Robust Character Labeling in Movie Videos: Data Resources and Self-supervised Feature Adaptation

Krishna Somandepalli *Member, IEEE*, Rajat Hebbar and Shrikanth Narayanan *Fellow, IEEE*

***Abstract*—Robust face clustering is a vital step in enabling computational understanding of visual character portrayal in media. Face clustering for long-form content is challenging because of variations in appearance and lack of supporting large-scale labeled data. Our work in this paper focuses on two key aspects of this problem: the lack of domain-specific training or benchmark datasets, and adapting face embeddings learned on web images to long-form content, specifically movies. First, we present a dataset of over 169,000 face tracks curated from 240 Hollywood movies with weak labels on whether a pair of face tracks belong to the same or a different character. We propose an offline algorithm based on nearest-neighbor search in the embedding space to mine hard-examples from these tracks. We then investigate triplet-loss and multiview correlation-based methods for adapting face embeddings to hard-examples. Our experimental results highlight the usefulness of weakly labeled data for domain-specific feature adaptation. Overall, we find that multiview correlation-based adaptation yields more discriminative and robust face embeddings. Its performance on downstream face verification and clustering tasks is comparable to that of the state-of-the-art results in this domain. We also present the SAIL-Movie Character Benchmark corpus developed to augment existing benchmarks. It consists of racially diverse actors and provides face-quality labels for subsequent error analysis. We hope that the large-scale datasets developed in this work can further advance automatic character labeling in videos. All resources are available freely at https://sail.usc.edu/~ccmi/multiface.**

***Index Terms*—video character labeling; self-supervision; multiview correlation; triplet loss; face diarization, face clustering, computational media understanding**

I. Introduction

MEDIA is created by humans, for humans: to tell stories that educate, entertain, inform, or call us to action. When we watch a TV series or a movie, the onscreen characters shape our point of view by providing a window into the narrative. These characters advance the plot and play a vital role in effective storytelling [1]. Advances in machine learning can help automatically identify the "*who, where*, and *when*" in a story and help build a computational understanding of character representations, portrayal, and behavior in media content [2]. Such research methods and tools can also contribute to other application domains such as understanding social interactions in video [3], automatic video captioning [4] and developing computational narratology [5].

Character-level analysis of media content is of interest to a variety of stakeholders – from content creators and curators to engineers, media scholars and consumers. Consider, for example, video streaming platforms, which are able to tailor recommendations based on the cast of characters and the settings in which they appear [6]. Another example, particularly for content curators, is the *X-ray* feature [7] on Amazon's Prime Video platform. Aimed at viewers that want to learn more about who they are watching, X-ray, among other things, identifies the cast and characters in some Prime Video content, enriching overall user experience. The automated tools also allow media scholars and content creators to conduct large-scale studies of media trends such as in TV shows, movies, and advertisements, using character on-screen presence to shed light on a variety of relevant topics such as diversity and inclusion in representation and portrayals [2], [8].

A first step toward developing such technology is the ability to automatically identify the characters in the visual modality, i.e., characterize the *who* in a video. This is a challenging task because of both the wide variability within and across individuals and their portrayals across contexts, and due to the rich variety in the design of different media forms (e.g., automatic character labeling in animated content [9]). Media forms also vary with respect to duration, style, and production quality. For example, traditional feature-length movies versus micro-videos on platforms such as TikTok [10], [11]. Micro-videos are typically short-form, realistic videos unlike movies which are long-form and highly produced. In this work, we focus on live-action movie content where a person's face is used as the primary signal of identification. This is typically achieved through a two-step process: i) face *detection* to localize the face of a person in a frame, ii) clustering the detected faces to *recognize* a person irrespective of where and how they appear in the content. Recent advances in face detection can localize faces with near-perfect precision, even in extreme conditions of illumination and pose [12]. Similarly, advances in learning face representations (embeddings) and rich open-source face datasets (e.g., [13], [14]) have provided powerful frameworks to identify a person by their face.

However, face recognition in videos in the absence of domain-matched training data remains a challenging problem. We need to robustly identify the characters regardless of changes in appearance, background imagery, facial expression, size (resolution), view points (pose), illumination, partial detection (occlusion), and in some cases, age [15]. Figure 1 highlights the variability in the appearance of characters making the character labeling task challenging in the presence of visual distractors. The task is further complicated in long-form content such as movies, where characters occur at varying frequencies and suitable exemplars of actors playing them are not always available. In this setup, effective face recognition

Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA, 90089; K. Somandepalli and R. Hebbar contributed equally to this work.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3096155, IEEE Transactions on Multimedia

Fig. 1. Challenging instances of faces detected in a movie for character labeling task. The example shows the prominent characters from a 2016 Academy award winning movie *Hidden Figures*. Face quality labels associated with each track are also shown to tag some of the visual distractors. The images in the first column are character label exemplars taken from the IMDb page: www.imdb.com/title/tt4846340

not only requires face embeddings which remain robust to visual distractors, but possibly unsupervised methods such as clustering to accurately identify every character in a movie. This is the main focus of this work with Hollywood movie videos as our exemplary application domain of interest.

Unlike photo albums or image datasets, the temporal nature of videos can be used to group time-consecutive faces detected from the same person into *face tracks*, using simple heuristics based on the detection overlap [16] agnostic to the face identity. Thus, face recognition in video is generally performed at the *track-level*. Face tracking in video content such as movies—where there are typically multiple characters appearing in a scene—also offers *self-supervised* means of mining pairs of faces that belong to the same person (all faces within a track) and faces that belong to different people (all faces co-occurring in a frame). This process exploits the co-occurring nature of faces (both spatially and temporally) inherently present in videos such as movies and does not require additional supervision; hence, the term *self-supervised*. In the context of video face clustering, several past studies have shown self-supervision frameworks to be effective in learning robust domain-specific face embeddings [17], [18] as well as in improving face clustering [19], [20].

In this paper, we use self-supervision toward two critical aspects of face clustering in long-form content: addressing the lack of domain-matched training data and adapting deep face embeddings learned from static images to movie face tracks. First, we present a large-scale weakly labeled dataset that we curated by mining instances of spatially and temporally co-occurring face tracks in movies, called *SAIL-MultiFace*. Second, we present an offline method based on nearest-neighbor search to identify challenging cases in the weakly labeled data: that is, faces belonging to the same person which are "far apart" (*hard-positives*) and faces of different people that are "close together" (*hard-negatives*) in the embedding space. Then, to improve the overall face recognition in the movie domain using weakly labeled data, we explore triplet loss [13], [17] and multiview correlation [21] to adapt general-purpose face embeddings to the movie domain.

Finally, we present the *SAIL-Movie Character Benchmark (SAIL-MCB)*, a resource developed to evaluate and compare the performance of face clustering methodologies for movies. For this purpose, we considered movie/TV videos evaluated in other studies [17], [22], and in our own recent study [20]. Finally, to ensure that this benchmark is representative and inclusive of all actors in movies, we included two other movies with a more racially diverse cast of characters. For all six videos, we annotated nearly 10,000 face tracks with over half a million faces with character labels of both primary and secondary actors. We also annotated face quality labels (See an example in Figure 1) to understand the performance of face recognition frameworks in the presence of visual distractors. Our experimental results with face verification and clustering, and subsequent error analysis, demonstrate the benefit of using self-supervised adaptation techniques to improve character labeling in videos.

The rest of the paper is organized as follows: In section II, we discuss the relevant literature for automatic visual character labeling in movies. We then describe the data resources we developed: SAIL-MultiFace for adaptation and the SAIL-MCB for benchmarking and evaluation in section III; followed by a discussion of the feature adaptation methods in section IV. In section V, we consider four widely successful general-purpose face embedding frameworks to evaluate face clustering in movies with and without domain adaptation.

II. Related Work

We review here relevant existing work to contextualize the two challenges of face clustering in movies central to our work: 1. The lack of movie domain-specific data resources for training/evaluation and 2. Addressing the wide inherent variability in movie content through the ideas of self-supervision for domain adaptation.

A) Face recognition data resources. One of the earliest large-scale open source datasets called Labeled Faces in the Wild (LFW), was developed in 2008 [23]. LFW consists of over 13,000 faces from 5,749 people detected using the Viola

Jones face detector [24]. With the advent of deep learning methods for more robust face detection at scale, several large datasets have been created by automatically collecting face images from the Internet. For example, the CASIA WebFace dataset [25] contains nearly half a million images from over 10,000 people. Wider Face [26], a benchmark dataset for face detection, consists of around 400,000 faces in over 32,000 search engine retrieved images. CelebFaces Attributes (CelebA) is another large-scale resource with images from more than 20,000 celebrities. More recently, the VGGFace2 dataset [14] was released with nearly 3 million faces from over 9,000 celebrities. Another notable effort in this space was led by Microsoft in creating MSCeleb-1M [27] with 10 million face images from nearly 100,000 individuals. While this dataset is no longer publicly available, this effort highlights the feasibility of curating large-scale face recognition resources. At the same time, such efforts also raise important ethical and privacy concerns; see [28] for an excellent discussion.

Unlike the image domain, there have been very few fully labeled face video sources. A notable dataset in the video domain is YouTube Faces (YTFaces, [29]) with about 3,500 face tracks from over 1,500 different people sampled from interview recordings on YouTube. MovieNet [30] is another recent addition with over 1 million character labels annotated in key frames of over 1000 movies. The availability of such resources has led to the development of several supervised deep representations (embeddings) of faces (e.g., [13], [31], [32], [33], [34]) that have proven to be powerful in characterizing and distinguishing a person's identity.

B) Deep face embeddings. Deep face embeddings are typically trained with static images mined from web search or photo albums. Two major challenges remain in using these embeddings directly for character labeling for long-form content. First, the unit of analysis in videos is a *face track*. Image-level face embeddings are typically aggregated across all faces in the track using the mean operation [35]. This may not be robust to dynamic changes of a person's face within a track as shown in [35], resulting in unreliable track-level embeddings. Second, unlike web images, depictions in videos show a person's face in a wide variety of situations (backgrounds and visual distractors), particularly prevalent in long-form content such as movies [18], resulting in domain mismatch. Thus image-domain embeddings do not often yield robust face representations for video content [17]. One approach to address this domain mismatch is to train or adapt image-level face embeddings using labeled data from the domain-of-interest: in our case, perhaps using YTFaces dataset. However, for training deep learning models, such datasets are relatively small in size (3,500 face tracks) and the source videos (e.g., video interviews of celebrities with mostly frontal facing appearance) have fewer visual distractors than what is possible and expected . Although additional domain-specific resources may be collected by manually assigning character labels to video face tracks, this process is generally time-consuming and expensive [14].

C) Character labeling in videos. Fully automatic identification of characters in video using faces has been studied for over a decade. Some earlier works [22], [36], [37] have explored aligning speaker names available in movie screenplays with subtitles to obtain character labels at a timestamp. Character identification was then framed as a matching problem of the faces detected in a movie frame with the names extracted from screenplays. While effective in identifying at least the speaking/named characters in a video, these methods fail to scale up or generalize due to the limited access to final production screenplays as well as inaccurate time-alignment with the subtitles [38], [39]. Full-body person recognition was also studied for character identification in videos [40]. However, movies often use close-up shots of characters and show them in different attires [41], [42] throughout the movie, making the application of full-body recognition to this domain more challenging. Audio-visual character labeling methods, particularly for movies, have been somewhat less successful (e.g., [43], [44]); primarily because such efforts have mostly focused on identifying speaking characters (active speaker labeling [45], [46] and fail to identify nonspeaking characters.

Using external metadata for supervised matching of characters was also explored. A prominent example used IMDb images for TV series as labels for character labeling [47]. Such methods fail to generalize for movies because of the mismatch between the appearance of an actor's face on sources such as IMDb and their appearance in a movie (for example, the effect of makeup [48] or age [15]). Figure 1 illustrates an example of this mismatch in one of the movies in our dataset. The differences between an actor's appearance on the IMDb-curated image versus the actual appearance in the movie, are very noticeable. Whereas, in the present work, we focus on accurately identifying all faces belonging to a character using unsupervised methods. If necessary, the exemplars of the resulting clusters can be mapped to actors from casting lists with minimal manual effort. Thus, in order to robustly represent faces in movies, we explore self-supervision for domain adaptation. Self-supervision based research in this domain can be broadly categorized along two directions: (1) mining weakly labeled data from video content and (2) adapting face image embeddings for the domain-of-interest.

D) Self-supervision to collect weakly labeled data. Local tracking of faces detected in a video acts as high precision clustering to identify faces that *must* belong to the same person, and faces that *cannot* belong to the same person (when multiple faces co-occur in a frame); generating *must-link* and *cannot-link* constraints respectively [49]. Thus, without additional signals such as subtitles, metadata, or speaker labels, weakly labeled faces can be readily mined in movie content, agnostic to the character identity [20]. In the same spirit, we curate over 169,000 face tracks with weak labels from 240 movies to generate domain-matched data for feature adaptation.

While we were motivated by the success of face-tracking to mine a large number of weakly labeled tracks from movies, we used distance based methods to further identify challenging samples for downstream adaptation tasks. Recently, two methods, track-supervised Siamese network (TSiam) and self-supervised Siamese network (SSiam) were proposed in [18] which—besides face tracking—relied on the Euclidean distances in the embedding space to generate similar

and dissimilar face tracks. Similarly, we propose a nearest neighbor-based approach to further segment the curated tracks into smaller "tracklets". Hard-positives were then identified between tracklets with maximal distance, and hard-negatives, between cannot-link tracklets with minimal distance in the embedding space. This method is entirely offline and avoids the need for complex online hard-example mining, common in triplet-loss based systems [13].

E) Self-supervised feature adaptation. The weak labels generated from must-link and cannot-link constraints have been successfully used for feature adaptation using metric learning. Unsupervised logistic discriminative learning (ULDML [19]) was proposed to learn a metric such that must-link faces are closer to each other and cannot-link faces are further apart in the feature space. A Siamese network was trained with contrastive-loss using such weakly labeled data for face verification task in [50] and using triplet-loss in the development of FaceNet [13]. In the context of track-level face clustering in videos, similar and dissimilar faces were used to train face embeddings using a Siamese network with contrastive loss [51]. Triplet-loss was shown to generally perform better than contrastive loss for face recognition in videos [52]. An improved triplet (ImpTriplet) loss was proposed in [17] that performed better than the traditional triplet loss for track-level feature adaptation by additionally constraining the distance of the must-link pairs. Thus, in our work, we evaluated ImpTriplet to adapt face embeddings using the hard-positive and hard-negative tracklets automatically gathered from movies.

Unlike contrastive/triplet loss formulation which needs a negative sample, multiview methods such as canonical correlation analysis (CCA, [53]) and their deep variants [54] can be used to learn the shared information between a pair of positive samples; in our case, learning the shared character identity from different appearances (views) of a person. While deep CCA was applied effectively for face recognition [55] and reconstruction [56], multiview learning methods in general have not been explored for face clustering in videos. Perhaps the closest in this context is using linear discriminant analysis (LDA) for face recognition in videos [57]. But, unlike CCA-based methods, LDA needs labels for all classes and just weakly-labeled data is inadequate. In a recent work, we developed a neural-based approach called multiview correlation (MvCorr [58]) to generalize CCA for more than two views where all we know is that a set of observations come from the same source. In the domain of speaker recognition, we showed that MvCorr offers state-of-the-art performance for speaker clustering [58] capturing information regarding the person's identity irrespective of the spoken utterance. Analogous to speaker recognition, the hard-positive tracklets we extract from face tracks readily include faces of the same person in different views with respect to pose, illumination and occlusion. Thus, for feature adaptation experiments with weakly labeled data, we explore both ImpTriplet and MvCorr frameworks.

A. Benchmark datasets for movie face clustering.

The overarching goal of face clustering in movies is to identify the characters wherever and whenever they appear throughout the video. In the domain of movie video analysis in particular, there are very few open-source datasets available till date to benchmark related tasks. This is in part because feature-length movie videos are longer in duration compared to trailers and other video clips (e.g., YTFaces [29]). Large-scale movie datasets such as MovieNet [30] only provide annotations on key frames of movie videos, limiting the use of local face tracking. Labeling characters throughout the content requires expensive manual effort and can be time intensive.

Other benchmark datasets have episodes of TV series: *Buffy the vampire slayer* [22], [59], *Big Bang Theory* [60] and *Sherlock Holmes* [61]. However, TV episodes are generally shorter compared to movies and do not have as much variability with respect to the appearance of characters and the backgrounds or situations in which they appear. In the movie domain, a few widely used examples include the movies *Casablanca* and *American Beauty* compiled in [62], *Notting Hill* [63] and more recently ACCIO [15] which is a dataset of the *Harry Potter* movies collected with a focus on learning age-invariant face representations. In our recent work, we released labels for two other movies adding to these resources [20]. These movies and TV episodes mostly include white actors in prominent roles and are not entirely representative of the growing and desired trend of diverse casting in Hollywood (see reports [64], [65]). In this work, we address this limitation by developing a benchmark dataset that includes two movies with a more racially diverse casting. Together, we hope that these resources can enable a robust evaluation of automatic character labeling methods in the movie domain.

III. Data Resources

In this section we describe the two data resources we developed: (1) the SAIL Movie Character Benchmark (SAIL-MCB) and (2) SAIL-MultiFace: weakly labeled face tracks for training or adapting general purpose face embeddings. All movies were purchased in house.

A. SAIL-MCB: SAIL Movie Character Benchmark Dataset

We started with two widely used benchmark videos: a movie, *Notting Hill* (**NH**), and an episode from season 5 of the TV series, *Buffy the Vampire Slayer* (**BFF**. **NH** and **BFF** have been used to evaluate video face clustering, both online (e.g., [66]) and offline (e.g., [67], [68], [69]) methods. Although labels are publicly available for these videos, we repeated the annotation process to improve the overall coverage of the character labels: our system detected significantly more number of faces than those in recent reports. For example, Sharma et. al [18] reported 39,263 faces detected for **BFF**; compare this with 8,000 more faces (See Table I, row 2) in SAIL-MCB. We also included data from two movies which we made publicly available in our recent work [20]: *Dumb and Dumber To* (**DD2**) and *Maleficent* (**MT**). These two movies were chosen to include content produced more recently compared to **NH**. In addition, we included *About Last Night* (**ALN**) and *Hidden Figures* (**HF**) to include a more racially diverse cast of actors.

TABLE I
DETAILS OF MOVIE CHARACTER BENCHMARK (SAIL-MCB) DATASET. THE NUMBER OF CHARACTERS IN A GIVEN MOVIE THAT WERE CHOSEN TO BE LABELED ENSURED A COVERAGE OF AT LEAST 99% OF THE DETECTED FACES. THE RANGE OF NUMBER OF TRACKS-PER-CHARACTER SHOWS THAT WE LABEL BOTH PROMINENT AND MINOR CHARACTERS BASED ON THE AMOUNT OF THEIR APPEARANCE.

	Movie (year)	No. faces	No. tracks	No. faces-per-track mean ± std	No. characters	No. tracks-per-character (min, max)	% frontal face tracks
ALN	About Last Night (2014)	70,210	1,880	38.2 ± 34.1	10	(16, 491)	41.1
BFF	Buffy, the Vampire Slayer (2000)	47,870	634	66.3 ± 69.4	12	(17, 112)	16.3
DD2	Dumb and Dumber To (2014)	152,908	2,361	70.8 ± 66.3	10	(12, 557)	24.9
HF	Hidden Figures (2016)	101,438	1,902	59.8 ± 61.1	24	(8, 283)	47.3
MT	Maleficent (2014)	53,450	875	64.1 ± 57.9	10	(14, 254)	28.2
NH	Notting Hill (1999)	154,625	2,121	79.2 ± 84.5	12	(18, 585)	21.9

For all six videos, we first perform face detection and local tracking using Google's API obtained with an academic license. The face tracking used here was developed using simple heuristics based on analyzing the intersection-of-union of the successive bounding boxes of the detected faces [70]. A summary of the number of faces detected and face tracks as well as the density of faces per track is shown in Table I. We used a face track as the unit of annotation for the movie videos. The process included two tasks: character labeling and face quality labeling.

Character labeling: We used the names of the actors listed on the IMDb casting page corresponding to each movie/TV series as character labels. These labels were manually assigned to each face track by two human annotators independently and ties were resolved by a third person, thereby ensuring high annotation agreement. The number of characters labeled was not fixed across the movies and chosen to cover at least 99% of all faces (not face tracks) detected in each video. This was another reason why we re-annotated **NH** and **BFF** in house. Recent studies had only labeled for 5 characters (in **NH** [35]) or 6 characters (in **BFF** [35], [18]); compare this with 12 characters for both videos in SAIL-MCB.

As summarized in Table I, most movies only needed 10 or 12 character labels to cover 99% of the faces detected. Hidden Figures (HF) had the largest number of characters at 24. The number of face tracks per character varied widely within a movie. For example, in **HF**, the least frequent character has only eight face tracks while the most frequent character has 283 tracks (See Table I). Thus, we also pick up on some characters in a minor role besides the commonly appearing characters (lead/co-leads).

Face quality labeling: In order to facilitate a detailed performance and error analysis of face clustering methods on SAIL-MCB, we also obtained face quality labels for different visual distractors while labeling face tracks with their character IDs. These qualitative labels were annotated along six dimensions at the track level: (1) whether *all* faces in the track are facing the viewer (frontal:*F*); (2) at least one face in the track is shown facing sideways (profile; *P*); (3) at least one face in the track is blurry (*B*); (4) at least one face in the track is shown wearing glasses *(G)*; (5) at least one face in the track is only partially visible or occluded *(O)*; and (6) at least one face in the track is poorly lit *(L)*. These questions were presented sequentially to the annotators and they were instructed to tag with more than one label where applicable.

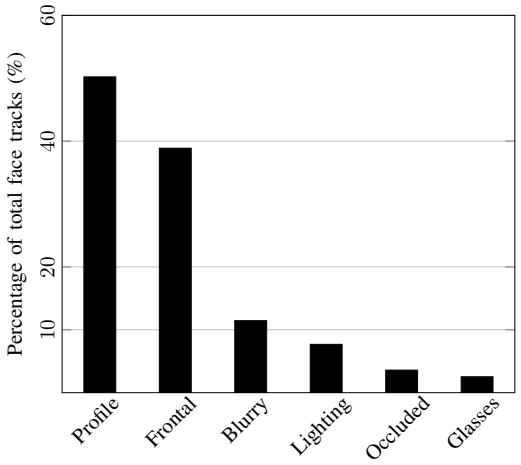

Fig. 2. Distribution of face quality labels in SAIL-MCB at the track-level. Over 50% of the face tracks were labeled as "Profile" which means that at least one face in the track was shown posing sideways.

A few exemplars of these labels are shown in Figure 1. The distribution of the quality labels across all videos in SAIL-MCB is shown in Fig. 2. In this figure, notice that the total does not sum to 100 as a single face track can have multiple distractor labels. Over 50% of the face tracks had at least on profile face. Additionally, we found that on average, only 30% of all face tracks were completely frontal facing in our dataset (See Table I). These annotations along with the track-level character labels have been made publicly available.

B. SAIL-MultiFace: Harvesting Weakly Labeled Face Tracks

Our methodology for gathering weakly labeled data from movies includes two steps: (1) Face tracking and preprocessing to generate must-link and cannot-link pairs of face tracks and (2) hard-example mining in the embedding space to identify "difficult" tracklets. We call this dataset *MultiFace* as it includes faces corresponding to multiple views of the same person in different poses or multiple settings or faces of different people in the same setting.

The spatial and temporal co-occurrence patterns of people in a movie scene can be used to generate associations of *must-link* and *cannot-link* constraints i.e., faces in a track *must* belong to the same person and multiple faces in a frame *cannot* belong to the same person, respectively. To gather such data in movie videos, we considered a set of 240 movies (24 fps video frame

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3096155, IEEE Transactions on Multimedia

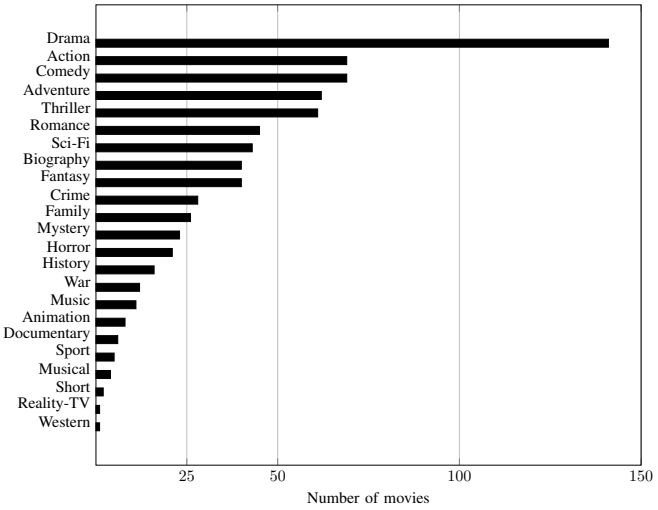

Fig. 3. Distribution of the genres for the 240 movies used for harvesting weakly labeled face tracks in SAIL-MultiFace. Movies may have multiple genres associated with them as listed on IMDb.

rate, 1280x720 resolution) released between 2014–2018 that were purchased in house.

These movies span a wide range of genres as shown in Figure 3, providing data from different movie styles. The movie titles and related details are provided in the supplementary methods, section I. We performed the following preprocessing steps to mine must-link faces and cannot link tracks from each movie:

1) Face detection and local tracking (as explained in the previous section).
2) Filter out face tracks which have a duration of less than 1s, i.e., minimum of 24 faces in a track. This limits the search space of tracks for the next step as well as provides opportunities to extract instances of the same person appearing in different conditions such as pose, expression and lighting.
3) Only retain tracks which co-occur with other tracks. We considered two face tracks to be co-occurring if they shared at least six frames (0.25s) in common. This threshold was chosen heuristically to minimize propagating errors from tracking.

The count statistics of the total number of tracks retained at each step of the process is shown in Table II for the 240 movies used for this task. Of the tracks identified in step (2), on average, $45.1 \pm 21.2\%$ of the face tracks had at least one co-occurring track. This statistic is consistent with the range of $35-70\%$ reported by a previous work [18] that also mined co-occurring face tracks in movies.

It is important to note that the final number of 169,000 tracks after preprocessing only represents the must-link instances that provide *positive samples* (faces of the same person). In order to extract *negative samples* (faces from different persons), we need to look at all cannot-link pairs associated with them. While every face track overlapped with at least one other track, the number of overlapping tracks varied between 1–94 with an average of 5.2 ± 9.1 overlaps per track; thus resulting in a large set of negatives to choose from.

TABLE II
COUNT STATISTICS OF THE TRACKS MINED AT EACH STAGE OF THE HARVESTING PROCESS. SAMPLE SIZE OF MOVIES = 240

Statistic	Total No. faces	No. tracks	No. tracks/movie mean $\pm$ std.
Track length $\geq$ 1s	23.2M	335845	1389.9 $\pm$ 760.3
Co-occurring tracks	10.2M	**169201**	726.2 $\pm$ 704.4

In such cases, past face clustering studies have emphasized the need for *hard-example* mining to not only reduce the search space but also improve the robustness of face representations to visual distractors (e.g., [34]). The goal here is to identify samples belonging to the same person that appear very different (hard-positives) and samples belonging to different persons that look similar to each other (hard-negatives). In the next section, we discuss the development of a hard-example mining approach for our use case.

C. Mining Hard-positive and Hard-negative Tracklets

Hard-example mining has been studied extensively in the computer vision literature for tasks such as object recognition (e.g., [71]). In the context of face clustering, mining hard-positive and hard-negative *faces* in an embedding space can be easily achieved by identifying the pairs of face samples that yield maximum and minimum distance respectively [18]. However, face clustering is performed at the track-level where we typically average the embeddings of all the faces in a track to provide a robust representation. Thus, we propose a nearest-neighbor based approach to identify the hard-positive and hard-negative *tracklets*.

The pseudocode for the proposed method is detailed in Algorithm 1. Let us assume that we have a semantic embedding space for the face tracks (e.g., VGGFace2) denoted by $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N] \in \mathbb{R}^{d \times N}$ with d-dimensional embeddings for N faces in the track. As described in the function `HardPositiveMining()`, we segment each track into multiple tracklets to form a hard-positive set. First, we find the pair of embeddings in $\mathbf{H}$ that are maximally distant from each other and assign one of them to be the *anchor* face $\mathbf{h}_a$ (line 3). Then k-nearest neighbors of $\mathbf{h}_a$ are accrued and averaged to form the corresponding *anchor tracklet* $\mathbf{v}_a$ (`NNTracklet`$_k$`()`, line 4) After fixing the anchor $\mathbf{h}_a$, we mine the hard-positive set $\mathcal{V}_p : \{\mathbf{h}_p^{(i)} | i \in [1, M-1]\}$ iteratively (lines 5–9) as follows:

(1) Find $\mathbf{h}_p$ farthest from $\mathbf{h}_a$ from the columns remaining in $\mathbf{H}$ after removing the nearest neighbors of $\mathbf{h}_a$ (line 7)

(2) Average the k-nearest neighbors of $\mathbf{h}_p$ to get its tracklet $\mathbf{v}_p$ (line 8)

(3) Remove these nearest neighbors from the matrix $\mathbf{H}$.

Repeat these steps $M-1$ times to obtain the set of hard-positive tracklets $\{\mathbf{v}_p^{(i)} | i \in [1, M-1]\}$ for anchor $\mathbf{v}_a$. The variable M is controlled by choosing an appropriate value for the parameter k in the nearest neighbor search (see **Parameters** in Algorithm 1). For example, if $M = 5$ from a track with N faces, we set $k = \lfloor \frac{N}{5} \rfloor$. This choice of k ensures that all tracklets are of same length and that most faces in a track

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3096155, IEEE Transactions on Multimedia

Algorithm 1: Mining Hard-positive and Hard-negative tracklets using Nearest Neighbor Search

Input: The d-dimensional embeddings of a face-track $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N] \in \mathbb{R}^{d \times N}$; Associated set of C cannot-link tracks $\mathcal{C} : \{\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(C)}\}$
Output: Anchor tracklet $\mathbf{v}_a$ with its hard-positive tracklets $\mathcal{V}_p : \{\mathbf{v}_p^{(1)}, \ldots, \mathbf{v}_p^{(M-1)}\}$ and hard-negative tracklet $\mathbf{v}_q$
Parameters: Number of tracklets per track M and nearest-neighbor parameter $k = \lfloor N/M \rfloor$
Initialize: The set of must-link tracklets including hard-positives $\mathcal{V}_p = \{\}$ and cannot-link tracklets $\mathcal{V}_q = \{\}$

1
2 **Function** `HardPositiveMining(`$\mathbf{H}$, M`)`**:**
3 $\quad \mathbf{h}_a = \arg\max_{a \in [1,N]} \|\mathbf{h}_a - \mathbf{h}_b\|^2$ // Find anchor $\mathbf{h}_a$ from pairwise distances in $\mathbf{H}$
4 $\quad (\mathbf{v}_a \in \mathbb{R}^d, \quad \mathbf{H}' \in \mathbb{R}^{d \times N-k}) \leftarrow$`NNTracklet`$_k(\mathbf{h}_a, \mathbf{H}, k)$ // Find anchor tracklet
5 $\quad$ **while** $|\mathcal{V}_p| < M$ *and* $|\mathbf{H}'\{:\}| > 0$ **do**
6 $\quad\quad \mathbf{H} \leftarrow \mathbf{H}'$
7 $\quad\quad \mathbf{h}_p = \arg\max_i \|\mathbf{h}_a - \mathbf{H}\{:, i\}\|^2$ // All hard-positives are mined with respect to the anchor $\mathbf{h}_a$
8 $\quad\quad (\mathbf{v}_p, \quad \mathbf{H}') \leftarrow$`NNTracklet`$_k(\mathbf{h}_p, \mathbf{H}, k)$
9 $\quad\quad \mathcal{V}_p \leftarrow \mathcal{V}_p \cup \{\mathbf{v}_p\}$ // Find hard-positives iteratively
10 $\quad$ **return** $(\mathbf{v}_a, \mathcal{V}_p)$
11
12 **Function** `HardNegativeMining(`$\mathbf{H}$, $\mathcal{C}$`)`**:**
13 $\quad (\mathbf{v}_a, \_) \leftarrow$`HardPositiveMining(`$\mathbf{H}$, M`)` // Find anchor tracklet as reference
14 $\quad$ **while** $|\mathcal{V}_q| < MC$ **do**
15 $\quad\quad c = |\mathcal{V}_q|/M + 1$ // Iterate through all the cannot-link tracks
16 $\quad\quad (\mathbf{v}'_a, \mathcal{V}') \leftarrow$`HardPositiveMining(`$\mathbf{H}^{(c)}$,$M$`)` // Segment cannot-link tracks into tracklets
17 $\quad\quad \mathcal{V}_q \leftarrow \mathcal{V}_q \cup \mathcal{V}' \cup \{\mathbf{v}'_a\}$ // Collect tracklets from all cannot-link tracks
18 $\quad \mathbf{v}_q = \arg\min_{\mathbf{v} \in \mathcal{V}_q} \|\mathbf{v}_a - \mathbf{v}\|^2$ // Find the closest cannot-link tracklet to anchor face
19 $\quad$ **return** $\mathbf{v}_q$
20
21 **Function** `NNTracklet`$_k$`(`$\mathbf{q}$, $\mathbf{X}$, k`)`**:**
22 $\quad \mathbf{Y} \in \mathbb{R}^{d \times k} \leftarrow$`knn(`$\mathbf{q}$, $\mathbf{X}$, k`)` // Find k-nearest neighbors of the query $\mathbf{q} \in \mathbb{R}^d$ from the matrix $\mathbf{X} \in \mathbb{R}^{d \times N}$
23 $\quad \mathbf{u} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{Y}\{:, i\}$ // Average nearest neigbors to form tracklet embedding
24 $\quad \mathbf{X} \leftarrow \mathbf{X}\{:, :\} - \mathbf{Y}\{:, i\}$ // Remove the nearest neighbor columns from the original matrix $\mathbf{X}$
25 $\quad$ **return** $(\mathbf{u}, \mathbf{X})$

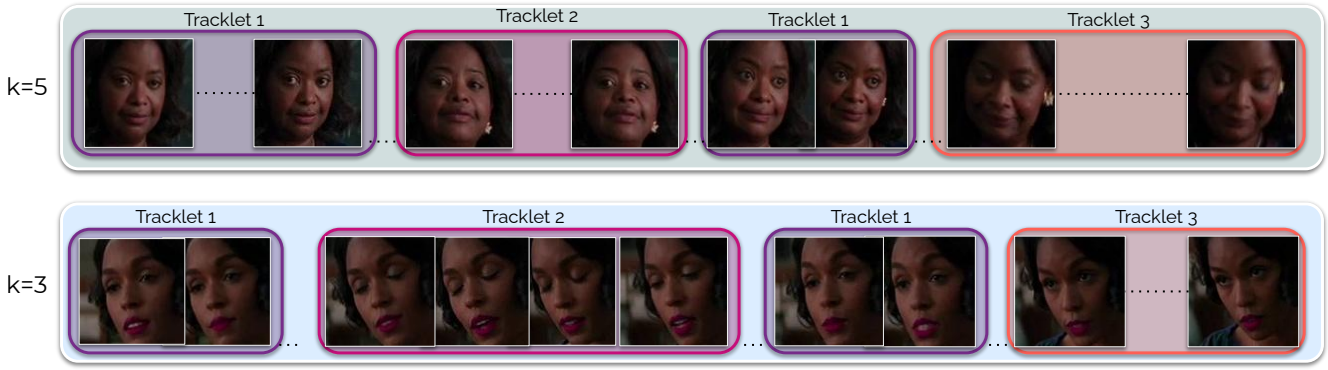

Fig. 4. Example of hard-positive tracklets resulting from our proposed method with the nearest neighbor parameter $k = 3, 5$. Each color indicates one tracklet. Notice that they differ from each other with respect to face orientation (e.g., Row 1: Tracklet 1 vs. Tracklet 2) or with eyes open/closed (e.g., Row 2: Tracklet 2 vs. Tracklet 3). A single hard-positive tracklet can be formed from faces that may not appear in a sequential order allowing us to mine *harder* positives (see Tracklet 1). The pair of cannot-link face tracks shown here are from the movie *Hidden Figures* (2016) at time 00 : 11 : 05.

are used up. We used the KDTree algorithm implementation in scikit-learn [72] for nearest neighbor search. An example of the resulting hard-positive tracklets is shown Figure 4. Notice how they differ from each other with respect to face orientation and eyes closed/open. An interesting consequence of using k-nearest neighbors in our approach—particularly without time-contiguous constraints—is that we can form tracklets with faces that are similar in the embedding space but may not be sequential in a face track, as highlighted in Figure 4.

Our proposed hard-negative mining is detailed in the function `HardNegativeMining()`. It takes two inputs: the parent track $\mathbf{H}$ of anchor $\mathbf{v}_a$ and the set of all $C \geq 1$ cannot-link tracks for $\mathbf{H}$ (see Step 3 in III-B); denoted by $\mathcal{C} : \{\mathbf{H}^{(i)} | i \in [1, C]\}$. Each track in $\mathcal{C}$ is segmented into M tracklets (line 15–16) to obtain a total of MC cannot-link tracklets $\mathcal{V}_q$ (line 17). A hard-negative is then identified as the tracklet in $\mathcal{V}_q$ that has the shortest distance to the anchor $\mathbf{v}_a$ (line 18). In Fig. 4, `Row1 : Tracklet3` was identified as the anchor and `Row2 : Tracklet2` as its hard-negative. Notice that both tracklets show the actors with heads tilted similarly and eyes closed in a similar (dark) background. Throughout this algorithm, we use the normalized Euclidean distance metric to estimate distances in the embedding space. Notice that both hard-positives and hard-negatives are mined with respect to an anchor which can be directly used for feature adaptation with triplet-loss based frameworks. For all subsequent feature adaptation methods, we use the hard-positives $\{\mathbf{v}_p^{(1)}, \ldots, \mathbf{v}_p^{(M-1)}\}$ and the hard-negative $\mathbf{v}_q$ corresponding to the anchor $\mathbf{v}_a$.

Furthermore, our proposed method is *adaptive* in nature. That is, we do not need a predetermined or tuned threshold for the distance metric to identify the hard-positives and hard-negatives. Instead, we choose the number of tracklets and let the nearest neighbor search determine the threshold. This approach ensures that every movie in our dataset has a different *minimum hard-negative distance* i.e., the closest or the most similar cannot-link pair, for which feature adaptation is necessary in the embedding space. The distribution of the minimum hard-negative distance per movie in the SAIL-MultiFace dataset is shown supplementary methods, section II, Figure 1. We also discuss a few movies for which the minimum hard-negative distance is small in the embedding space. The results suggest that our proposed hard-example mining algorithm not only successfully identifies the most challenging faces to distinguish in the embedding space, but also underscores the need for feature adaptation (see supplementary methods, section II).

IV. Self-supervised Feature Adaptation

As discussed in Section II, we investigate improved triplet loss (ImpTriplet, [17]) and multiview correlation (MvCorr, [58]). Here, we review these two methods in our context of adapting general-purpose face embeddings for video face tracks using the hard-positive and hard-negative tracklets.

A. ImpTriplet: Improved Triplet Loss

ImpTriplet [17] is an advanced version of the popular triplet loss formulation. For one triplet, the original triplet loss function is defined as:

$$L_o = \frac{1}{2}\max\left\{0, D(\mathbf{v}_a, \mathbf{v}_p^{(1)}) - D(\mathbf{v}_a, \mathbf{v}_q) + \alpha\right\} \quad (1)$$

where $\{\mathbf{v}_a, \mathbf{v}_p^{(1)}\}$ and $\{\mathbf{v}_a, \mathbf{v}_q\}$ are the hard-positive and hard-negative tracklet pairs respectively and α is the distance margin (typically, $\alpha = 1$). Minimizing this loss would cause the triplet embedding to push the negative tracklet $\mathbf{v}_q$ from the anchor $\mathbf{v}_a$. However there are two key limitations in this formulation per [17]: (1) $\mathbf{v}_q$ is pushed away only from $\mathbf{v}_a$ and not both $\mathbf{v}_a$ and $\mathbf{v}_p^{(1)}$, and (2) the distance margin of the positive pair $\mathbf{v}_a$ and $\mathbf{v}_p^{(1)}$ is not specified. ImpTriplet addresses these two issues by introducing *interclass constraints* Φ and *intra-class constraints* Ψ respectively, as follows:

$$\Phi(\mathbf{v}_a, \mathbf{v}_p^{(1)}, \mathbf{v}_q) = \alpha + D(\mathbf{v}_a, \mathbf{v}_p^{(1)}) - \frac{1}{2}\big(D(\mathbf{v}_a, \mathbf{v}_q) + D(\mathbf{v}_p^{(1)}, \mathbf{v}_q)\big) \quad (2)$$

$$\Psi(\mathbf{v}_a, \mathbf{v}_p^{(1)}) = D(\mathbf{v}_a, \mathbf{v}_p^{(1)}) - \hat{\alpha} \quad (3)$$

Here, $\hat{\alpha}$ ensures that the anchor and positive pair lie within a margin ($\hat{\alpha} = 0.1$) Finally the ImpTriplet loss is given as:

$$L_s = \max\left\{0, \Phi\{\mathbf{v}_a, \mathbf{v}_p^{(1)}, \mathbf{v}_q\}\right\} + \lambda \max\left\{0, \Psi\{\mathbf{v}_a, \mathbf{v}_p^{(1)}\}\right\} \quad (4)$$

where the parameter λ balances the contribution of intra-class constraints in the modified triplet formulation ($\lambda = 0.02$). The parameters $\hat{\alpha}$ and λ were identified in [17].

B. MvCorr: Multiview Correlation

MvCorr can successfully incorporate information from more than two views without the need for negative samples or additional labels to learn discriminative embeddings [21]. For face clustering, the hard-positive tracklets containing different visual distractors can be treated as multiple views of a person's facial identity. We first discuss the loss formulation and then its application within a neural network framework.

Let T be the total number of available face tracks. Each track is segmented into M tracklets (as described in Section III-C). We can collect the d-dimensional embeddings of each tracklet as columns to form a set of M hard-positive *matrices* $\{\mathbf{V}_a, \mathbf{V}_p^{(1)}, \ldots, \mathbf{V}_p^{(M-1)}\}$ with $\mathbf{V}_* \in \mathbb{R}^{d\times T}$. The *multiview correlation* matrix $\mathbf{\Lambda}$ is the normalized ratio of the sum of between-view covariances $\mathbf{R}_b$ and sum of within-view covariances $\mathbf{R}_w$ for M views as follows:

$$\mathbf{\Lambda} = \frac{1}{M-1}\frac{\mathbf{R}_b}{\mathbf{R}_w} = \frac{\sum_{l=1}^{M}\sum_{k=1,k\neq l}^{M}\bar{\mathbf{V}}_l(\bar{\mathbf{V}}_k)^\top}{(M-1)\sum_{l=1}^{M}\bar{\mathbf{V}}_l(\bar{\mathbf{V}}_l)^\top} \quad (5)$$

where $\bar{\mathbf{V}}_* = \mathbf{V}_* - \mathbb{E}(\mathbf{V}_*)$ are mean-centered data matrices. The common scaling factor $(T-1)^{-1}M^{-1}$ in the ratio is omitted. Our objective is to estimate a shared subspace, $\mathbf{W} \in \mathbb{R}^{d\times d}$ such that the multiview correlation is maximized. Thus, the loss function can be written as:

$$\rho^{(M)} = \max_{\mathbf{W}} \frac{1}{d(M-1)}\frac{\text{Tr}(\mathbf{W}^\top\mathbf{R}_b\mathbf{W})}{\text{Tr}(\mathbf{W}^\top\mathbf{R}_w\mathbf{W})} \quad (6)$$

where $\text{Tr}(\cdot)$ denotes the trace of a matrix. The subspace $\mathbf{W}$ can be estimated by solving the generalized eigenvalue problem to simultaneously diagonalize $\mathbf{R}_b$ and $\mathbf{R}_w$. Hence, the MvCorr objective is the average of the ratio of eigenvalues of between-view and within-view covariances. In other words, if $\mathbf{R}_w$ is invertible, then we wish to find the a transformation matrix $\mathbf{W}$ that maximizes the spectral norm of $\mathbf{R}_w^{-1}\mathbf{R}_b$. Thus, maximizing the between-view variability while minimizing the within-view variability, capturing the shared information across the views. This ratio of variances formulation is similar to that in linear discriminant analysis (LDA) but without the need for class labels.

We use neural networks to optimize MvCorr for large and complex datasets similar to our past work [58]. Here, data from M hard-positive matrices is input to M corresponding " subnetworks" with identical architecture but no shared weights across them. The embedding capturing the shared information across the views is the output of the last layer of the individual networks. The MvCorr model is trained using mini-batch SGD to minimize the loss $1-\rho^{(M)}$. During inference, we only need to extract embeddings from one of the subnetworks as the last-layer activations are maximally correlated across all the subnetworks by virtue of the loss, as demonstrated in [21].

V. Experiments and Results

One of the goals in this work is to empirically study whether feature adaption using weakly labeled data can lead to robust face clustering in the movie domain. For feature adaptation, we first need to choose an embedding space. To this end, we

TABLE III
COMPARISON OF FACE VERIFICATION PERFORMANCE (TPR @ 0.1 FPR %) OF BASELINE MODELS FOR STANDARD VIDEO DATASETS. METRIC

Model / Dataset	IJB-B Aligned	IJB-B Unaligned	YTFaces
VGGFace2 [14]	97.0	97.7	**96.6**
FaceNet [13]	**98.5**	**98.7**	95.7
SphereFace [32]	94.4	52.2	69.2
PFE [34]	**98.3**	73.3	94.7

compare and evaluate the four of the best opensource frameworks proposed over the recent years [13], [14], [32], [34]. Specifically, we setup face verification experiments to evaluate which model performs the best for the video domain, followed by feature adaptation experiments on the harvested track data SAIL-MultiFace (see Sec. III-B). Finally, using the SAIL-MCB dataset, we benchmark face clustering performance with and without adaptation along with a detailed error analysis using the associated face quality labels (see Fig. 2).

A. Face Verification for video data

We refer to the face embedding frameworks we tested as *baseline models* since we do not additionally adapt them for the movie domain. We compared *FaceNet* (2015, [13]), *SphereFace* (2017, [32]), *VGGFace2* (2018, [14]) and *Probabilistic Face Embeddings* (*PFE*, 2019, [34]). While these frameworks have been extensively benchmarked against image datasets such as LFW [23], performance comparison in the video domain is lacking. Thus, we set up face verification experiments using two video face benchmark datasets: IJB-B [73] and YTFaces [29]. IJB-B is commonly used for benchmarking face-verification methods in videos. It consists of around 77,000 faces detected from 21,000 still images and 55,000 video-frames. YTFaces is widely used for face recognition and verification in video, consisting of 3425 videos of approximately 1600 unique identities. While IJB-B, acquired in relatively controlled conditions helps validate if face embeddings trained on images perform well for video, YTFaces (mined from Youtube) helps evaluate their use for videos-in-the-wild.

Baseline models. *FaceNet* [13] was trained using triplet loss and Inception-v1 architecture, for downstream tasks such as face verification and clustering.

SphereFace [32] is a metric learning method that combines the ideas of cross-entropy and angular margin loss to improve classification performance. Notably, one of the benefits of the angular margin loss used here is its ease of training compared to triplet loss-based methods.

Probabilistic face embeddings (PFE [34]) models different faces of the same person as multivariate Gaussian distributions where the mean captures information about the identity.

VGGFace2 is a ResNet-50 network trained on the large-scale dataset, also called as VGGFace2 [14] with over 9,000 face identities mined from YouTube for the task of face classification. In related work, *VGGFace2* showed state-of-the-art face verification and clustering performance for standard datasets such as IJB-B [74]. For all baseline models, we used publicly available code and models pretrained on VGGFace2 dataset. See supplementary methods, section III for preprocessing, implementation details specific to each model.

Verification setup. Most face recognition methods typically perform face alignment based on facial landmarks as a pre-processing step to align faces in different poses. In movies however, these landmarks can be difficult to detect due to occlusion, pose, etc (See Fig. 1). Hence, we evaluate the sensitivity of the different methods to alignment by performing two sets of verification experiments: on raw face images and *aligned* face images. For details on alignment, please see supplementary methods, section III. Consistent with past video face experiments, we use the 1:1 verification setup described in [14] for IJB-B. For YTFaces, we create mean track-level embeddings (without alignment) and report results averaged over the 10-fold splits[1] [29]. For all methods, we use the cosine distance between l_2-normalized embeddings as the similarity metric.

Results. The ROC curves for all verification results in IJB-B with and without face alignment and YTFaces are shown in supplementary methods, section III, Figure 2. For the purposes of comparison, we choose to report TPR @ 0.1 FPR as the evaluation metric, consistent with previous reports (for example [34]). The results are summarized in Table III. PFE and SphereFace are heavily reliant on face alignment and perform poorly on unaligned face images (columns 1–2, Table III). Without face alignment, SphereFace performed the poorest for YTFaces among the methods considered. On the other hand, both VGGFace2 and FaceNet performed consistently well on IJB-B and YTFaces. While FaceNet performed slightly better than VGGFace2 in IJB-B, overall, VGGFace2 performed the best considering that face verification is more challenging for videos in-the-wild as in YTFaces. Our findings are also comparable to the results reported elsewhere (e.g., [74]). Based on these observations, we chose VGGFace2 as the embedding space to perform movie-domain adaptation using the weakly labeled data we harvested from 240 movies (see Sec. III-B).

B. Self-supervised Feature Adaptation

The network architecture for ImpTriplet consists of three subnetworks; one each for anchor $\mathbf{v}_a$, hard-positive $\mathbf{v}_p^{(1)}$ and hard-negative $\mathbf{v}_q$. Each subnetwork is a fully-connected network (FCN) of identical architecture with shared weights across the subnetworks. We used publicly available code[2] to implement the loss as described in Sec.IV-A. Similarly, for MvCorr adaptation, we also use three subnetworks but without the need of hard-negatives. We chose $M = 3$, and used the anchor $\mathbf{v}_a$ and two hard-positives, $\mathbf{v}_p^{(1)}, \mathbf{v}_p^{(2)}$ as the three multiview inputs to the network as described in IV-B. We set the number of views M to three in order to be comparable with the ImpTriplet adaptation which uses three tracklets per face track. In MvCorr, the weights are not shared across the subnetworks, unlike the ImpTriplet model. We used our publicly released code to train MvCorr models[3]. To

[1] YTFaces evaluation splits: www.cs.tau.ac.il/wolf/ytfaces
[2] ImpTriplet code: github.com/manutdzou/Strong_Person_ReID_Baseline
[3] MvCorr code: github.com/usc-sail/gen-dmcca

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3096155, IEEE Transactions on Multimedia

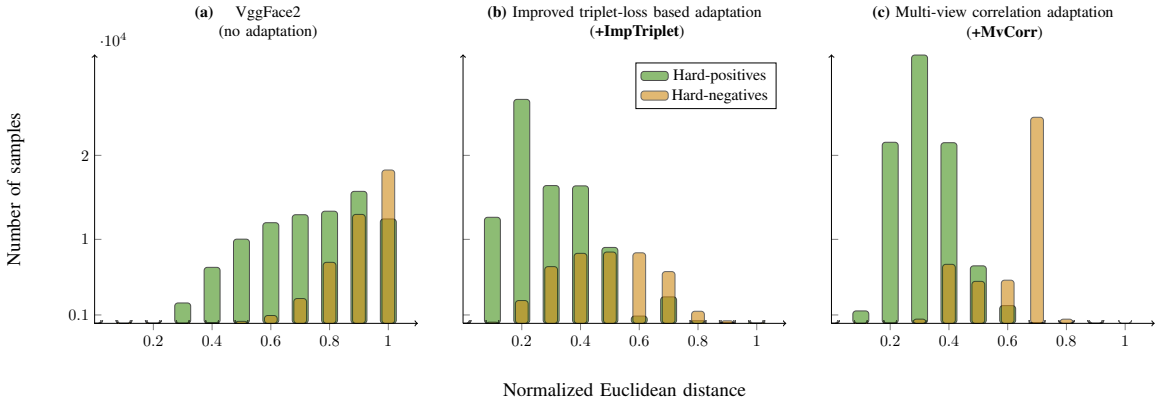

Fig. 5. Effect of feature adaption with weakly labeled data. Feature adaptation is expected to bring positive samples closer to each other and pull apart negative samples further in the transformed embedding space. Qualitative comparison of the distributions of hard-positive and hard-negative distances in SAIL-MultiFace development set shows the benefit of adaptation with (b) ImpTriplet and (c) MvCorr over the (a) original embeddings without adaptation.

choose the sub-network architecture, we explored three FCN configurations:

C1: **INP**$[(512)] \rightarrow$ **FC**$[512]$, **DO**$(0.2) \rightarrow$ **FC**$[256]$

C2: **INP**$[(512)] \rightarrow$ **FC**$[1024]$, **DO**$(0.2) \rightarrow$ **FC**$[512]$

C3: **INP**$[(512)] \rightarrow$ **FC**$[1024]$, **DO**$(0.2) \rightarrow$ **FC**$[512]$, **DO**$(0.2) \rightarrow$ **FC**$[256]$

where **INP** = Input, **FC** = Fully connected layer with ReLu/sigmoid activation followed by batch normalization. The number of nodes in each layer is indicated inside $[\cdot]$. A dropout (**DO**) of 0.2 was added for all intermediate FC layers. Dropout was tuned over the range $\{0.1, 0.2, 0.4\}$.

Training and model choice. All adaptation experiments were conducted on $169,201$ tracks in SAIL-MultiFace. For the training set, we used the data from 180 movies resulting in $126,435$ samples each for hard-positives and hard-negatives. The remaining $42,766$ samples were used as the development set. Both adaptation networks were trained with a batch size of 1024 using SGD (momentum=0.9, decay=$1e-6$) with a learning rate of 0.001 for ImpTriplet and 0.01 for MvCorr. To determine model convergence, we applied early stopping criteria (stop training if the loss on the development set at the end of a training epoch did not decrease by 10^{-3} for 5 consecutive epochs). Of the *C1–C3* configurations tested, we chose *C2* for ImpTriplet and *C1* for MvCorr as they showed the smallest loss on the development set at convergence. Increasing the model size beyond *C3* with additional layers or changing the embedding size from 256 to 128 or 1024 did not appear to further improve the loss at convergence. All configurations showed slightly better performance with ReLu over sigmoid activation. All models were implemented in TensorFlow[4] and trained on GeForce GTX 1080 Ti GPU.

Adaptation results. First, we examine the distribution of the normalized Euclidean distance for all hard-positive pairs $\left\|\mathbf{v}_a - \mathbf{v}_p^{(1)}\right\|^2$ and hard-negative pairs $\|\mathbf{v}_a - \mathbf{v}_q\|^2$, in our development set of 60 movies. For hard-positives, we expect a smaller pairwise distance and the distribution to skew left, and for hard-negatives, we expect larger distances and the distribution to skew right. The distributions for VGGFace2 embeddings without any adaptation is shown in Figure 5a.

[4]TensorFlow 2.1: tensorflow.org

TABLE IV
FACE VERIFICATION PERFORMANCE WITH ADAPTATION AVERAGED ACROSS ALL VIDEOS ON THE SAIL-MCB BENCHMARK DATASET.

Metric/ Model	FaceNet	VGGFace2	+ImpTriplet	+MvCorr
TPR @ 0.1FPR %	90.2 ± 3.5	92.5 ± 1.7	91.4 ± 2.0	**93.7 ± 1.2**

The distances generally skew right regardless of whether the samples were positives (similar) or negatives (dissimilar). The distribution of hard-positive distances is fairly uniform in the range 0.6–1 showing that the face tracklets belonging to the same person can be far apart in the embedding space. It suggests that we indeed encounter domain mismatch on direct application of VGGFace2 embeddings for face tracks in movies. This result also highlights the effectiveness of our nearest neighbor-based hard-example mining in identifying "difficult" samples in the embedding space.

Next, we examine the distribution of hard-positive and hard-negative distances for ImpTriplet and MvCorr adaptation. As shown in Figure 5b–c, both models skew the hard-positive distances further to the left compared to VGGFace2. This suggests that both methods help reduce the distance between the hard-positives as desired. Compared to the original embeddings, ImpTriplet adaptation appears to reduce the distance between dissimilar faces (Figure 5b) which could prove to be detrimental to downstream verification/clustering tasks. However, MvCorr skews the distribution of hard-negative distances further to the right than ImpTriplet. Although hard-negatives are not used in MvCorr adaptation, it appears to pull the cannot-link tracks (dissimilar faces) far apart from each other in the transformed embedding space; suggesting improved discriminability. This is consistent with our past multiview representation learning work where multiview embeddings were able to robustly classify whether two speech segments belonged to the same person or not [21].

Face verification results. A possible drawback of metric-learning based adaptation is that it may transform the embedding space to optimize only for the distances between hard examples while losing the discriminability of the input embeddings. In other words, it could overfit to the adaptation

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3096155, IEEE Transactions on Multimedia

TABLE V
V-MEASURE FOR HIERARCHICAL AGGLOMERATIVE CLUSTERING (HAC) AND AFFINITY PROPAGATION (AP) WITH OVER-CLUSTERING INDEX (OCI)

	Method	ALN	BFF	DD2	HF	MT	NH	Mean (OCI)
HAC	VGGFace2	83.2	95.3	82.6	76.7	88.6	79.5	84.3 (1.0)
	+ImpTriplet	81.2	96.6	83.9	78.0	89.7	81.1	85.1 (1.0)
	+MvCorr	**86.8**	**97.6**	**85.7**	**82.2**	**92.1**	**84.0**	**88.1** (1.0)
AP	VGGFace2	56.3	76.9	57.1	65.9	67.5	59.5	63.9 (5.2)
	+ImpTriplet	57.4	77.9	58.8	67.7	68.9	60.6	65.2 (6.0)
	+MvCorr	**60.4**	**84.3**	**60.1**	**70.1**	**69.6**	**62.0**	**67.7** (6.7)

TABLE VI
COMPARISON OF AVERAGE CLUSTERING ACCURACY WITH STATE-OF-THE-ART METHODS BASED ON SELF-SUPERVISION.

Method/ Dataset	BFF	NH
ULDML (2011) [19]	41.6	73.2
HMRF (2013) [49]	50.3	84.4
WBSLRR (2014) [63]	62.7	96.3
Zhang et. al. (2016) [59]	92.1	**99.0**
CP-SSC (2019) [20]	65.2	54.3
TSiam (2019) [18]	92.5	-
SSiam (2019) [18]	90.9	-
+MvCorr (ours)	**97.7**	96.3

dataset. To assess overfitting, we repeat the face verification experiments for SAIL-MCB benchmark dataset using the adapted VGGFace2 embeddings. We generate verification pairs by exhaustively mining all combinations of face tracks using the ground-truth character labels. This resulted in an average of about $993,000 \pm 372,000$ pairs ($26 \pm 9\%$ matching pairs) across the six videos in SAIL-MCB. We report TPR at 0.1 FPR averaged across all videos as the performance metric. As shown in Table IV, the performance of ImpTriplet is comparable to that of VGGFace2. With MvCorr adaptation, we observed a small but significant (1.2%) improvement in the true positive rate at 0.1 FPR. The corresponding ROC curves for face verification performance on the SAIL-MCB dataset are shown in supplementary methods, section IV, Fig. 3.

C. Face clustering experiments

To test the applicability of our system for unsupervised automatic character labeling in videos, we compare the unsupervised clustering performance for VGGFace2 embeddings with and without adaptation. For a fair comparison with past works [17], [59], we use hierarchical agglomerative clustering (HAC [72]) and assume the number of desired clusters (unique characters) to be known. However, in practice, the number of unique characters in a movie is often not available. Hence, we also experiment with affinity propagation clustering (AP [75]) which does not require the number of clusters before running the algorithm. Similar to the k-medoids algorithm, AP first finds representative *exemplars* to cluster all the points in the dataset. The exemplar count determines the number of clusters.

We evaluate the performance using several clustering metrics: homogeneity, completeness, V-measure, purity, and accuracy. In Table V, we report the V-measure scores on the benchmark dataset. Performance evaluation with respect to other metrics is presented in the supplementary methods, section V. For both HAC and AP, we achieve nearly 3% improvements using ImpTriplet and 4% improvement using MvCorr (See Table V). The V-measure scores for MvCorr were significantly better than VGGFace2 across all videos in our dataset[5]. In contrast to HAC, the V-measure scores for AP were low, but it yields clusters which are nearly 100% pure where multiple clusters may belong to the same character. We quantify this by reporting the over-clustering index [9] which is the average number of clusters assigned to a character (the mean OCI across all movies is shown in Table V). Mean OCI for the HAC method is 1 because the number of clusters is provided to the clustering algorithm.

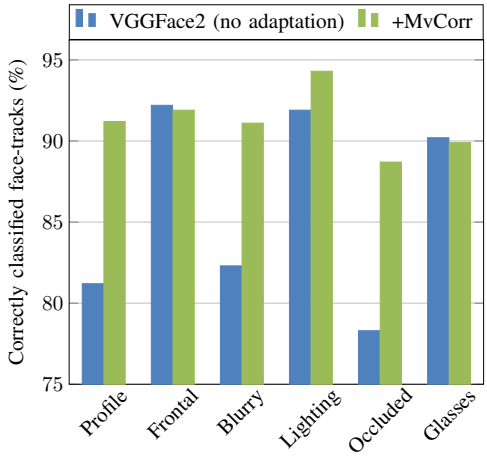

Fig. 6. Face quality error analysis in SAIL-MCB. For tracks with faces either wearing glasses (Glasses) or always facing the camera (frontal), MvCorr adaptation (+MvCorr) performed on-par with VGGFace2 pre-adaptation. In all other cases, +MvCorr significantly improved clustering accuracy.

Comparison with state-of-the-art. Finally, we compare the HAC performance for clustering characters in **BFF** and **NH** to the results reported in existing works. It is important to note that although the number of characters in our dataset is greater, the comparison metric is *mean* clustering accuracy, which is generally robust to these differences. As shown in Table VI, our proposed approach is comparable to the state-of-the-art methods for the two videos.

Face clustering error analysis. As part of the SAIL-MCB benchmarking dataset, we also obtained face quality labels along 6 dimensions as described in Sec. III-A. We perform error analysis of the clusters obtained using HAC along these dimensions. We use the percentage of total face tracks tagged

[5]Significance testing: permutation test $n = 10^5$, $p = 0.007$

with a particular quality label that were correctly classified as the metric of analysis. To determine if a face track is correctly classified, we applied the Hungarian algorithm to the HAC output using the ground-truth character labels. Results comparing this metric for VGGFace2 embeddings without adaptation and with MvCorr adaptation are presented in Fig. 6. For frontal (all faces in a track facing the camera) and glasses (at least one face wearing glasses in a track), there was no significant change in performance with MvCorr adaptation. This is consistent with previous evaluation of VGGFace2 on datasets such as CelebA [14] which showed the model to be robust to the attribute of glasses and frontal facing images. However with all other cases (profile, blurry, poor lighting and occluded/obstructed), MvCorr adaptation significantly improved the clustering performance[6]. These results highlight the importance of the need for domain-matched data and subsequent adaptation to improve face clustering in movies to account for visual distractors.

VI. Discussion

A key component of a robust, fully unsupervised, and automatic face clustering framework for movies is the ability to discover the number of characters without using additional metadata. In our clustering experiments, we either assumed this number to be known apriori (in HAC) or allowed for over-clustering (with AP) which requires additional heuristics or manual intervention to merge these clusters. In this context, we want to highlight two recent and promising directions of research.

An iterative merging algorithm called ball clustering (BCL, [76]) was developed to jointly estimate the number of clusters as well as resolve the assignment issue of the face tracks in a video. However, BCL was only evaluated on TV series. While the results are encouraging with respect to clustering background/secondary characters, movies tend to have more intermittent characters than in TV series. As such, the generalizability of BCL for long-form content remains to be explored.

Recent studies of online diarization for videos in [66], [77] proposed a shot-specific character interaction graph to incorporate constraints mined from movies. One of the benefits of online methods is that as diarization progresses new characters may be "discovered" as part of the process, creating new clusters. Our future work will focus on studying these methods for face clustering in movies, particularly in a zero-shot learning framework, which has been effectively used for person re-identification in the image domain [78]. In the context of real-time vision applications, it is worth highlighting two recent works on video event summarization. An alignment based method called F-DES [79] and an AdaBoost-based approach called DELTA [80] were developed to identify local and global similarity patterns to identify key events from multiview videos. Our future work will consider incorporating such ideas to improve hard-example mining from weakly labeled face tracks.

Finally, although SAIL-MCB included more racially diverse movies, the face quality labels used in the error analysis only included dimensions related to visual distractors. These labels were used to evaluate the robustness of face clustering methods. However, these dimensions did not include demographic variables such as gender, age and race. We are currently working along this direction to contribute to a growing list of resources such as *FairFace* [81] which is a face image dataset with demographic variables. These resources can help assess the fairness of algorithms along with their robustness.

[6]Significance testing: Permutation test $n = 10^4, p \leq 0.01$

VII. Conclusion

In this work, we investigated robust face clustering in the movie domain using ideas of self-supervision. First, we developed the SAIL-Movie Character Benchmark data set (SAIL-MCB) with character labels for six movie/TV videos, and SAIL-MultiFace with weakly labeled data from 240 movies, to offer domain-specific resources for feature adaptation and benchmarking. Next, we proposed a nearest-neighbor approach to identify hard-positive and hard-negative tracklets from the must-link and cannot-link faces mined in SAIL-MultiFace. Finally, using these tracklets, we explored triplet-loss and multiview correlation based frameworks to adapt face embeddings learned from web images to long-form content such as movies. Our face verification/clustering experiments and error analysis highlight the benefits of self-supervised feature adaption for robust automatic character labeling in movies. The SAIL-MCB and SAIL-MultiFace datasets have been made publicly available. We hope that these resources will help advance the research in understanding character portrayals in media content.

References

[1] J. Cohen, "Defining identification: A theoretical look at the identification of audiences with media characters," *Mass communication & society*, vol. 4, no. 3, pp. 245–264, 2001.

[2] K. Somandepalli, T. Guha, V. R. Martinez, N. Kumar, H. Adam, and S. Narayanan, "Computational media intelligence: Human-centered machine analysis of media," *Proceedings of the IEEE*, 2021.

[3] P. Vicol, M. Tapaswi, L. Castrejon, and S. Fidler, "Moviegraphs: Towards understanding human-centric situations from videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8581–8590.

[4] A. Rohrbach, M. Rohrbach, S. Tang, S. Joon Oh, and B. Schiele, "Generating descriptions with grounded and co-referenced people," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4979–4989.

[5] M. Kim and Y. Y. Doh, "Computational modeling of players' emotional response patterns to the story events of video games," *IEEE Transactions on Affective Computing*, vol. 8, no. 2, pp. 216–227, 2017.

[6] A. El Bolock, A. El Kady, C. Herbert, and S. Abdennadher, "Towards a character-based meta recommender for movies," in *Computational Science and Technology*. Springer, 2020, pp. 627–638.

[7] "4 ways to use X-Ray in Prime Video." [Online]. Available: https://www.amazon.com/primeinsider/video/pv-xray-tips.html

[8] T. Guha, C.-W. Huang, N. Kumar, Y. Zhu, and S. S. Narayanan, "Gender representation in cinematic content: A multimodal approach," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, 2015, pp. 31–34.

[9] K. Somandepalli, N. Kumar, T. Guha, and S. S. Narayanan, "Unsupervised discovery of character dictionaries in animation movies," *IEEE Transactions on Multimedia*, pp. 539–551, 2017.

[10] L. Nie, M. Liu, and X. Song, 2019.

[11] L. Nie, X. Wang, J. Zhang, X. He, H. Zhang, R. Hong, and Q. Tian, "Enhancing micro-video understanding by harnessing external sounds," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1192–1200.

[12] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou, "Retinaface: Single-stage dense face localisation in the wild," *arXiv preprint arXiv:1905.00641*, 2019.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE CVPR*, 2015, pp. 815–823.

[14] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, 2018, pp. 67–74.

[15] E. Ghaleb, M. Tapaswi, Z. Al-Halah, H. K. Ekenel, and R. Stiefelhagen, "Accio: A data set for face track retrieval in movies across age," in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, 2015, pp. 455–458.

[16] Y. Xiang, A. Alahi, and S. Savarese, "Learning to track: Online multi-object tracking by decision making," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4705–4713.

[17] S. Zhang, Y. Gong, and J. Wang, "Deep metric learning with improved triplet loss for face clustering in videos," in *Pacific Rim Conference on Multimedia*. Springer, 2016, pp. 497–508.

[18] V. Sharma, M. Tapaswi, M. S. Sarfraz, and R. Stiefelhagen, "Self-supervised learning of face representations for video face clustering," in *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*. IEEE, 2019, pp. 1–8.

[19] R. G. Cinbis, J. Verbeek, and C. Schmid, "Unsupervised metric learning for face identification in tv video," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1559–1566.

[20] K. Somandepalli and S. Narayanan, "Reinforcing self-expressive representation with constraint propagation for face clustering in movies," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 4065–4069.

[21] K. Somandepalli, N. Kumar, R. Travadi, and S. Narayanan, "Multimodal representation learning using deep multiset canonical correlation," *arXiv preprint arXiv:1904.01775*, 2019.

[22] M. Everingham, J. Sivic, and A. Zisserman, "Hello! my name is... buffy"–automatic naming of characters in tv video." in *BMVC*, vol. 2, 2006, p. 6.

[23] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments," in *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*, Oct. 2008.

[24] P. Viola and M. Jones, "Robust real-time object detection," in *International Journal of Computer Vision*, 2001.

[25] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014.

[26] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5525–5533.

[27] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, "Ms-celeb-1m: A dataset and benchmark for large-scale face recognition," in *European conference on computer vision*. Springer, 2016, pp. 87–102.

[28] R. Van Noorden, "The ethical questions that haunt facial-recognition research," *Nature*, vol. 587, no. 7834, pp. 354–358, 2020.

[29] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *CVPR 2011*. IEEE, 2011, pp. 529–534.

[30] Q. Huang, Y. Xiong, A. Rao, J. Wang, and D. Lin, "Movienet: A holistic dataset for movie understanding," *arXiv preprint arXiv:2007.10937*, 2020.

[31] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," *British Machine Vision Association*, 2015.

[32] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE CVPR*, 2017, pp. 212–220.

[33] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE CVPR*, 2019, pp. 4690–4699.

[34] Y. Shi and A. K. Jain, "Probabilistic face embeddings," in *Proceedings of the IEEE ICCV*, 2019, pp. 6902–6911.

[35] V. Sharma, M. S. Sarfraz, and R. Stiefelhagen, "A simple and effective technique for face clustering in TV series," 2017.

[36] V. Ramanathan, A. Joulin, P. Liang, and L. Fei-Fei, "Linking people in videos with "their" names using coreference resolution," in *European conference on computer vision*. Springer, 2014, pp. 95–110.

[37] J. Sivic, M. Everingham, and A. Zisserman, ""who are you?"-learning person specific classifiers from video," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1145–1152.

[38] T. Cour, B. Sapp, A. Nagle, and B. Taskar, "Talking pictures: Temporal grouping and dialog-supervised person recognition," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 1014–1021.

[39] M.-L. Haurilet, M. Tapaswi, Z. Al-Halah, and R. Stiefelhagen, "Naming tv characters by watching and analyzing dialogs," in *2016 IEEE WACV*. IEEE, 2016, pp. 1–9.

[40] K. Kim, Z. Yang, I. Masi, R. Nevatia, and G. Medioni, "Face and body association for video-based face recognition," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 39–48.

[41] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. Pal, H. Larochelle, A. Courville, and B. Schiele, "Movie description," *International Journal of Computer Vision*, vol. 123, no. 1, pp. 94–120, 2017.

[42] Y. Yu, J. Kim, H. Yun, J. Chung, and G. Kim, "Character grounding and re-identification in story of videos and text descriptions," in *European Conference on Computer Vision*. Springer, 2020, pp. 543–559.

[43] E. El Khoury, C. Sénac, and P. Joly, "Audiovisual diarization of people in video content," *Multimedia tools and applications*, vol. 68, no. 3, pp. 747–775, 2014.

[44] I. Kapsouras, A. Tefas, N. Nikolaidis, G. Peeters, L. Benaroya, and I. Pitas, "Multimodal speaker clustering in full length movies," *Multimedia Tools and Applications*, vol. 76, no. 2, pp. 2223–2242, 2017.

[45] F. Vallet, S. Essid, and J. Carrive, "A multimodal approach to speaker diarization on tv talk-shows," *IEEE transactions on multimedia*, vol. 15, no. 3, pp. 509–520, 2012.

[46] R. Sharma, K. Somandepalli, and S. Narayanan, "Toward visual voice activity detection for unconstrained videos," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 2991–2995.

[47] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Who's that actor? automatic labelling of actors in tv series starting from imdb images," in *ACCV*. Springer, 2016, pp. 467–483.

[48] N. Kose, L. Apvrille, and J. Dugelay, "Facial makeup detection technique based on texture and shape analysis," in *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, vol. 1, 2015, pp. 1–7.

[49] B. Wu, Y. Zhang, B.-G. Hu, and Q. Ji, "Constrained clustering and its application to face clustering in videos," in *Proceedings of the IEEE CVPR*, 2013, pp. 3507–3514.

[50] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[51] S. Datta, G. Sharma, and C. Jawahar, "Unsupervised learning of face representations," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition*. IEEE, 2018, pp. 135–142.

[52] J. Huo and T. L. van Zyl, "Unique faces recognition in videos," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. IEEE, 2020, pp. 1–7.

[53] H. Hotelling, "Relations between two sets of variates," in *Breakthroughs in statistics*. Springer, 1992, pp. 162–190.

[54] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *International Conference on Machine Learning*, 2013, pp. 1247–1255.

[55] X. Chang, T. Xiang, and T. M. Hospedales, "Scalable and effective deep cca via soft decorrelation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1488–1497.

[56] Z. Zhang, Y. Yuan, X. Shen, and Y. Li, "Low resolution face recognition and reconstruction via deep canonical correlation analysis," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 2951–2955.

[57] A. Pnevmatikakis and L. Polymenakos, "Subclass linear discriminant analysis for video-based face recognition," *Journal of Visual Communication and Image Representation*, pp. 543–551, 2009.

[58] K. Somandepalli, N. Kumar, A. Jati, P. Georgiou, and S. Narayanan, "Multiview shared subspace learning across speakers and speech commands," *Proc. Interspeech 2019*, pp. 2320–2324, 2019.

[59] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Joint face representation adaptation and clustering in videos," in *European conference on computer vision*. Springer, 2016, pp. 236–251.

[60] M. Tapaswi, M. Bäuml, and R. Stiefelhagen, ""knock! knock! who is it?" probabilistic person identification in tv-series," in *2012 IEEE CVPR*. IEEE, 2012, pp. 2658–2665.

[61] A. Nagrani and A. Zisserman, "From benedict cumberbatch to sherlock holmes: Character identification in TV series without a script," *arXiv preprint arXiv:1801.10442*, 2018.

[62] P. Bojanowski, F. Bach, I. Laptev, J. Ponce, C. Schmid, and J. Sivic, "Finding actors and actions in movies," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2280–2287.

[63] S. Xiao, M. Tan, and D. Xu, *Weighted Block-Sparse Low Rank Representation for Face Clustering in Videos*. Springer International Publishing, 2014, pp. 123–138.

[64] "2020 Hollywood Diversity Report: A different story behind the scenes." [Online]. Available: https://newsroom.ucla.edu/releases/2020-hollywood-diversity-report

[65] "2020 Film: Historic Gender Parity in Family Films." [Online]. Available: https://seejane.org/research-informs-empowers/2020-film-historic-gender-parity-in-family-films/

[66] P. Kulshreshtha and T. Guha, "Dynamic character graph via online face clustering for movie analysis," *Multimedia Tools and Applications*, vol. 79, no. 43, pp. 33 103–33 118, 2020.

[67] C. Zhou, C. Zhang, X. Li, G. Shi, and X. Cao, "Video face clustering via constrained sparse representation," in *2014 IEEE International Conference on Multimedia and Expo (ICME)*. Los Alamitos, CA, USA: IEEE Computer Society, jul 2014, pp. 1–6.

[68] J. Bian, X. Mei, and J. Zhang, "A video face clustering approach based on sparse subspace representation," in *Tenth International Conference on Digital Image Processing (ICDIP 2018)*, vol. 10806. International Society for Optics and Photonics, 2018, p. 1080645.

[69] I. U. Haq, K. Muhammad, A. Ullah, and S. W. Baik, "Deepstar: Detecting starring characters in movies," *IEEE Access*, vol. 7, pp. 9265–9272, 2019.

[70] A. Bugeau and P. Pérez, "Track and cut: simultaneous tracking and segmentation of multiple objects with graph cuts," *EURASIP Journal on Image and Video Processing*, p. 317278, 2008.

[71] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proceedings of the IEEE CVPR*, 2016, pp. 761–769.

[72] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[73] C. Whitelam, E. Taborsky, A. Blanton, B. Maze, J. Adams, T. Miller, N. Kalka, A. K. Jain, J. A. Duncan, K. Allen *et al.*, "Iarpa janus benchmark-b face dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 90–98.

[74] F.-J. Chang, A. Tuan Tran, T. Hassner, I. Masi, R. Nevatia, and G. Medioni, "Faceposenet: Making a case for landmark-free face alignment," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 1599–1608.

[75] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, pp. 972–976, 2007.

[76] M. Tapaswi, M. T. Law, and S. Fidler, "Video face clustering with unknown number of clusters," in *Proceedings of the IEEE ICCV*, 2019, pp. 5027–5036.

[77] P. Kulshreshtha and T. Guha, "An online algorithm for constrained face clustering in videos," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2670–2674.

[78] Z. Wang, R. Hu, C. Liang, Y. Yu, J. Jiang, M. Ye, J. Chen, and Q. Leng, "Zero-shot person re-identification via cross-view consistency," *IEEE Transactions on Multimedia*, vol. 18, no. 2, pp. 260–272, 2015.

[79] K. Kumar and D. D. Shrimankar, "F-des: Fast and deep event summarization," *IEEE Transactions on Multimedia*, vol. 20, no. 2, pp. 323–334, 2017.

[80] ——, "Deep event learning boost-up approach: Delta," *Multimedia Tools and Applications*, vol. 77, no. 20, pp. 26 635–26 655, 2018.

[81] K. Karkkainen and J. Joo, "Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1548–1558.