp. 1145–1152.

[114] Song, Yang, and Leung, Thomas. Context-aided human recognition–clustering. In *European Conference on Computer Vision* (2006), Springer, pp. 382–395.

[115] Sonntag, Daniel, Barz, Michael, Zacharias, Jan, Stauden, Sven, Rahmani, Vahid, Fóthi, Áron, and Lőrincz, András. Fine-tuning deep cnn models on specific ms coco categories. *arXiv preprint arXiv:1709.01476* (2017).

[116] Srivastava, Nitish, Hinton, Geoffrey E, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdi-nov, Ruslan. Dropout: a simple way to prevent neural networks from overfitting. In *Journal of Machine Learning Research* (2014), vol. 15.

[117] Stalder, Severin, Grabner, Helmut, and Van Gool, Luc. Cascaded confidence filtering for improved tracking-by-detection. In *European Conference on Computer Vision* (2010), Springer, pp. 369–382.

[118] Sun, Xudong, Wu, Pengcheng, and Hoi, Steven CH. Face detection using deep learning: An improved faster rcnn approach. *arXiv preprint arXiv:1701.08289* (2017).

[119] Sung, Kah-Kay, and Poggio, Tomaso. Learning and example selection for object and pattern detection. *MIT* (1994).

[120] Sutton, Charles, and McCallum, Andrew. *An introduction to conditional random fields for relational learning*, vol. 2. Introduction to statistical relational learning. MIT Press, 2006.

[121] Tang, Kevin, Ramanathan, Vignesh, Fei-Fei, Li, and Koller, Daphne. Shifting weights: Adapting object detectors from image to video. In *Advances in Neural Information Processing Systems* (2012), pp. 638–646.

[122] Tang, Xu, Du, Daniel K., He, Zeqiang, and Liu, Jingtuo. Pyramidbox: A context-assisted single shot face detector. In *European Conference on Computer Vision (ECCV)* (September 2018).

[123] Tapaswi, Makarand, Bäuml, Martin, and Stiefelhagen, Rainer. knock! knock! who is it? probabilistic person identification in tv-series. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2012), IEEE, pp. 2658–2665.

[124] Tapaswi, Makarand, Çağn Çörez, Cemal, Bäuml, Martin, Ekenel, Hazim Kemal, and Stiefelhagen, Rainer. Cleaning up after a face tracker: False positive removal. In *ICIP* (2014).

[125] Tapaswi, Makarand, Law, Marc T, and Fidler, Sanja. Video face clustering with unknown number of clusters. In *Proceedings of the IEEE International Conference on Computer Vision* (2019), pp. 5027–5036.

[126] Tapaswi, Makarand, Parkhi, Omkar M., Rahtu, Esa, Sommerlade, Eric, Stiefelhagen, Rainer, and Zisserman, Andrew. Total cluster: A person agnostic clustering method for broadcast videos. In *ICVGIP* (2014).

[127] van der Maaten, Laurens, and Hinton, Geoffrey. Visualizing data using t-sne. In *Journal of Machine Learning Research* (2008), vol. 9.

[128] Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Ł ukasz, and Polosukhin, Illia. Attention is all you need. In *Advances in Neural Information Processing Systems 30* (2017), I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., pp. 5998–6008.

[129] Vicol, Paul, Tapaswi, Makarand, Castrejon, Lluis, and Fidler, Sanja. Moviegraphs: Towards understanding human-centric situations from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 8581–8590.

[130] Wagstaff, Kiri, Cardie, Claire, Rogers, Seth, Schrödl, Stefan, et al. Constrained k-means clustering with background knowledge. In *International Conference on Machine Learning* (2001).

[131] Wan, Shaohua, Chen, Zhijun, Zhang, Tao, Zhang, Bo, and Wong, Kongkat. Bootstrapping face detection with hard negative examples. *arXiv preprint arXiv:1608.02236* (2016).

[132] Wang, Xiaolong, Girshick, Ross, Gupta, Abhinav, and He, Kaiming. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 7794–7803.

[133] Wang, Xiaolong, and Gupta, Abhinav. Videos as space-time region graphs. In *European Conference on Computer Vision (ECCV)* (2018), pp. 399–417.

[134] Wang, Xiaolong, Shrivastava, Abhinav, and Gupta, Abhinav. A-fast-rcnn: Hard positive generation via adversary for object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[135] Wang, Yitong, Ji, Xing, Zhou, Zheng, Wang, Hao, and Li, Zhifeng. Detecting faces using region-based fully convolutional networks. *arXiv preprint arXiv:1709.05256* (2017).

[136] WESTON, Jason. Large-scale semi-supervised learning.

[137] Wikipedia contributors. Shot (filmmaking) — Wikipedia, the free encyclopedia, 2019. [Online; accessed 02-November-2019].

[138] Wu, Baoyuan, Lyu, Siwei, Hu, Bao-Gang, and Ji, Qiang. Simultaneous clustering and tracklet linking for multi-face tracking in videos. In *IEEE International Conference on Computer Vision (ICCV)* (2013), pp. 2856–2863.

[139] Wu, Baoyuan, Zhang, Yifan, Hu, Bao-Gang, and Ji, Qiang. Constrained clustering and its application to face clustering in videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2013), pp. 3507–3514.

[140] Xiao, Shijie, Tan, Mingkui, and Xu, Dong. Weighted block-sparse low rank representation for face clustering in videos. In *European Conference on Computer Vision (ECCV)* (2014), Springer, pp. 123–138.

[141] Xie, Saining, Sun, Chen, Huang, Jonathan, Tu, Zhuowen, and Murphy, Kevin. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *European Conference on Computer Vision (ECCV)* (2018), pp. 305–321.

[142] Xu, Kelvin, Ba, Jimmy, Kiros, Ryan, Cho, Kyunghyun, Courville, Aaron, Salakhudinov, Ruslan, Zemel, Rich, and Bengio, Yoshua. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning* (2015), pp. 2048–2057.

[143] Yang, Bo, and Nevatia, Ram. An online learned crf model for multi-target tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2012), IEEE, pp. 2034–2041.

[144] Yang, Jiaolong, Ren, Peiran, Zhang, Dongqing, Chen, Dong, Wen, Fang, Li, Hongdong, and Hua, Gang. Neural aggregation network for video face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 4362–4371.

[145] Yang, Shuo, Luo, Ping, Loy, Chen Change, and Tang, Xiaoou. From facial parts responses to face detection: A deep learning approach. In *IEEE International Conference on Computer Vision (ICCV)* (2015), pp. 3676–3684.

[146] Yang, Shuo, Luo, Ping, Loy, Chen Change, and Tang, Xiaoou. WIDER FACE: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

[147] Yang, Shuo, Luo, Ping, Loy, Chen Change, and Tang, Xiaoou. Wider face: A face detection benchmark. In *CVPR* (2016).

[148] Yu, Jiahui, Jiang, Yuning, Wang, Zhangyang, Cao, Zhimin, and Huang, Thomas. Unitbox: An advanced object detection network. In *Proceedings of the 2016 ACM on Multimedia Conference* (2016), ACM, pp. 516–520.

[149] Zhang, Kaipeng, Zhang, Zhanpeng, and Li, Zhifeng. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters 23*, 10 (Oct 2016), 1499–1503.

[150] Zhang, Liliang, Lin, Liang, Liang, Xiaodan, and He, Kaiming. Is faster r-cnn doing well for pedestrian detection? In *European Conference on Computer Vision* (2016), Springer, pp. 443–457.

[151] Zhang, Ning, Paluri, Manohar, Taigman, Yaniv, Fergus, Rob, and Bourdev, Lubomir. Beyond frontal faces: Improving person recognition using multiple cues. In *arXiv:1501.05703* (2015).

[152] Zhang, Shanshan, Benenson, Rodrigo, Omran, Mohamed, Hosang, Jan, and Schiele, Bernt. How far are we from solving pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 1259–1267.

[153] Zhang, Shifeng, Zhu, Xiangyu, Lei, Zhen, Shi, Hailin, Wang, Xiaobo, and Li, Stan Z. S3fd: Single shot scale-invariant face detector. *arXiv preprint arXiv:1708.05237* (2017).

[154] Zhang, Tian, Ramakrishnan, Raghu, and Livny, Miron. Birch: an efficient data clustering method for very large databases. In *SIGMOD* (1996), ACM.

[155] Zhang, Zhanpeng, Luo, Ping, Loy, Chen Change, and Tang, Xiaoou. Joint face representation adaptation and clustering in videos. In *European Conference on Computer Vision (ECCV)* (2016), pp. 236–251.

[156] Zhao, Rui, Ouyang, Wanli, and Wang, Xiaogang. Unsupervised salience learning for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2013), pp. 3586–3593.

[157] Zheng, Liang, Yang, Yi, and Hauptmann, Alexander G. Person re-identification: Past, present and future. *arXiv preprint arXiv:1610.02984* (2016).

[158] Zhu, Chunhui, Wen, Fang, and Sun, Jian. A rank-order distance based clustering algorithm for face tagging. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2011).

[159] Zitnick, C Lawrence, and Dollár, Piotr. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision* (2014), Springer, pp. 391–405.