Movie character re-identification by agglomerative clustering of deep features

Samuel Ducros, Gérard Subsol, Mathieu Lafourcade, Jean-Marie Barthélémy, William Puech

Movie character re-identification by agglomerative clustering of deep features

Samuel Ducros[1,2], Gérard Subsol[1], Mathieu Lafourcade[1], Jean-Marie Barthélémy[2], William Puech[1]
[1] LIRMM, Univ Montpellier, CNRS, Montpellier, France
[2] ECOSM (European Company of Softwares for Media), Montpellier, France

Abstract

In this paper, we present a method for agglomerative clustering of characters in a video. Given a video edited with humans, we seek to identify each person with the character they represent. The proposed method is based on agglomerative clustering of deep face features, using first neighbour relations. First, the heads and faces of each person are detected and tracked in each shot of the video. Then, we create a feature vector of a tracked person in a shot. Finally, we compare the feature vectors and we use first neighbour relations to group them into distinct characters. The main contribution of this work is a person re-identification framework based on an agglomerative clustering method, and applied to edited videos with large scene variations.

Key words

- Person re-identification
- Agglomerative clustering
- Movie character identification
- Deep face features

Introduction

Several processes in the film industry require to follow the characters all along the movie. For example, knowing which characters are present in a shot is useful to detect and localize who is speaking. These last years, many methods based on Deep Learning techniques have been proposed to detect or recognize people in an image or a sequence of images. For example, Amazon Rekognition[1] uses an actor database to detect and recognize celebrity all along a video. But if we have no information on the identity or the number of characters, the problem becomes more complex and is known as ”person re-identification” [11]. This kind of algorithms are mostly focused on video surveillance applications [13, 5, 8]. But we can find in [6] a method dedicated to the detection of all the characters present in a movie. It is based on face detection for each shot and a clustering method to link the similar faces along the movie. The key problem remains the robustness with respect to the diversity of pose, the style of the character appearance, and the occlusion when two characters are in the same images.

In this paper, we propose an agglomerative clustering method, that we integrate into a pipeline that re-identifies people detected in a video, without any knowledge about the identity or the number of characters. Our method is aimed to be more robust as we use head detection combined with face detection, a robust tracking algorithm and an agglomerative clustering method on feature vectors of the detected faces. We intend to apply this method on TV shows, news or movies to detect, identify and track all the characters shot by shot.

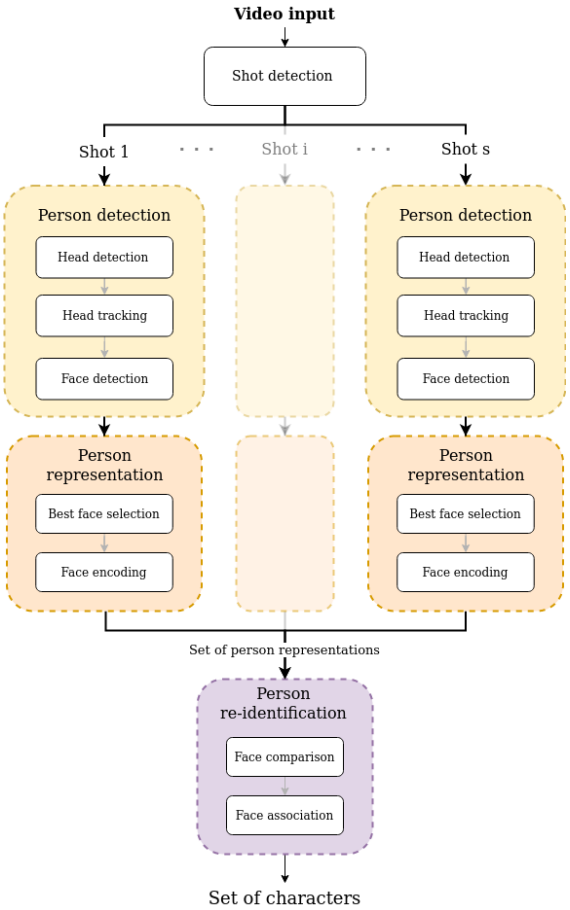


Figure 1: Overview of the proposed method.

Method

In this section, we describe our proposed agglomerative movie character re-identification algorithm (see **Figure 1**). We first parse the video into shots. Each shot corresponds to a continuous sequence of frames. For each shot, we detect, localize and track what we call ”persons”, who are unknown individuals appearing along a continuous sequence of frames. The work presented here focuses on the re-identification of the persons of

[1] `https://docs.aws.amazon.com/rekognition/latest/dg/celebrities.html`

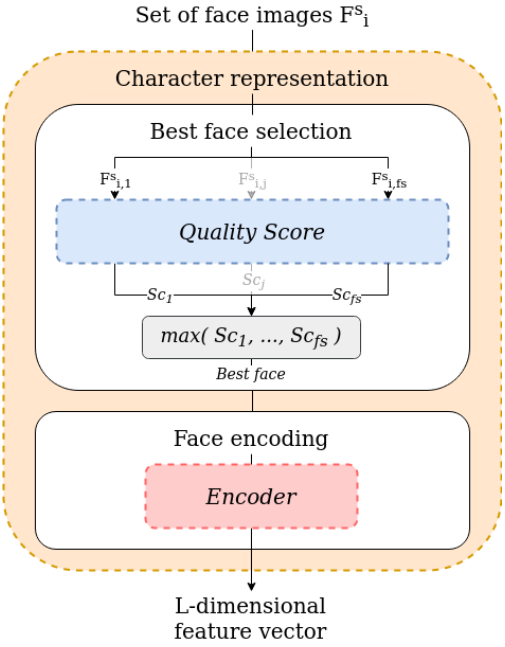


Figure 2: Character representation process.

different shots in order to retrieve and follow a "character" (actual actor) in the whole movie, without any previous knowledge about its identity. The last step of the process is for a human agent to annotate each one of the re-identified persons to give them the identity of the character (e.g. the name). Each component of our algorithm is described and explained in the next sections.

Person detection and tracking per shot

To detect and track a person in a shot, we assume they can be represented by their facial features. But persons can move and turn around and their face may no more be present. So in order to get a robust tracking of a person in the shot, we choose to use first a head detector algorithm which can localize the head of a person in all position, from front, profile or back. We combine this head detector with a tracking method to get a head tracklet.

We then use a face detector to search for faces in the head tracklet. We select then only the images of the head tracklet where the face is visible to get a set of faces $\mathscr{F}_i^s$ where s is the shot number and i is the detected person index in the current shot

Person representation per shot

We assess then the quality of the faces. We search for the "best" face in the set of faces $\mathscr{F}_i^s$ by using the SDD-FIQA method (Similarity Distribution Distance for Face Image Quality Assessment), based on a Neural Network trained in an unsupervised fashion [9]. We obtain a score between 0 and 100, 100 being the best quality and we choose the image F_i^s with the highest score. We set a minimum score, beneath which we suppose there is no face in the tracklet.

In order to compare faces for re-identification, we need to compute a feature vector $\mathbf{r}_i^s$ for the best face F_i^s. We can use methods such as FaceNet-512 [10] and VGG-Face [7] which results in feature vectors of size $L = 512$ for FaceNet-512, while VGG-Face gives a $L = 2622$-dimensional feature vector. **Figure 2** illustrates this process.

Agglomerative person clustering between shots

Let us say that each shot $s \in [\![1,S]\!]$ contains n_s different persons. We state that two detected persons in the same shot can not be the same character nor person (this excludes videos with one actor that plays several roles at the same time). We then have n_s different characters, with $\sum_{s=1}^{S} n_s = N$.

We have now N detected persons (and so N feature vectors) distributed into the S shots of the video. If there are actually K characters, we should have $K \leqslant N$ (if we do not have detected false persons). The aim here is to cluster persons detected in different shots to retrieve the character which may appear all along the movie. Notice two particular cases: if $K = 1$, there is only one character who is present in the S shots. This video could be for example a YouTuber alone in front of the camera. On the contrary, if we have before any processing $K = N$, this means that there are as many persons (which are detected in only one shot) as real characters. In other words, each character is present in only one shot. This can appear for example in a street interviews video, where each interviewee is present in one shot only.

Given the feature vectors $\mathbf{r}_i^s$ of all the persons i in all the shots s of the video, we compare them to each other, according to a distance metric (such as euclidean L1, euclidean L2 or cosine similarity metric). Finally, we associate these representations together using a new agglomerative clustering method based on AGNES (Agglomerative Nesting) [4], and we obtain clusters of re-identified detected persons corresponding to the same character.

In order to create clusters of representations, the first step of our method is to compute distances between all the feature vectors. These distances define the proximity of two representations and are the basis of our clustering method. To do that, we choose to use 3 metrics, Euclidean L1, Euclidean L2 and Cosine similarity, but we could use other distance metrics.

Let $\mathbf{D_{st}} \in \mathscr{M}_{n_s,n_t}(\mathbb{R})$ be the matrix of distances between the persons detected in the shot s and the persons detected in the shot t. The matrix $\mathbf{D_{st}}$ is:

$$\mathbf{D_{st}} = \left(d_{i,j}^{s,t}\right)_{\substack{1 \leqslant i \leqslant n_s \\ 1 \leqslant j \leqslant n_t}}, \tag{1}$$

with $d_{i,j}^{s,t} = dist\left(\mathbf{r}_i^s,\ \mathbf{r}_j^t\right)$ the distance between the representations of the detected person $i \in [\![1,n_s]\!]$ and the detected person $j \in [\![1,n_t]\!]$.

As the distance function is symmetric, we have $\mathbf{D_{st}} = \mathbf{D_{ts}}^{\mathrm{T}},\ \forall s,t \in [\![1,S]\!]$. If $s = t$, the distance matrix D_{ss} contains the distances between the persons detected in the same shot s. But as we stated that two detected persons in the same shot can not be the same character, we set the distance between all the representations in the same shot to $+\infty$.

Let then $\mathbf{D} \in \mathscr{M}_{N,N}(\mathbb{R})$ be the matrix of distance between all the representations. We can decompose this matrix with the sub-matrices $\mathbf{D_{st}},\ \forall s,t \in [\![1,S]\!]$:

$$\mathbf{D} = \begin{bmatrix} \mathbf{D_{11}} & \mathbf{D_{12}} & \cdots & \mathbf{D_{1S}} \\ \mathbf{D_{12}}^{\mathrm{T}} & \mathbf{D_{22}} & \cdots & \mathbf{D_{2S}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D_{1S}}^{\mathrm{T}} & \mathbf{D_{2S}}^{\mathrm{T}} & \cdots & \mathbf{D_{SS}} \end{bmatrix} \in \mathscr{M}_{N,N}(\mathbb{R}). \quad (2)$$

Algorithm 1: Our proposed agglomerative algorithm, based on AGNES [4]

Data: $N \geqslant 0$
Data: $\mathbf{D}$
Data: $\{C_i = (\mathbf{r}_i,\ T_i,\ \mathscr{E}_i = \{\mathbf{r}_i\}),\ i \in [\![1,N]\!]\}$
`// Triplet representing a cluster:`
`// r_i: feature vector.`
`// T_i: associated threshold.`
`// E_i: set of feature vectors composing the cluster`
Result: $\{C_k,\ k \in [\![1,K]\!]\}$
`// Set of final clusters representing all the characters.`
$N_c \leftarrow N$;
while *matrix* $\mathbf{D}$ *contains a finite and strictly positive value* **do**
  $d_{min} \leftarrow \min\{d_{i,j} \in \mathbf{D} \mid 0 < d_{i,j} < +\infty\}$
  $(i_{min}, j_{min}) \leftarrow \operatorname{argmin}\{d_{i,j} \in \mathbf{D} \mid 0 < d_{i,j} < +\infty\}$
  `// i_min is the row cluster id and j_min is the column cluster id`
  if $d_{min} \leqslant \min(T_{i_{min}}, T_{j_{min}})$ **then**
    $\mathscr{E}_{i_{min}} \leftarrow \mathscr{E}_{i_{min}} \cup \{\mathscr{E}_{j_{min}}\}$ `// Clusters merge`
    $\mathbf{r}_{i_{min}} \leftarrow \operatorname{mean}(\mathscr{E}_{i_{min}})$
    `// New representation vector`
    $T_{i_{min}} \leftarrow$ `ComputeThresh`$(d_{min},\ T_{i_{min}},\ T_{j_{min}})$
    `// New threshold`
    for *k from* 1 *to* N_c, *except* $k = i_{min}$ *and* $k = j_{min}$ **do**
      if $\mathbf{D}_{i_{min},k} = +\infty$ *or* $\mathbf{D}_{j_{min},k} = +\infty$ **then**
        $\mathbf{D}_{i_{min},k} \leftarrow +\infty$
        $\mathbf{D}_{k,i_{min}} \leftarrow +\infty$
      else
        $\mathbf{D}_{i_{min},k} \leftarrow dist(\mathbf{r}_{i_{min}}, \mathbf{r}_k)$
        $\mathbf{D}_{k,i_{min}} \leftarrow dist(\mathbf{r}_{i_{min}}, \mathbf{r}_k)$
      end
    end
    Delete the column $\mathbf{D}_{:,j_{min}}$ and the row $\mathbf{D}_{j_{min},:}$
    $N_c \leftarrow N_c - 1$
  else
    $\mathbf{D}_{i_{min},j_{min}} \leftarrow +\infty$
  end
end

$\mathbf{D}$ is then a symmetric matrix. The matrix $\mathbf{D}$ is also characterized by 0 for all values of the diagonal, because the distance between a representation and itself is equal to 0. We have:

$$\mathbf{D} = (d_{r,u})_{1 \leqslant r \leqslant u \leqslant N}, \quad (3)$$

with $d_{r,u}$ the distance between the person indexed r among the N detected persons, and the person indexed u. This notation does not take the shot numbers s and t into account, but these are related by

$$d_{i,j}^{s,t} = d_{i+\sum_{k=1}^{s-1} n_k,\ j+\sum_{l=1}^{t-1} n_l}. \quad (4)$$

Our strategy is to associate the detected persons sequentially, from the closest representations to the farthest ones, and to create clusters of representations by using the Agglomerative Nesting algorithm (AGNES) [4]. We suppose first that each detected person $i \in [\![1,N]\!]$ initially composes its own cluster, and we define these clusters by $C_i = \{\mathbf{r}_i,\ T_i,\ \mathscr{E}_i\}$, with $\mathscr{E}_i$ the set of face feature vectors composing the cluster, T_i the threshold associated with the cluster, and $\mathbf{r}_i$ the characteristic vector of the cluster, corresponding to the barycentre of the set of face vectors $\mathscr{E}_i$. Each recognition method applies a threshold T on the distance between two representations to check their similarity. Then, as input of the algorithm, each $\mathscr{E}_i$ is a singleton containing the face vector $\mathbf{r}_i$ (and so cluster feature vector), and $T_i = T,\ \forall i \in [\![1,N]\!]$. We have at the beginning N clusters. The aim is to iteratively reduce the number of clusters to find the K clusters that represent the K true characters, with K unknown. Our algorithm is described in **Algorithm 1**.

To compute the new threshold of two merged clusters $C_\alpha = \{\mathbf{r}_\alpha,\ T_\alpha,\ \mathscr{E}_\alpha\}$ and $C_\beta = \{\mathbf{r}_\beta,\ T_\beta,\ \mathscr{E}_\beta\}$, we compute the radius of the intersection of the L-dimensional sphere S_α with center $\mathbf{r}_\alpha \in \mathbb{R}^L$ and radius $T_\alpha \in \mathbb{R}^+$, and the L-dimensional sphere S_β with center $\mathbf{r}_\beta \in \mathbb{R}^L$ and radius $T_\beta \in \mathbb{R}^+$. We also take into account the euclidean L2 distance $d = \|\mathbf{r}_\alpha - \mathbf{r}_\beta\|_2$ and we obtain the new threshold T' of the new cluster C' merging C_α and C_β:

$$T' = \frac{1}{2d} \times \sqrt{4T_\alpha^2 d^2 - \left(T_\alpha^2 - T_\beta^2 + d^2\right)^2}. \quad (5)$$

Experimental results

Let us take a 1min-passage from the episode S04E02 of Friends[2] as an illustration for the results. This video contains $S = 16$ shots, and a total of $N = 45$ characters detected, that we have to cluster into $K = 6$ groups to represent the 6 true characters present in the scene. For our approach, we assume that the value $K = 6$ is unknown.

[2]Friends: `https://en.wikipedia.org/wiki/Friends`

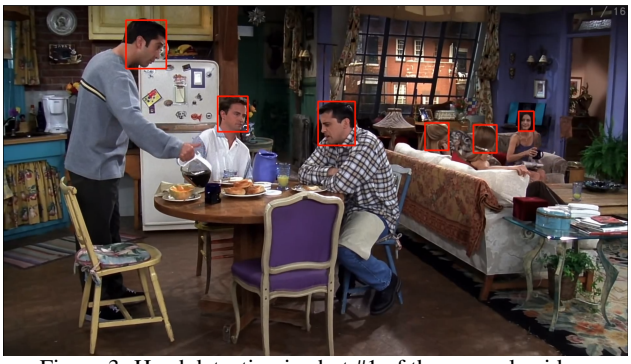
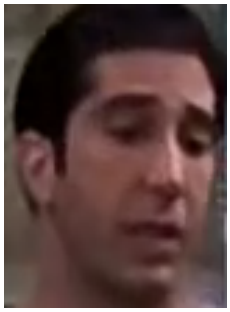
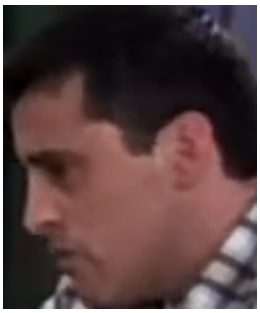


Figure 3: Head detection in shot #1 of the example video.

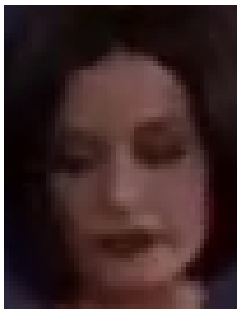


(a) Ross best face, Q=40.3

(b) Chandler best face, Q=52.7



(c) Joey best face, Q=37.2



(d) Monica best face, Q=37.5

Figure 4: Best faces of the 4 persons with a front or a profile face visible at least on one frame in the shot 1. As the face of Chandler (4b) is better enlightened and straight, it has a better score than the other. On the contrary, Joey (4c) is in profile and his score is lower.

We first cut the video into shots using PySceneDetect for its fastness and its performance [1]. We obtain all the shot cuts in a video, and we get $S = 16$ videos without camera change, as we expected. Let's take the first shot $s = 1$. The head detector Yolov5-CrowdHuman [3] and the tracking method DeepSort [12] return correctly all the heads tracklets of the 6 persons in the shot. **Figure 3** illustrates the head detection, on the frame #117 of the first shot. We notice 2 persons with a front face (Chandler and Monica), 2 with a profile head (Ross and Joey) and 2 others showing their back (Phoebe and Rachel). Among all the frames of the shot #1 where faces are detected, we select the front face with the best score according to the image quality assessment factor SDD-FIQA [9]. **Figure 4** illustrates the best faces chosen by the algorithm and the associated score, for 4 persons. Unfortunately, for the two persons from the back, we can not select a face, so they are not considered for this shot.

We then encode the obtained best faces with the encoder FaceNet-512 [7] and we obtain a $L = 512$-dimensional feature vector, which we compare with all the other feature vectors of the best faces in the other shots, using the euclidean L2 distance. We obtained then the matrix **D**, on which we compute our **Algorithm 1**. This outputs the clusters of the detected person along all the shots of the video, and we illustrate the result on **Figure 5**. We notice for example in **Figure 5a** that the character Ross appears in 5 shots. We can also notice that the clusters illustrated in **Figure 5d** and **Figure 5e** represent the same character but they are separated into two distinct clusters.

After manually annotating each cluster, we then obtain the

(a) Ross cluster

(b) Chandler cluster

(c) Joey cluster (d) Monica cluster 1 (e) Monica cluster 2

Figure 5: Obtained clusters with our approach, for characters Ross, Chandler, Joey and Monica. Two clusters have been created for Monica, which is due to the low quality of the face images, especially the face image of the cluster **Figure 5d** which is only 50 pixels wide.

Figure 6: Named characters in the shot #1 of the example video of Friends. As we use facial recognition in the process, the back of the heads of Phoebe and Rachel are not identified.

video results available on this link[3]. **Figure 6** illustrates the correct annotation of the frame 117 of the shot #1, obtained with our algorithm.

Discussion

Some limitations can occur with our algorithm. First, during the person re-identification phase, two different characters can be associated to the same cluster. A solution would be for the end-user to visually check that all the best face images of the persons tracklets which are aggregated to a cluster (as the 5 faces of Ross cluster in **Figure 5a**) really represent the same character. This is a quick process that would require, in case of an error, to identify manually each person of the cluster.

The opposite case is when several clusters are created whereas they represent the same character (as in **Figure 5d and**

[3]https://seafile.lirmm.fr/f/f4cd1a0fcc8b4b079d8d/

Figure 7: Case where one character occludes another one. We present 3 frames of the same shot ordered by time from top to bottom. Initially, the two characters (the man with grey shirt and Joey) are correctly tracked. But then, the occlusion of Joey by the other man, combined with the fact that the man moves too fast to be correctly detected by the head detector, leads to a confusion between the two tracklets of both characters. The method keeps only the tracking of the man in grey before the occlusion, and the tracking of Joey after the occlusion. The best face in this mixed tracklet is Joey's (as it is visible from front and motionless), so after annotation, his name was propagated to the other character.

5e). In fact, low image quality, variations of lightning or face orientation may change the feature vector of the face encoder. The distance between two representations of the same face becomes then too large to aggregate the two persons. Notice that this causes only more clusters than it should be; the result is not wrong but it will require the end-user to annotate more data.

An other limitation is head overlapping, which may impact the tracking phase. Let imagine a shot where the head of the character A passes in front of the head of the character B at some time t. Before time t, the method creates a tracklet for each character. Then at time t, our method only captures one head in the frame, and then the two tracklets arrive at the same place. If after time t, the character A goes off camera or is not clearly detectable (for example because we can not distinguish his face), the tracking algorithm will follow only the character B, resulting in a unique tracklet containing two different characters. **Figure 7** shows an example of this problem.

Some objects might also be recognized as human faces during the face detection phase, and then be tracked and clustered. Most of the time, these objects are tracked only during a small number of frames as they are only confused with human head with the right conditions (for example the right angle or the right light). Then there is not even enough frames where a face is detected to create an actual tracklet. Indeed, we introduce a threshold f_{min} based on the "minimum number of frames" in the tracking method to remove these artefactual tracklets. So in general, the object is discarded during the tracking phase. Examples of objects mistaken with human faces are presented in **Figure 8**

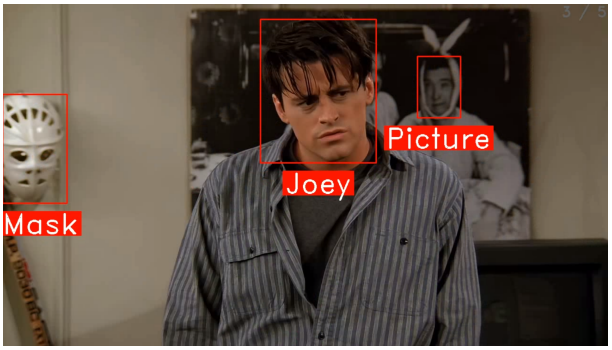


Figure 8: Example of objects mistaken for human faces: a photo portrait and a mask.

Conclusion

In this work, we have presented an agglomerative movie character clustering algorithm based on first neighbour relations, and using head and face detection. While most work on person re-identification focus on deep face features recognition only, we add a head detection to improve the tracking of the characters. Moreover, our method of face-clustering is based on first neighbour relations, which aims to overcome errors due to the application of a threshold alone.

In the future, we plan to apply our method on the Video Person-Clustering dataset VPCD, and compare our results with the state of the art Mu-HPC [2].

Acknowledgments

This work is supported by a CIFRE convention of the ANRT and the French Ministry of Higher Education and Research.

References

[1] Castellano B. Pyscenedetect, 2022. http://github.com/Breakthrough/PySceneDetect.

[2] Kalogeiton V. Brown A. and Zisserman A. Face, Body, Voice: Video Person-Clustering with Multiple Modalities. In *International Conference on Computer Vision (ICCV) 2021 Workshop on AI for Creative Video Editing and Understanding*, Montreal (virtual), Canada, October 2021.

[3] Deepak. Yolov5-crowdhuman, Last updated in July 2021. https://github.com/deepakcrk/yolov5-crowdhuman.

[4] L. Deng, T. Tan, J. Han, and T. Tian. Iagnes algorithm for protocol recognition. *High Technology Letters*, 24:408–416, 12 2018.

[5] Bipin G. and Abhijit K. End-to-end person re-identification: Real-time video surveillance over edge-cloud environment. *Computers and Electrical Engineering*, 99:107824, 2022.

[6] Ijaz Ul Haq, Khan Muhammad, Amin Ullah, and Sung Wook Baik. Deepstar: Detecting starring characters in movies. *IEEE Access*, 7:9265–9272, 2019.

[7] Serengil Sefik I. and Ozpinar A. Lightface: A hybrid deep face recognition framework. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 23–27. IEEE, 2020.

[8] Furqan M. Khan and Francois Bremond. Person re-identification for real-world surveillance systems, 2016.

[9] Zhang R. Huang Y. Li S. Li J. Li Y. Cao L. Ou F., Chen X. and Wang Y. Sdd-fiqa: Unsupervised face image quality assessment with similarity distribution distance. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7666–7675, 2021.

[10] Kalenichenko D. Schroff F. and Philbin J. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.

[11] Zimin Tian, Si Chen, Da-Han Wang, and Junwen Lu. A survey of person re-identification based on deep learning. In *2021 10th International Conference on Computing and Pattern Recognition*, ICCPR 2021, page 36–42, New York, NY, USA, 2021. Association for Computing Machinery.

[12] Bewley A. Wojke N. and Paulus D. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649, 2017.

[13] Shaoxiong Zhang, Yunhong Wang, Tianrui Chai, Annan Li, and Anil K. Jain. Realgait: Gait recognition for person re-identification, 2022.