Unsupervised Discovery of Character Dictionaries in Animation Movies

Krishna Somandepalli, *Member, IEEE*, Naveen Kumar, *Member, IEEE*, Tanaya Guha, *Member, IEEE*, and Shrikanth S. Narayanan, *Fellow, IEEE*

***Abstract*—Automatic content analysis of animation movies can enable an objective understanding of character (actor) representations and their portrayals. It can also help illuminate potential markers of unconscious biases and their impact. However, multimedia analysis of movie content has predominantly focused on live-action features. A dearth of multimedia research in this field is because of the complexity and heterogeneity in the design of animated characters—an extremely challenging problem to be generalized by a single method or model. In this paper, we address the problem of automatically discovering characters in animation movies as a first step toward automatic character labeling in these media. Movie-specific character dictionaries can act as a powerful first step for subsequent content analysis at scale. We propose an unsupervised approach which requires no prior information about the characters in a movie. We first use a deep neural network-based object detector that is trained on natural images to identify a set of initial character candidates. These candidates are further pruned using saliency constraints and visual object tracking. A character dictionary per movie is then generated from exemplars obtained by clustering these candidates. We are able to identify both anthropomorphic and nonanthropomorphic characters in a dataset of 46 animation movies with varying composition and character design. Our results indicate high precision and recall of the automatically detected characters compared to human-annotated ground truth, demonstrating the generalizability of our approach.**

***Index Terms*—Animation movies, deep neural networks, object tracking, saliency, unsupervised clustering, video diarization.**

I. Introduction

AUTOMATIC analysis of movie content is of growing interest in the multimedia research community. One of the driving factors for this research is the large number of movies that are produced, disseminated and consumed annually. Besides being of entertainment value, movies often have an effect on certain social and economic aspects, as well as have a global reach and audience.

Researchers have addressed movie content analysis with different objectives and outlooks. Such efforts are often based on efficient indexing and organization of the media content for easy user navigation. They include shot boundary detection for movie segmentation [1], [2], video summarization [3] and abstraction [4]. The study in [2] builds a generative model that incorporates contextual information in order to reorganize interleaved shots into multiple plot threads. Approaches such as in [5] combines the aspects of video summarization, i.e., *who*, *what*, *where* and *when* for a semantic understanding of the movie content and structure. *RoleNet* proposed in [6] examines the movie content from a social network analysis perspective of the movie character roles rather than using audiovisual features. In general, movie content is a rich source of data that includes audio, video and text (dialogs) that enables such multimodal analysis.

Complementary to the aforementioned studies which attempt to achieve a high-level understanding of movies, efforts for a fine-grained (frame level or scene level statistics) analysis of video content have also been emerging. One such application is to quantify the amount of time a character appears on screen in a movie. The study in [7] examined these aspects with respect to gender revealing skewed distributions for the onscreen time of female characters. In order to advance from gender-level statistics to character-level statistics, person identification or character labeling is a crucial step in this direction. We refer to this problem as automatic video diarization – partitioning the video stream into actor-homogeneous segments, i.e., *who appeared*, *when* and for *how long*. Character labeling in live-action TV and movies has been achieved with modest success in [8]–[12]. This is typically performed by clustering the detected faces (e.g., [8]) or by multimodal approaches (e.g., [9], [10]) that model audio and subtitles or scripts alongside the detected faces from video.

It is important to note that all these studies exclusively focus on live-action TV and do not generalize to animated media content. Digital animation movies have contributed to over 10% of the box office market shares in the past decade [13]. Multimedia research in this domain is extremely scarce and technology developed for live-action TV content fails for animated content. Human face detection is the crux of character labeling methods for live action TV. Since human-characters can be uniquely identified by their faces, this method performs adequately well. But, such methods developed for human faces do not work

Manuscript received November 28, 2016; revised July 26, 2017; accepted August 4, 2017. This work is based upon work supported by the National Science Foundation. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xilin Chen. *(Corresponding author: Krishna Somandepalli.)*

K. Somandepalli, N. Kumar, and T. Guha is with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90047 USA (e-mail: krishna.somandepalli@gmail.com; knaveen87@gmail.com; tanayaguha@gmail.com).

S. S. Narayanan is with the Signal & Image Processing Institute, University of Southern California, Los Angeles, CA 90089 USA (e-mail: shri@sipi.usc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2745712

Fig. 1. Examples illustrating the heterogeneity of animated characters. (a): human-like (Frozen) (b): anthropomorphic (Frozen) (c) and (d): abstract (How to Train your Dragon, and Cars).

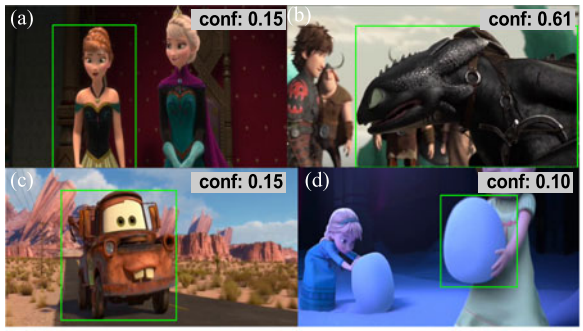

Fig. 2. Character candidates chosen by the Multibox object detector. Conf. indicates the confidence score of the network for the detected object.

for the digital animation genre. Animated characters, though mostly anthropomorphic (having human characteristics) are not always human-like in appearance. They can be fictional animals, inanimate objects or abstract in design (see Fig. 1 for a few examples).

A major obstacle for automating content analysis of animated media is the lack of a model that generalizes across different characters with varying composition and design. This task becomes extremely complex given that all the characters even within a single movie may not share the same structural characteristics (e.g., human-like and non-human characters from the same movie—Fig. 1(a) and (b) from the movie Frozen).

In the context of video diarization, when the characters that appear on screen are generally not known a priori, a key step is to provide a list of characters that form the *who appeared* component of the system. We refer to such a list of characters specific to each movie as a *character dictionary*. The automatic discovery of these character dictionaries is the primary objective in this paper. Our overarching goal is to engineer a model for animation movie video diarization. With the proposed character dictionaries, animation character labeling may be achieved by techniques such as [14] that can retrieve frames and shots given an object of interest.

In content analysis of animated media, researchers have thus far focused on problems such as cut detection [15], color-based video categorization [16] and movie abstraction [17], [18]. One method proposed in [19] performs human-like face detection from cartoon images using skin-segmentation techniques. Considering the variation in texture, color and shape of animated characters in general (as illustrated in Fig. 1), these methods do not generalize well. To the best of our knowledge, no work to date has specifically addressed the problem of automatic discovery of characters from animated media in a scalable manner.

In contrast to live-action movies, animation movies are completely artist generated. Sketches of the characters are designed by the artists or the animators, generally referred to as *model sheets* from which character-specific 3D models are generated. Sketch based image retrieval systems such as [20] can be used to achieve video diarization when model sheets are available. However, model sheets are copyrighted material and mostly owned by the animation studio which produced the movie. As such, they are not publicly available and approaches which are based on model sheets will not be scalable for all movies.

In 1981, Frank Thomas and Ollie Johnston published *The Illusion of Life* [21]; it outlines a set of twelve basic principles of animation. Animators have been using this as a cookbook for designing characters in order for the viewers to appreciate "animation" over mere "movement". While most of these principles aid animators in adding semantic or artistic value (e.g., anticipation, exaggeration), a few can be exploited in a computer vision context (e.g., *Solid Drawing*: drawing volume solidity and illusion of three dimensions; *Staging*: Distinctive color, depth of field and positioning in the frame to highlight the character). Defining an *animated character* in a complete sense would involve delineating abstract concepts such as life (or sentience even) from movie content. In this paper, we only analyze the video stream from animation movies and leverage some of the aforementioned principles of animation as proxies to identify the characters.

At the outset, we pose our problem as an object detection task where any object can be a possible *character candidate*. Animation movie frames are comparable with natural photographic images, especially in their similarities of depth of field and the character presentation in a frame. Additionally, we assume no prior models with respect to shape, size, color, or texture for these candidates in order for the proposed system to generalize.

A few prominent examples of state-of-the-art object detection systems include discriminatively trained deformable parts-based model (DPM, [22], [23]) and deep neural network (DNN) models such as [24]–[26], both of which are supervised and trained over a predefined set of object classes. DPMs need a carefully designed part-decomposition model of an object which makes it unsuitable given the heterogeneity of characters within just a single movie. In contrast, DNN-based methods such as [24] can detect objects in real-time and outperform DPMs. Specifically, DNN models that are saliency-inspired in design [25] are of interest for our problem statement. Although supervised with a finite set of object classes, they have been shown to detect objects in a *class-agnostic* manner [26] i.e., detect classes of objects not used for training the model.

Movies in general, portray only a handful of *prominent* characters. They are more likely to appear frequently in order for the viewer to easily comprehend the content and the plot of

the movie. Additionally in movies, characters or the objects-of-interest tend to remain on screen for up to a few seconds depending on the situation. Visual object tracking can be used as an effective method to segment characters locally in time. Several previous works have used tracking as a means to automatically detect a class of objects (e.g., pedestrians, [27]). Object tracking algorithms can be error-prone in a movie video environment because of object deformation, background clutter, changes in illumination, occlusion and lack of a stationary backgrounds. However, visual tracking can minimize the number of detected objects to be considered by accounting for minor deformation or linear motion of the object. Furthermore, tracking also provides time information that can be used for diarization subsequently. For example, in [11], supervisory information available on a profile face is used to learn the appearance of a frontal face from faces tracked in TV series. A reasonable assumption in describing animated character is that the prominent characters are not transient when presented on-screen and appear frequently in the movie. In our method, we use this aspect of character presentation in movies to select character candidates. As a result, the character dictionaries consist of only the frequently occurring characters.

In this paper, we propose a novel approach to automatically discover characters that appear in an animation movie. Our proposed method is unsupervised in the sense that we do not train any aspect of our system with data from animated media content. Furthermore, we use no specific knowledge of the animation style or the physical attributes of the animated characters, thereby ensuring that our system can scale and generalize through the whole spectrum of animation movie content.

The rest of the paper is organized as follows: Section II describes the proposed system for selecting character candidates from an animation movie. In Section III, we present the experiments performed and the creation of an evaluation database. Section IV contains the experimental results and final considerations followed by conclusions and future work in Section V.

II. Methods

In this section, we first introduce the different systems that we use to identify and prune the detected objects to obtain a set of possible character candidates. We then use a clustering approach to identify character exemplars that constitute the final character dictionary. The overview of the proposed system is shown in the Fig. 3.

Our animation movie database consisted of forty-six movies, for which we annotated their prominent characters. We then conducted a detailed performance evaluation on eight animation movies which were chosen to represent varying degrees of heterogeneity in character design and composition. The movie-cast data from forty-six movies used for our system evaluation and the output from our system has been released as part of the SAIL Animation Movie character Database (SAIL-AMDb).[1] We have also made the code publicly available.[2]

[1]https://github.com/usc-sail/mica-animation/wiki
[2]https://github.com/usc-sail/mica-animation

A. Coarse Detection of Character Candidates

Animated characters are often designed to have the appearance of a 3D object and characterized by shallow focus where the image plane of the character is in focus while the rest of the frame is out of focus [21]. In other words, they are the salient objects in a given frame. Capitalizing on this, we define a *character candidate* as any object that can be detected by a general-purpose object detector.

We use a pre-trained deep neural network (DNN) called *MultiBox* [25], [26], designed for object detection. Our preliminary experiments with other region proposal networks such as [24] yielded similar results. We chose *MultiBox* since our motivation for using an object detector was only to generate an initial set of potential character candidates.

MultiBox is a convolutional neural network (CNN) with an inception-style architecture [28] trained with the full 200-category object detection challenge data set from ImageNet Large Scale Visual Recognition Challenge 2014 (ILSVRC-2014) [29]. This model generates multiple bounding boxes and an associated confidence score that quantifies the network's confidence of each box containing an object. The model has been shown to perform object localization in a *class-agnostic* manner and achieve state-of-the-art performance in object detection tasks [25]. Furthermore, since the network is tailored towards the localization problem, it achieves a scalable representation of multiple salient objects in an image. These features make this model uniquely suitable for our problem. It is important to note that this model is trained with natural images of distinct object classes. Although the authors in [25] have shown that the model generalizes over unseen classes, here we apply the pre-trained DNN for images sampled from animation movies. We refer to this discrepancy as *DNN training bias*. This results in detecting objects that are not characters in a movie (e.g., traffic-light, chair). We refer to such objects as *noisy objects*.

In order to reduce the computational time, we downsample a movie (originally encoded at 23.98 fps) by one frame every 0.42 s (every 10th frame). The resulting frames are input to MultiBox [25] to obtain all possible bounding boxes for each image. The confidence score that is returned with each of these boxes was originally optimized in the DNN to match the ground truth object boxes from natural images.

Because of the aforementioned *DNN training bias*, we generally observed lower range of confidence scores for objects detected that were animated characters. We chose to retain objects with a confidence score greater than 0.1. In order to determine this threshold, we randomly sampled 100,000 frames from the movie Frozen (2013) in our movie database. We first assumed to have at most five possibly overlapping objects of interest in one frame and obtained the confidence scores for the five most confident objects in each frame. We then examined the distribution of the confidence scores for all the objects detected. We set the confidence threshold to 75th percentile of the distribution of confidence scores which is equal to 0.1002, thus retaining all objects with confidence score greater than 0.1. We apply this confidence threshold for all the movies in our database. A few examples of objects detected and their confidence scores returned by the network are shown in Fig. 2.

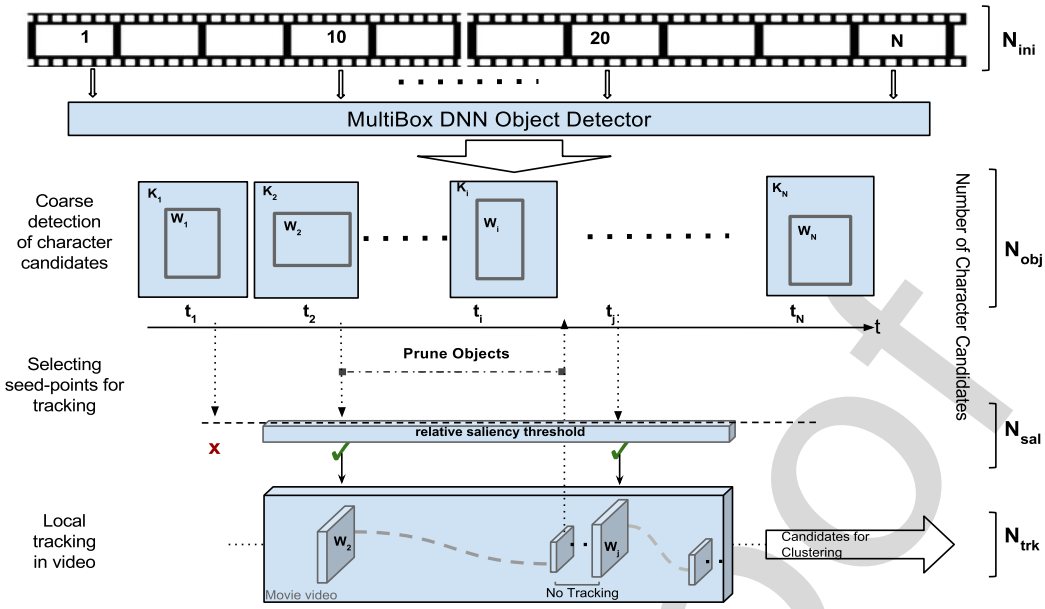

Fig. 3. Schematic diagram of the proposed method.

We also computed the area of each bounding box of an object relative to the image frame and excluded objects in bounding boxes with an area less than 1% or greater than 99% of the entire frame. This ensures that very small objects and holistic scenes are excluded as character candidates. When multiple objects were detected in a single frame, we pruned them to obtain at most one object per frame following the approach in [25]. We performed non-maximum-suppression with a Jaccard similarity [30] threshold of 0.5 and, chose the object with the maximum area in that frame. We identified only a single object per frame in order to simplify the subsequent step of single-target visual object tracking. We refer to a chosen frame containing a character candidate as a *candidate frame*.

A schematic of the proposed approach is illustrated in Fig. 3. Let N_{ini} be the initial number of images (movie frames) input to *MultiBox* and N_{obj} be the number of character candidates chosen. We denote the candidate frame K and the bounding box W enclosing the object as a set $\mathbf{M_{obj}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{obj}]\}$ and t_i refers to the time (or frame number) in the movie at which the object i occurs. Qualitative analyses showed that this step captures most of the characters in an animation movie at least once (e.g., images shown in Fig. 2(a)–(c)). However, this set also contains redundant and noisy objects which include non-characters or background objects (e.g., Fig. 2(d)).

B. Saliency Constraints and Object Tracking

In the next phase of our system, we used the saliency of the detected object as a constraint to prune the set of character candidates obtained in the previous step. We use this pruned set of candidate frames as seed-points for tracking. During tracking, we do not distinguish camera motion from object motion, thereby ensuring that a sufficient condition for a character candidate is its presence on the screen rather than motion (e.g., a talking tree).

Fig. 4. (a) Example for DNN training bias and saliency constraint; (b) Masked regions showing saliency, here *relative saliency score* $R_s(W_1) = 9.2\%$.

1) Saliency-Constrained Pruning: As described in Section II-A, the DNN training bias may result in choosing objects that, although salient, may not be the characters of interest (e.g., detected lamp in a scene with two characters—see Fig. 4(a)). To quantify this, we use a saliency measure proposed in [31] for the character candidate with respect to the entire frame. Unsupervised methods that estimate saliency typically use pixel-level features such as color, intensity (e.g., [32]) or background-detection in dynamic scenes (e.g., [33]). In contrast, the measure proposed in [31] estimates saliency of local areas (instead of pixel level) in static images and requires no training. This method uses a kernel-based approach where the size of the window relates to the scale of the target objects. The saliency of a pixel inside the window is estimated using the conditional probability of that pixel drawn from the distribution estimated inside that window versus the distribution of the surrounding area.

We first converted the RGB images to CIELAB color space (because of the perceptual uniformity of the CIE color space[3]) to estimate a saliency map for the entire candidate frame by

[3]http://www.brucelindbloom.com

choosing window sizes at different scales as described in [34]. The resulting saliency maps are binarized by setting values greater than 0.7 to 1 as recommended in [34]. An example of the saliency map is shown in Fig. 4(b). Let $A_s(W)$ be the area of the salient region contained within a bounding box, W in an image frame K. We define a *relative saliency score*, $R_s(W)$ of an object enclosed by the box W as the percentage salient area it contributes to the frame, K:

$$R_s(W) = \frac{A_s(W)}{A_s(K)} \times 100. \quad (1)$$

We obtained the *relative saliency score*, $R_s(W_{t_i}^{(i)})$ for every character candidate in the set $\mathbf{M_{obj}}$ from the *MultiBox* object detector. We used a threshold of 10% and retain only those character candidates which have a relative saliency score greater than this threshold. These candidates are next used as seed-points for tracking. This threshold was initially decided based on qualitative observation. We then conducted additional experiments to assess the effect of this threshold parameter as described in the Section III-C. The resulting set of *salient* character candidates is denoted as $\mathbf{M_{sal}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{sal}]\}$, where N_{sal} is the total number of objects deemed salient after this step with $|\mathbf{M_{sal}}| \leq |\mathbf{M_{obj}}|$ where $|\cdot|$ indicates the cardinality of the set.

2) Deformable Object Tracking: An important property of animated characters is their appearance on screen for up to a few seconds depending on the context. We utilized this property by performing a single-target visual tracking of the salient character candidates. Since animated characters are mostly deformable bodies, the rigidity assumption that most tracking algorithms employ in their motion models (for review, see [35]) does not hold. We employ a deformable object tracking algorithm [36] which does not impose rigidity assumptions on the object-of-interest while tracking.

This method first builds a static-appearance model of the object by clustering the key-points into sets of *inliers* (for the object body) and *outliers* (for the background) using a dissimilarity measure that quantifies the correspondences between key-points. The dissimilarity measure is estimated by computing the distance between the initial set of corresponding key-points and the transformed version. The model is then adaptively updated in time by propagating only the *inlier* correspondences by estimating the optical flow of the key-points. The degree of tolerance towards the deformation of the object is factored into the model by setting a parameter in the tracking algorithm which ensures that the cluster of inlier points are spatially localized. We used the BRISK [37] features for key-point detection and the parameters were set according to [36] after histogram equalization of the images.

Tracking every object from the set of salient character candidates for the full length of the movie is computationally expensive and may lead to accumulated tracking errors. Hence, we performed *local-tracking* in a serial and progressive fashion as described in **Algorithm 1**. We refer to the first *candidate frame* and the corresponding bounding box for the object of each track as a *seed-point*. *Local-tracking* substantially reduced the number of character candidates by eliminating objects that were successfully tracked in consecutive frames. As we performed single-target visual tracking, this process may also exclude other characters that co-occur within a given track. However, since *prominent* characters occur quite frequently in a movie, the issue of losing certain characters was not significantly noted. The duration of time for which an object is tracked is used as a threshold for retaining objects. We refer to this as the *track duration threshold*, τ and initially set to one frame. This would only eliminate the transient and/or spurious object detections. Additional experiments varying the τ parameter are conducted as discussed later. We denote the set of character candidates returned after tracking as $\mathbf{M_{trk}}$ with $|\mathbf{M_{trk}}| = N_{trk}$ such that $N_{trk} \leq N_{sal} \leq N_{obj}$ The number of character candidates obtained after pruning at each step as a percentage of the initial number of input frames is shown in Table II.

Algorithm 1: Local Tracking

Input: Set of salient character candidates: $\mathbf{M_{sal}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{sal}]\}$; movie, $\mathbf{V}$
Output: Set of track *seed-points*; $\mathbf{M_{trk}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{sal}]\}$ and corresponding track duration T_i
Parameters: Track duration threshold: τ

```
while ( V open) do
    M_trk = {}
    while ( M_sal ≠ ∅) do
        Begin tracking at the earliest time frame, i.e.,
        {(K_{t_j}^{(j)}, W_{t_j}^{(j)})} ← min_{∀t_j : j ≤ |M_sal|} {M_sal}
        Object tracking lost at t_k ≥ t_j
        Track duration, T_j ← t_k − t_j
        if ( T_j > τ) then
            Update tracked seed-points
            M_trk ← M_trk ∪ {(K_{t_j}^{(j)}, W_{t_j}^{(j)})}
            Prune character candidates
            M_sal ← M_sal ∩ {(K_{t_m}^{(m)}, W_{t_m}^{(m)}) | ∀m > k}
        else
            M_sal ← M_sal ∩ {(K_{t_m}^{(m)}, W_{t_m}^{(m)}) | ∀m > j}
        end
    end
    N_trk = |M_trk|
end
```

C. Exemplars for Character Representation

The character candidates chosen thus far may be redundant to some extent, and may contain multiple images with varying view-point or segments of the same object. In order to group similar objects together, we pose this as an unsupervised clustering problem with an unknown number of clusters. A suitable approach to represent such data is to identify a smaller set of samples, referred to as *exemplars*. We use affinity propagation (AP) clustering [38] to obtain exemplars which constitute the final *character dictionary* for a given movie. AP clustering is well suited for this problem because it is deterministic, achieves

TABLE I
DETAILS OF THE EVALUATION DATASET

ID	Movie (US Release year)	Duration(mins)	Prominent Characters†	Production Studio	Grossing (in $ millions)
V1	Cars 2 (2011)	107	10 (3)	Pixar	191
V2	Free Birds (2011)	91	11 (4)	Reel FX Creative	55
V3	Frozen (2013)	102	9 (4)	Walt Disney	400
V4	How to Train your Dragon 2 (2014)	102	12 (4)	DreamWorks	177
V5	Shrek Forever After (2010)	93	9 (5)	DreamWorks	238
V6	Tangled (2010)	100	9 (4)	Walt Disney	200
V7	The Lego Movie (2014)	101	12 (3)	Warner Animation	257
V8	Toy Story 3 (2010)	103	18 (9)	Pixar	415

† ( ) indicates number of minor characters.

TABLE II
PERCENTAGE OF INITIAL NUMBER OF OBJECTS AFTER EACH STEP OF PRUNING ON THE EVALUATION DATASET

Movie ID	N_{ini}	$N_{obj}(\%)$	N_{sal}**(%)***	N_{trk}**(%)**$^+$
V1	15395	19.88	16.99	5.61
V2	13102	14.08	12.50	5.01
V3	14676	9.36	6.83	2.56
V4	14676	9.25	6.32	3.17
V5	13406	10.61	8.06	3.32
V6	14372	9.42	8.22	3.05
V7	14460	9.37	6.96	2.92
V8	14748	11.80	9.79	3.79

*relative saliency threshold = 10%. + track duration threshold = 1 frame.

a lower clustering error compared to other clustering methods such as k-means [39] and does not require a predetermined number of clusters.

We used the *ImageNet* model proposed in [40] to extract features to cluster the character candidates. Several previous works (e.g., [41]) have shown that feature representations from fully-connected layers in a CNN generalize well for various image recognition tasks. Specifically, we use a 4096-dimensional feature from the second fully connected layer, "FC7" from the ImageNet model which was trained with ILSVRC-2012 [29] competition data.

Because the FC7 features are sparse, we use cosine distance to compute a pairwise similarity matrix, $\mathbf{S_{ij}}$ between the feature vectors, $\{\mathbf{v_i}\}$

$$\mathbf{S_{ij}} = \frac{\mathbf{v_i}\mathbf{v_j}^T}{\|\mathbf{v_i}\|\|\mathbf{v_j}\|} \forall i, j \in [1, N_{trk}]. \quad (2)$$

The appearance of most characters is somewhat homogeneous (except for variations in pose and deformation) throughout a movie in terms of shape, color or attire of the character. Leveraging this observation, we also used GIST descriptors [42] for clustering. GIST features provide a low dimensional representation that describes the prominent spatial structure in an image. GIST features have been used for clustering tasks such as scene clustering (e.g., [43]) with some success. We obtained a 960-dimensional GIST descriptor for the character candidates using *pyleargist*[4] package in Python. We then computed negative Euclidean distance between all the candidates from a movie to form a similarity matrix for clustering. Additionally, we also evaluated the clustering performance of GIST and FC7 features.

We used the AP algorithm proposed in [44] to cluster the similarity matrices obtained from the character candidates. The goal of AP clustering is to choose a character candidate j to be the exemplar of the $i$thcandidate. Define *responsibility* $r(i, j)$: degree of support that the candidate j should be the exemplar of i and *availability* $a(i, j)$: degree of support by which the candidate i should choose j to be its exemplar. Initialize $r(i, j), a(i, j) = 0; \forall i, j$ and update responsibility and availability as below:

$$r(i, j) \leftarrow \mathbf{S_{ij}} - \max_{k:k\neq j}(a(k, i) + \mathbf{S_{ik}}) \quad (3)$$

$$a(j, j) \leftarrow \sum_{k:k\neq j} \max[0, r(k, j)] \quad (4)$$

$$a(j, i) \leftarrow \min\big(0, r(j, j) + \sum_{k:k\notin(j,i)} \max[0, r(k, j)]\big). \quad (5)$$

Introduce a damping factor, $\lambda \in [0, 1)$ to account for numerical oscillations over iterations in time t

$$r(j, i)_t \leftarrow (1 - \lambda)r(j, i)_t + \lambda r(j, i)_{t-1} \quad (6)$$

$$a(j, i)_t \leftarrow (1 - \lambda)a(j, i)_t + \lambda a(j, i)_{t-1}. \quad (7)$$

Pick j to be an exemplar of i if

$$\arg\max_j (r(i, j) + a(j, i)). \quad (8)$$

We set the damping factor, λ which controls the update of $r(i, j)$ and $a(i, j)$ in each step to 0.5. Changing this parameter had no effect on the exemplars we obtain. Let N_{xmp} be the total number of exemplars returned.

AP clustering works well with animation movies since the appearance of most characters (e.g., attire) is consistent within a given movie and the features we used for clustering can capture these attributes. An additional benefit of using AP clustering is that the number of exemplars (i.e., the size of character dictionary) need not be pre-specified. On the other hand, we risk *over-clustering*, i.e., a single character may be represented by multiple exemplars since the features we use are generic and not designed to capture variation in scale, orientation or view-point of a character. This was evident when we performed a second pass of AP clustering on the exemplars obtained here and failed to cluster the *perceptually identical* characters together. In

[4]https://pypi.python.org/pypi/pyleargist

order to penalize for over-clustering, we define an *over-clustering index* in our performance evaluation measures as described in Section III-C.

III. Experiments

The problem of identifying character dictionaries for animation movies addressed in this paper is unique. Due to the lack of existing performance evaluation frameworks for this task, we first created a *reference character dictionary* (movie-cast) for each movie in our database. We then used these reference character dictionaries as ground truth to evaluate the character dictionaries output by the proposed method. These reference character dictionaries have been made publicly available as a part of the SAIL-AMDb[5] along with outputs used for our system evaluation.

A. Evaluation Database

Our animation movie database consisted of a total of forty-six movies produced between 2010–2014. Of the forty-six movies available, we chose eight top-grossing movies to evaluate the performance of our method in greater detail and to determine the best parameter choices for *relative saliency threshold* and the *track duration threshold*. The year of release, duration, production company and size of the reference character dictionary are shown in Table I. For brevity, we refer to these movies as V1–V8.

These eight movies were chosen to test the generalizability of the proposed system. They represent a diverse set of characters in terms of design and composition produced by prominent animation studios. These movies include instances of human or human-like characters (V3, V5, V6), non-human but anthropomorphic (V3, V5), toy-like (V7, V8) and animals (V2, V4, V5). All movies (except V6) include at least one instance of a character which is abstract in design. The dataset includes movies with varying degrees of illumination, background/environment and motion of the characters. For example, V1, V6 and V8 have overall higher illumination compared to V3, V4 and V5. The movies V1 and V4 have faster moving characters (e.g., dragons and cars) compared to the others. Quantitative analyses to evaluate the diversity of this dataset (e.g., variation in color, illumination or other characteristics) are beyond the scope of this paper (and an objective of our future work).

As described in Section II-C, the character dictionary output by the proposed system for each movie are the exemplars identified by AP clustering. The character candidates on which the clustering is performed are obtained by optimizing two system parameters using a grid search: relative saliency threshold and track duration threshold. The settings used for the two parameters are $R_s(X) = \{0, 10, 20, 50, 80, 90\}$ and $\tau = \{1, 12, 24, 48, 120\}$. The values for τ (in frames) correspond to the least possible value (one frame), and approximately 0.5 s, 1 s, 2 s and 5 s of the movie duration respectively.[6]

B. Reference Character Dictionaries

We borrow the same definitions for a character as described in [45] and [46] to create a movie-specific *reference character dictionary*. All named characters (speaking and non-speaking) displayed on-screen were included. Similar to [46], we first used the set of prominent characters as listed by a leading online box-office reporting service.[7] The designation of a *minor character* available in this resource was retained. This list however, does not include non-speaking characters (e.g., dragons). Hence, if a character was given a specific name in the movie (as opposed to generic names such as *a Spanish ambassador*), we included them in the reference. For each of these characters, we obtained a representative full-body image from the movie posters or DVD covers available online. If the said character was absent in these sources, a representative image was manually obtained from the internet. The number of prominent characters including the number of minor characters are listed in Table I. For annotation purposes, all characters in the reference dictionaries are assigned a unique ID to preserve character anonymity.

We use annotations from Mechanical Turk workers (MTurk; a crowdsourcing platform by Amazon Web Services) to compare the *proposed* and *reference* character dictionaries. As discussed in Section II-C, the exemplars in the proposed dictionaries may vary from the representative image used to construct the reference. Hence, by using MTurk, we leverage the human perceptual ability to match the exemplars with the items in the reference. The annotators are instructed to consider an exemplar to be a match if 1) it is identifiable regardless to variation in scale, illumination, orientation or viewpoint or 2) an identifiable segment of the reference character is present in the exemplar or 3) if the exemplar consists of the said reference character. The annotators indicate a match with a unique ID available for every character in the *reference*. Furthermore, if an exemplar consists of multiple reference characters, the annotators are instructed to list all the relevant IDs. Three different annotations were acquired for each of the exemplars from unique annotators. In order to check for possible confounding factors, additional information on whether the annotator had watched the movie prior to annotating was also collected.

We performed an inter-rater reliability analysis to ensure that the MTurk annotations were reliable. Since we obtained more than two annotations, inter-rater agreement (more specifically, inter-annotation agreement) was quantified using Krippendorf's alpha [47] for each movie. The categorical values that were used to compute this measure were the unique IDs assigned to each character from the reference. Krippendorf's alpha was high for the eight movies used in our system evaluation with mean/standard deviation of $\alpha = 0.81 \pm 0.05$ indicating strong agreement. Across all forty-six movies, Krippendorf's Alpha was similarly high (0.82). Furthermore, no difference in agreement was observed between the set of annotations performed by workers who had watched the movie and those who had not. Following high agreement, we obtained a single annotation per exemplar by performing simple majority voting on the

[5] https://goo.gl/WbESbz

[6] Frame rate for all movies in the dataset was 23.98 fps

[7] www.boxofficemojo.com

three annotations. Three-way ties were resolved with random assignment.

C. Performance Evaluation

The performance of our method for different experiments was quantified by comparing the reference character dictionaries with the output dictionaries from the proposed method. We refer to the set of exemplars in the proposed dictionary that were successfully matched to a character in the reference as the *relevant exemplars* and the remaining as, the *noisy exemplars*. As described earlier, multiple exemplars can represent a single character. Therefore, we examine the unique set of character IDs in the proposed dictionary (*matched characters*) and those never identified (*missed characters*). Following this, we compute three measures; precision, P, recall, R and F1 score, F_1 as follows:

$$P = \frac{|\{relevant\ exemplars\}|}{|\{relevant\ exemplars\} \cup \{noisy\ exemplars\}|} \quad (9)$$

$$R = \frac{|\{matched\ characters\}|}{|\{matched\ characters\} \cup \{missed\ characters\}|} \quad (10)$$

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}. \quad (11)$$

Additionally, we define *over-clustering index* as a measure to quantify the extent to which multiple exemplars per character appear in our character dictionaries. In other words, the extent to which we *over-cluster* the relevant characters. Over-clustering index for a movie is computed as the median of number of exemplars per character in the set of the relevant exemplars. Since this metric is defined only over the set of relevant exemplars, it is independent of precision. It is bounded below by 1 (one exemplar per character) and bounded above by N_{xmp} (all exemplars represent just one character).

In order to compare the clustering performance of GIST and FC7 features, we measure the *purity* of clustering as described in [48]. We assign each cluster to the most frequently occurring character in that cluster. Then, the we measure purity by counting the total number of correctly assigned characters, across all clusters and dividing by the total number of candidates clustered (N_{trk}) as below:

$$purity = \frac{1}{N_{trk}} \sum_{k} \max_{j} |\omega_k \cap c_j| \quad (12)$$

where *purity* $\in [0, 1]$, $\Omega = \{\omega_1, \omega_2, ..., \omega_k\}$ is the set of all clusters and $\mathcal{C} = \{c_1, c_2, ..., c_j\}$ is the set of all relevant exemplars.

By our definition of precision (9), a lower value would indicate that character candidates which are not listed in the reference were identified as exemplars. These *noisy exemplars* could either be a result of minor characters not being listed in the reference or background objects being identified as exemplars. Similarly, a high recall (10) would reflect the ability to identify all the prominent characters at least once. Ideally, recall $= 1.0$ and over-clustering index $= 1$ would indicate that every character in the reference was detected by exactly one relevant exemplar. Higher values of the over-clustering index reflect on the failure to cluster similar character candidates. This is likely a consequence of the features not being invariant to the orientation, view-point or scale of the character candidates. Complementary to precision, recall and F1 score which measure the performance of clustering with respect to a reference, purity (12) measures the extent to which clusters belonged to a single character, thus evaluating the features (FC7 versus GIST) used for clustering.

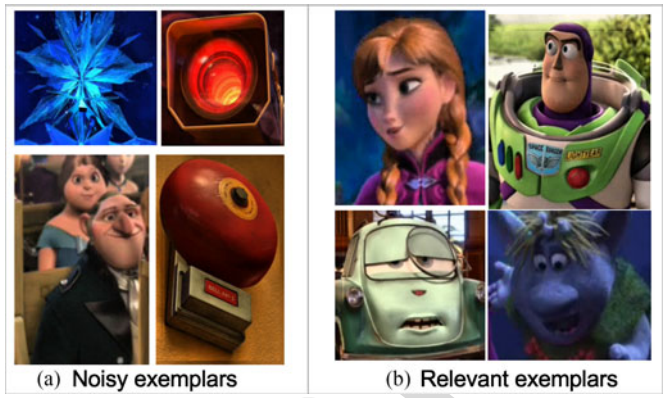

Fig. 5. Examples of noisy and relevant exemplars.

The F1 score, precision and recall measures for all eight movies are averaged for each experiment to determine the best choice of relative saliency threshold and track duration threshold. These optimal parameters were used to obtain character dictionaries for the remaining thirty-eight movies in our evaluation dataset.

IV. Results and Discussion

A few examples of the *relevant* and *noisy* exemplars from the proposed character dictionaries are shown in Fig. 5. As described earlier, exemplars are categorized as relevant or noisy based on a reference dictionary constructed for each movie. One source of noisy exemplars is how we construct these reference dictionaries. Since the reference consists of only the prominent characters, it may result in some minor characters being categorized as noisy (See bottom-left image in Fig. 5(a)).

The second source of noisy exemplars is the training data used for the *MultiBox* object detector which comprised only of natural images. Characters which belong to object classes that the DNN was trained on tend to get detected more often and consistently (e.g., traffic lights, bell). The subsequent steps in our method that use relative saliency score and local-tracking attempt to eliminate some of these noisy exemplars. However, depending on the frequency of occurrence or saliency of the character candidates, they may not always be successfully pruned. Table II shows the percentage of the input frames pruned at each step. The proposed character dictionaries for three movies; V1, V2 and V3 are shown in Fig. 10–12 in Appendix B.

The precision, recall and F1 score measures that we used to quantify the performance of our method are shown in Fig. 6. The relative saliency threshold and track duration threshold were chosen corresponding to the best F1 score (highlighted in Fig. 6(a)). These measures were averaged across the eight movies for each setting of two parameters, relative saliency

Fig. 6. Average (a) F1 score, (b) Precision and (c) Recall for all experiments.

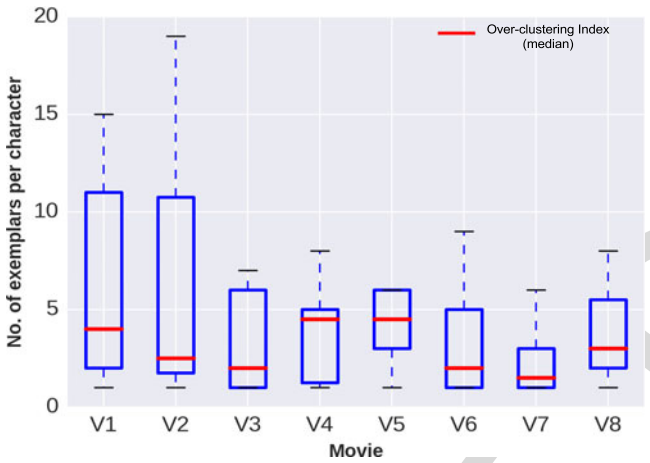

Fig. 7. Distribution of number of exemplars per character in each movie for $R_s(X) = 10\%$ and $\tau = 1$.

TABLE III
F1 SCORE AND PURITY FOR FC7 AND GIST FEATURES USED IN CLUSTERING

Movie ID	FC7 features		GIST descriptors	
	F1 score	Purity	F1 score	Purity
V1	0.691	0.708	0.713	0.414
V2	0.773	0.651	0.769	0.345
V3	0.825	0.842	0.821	0.304
V4	0.764	0.598	0.693	0.322
V5	0.532	0.712	0.653	0.408
V6	0.740	0.677	0.732	0.398
V7	0.732	0.693	0.743	0.438
V8	0.752	0.745	0.799	0.392
Average:	**0.726**	**0.703**	**0.740**	**0.378**

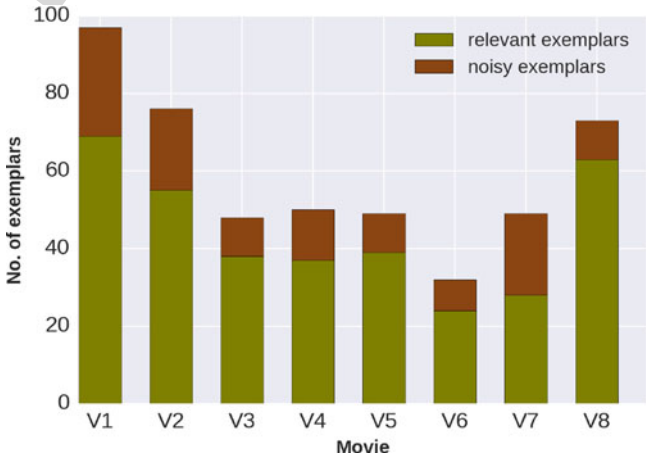

Fig. 8. Number of relevant and noisy exemplars for each movie with $R_s(X) = 10\%$ and $\tau = 1$.

threshold, $R_s(X)$ and track duration threshold, τ. Overall, recall is high (over 80% for $\tau = 1$ and $R_s(X) = 10\%$) which indicates that our proposed character dictionaries were able to identify most of the characters in the reference at least once. Precision ranges between 70% and 90% indicating that less than one-third of exemplars in our proposed dictionaries are noisy.

We note that the recall measure defined here has to be interpreted alongside over-clustering index; a metric that captures the extent to which multiple exemplars represent a single reference character. The distribution of number of relevant exemplars per character for the eight movies is shown in Fig. 7. The median number of exemplars per character, i.e., the over-clustering index is less than 5 for all the eight movies. As described in Section III-C, this measure lies between 1 and the number of exemplars. Here, the number of exemplars range between 35 and 95 (with $R_s(X) = 10\%; \tau = 1$) but the over-clustering index is less than 5 which reflects on the effective performance of the affinity propagation (AP) algorithm used for clustering.

Additionally, we compared the F1 score and purity of clustering for the eight movies, in order to evaluate the features used in clustering, as shown in Table III. Although the F1 scores (computed by comparing the exemplars to the reference) were similar between the two descriptors, the clustering purity using FC7 features was significantly higher (paired t-test, $p \ll 0.01$ to reject $H_0 : \mu_0 \leq \mu_1$) than that of GIST descriptors. This indicates that FC7 features yield less noisy and more homogeneous clusters from AP clustering. Furthermore, FC7 features perform better for clustering than GIST features, perhaps because *ImageNet* was trained to classify objects robust to variation in the the background or view-point and occlusions, whereas GIST descriptors capture the holistic shape information in an image.

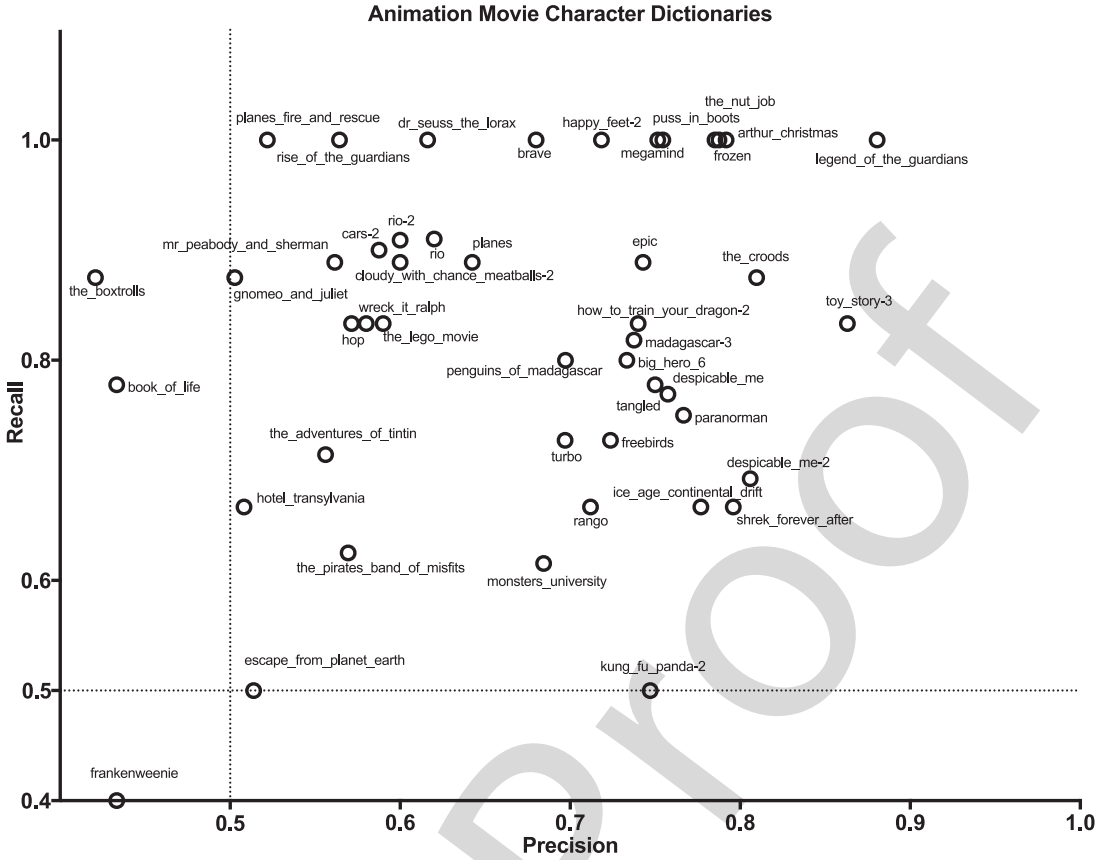

Fig. 9. Precision versus recall for forty-six movies.

As shown in Fig. 6(c), recall drops with an increase in τ as expected. Since, by increasing τ we retain only those character candidates which remain longer on-screen and do not always co-occur with other salient objects. This results in excluding some relevant exemplars. In contrast, an increase in precision (See Fig. 6(b)) is noticed on increasing τ since a few noisy character candidates that are infrequent get pruned successfully. Relative saliency threshold had the desired effect on the system output i.e., increasing $R_s(X)$ results in an increase in precision. However, these gains in precision by increasing $R_s(X)$ beyond 10% were not substantial.

In order to determine a good choice of the system parameters, we examine F1 score for different combinations of $R_s(X)$ and τ as shown in Fig. 6(a). $R_s(X) = 10\%$ and $\tau = 1$ would be the best choice of settings. For these settings, the number of relevant and noisy exemplars for the eight movies are shown in Fig. 8. It is interesting to note that movies V1 and V2 have relatively larger character dictionaries and a higher range of number of exemplars per character (See Appendix Fig. 10–11). All the characters in the movies, V1 and V2 are similar to cars and birds in appearance. The results at a glance show that all instances of these characters in different scenes were detected in these movies (which include the minor characters and different appearances of the same character with respect to view-point). This is likely because both cars and birds are among the object classes in ILSVRC-2014 data used to train *MultiBox*.

Character dictionaries for the remaining thirty-eight movies were obtained with the choice of $R_s(X)$ and τ determined above. The range of precision was 0.45–0.89 (mean/standard deviation: 0.66 ± 0.12) and recall: 0.42–1.0 (0.83 ± 0.16). The range of over-clustering index was 1.5–6.0 (3.5 ± 1.5). See Fig. 9 (Appendix A) for precision and recall measures of all the forty six movies in our dataset. Further error analysis considering different aspects of all the movies (e.g., character design, color, illumination) is warranted and will be a part of our future work.

We note that the movie Frankenweenie (2013) which was produced in black and white has the lowest precision and recall in our dataset. This indicates that color rendering is an important factor since the DNNs we employ were trained with RGB images. The movies, Boxtrolls (2014) and The Book of Life (2014) both have a low precision and high recall indicating a larger number of noisy exemplars. On analyzing the errors in these samples, we observed that the local tracking method pruned approximately 42% of the initial character candidates (c.f. the average percentage of candidates pruned by local tracking for the rest of the movies was 62.23%). This is likely because these movies, unlike the others in the dataset use a *rapid-fire* film editing style which includes fast-action scene cuts and rapidly changing backgrounds which are not ideally suited for visual object tracking.

On the other hand, movies like Kung Fu Panda 2 (2011) and Escape from Planet Earth (2013) yield high precision and low

Fig. 10. Character dictionary of the movie Frozen: precision = 0.81, recall = 1.0; over-clustering index=2.

recall. This is likely because these movies feature only a small number of prominent characters and a larger number of unnamed characters which are not included in the reference dictionaries that we created. As expected, movies that feature distinct lifelike animals or humans, generally performed the best. For example, the movie Legend of Guardians (2010) featured only birds and The Nut Job (2014) featured animals – both animals and birds are included in the set of object categories of the ILSVRC datasets.

V. Conclusions and Future Work

In this paper, we proposed an unsupervised method to automatically create a dictionary of characters from an animation movie. We evaluated our method on a set of eight movies with diverse character styles and demonstrated high precision and recall on a dataset of forty-six movies. We also showed that the proposed method generalizes for animation movies at scale. These character dictionaries can serve as a powerful tool for character labeling to delineate aspects of *who appeared*, *when* and for *how long* in a movie (video diarization). We believe that our efforts can lay a foundation to provide an impetus for multimedia research endeavors specifically involving animated media content.

One of the drawbacks of the proposed method is that we use an object detector that was trained with natural images. We plan to address this issue using transfer learning to adapt the existing models to specialize the network for detecting characters from animation movies. The relevant and noisy exemplars that we annotated for the system evaluation can potentially be used for these methods. Our future work would also include using the relevant exemplars and associated cluster members as a single unit to facilitate robust video diarization of animation movies.

Fig. 11. Character dictionary of the movie Free Birds: precision = 0.72, recall = 0.72; over-clustering index = 2.5.

Appendix A
Precision and Recall of our Proposed Method for the Forty-Six Movies

The Fig. 9 plots precision vs. recall for all the movies in our dataset. The relative saliency threshold and track duration threshold was set to 10% and one frame respectively (tuned on a subset of 8 movies as described in Section IV).

Appendix B
Examples

Fig. 10–12 illustrate the proposed character dictionaries for three movies with the settings of relative saliency threshold =

Fig. 12. Character dictionary of the movie Cars-2: precision=0.61, recall = 0.9; over-clustering index = 4.

10% and track duration threshold = 1 frame. The exemplars here are arranged in no particular order to maintain their aspect ratios.

REFERENCES

[1] P. P. Mohanta, S. K. Saha, and B. Chanda, "A model-based shot boundary detection technique using frame transition parameters," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 223–233, Feb. 2012.

[2] C. Liu, D. Wang, J. Zhu, and B. Zhang, "Learning a contextual multi-thread model for movie/TV scene segmentation," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 884–897, Jun. 2013.

[3] B. W. Chen, J. C. Wang, and J. F. Wang, "A novel video summarization based on mining the story-structure and semantic relations among concept entities," *IEEE Trans. Multimedia*, vol. 11, no. 2, pp. 295–312, Feb. 2009.

[4] Y. Li, S.-H. Lee, C.-H. Yeh, and C. C. J. Kuo, "Techniques for movie content analysis and skimming: tutorial and overview on video abstraction techniques," *IEEE Signal Process. Mag.*, vol. 23, no. 2, pp. 79–89, Mar. 2006.

[5] K. Kurzhals, M. John, F. Heimerl, P. Kuznecov, and D. Weiskopf, "Visual movie analytics," *IEEE Trans. Multimedia*, vol. 18, no. 11, pp. 2149–2160, Nov. 2016.

[6] C. Y. Weng, W. T. Chu, and J. L. Wu, "Rolenet: Movie analysis from the perspective of social networks," *IEEE Trans. Multimedia*, vol. 11, no. 2, pp. 256–271, Feb. 2009.

[7] T. Guha, C.-W. Huang, N. Kumar, Y. Zhu, and S. S. Narayanan, "Gender representation in cinematic content: A multimodal approach," in *Proc. ACM Int. Conf. Multimodal Interaction*, 2015, pp. 31–34.

[8] A. Fitzgibbon and A. Zisserman, "On affine invariant clustering and automatic cast listing in movies," *Proc. 7th Eur. Conf. Comput. Vis.*, 2002, pp. 304–320.

[9] F. Vallet, S. Essid, and J. Carrive, "A multimodal approach to speaker diarization on TV talk-shows," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 509–520, Apr. 2013.

[10] J. Sivic, M. Everingham, and A. Zisserman, "Who are you?"—Learning person specific classifiers from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2009, pp. 1145–1152.

[11] M. Everingham, J. Sivic, and A. Zisserman, "Hello! My name is... Buffy"—automatic naming of characters in TV video," in *Proc. Brit. Mach. Vis. Conf.*, 2006.

[12] M. Everingham, J. Sivic, and A. Zisserman, "Taking the bite out of automated naming of characters in TV video," *Image Vis. Comput.*, vol. 27, no. 5, pp. 545–559, Apr. 2009.

[13] "Box office history for digital animation" [Online]. Available: http://www.the-numbers.com/market/production-method/digital-animation

[14] Z. Aghbari, K. Kaneko, and A. Makinouchi, "Content-trajectory approach for searching video databases," *IEEE Trans. Multimedia*, vol. 5, no. 4, pp. 516–531, Dec. 2003.

[15] B. Ionescu, V. Buzuloiu, P. Lambert, and D. Coquin, "Improved cut detection for the segmentation of animation movies," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2006, vol. 2, pp. II–II.

[16] B. Ionescu, D. Coquin, P. Lambert, and V. Buzuloiu, "Fuzzy color-based approach for understanding animated movies content in the indexing task," *J. Image Video Process.*, vol. 2008, pp. 8:1–8:17, Jan. 2008.

[17] L. Ott, P. Lambert, B. Ionescu, and D. Coquin, "Animation movie abstraction: Key frame adaptative selection based on color histogram filtering," in *Proc. 14th Int. Conf. Image Anal. Process.*, 2007, pp. 206–211.

[18] B. Ionescu, P. Lambert, D. Coquin, L. Ott, and V. Buzuloiu, "Animation movies trailer computation," in *Proc. 14th ACM Int. Conf. Multimedia*, 2006, pp. 631–634.

[19] K. Takayama, H. Johan, and T. Nishita, "Face detection and face recognition of cartoon characters using feature extraction," in *Proc. Image, Electron. Visual Comput. Workshop*, 2012, p. 48.

[20] S. Wang, J. Zhang, T. X. Han, and Z. Miao, "Sketch-based image retrieval through hypothesis-driven object boundary selection with HLR descriptor," *IEEE Trans. Multimedia*, vol. 17, no. 7, pp. 1045–1057, Jul. 2015.

[21] *The Illusion of Life: Disney Animation*. New York, NY, USA: Abbeville Press, 1981.

[22] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[23] Y. Tang, X. Wang, E. Dellandrea, and L. Chen, "Weakly supervised learning of deformable part-based models for object detection via region proposals," *IEEE Trans. Multimedia*, vol. 19, no. 2, pp. 393–407, Feb. 2017.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2016, pp. 779–788.

[25] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2014, pp. 2155–2162.

[26] C. Szegedy, S. E. Reed, D. Erhan, and D. Anguelov, "Scalable, high-quality object detection," arXiv:1412.1441, 2014.

[27] K. H. Lee and J. N. Hwang, "On-road pedestrian tracking across multiple driving recorders," *IEEE Trans. Multimedia*, vol. 17, no. 9, pp. 1429–1438, Sep. 2015.

[28] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Jun. 2015.

[29] O. Russakovsky *et al.*, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[30] P. Jacard, "The distribution of the flora in the alpine zone," *New Phytologist*, vol. 11, pp. 37–50, 1912.

[31] E. Rahtu and J. Heikkilä, "A simple and efficient saliency detector for background subtraction," in *Proc. 12th Int. Conf. Comput. Vis. Workshops*, Sep. 2009, pp. 1137–1144.

[32] Y. Hu, X. Xie, W.-Y. Ma, L.-T. Chia, and D. Rajan, "Salient region detection using weighted feature maps based on the human visual attention model," in *Proc. 5th Pacific Rim Conf. Multimedia Adv. Multimedia Inf. Process.*, 2005, pp. 993–1000.

[33] V. Mahadevan and N. Vasconcelos, "Background subtraction in highly dynamic scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Jun. 2008, pp. 1–6.
[34] E. Rahtu, J. Kannala, M. Salo, and J. Heikkilä, "Segmenting salient objects from images and videos," *Proc. 11th Eur. Conf. Comput. Vis. Comput. Vis.,* 2010, pp. 366–379.
[35] M. Kristan *et al.*, "The visual object tracking VOT2015 challenge results," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2015.
[36] G. Nebehay and R. Pflugfelder, "Clustering of static-adaptive correspondences for deformable object tracking," *Comput. Vis. Pattern Recogn.*, Jun. 2015, pp. 2784–2791.
[37] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2548–2555.
[38] D. Dueck and B. J. Frey, "Non-metric affinity propagation for unsupervised image categorization," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.
[39] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.
[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[41] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn. Workshops*, 2014, pp. 512–519.
[42] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, May 2001.
[43] M. Merler, B. Huang, L. Xie, G. Hua, and A. Natsev, "Semantic model vectors for complex video event recognition," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 88–101, Feb. 2012.
[44] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, pp. 972–977, 2007.
[45] "Comprehensive Annenberg Report on Diversity in Entertainment," 2016.
[46] S. L. Smith *et al.*, "Media, diversity, & social change initiative," 2016.
[47] K. Krippendorff, "Reliability in content analysis," *Human Commun. Res.*, vol. 30, no. 3, pp. 411–433, 2004.
[48] D. M. Christopher, R. Prabhakar, and S. Hinrich, "Introduction to information retrieval," *Introduction Inf. Retrieval*, vol. 151, p. 177, 2008.

Krishna Somandepalli received the Master's degree from University of California at Santa Barbara, CA, USA, in electrical and computer engineering. He received the Bachelor's degree in electronics and communication engineering from University Visvesvaraya College of Engineering, Bangalore, India. Following his Master's degree, he worked as an Assistant Research Scientist at NYU Langone Medical Center, New York, NY, USA. His research interests include multimodal analysis with image and signal data. He is currently working toward the Ph.D. degree in the Signal Analysis and Interpretation Laboratory Group, Department of Electrical Engineering, University of Southern California, Los Angeles, CA, USA.

Naveen Kumar received the Ph.D. degree in electrical engineering from the USC Viterbi School of Engineering, where he was a member of the Media Informatics and Content Analysis Group, Signal Analysis and Interpretation Lab. He received the B.Tech. degree in instrumentation engineering from the Indian Institute of Technology, Kharagpur, India, in 2009. He currently works at the Sony PlayStation R&D in San Mateo, CA, USA. His broad research interests include machine learning and signal processing for speech, multimedia and multimodal applications.

Tanaya Guha received the Ph.D. degree in electrical and computer engineering from the University of British Columbia (UBC), Vancouver, BC, Canada. She is an Assistant Professor in the Department of Electrical Engineering, Indian Institute of Technology (IIT) Kanpur. Prior to joining IIT Kanpur, she was a Postdoctoral Fellow at the Signal Analysis and Interpretation Lab (SAIL), University of Southern California.

Her current research interests include social and affective computing, multimedia analysis, and multimodal signal processing. Dr. Guha received Mensa Canada Woodhams Memorial Scholarship, Google Anita Borg Scholarship, and Amazon Grace Hopper celebration scholarship.

Shrikanth S. Narayanan (SM'88–M'95–SM'02–F'09) is the Niki & C. L. Max Nikias Chair in Engineering at the University of Southern California (USC), Los Angeles, CA, USA, and holds appointments as a Professor of Electrical Engineering, Computer Science, Linguistics, Psychology, Neuroscience and Pediatrics, the Research Director of the Information Science Institute, and as the Founding Director of the Ming Hsieh Institute. Prior to USC, he was with AT&T Bell Labs and AT&T Research from 1995–2000. At USC, he directs the Signal Analysis and Interpretation Laboratory (SAIL). His research focuses on human-centered signal and information processing and systems modeling with an interdisciplinary emphasis on speech, audio, language, multimodal and biomedical problems and applications with direct societal relevance. Prof. Narayanan is a Fellow of the National Academy of Inventors, the Acoustical Society of America, the International Speech Communication Association (ISCA) and the American Association for the Advancement of Science (AAAS), and a member of Tau Beta Pi, Phi Kappa Phi, and Eta Kappa Nu. He is the Editor-in-Chief for IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, an Editor for the *Computer Speech and Language Journal* and an Associate Editor for the APSIPA *Transactions On Signal And Information Processing*. He was also previously an Associate Editor of the IEEE TRANSACTIONS OF SPEECH AND AUDIO PROCESSING (2000–2004), IEEE SIGNAL PROCESSING MAGAZINE (2005–2008), IEEE TRANSACTIONS ON MULTIMEDIA, (2008–2011), the IEEE TRANSACTIONS ON SIGNAL AND INFORMATION PROCESSING OVER NETWORKS (2014–2015), IEEE TRANSACTIONS ON AFFECTIVE COMPUTING (2010–2016), and the *Journal of the Acoustical Society of America* (2009–2017). He received several honors including Best Transactions Paper awards from the IEEE Signal Processing Society in 2005 (with A. Potamianos) and in 2009 (with C. M. Lee) and selection as an IEEE Signal Processing Society Distinguished Lecturer for 2010–2011 and ISCA Distinguished Lecturer for 2015–2016. Papers coauthored with his students have received awards including the 2014 Ten-year Technical Impact Award from ACM ICMI and at several conferences. He has published more than 750 papers and has been granted 17 U.S. patents.

993 **QUERIES**

994 Q1. Author: Please check whether the affiliation of authors is okay as set.
995 Q2. Author: Please provide year information in Ref. [13].
996 Q3. Author: Please update Ref. [26], if already published in prints.
997 Q4. Author: Please provide page range in Refs. [19], [28], and [35].
998 Q5. Author: Please provide full bibliographic details in Ref. [46].

Unsupervised Discovery of Character Dictionaries in Animation Movies

Krishna Somandepalli, *Member, IEEE*, Naveen Kumar, *Member, IEEE*, Tanaya Guha, *Member, IEEE*, and Shrikanth S. Narayanan, *Fellow, IEEE*

***Abstract*—Automatic content analysis of animation movies can enable an objective understanding of character (actor) representations and their portrayals. It can also help illuminate potential markers of unconscious biases and their impact. However, multimedia analysis of movie content has predominantly focused on live-action features. A dearth of multimedia research in this field is because of the complexity and heterogeneity in the design of animated characters—an extremely challenging problem to be generalized by a single method or model. In this paper, we address the problem of automatically discovering characters in animation movies as a first step toward automatic character labeling in these media. Movie-specific character dictionaries can act as a powerful first step for subsequent content analysis at scale. We propose an unsupervised approach which requires no prior information about the characters in a movie. We first use a deep neural network-based object detector that is trained on natural images to identify a set of initial character candidates. These candidates are further pruned using saliency constraints and visual object tracking. A character dictionary per movie is then generated from exemplars obtained by clustering these candidates. We are able to identify both anthropomorphic and nonanthropomorphic characters in a dataset of 46 animation movies with varying composition and character design. Our results indicate high precision and recall of the automatically detected characters compared to human-annotated ground truth, demonstrating the generalizability of our approach.**

***Index Terms*—Animation movies, deep neural networks, object tracking, saliency, unsupervised clustering, video diarization.**

I. Introduction

AUTOMATIC analysis of movie content is of growing interest in the multimedia research community. One of the driving factors for this research is the large number of movies that are produced, disseminated and consumed annually. Besides being of entertainment value, movies often have an effect on certain social and economic aspects, as well as have a global reach and audience.

Researchers have addressed movie content analysis with different objectives and outlooks. Such efforts are often based on efficient indexing and organization of the media content for easy user navigation. They include shot boundary detection for movie segmentation [1], [2], video summarization [3] and abstraction [4]. The study in [2] builds a generative model that incorporates contextual information in order to reorganize interleaved shots into multiple plot threads. Approaches such as in [5] combines the aspects of video summarization, i.e., *who*, *what*, *where* and *when* for a semantic understanding of the movie content and structure. *RoleNet* proposed in [6] examines the movie content from a social network analysis perspective of the movie character roles rather than using audiovisual features. In general, movie content is a rich source of data that includes audio, video and text (dialogs) that enables such multimodal analysis.

Complementary to the aforementioned studies which attempt to achieve a high-level understanding of movies, efforts for a fine-grained (frame level or scene level statistics) analysis of video content have also been emerging. One such application is to quantify the amount of time a character appears on screen in a movie. The study in [7] examined these aspects with respect to gender revealing skewed distributions for the onscreen time of female characters. In order to advance from gender-level statistics to character-level statistics, person identification or character labeling is a crucial step in this direction. We refer to this problem as automatic video diarization – partitioning the video stream into actor-homogeneous segments, i.e., *who appeared*, *when* and for *how long*. Character labeling in live-action TV and movies has been achieved with modest success in [8]–[12]. This is typically performed by clustering the detected faces (e.g., [8]) or by multimodal approaches (e.g., [9], [10]) that model audio and subtitles or scripts alongside the detected faces from video.

It is important to note that all these studies exclusively focus on live-action TV and do not generalize to animated media content. Digital animation movies have contributed to over 10% of the box office market shares in the past decade [13]. Multimedia research in this domain is extremely scarce and technology developed for live-action TV content fails for animated content. Human face detection is the crux of character labeling methods for live action TV. Since human-characters can be uniquely identified by their faces, this method performs adequately well. But, such methods developed for human faces do not work

Manuscript received November 28, 2016; revised July 26, 2017; accepted August 4, 2017. This work is based upon work supported by the National Science Foundation. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xilin Chen. *(Corresponding author: Krishna Somandepalli.)*

K. Somandepalli, N. Kumar, and T. Guha is with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90047 USA (e-mail: krishna.somandepalli@gmail.com; knaveen87@gmail.com; tanayaguha@gmail.com).

S. S. Narayanan is with the Signal & Image Processing Institute, University of Southern California, Los Angeles, CA 90089 USA (e-mail: shri@sipi.usc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2745712

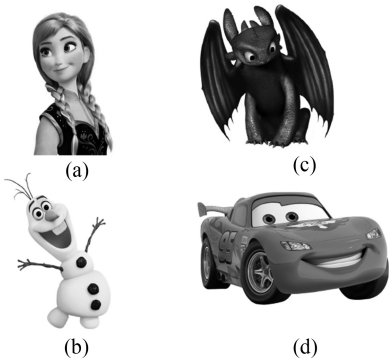

Fig. 1. Examples illustrating the heterogeneity of animated characters. (a): human-like (Frozen) (b): anthropomorphic (Frozen) (c) and (d): abstract (How to Train your Dragon, and Cars).

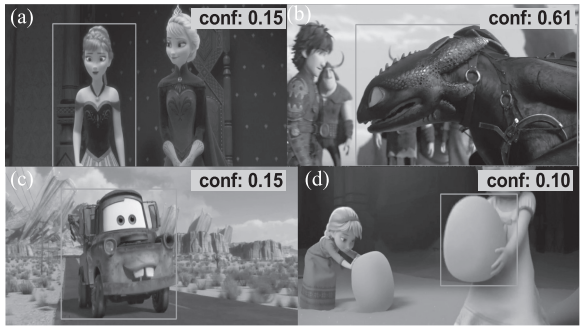

Fig. 2. Character candidates chosen by the Multibox object detector. Conf. indicates the confidence score of the network for the detected object.

for the digital animation genre. Animated characters, though mostly anthropomorphic (having human characteristics) are not always human-like in appearance. They can be fictional animals, inanimate objects or abstract in design (see Fig. 1 for a few examples).

A major obstacle for automating content analysis of animated media is the lack of a model that generalizes across different characters with varying composition and design. This task becomes extremely complex given that all the characters even within a single movie may not share the same structural characteristics (e.g., human-like and non-human characters from the same movie—Fig. 1(a) and (b) from the movie Frozen).

In the context of video diarization, when the characters that appear on screen are generally not known a priori, a key step is to provide a list of characters that form the *who appeared* component of the system. We refer to such a list of characters specific to each movie as a *character dictionary*. The automatic discovery of these character dictionaries is the primary objective in this paper. Our overarching goal is to engineer a model for animation movie video diarization. With the proposed character dictionaries, animation character labeling may be achieved by techniques such as [14] that can retrieve frames and shots given an object of interest.

In content analysis of animated media, researchers have thus far focused on problems such as cut detection [15], color-based video categorization [16] and movie abstraction [17], [18]. One method proposed in [19] performs human-like face detection from cartoon images using skin-segmentation techniques. Considering the variation in texture, color and shape of animated characters in general (as illustrated in Fig. 1), these methods do not generalize well. To the best of our knowledge, no work to date has specifically addressed the problem of automatic discovery of characters from animated media in a scalable manner.

In contrast to live-action movies, animation movies are completely artist generated. Sketches of the characters are designed by the artists or the animators, generally referred to as *model sheets* from which character-specific 3D models are generated. Sketch based image retrieval systems such as [20] can be used to achieve video diarization when model sheets are available. However, model sheets are copyrighted material and mostly owned by the animation studio which produced the movie. As such, they are not publicly available and approaches which are based on model sheets will not be scalable for all movies.

In 1981, Frank Thomas and Ollie Johnston published *The Illusion of Life* [21]; it outlines a set of twelve basic principles of animation. Animators have been using this as a cookbook for designing characters in order for the viewers to appreciate "animation" over mere "movement". While most of these principles aid animators in adding semantic or artistic value (e.g., anticipation, exaggeration), a few can be exploited in a computer vision context (e.g., *Solid Drawing*: drawing volume solidity and illusion of three dimensions; *Staging*: Distinctive color, depth of field and positioning in the frame to highlight the character). Defining an *animated character* in a complete sense would involve delineating abstract concepts such as life (or sentience even) from movie content. In this paper, we only analyze the video stream from animation movies and leverage some of the aforementioned principles of animation as proxies to identify the characters.

At the outset, we pose our problem as an object detection task where any object can be a possible *character candidate*. Animation movie frames are comparable with natural photographic images, especially in their similarities of depth of field and the character presentation in a frame. Additionally, we assume no prior models with respect to shape, size, color, or texture for these candidates in order for the proposed system to generalize.

A few prominent examples of state-of-the-art object detection systems include discriminatively trained deformable parts-based model (DPM, [22], [23]) and deep neural network (DNN) models such as [24]–[26], both of which are supervised and trained over a predefined set of object classes. DPMs need a carefully designed part-decomposition model of an object which makes it unsuitable given the heterogeneity of characters within just a single movie. In contrast, DNN-based methods such as [24] can detect objects in real-time and outperform DPMs. Specifically, DNN models that are saliency-inspired in design [25] are of interest for our problem statement. Although supervised with a finite set of object classes, they have been shown to detect objects in a *class-agnostic* manner [26] i.e., detect classes of objects not used for training the model.

Movies in general, portray only a handful of *prominent* characters. They are more likely to appear frequently in order for the viewer to easily comprehend the content and the plot of

the movie. Additionally in movies, characters or the objects-of-interest tend to remain on screen for up to a few seconds depending on the situation. Visual object tracking can be used as an effective method to segment characters locally in time. Several previous works have used tracking as a means to automatically detect a class of objects (e.g., pedestrians, [27]). Object tracking algorithms can be error-prone in a movie video environment because of object deformation, background clutter, changes in illumination, occlusion and lack of a stationary backgrounds. However, visual tracking can minimize the number of detected objects to be considered by accounting for minor deformation or linear motion of the object. Furthermore, tracking also provides time information that can be used for diarization subsequently. For example, in [11], supervisory information available on a profile face is used to learn the appearance of a frontal face from faces tracked in TV series. A reasonable assumption in describing animated character is that the prominent characters are not transient when presented on-screen and appear frequently in the movie. In our method, we use this aspect of character presentation in movies to select character candidates. As a result, the character dictionaries consist of only the frequently occurring characters.

In this paper, we propose a novel approach to automatically discover characters that appear in an animation movie. Our proposed method is unsupervised in the sense that we do not train any aspect of our system with data from animated media content. Furthermore, we use no specific knowledge of the animation style or the physical attributes of the animated characters, thereby ensuring that our system can scale and generalize through the whole spectrum of animation movie content.

The rest of the paper is organized as follows: Section II describes the proposed system for selecting character candidates from an animation movie. In Section III, we present the experiments performed and the creation of an evaluation database. Section IV contains the experimental results and final considerations followed by conclusions and future work in Section V.

II. Methods

In this section, we first introduce the different systems that we use to identify and prune the detected objects to obtain a set of possible character candidates. We then use a clustering approach to identify character exemplars that constitute the final character dictionary. The overview of the proposed system is shown in the Fig. 3.

Our animation movie database consisted of forty-six movies, for which we annotated their prominent characters. We then conducted a detailed performance evaluation on eight animation movies which were chosen to represent varying degrees of heterogeneity in character design and composition. The movie-cast data from forty-six movies used for our system evaluation and the output from our system has been released as part of the SAIL Animation Movie character Database (SAIL-AMDb).[1] We have also made the code publicly available.[2]

[1]https://github.com/usc-sail/mica-animation/wiki
[2]https://github.com/usc-sail/mica-animation

A. Coarse Detection of Character Candidates

Animated characters are often designed to have the appearance of a 3D object and characterized by shallow focus where the image plane of the character is in focus while the rest of the frame is out of focus [21]. In other words, they are the salient objects in a given frame. Capitalizing on this, we define a *character candidate* as any object that can be detected by a general-purpose object detector.

We use a pre-trained deep neural network (DNN) called *MultiBox* [25], [26], designed for object detection. Our preliminary experiments with other region proposal networks such as [24] yielded similar results. We chose *MultiBox* since our motivation for using an object detector was only to generate an initial set of potential character candidates.

MultiBox is a convolutional neural network (CNN) with an inception-style architecture [28] trained with the full 200-category object detection challenge data set from ImageNet Large Scale Visual Recognition Challenge 2014 (ILSVRC-2014) [29]. This model generates multiple bounding boxes and an associated confidence score that quantifies the network's confidence of each box containing an object. The model has been shown to perform object localization in a *class-agnostic* manner and achieve state-of-the-art performance in object detection tasks [25]. Furthermore, since the network is tailored towards the localization problem, it achieves a scalable representation of multiple salient objects in an image. These features make this model uniquely suitable for our problem. It is important to note that this model is trained with natural images of distinct object classes. Although the authors in [25] have shown that the model generalizes over unseen classes, here we apply the pre-trained DNN for images sampled from animation movies. We refer to this discrepancy as *DNN training bias*. This results in detecting objects that are not characters in a movie (e.g., traffic-light, chair). We refer to such objects as *noisy objects*.

In order to reduce the computational time, we downsample a movie (originally encoded at 23.98 fps) by one frame every 0.42 s (every 10th frame). The resulting frames are input to MultiBox [25] to obtain all possible bounding boxes for each image. The confidence score that is returned with each of these boxes was originally optimized in the DNN to match the ground truth object boxes from natural images.

Because of the aforementioned *DNN training bias*, we generally observed lower range of confidence scores for objects detected that were animated characters. We chose to retain objects with a confidence score greater than 0.1. In order to determine this threshold, we randomly sampled 100,000 frames from the movie Frozen (2013) in our movie database. We first assumed to have at most five possibly overlapping objects of interest in one frame and obtained the confidence scores for the five most confident objects in each frame. We then examined the distribution of the confidence scores for all the objects detected. We set the confidence threshold to 75th percentile of the distribution of confidence scores which is equal to 0.1002, thus retaining all objects with confidence score greater than 0.1. We apply this confidence threshold for all the movies in our database. A few examples of objects detected and their confidence scores returned by the network are shown in Fig. 2.

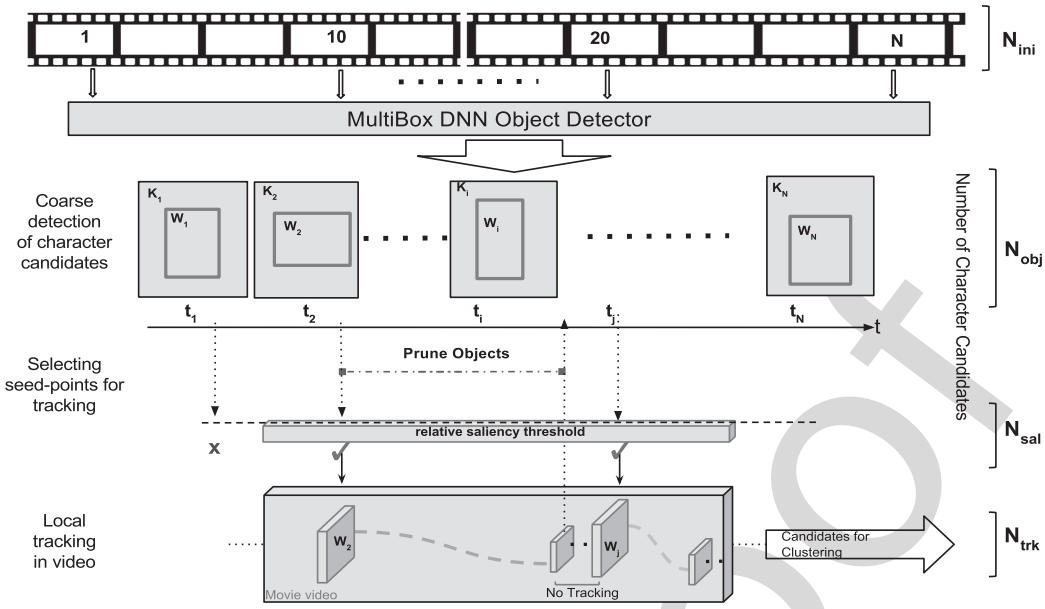

Fig. 3. Schematic diagram of the proposed method.

We also computed the area of each bounding box of an object relative to the image frame and excluded objects in bounding boxes with an area less than 1% or greater than 99% of the entire frame. This ensures that very small objects and holistic scenes are excluded as character candidates. When multiple objects were detected in a single frame, we pruned them to obtain at most one object per frame following the approach in [25]. We performed non-maximum-suppression with a Jaccard similarity [30] threshold of 0.5 and, chose the object with the maximum area in that frame. We identified only a single object per frame in order to simplify the subsequent step of single-target visual object tracking. We refer to a chosen frame containing a character candidate as a *candidate frame*.

A schematic of the proposed approach is illustrated in Fig. 3. Let N_{ini} be the initial number of images (movie frames) input to *MultiBox* and N_{obj} be the number of character candidates chosen. We denote the candidate frame K and the bounding box W enclosing the object as a set $\mathbf{M_{obj}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{obj}]\}$ and t_i refers to the time (or frame number) in the movie at which the object i occurs. Qualitative analyses showed that this step captures most of the characters in an animation movie at least once (e.g., images shown in Fig. 2(a)–(c)). However, this set also contains redundant and noisy objects which include non-characters or background objects (e.g., Fig. 2(d)).

B. Saliency Constraints and Object Tracking

In the next phase of our system, we used the saliency of the detected object as a constraint to prune the set of character candidates obtained in the previous step. We use this pruned set of candidate frames as seed-points for tracking. During tracking, we do not distinguish camera motion from object motion, thereby ensuring that a sufficient condition for a character candidate is its presence on the screen rather than motion (e.g., a talking tree).

Fig. 4. (a) Example for DNN training bias and saliency constraint; (b) Masked regions showing saliency, here *relative saliency score* $R_s(W_1) = 9.2\%$.

1) Saliency-Constrained Pruning: As described in Section II-A, the DNN training bias may result in choosing objects that, although salient, may not be the characters of interest (e.g., detected lamp in a scene with two characters—see Fig. 4(a)). To quantify this, we use a saliency measure proposed in [31] for the character candidate with respect to the entire frame. Unsupervised methods that estimate saliency typically use pixel-level features such as color, intensity (e.g., [32]) or background-detection in dynamic scenes (e.g., [33]). In contrast, the measure proposed in [31] estimates saliency of local areas (instead of pixel level) in static images and requires no training. This method uses a kernel-based approach where the size of the window relates to the scale of the target objects. The saliency of a pixel inside the window is estimated using the conditional probability of that pixel drawn from the distribution estimated inside that window versus the distribution of the surrounding area.

We first converted the RGB images to CIELAB color space (because of the perceptual uniformity of the CIE color space[3]) to estimate a saliency map for the entire candidate frame by

[3]http://www.brucelindbloom.com

choosing window sizes at different scales as described in [34]. The resulting saliency maps are binarized by setting values greater than 0.7 to 1 as recommended in [34]. An example of the saliency map is shown in Fig. 4(b). Let $A_s(W)$ be the area of the salient region contained within a bounding box, W in an image frame K. We define a *relative saliency score*, $R_s(W)$ of an object enclosed by the box W as the percentage salient area it contributes to the frame, K:

$$R_s(W) = \frac{A_s(W)}{A_s(K)} \times 100. \quad (1)$$

We obtained the *relative saliency score*, $R_s(W_{t_i}^{(i)})$ for every character candidate in the set $\mathbf{M_{obj}}$ from the *MultiBox* object detector. We used a threshold of 10% and retain only those character candidates which have a relative saliency score greater than this threshold. These candidates are next used as seed-points for tracking. This threshold was initially decided based on qualitative observation. We then conducted additional experiments to assess the effect of this threshold parameter as described in the Section III-C. The resulting set of *salient* character candidates is denoted as $\mathbf{M_{sal}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{sal}]\}$, where N_{sal} is the total number of objects deemed salient after this step with $|\mathbf{M_{sal}}| \leq |\mathbf{M_{obj}}|$ where $|\cdot|$ indicates the cardinality of the set.

2) Deformable Object Tracking: An important property of animated characters is their appearance on screen for up to a few seconds depending on the context. We utilized this property by performing a single-target visual tracking of the salient character candidates. Since animated characters are mostly deformable bodies, the rigidity assumption that most tracking algorithms employ in their motion models (for review, see [35]) does not hold. We employ a deformable object tracking algorithm [36] which does not impose rigidity assumptions on the object-of-interest while tracking.

This method first builds a static-appearance model of the object by clustering the key-points into sets of *inliers* (for the object body) and *outliers* (for the background) using a dissimilarity measure that quantifies the correspondences between key-points. The dissimilarity measure is estimated by computing the distance between the initial set of corresponding key-points and the transformed version. The model is then adaptively updated in time by propagating only the *inlier* correspondences by estimating the optical flow of the key-points. The degree of tolerance towards the deformation of the object is factored into the model by setting a parameter in the tracking algorithm which ensures that the cluster of inlier points are spatially localized. We used the BRISK [37] features for key-point detection and the parameters were set according to [36] after histogram equalization of the images.

Tracking every object from the set of salient character candidates for the full length of the movie is computationally expensive and may lead to accumulated tracking errors. Hence, we performed *local-tracking* in a serial and progressive fashion as described in **Algorithm 1**. We refer to the first *candidate frame* and the corresponding bounding box for the object of each track as a *seed-point*. *Local-tracking* substantially reduced the number of character candidates by eliminating objects that were successfully tracked in consecutive frames. As we performed single-target visual tracking, this process may also exclude other characters that co-occur within a given track. However, since *prominent* characters occur quite frequently in a movie, the issue of losing certain characters was not significantly noted. The duration of time for which an object is tracked is used as a threshold for retaining objects. We refer to this as the *track duration threshold*, τ and initially set to one frame. This would only eliminate the transient and/or spurious object detections. Additional experiments varying the τ parameter are conducted as discussed later. We denote the set of character candidates returned after tracking as $\mathbf{M_{trk}}$ with $|\mathbf{M_{trk}}| = N_{trk}$ such that $N_{trk} \leq N_{sal} \leq N_{obj}$ The number of character candidates obtained after pruning at each step as a percentage of the initial number of input frames is shown in Table II.

Algorithm 1: Local Tracking

Input: Set of salient character candidates: $\mathbf{M_{sal}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{sal}]\}$; movie, $\mathbf{V}$
Output: Set of track *seed-points*; $\mathbf{M_{trk}} = \{(K_{t_i}^{(i)}, W_{t_i}^{(i)}) | i \in [1, N_{sal}]\}$ and corresponding track duration T_i
Parameters: Track duration threshold: τ

```
while ( V open) do
    M_trk = {}
    while ( M_sal ≠ ∅) do
        Begin tracking at the earliest time frame, i.e.,
        {(K_{t_j}^{(j)}, W_{t_j}^{(j)})} ← min_{∀t_j : j ≤ |M_sal|} {M_sal}
        Object tracking lost at t_k ≥ t_j
        Track duration, T_j ← t_k − t_j
        if ( T_j > τ) then
            Update tracked seed-points
            M_trk ← M_trk ∪ {(K_{t_j}^{(j)}, W_{t_j}^{(j)})}
            Prune character candidates
            M_sal ← M_sal ∩ {(K_{t_m}^{(m)}, W_{t_m}^{(m)}) | ∀m > k}
        else
            M_sal ← M_sal ∩ {(K_{t_m}^{(m)}, W_{t_m}^{(m)}) | ∀m > j}
        end
    end
    N_trk = |M_trk|
end
```

C. Exemplars for Character Representation

The character candidates chosen thus far may be redundant to some extent, and may contain multiple images with varying view-point or segments of the same object. In order to group similar objects together, we pose this as an unsupervised clustering problem with an unknown number of clusters. A suitable approach to represent such data is to identify a smaller set of samples, referred to as *exemplars*. We use affinity propagation (AP) clustering [38] to obtain exemplars which constitute the final *character dictionary* for a given movie. AP clustering is well suited for this problem because it is deterministic, achieves

TABLE I
DETAILS OF THE EVALUATION DATASET

ID	Movie (US Release year)	Duration(mins)	Prominent Characters†	Production Studio	Grossing (in $ millions)
V1	Cars 2 (2011)	107	10 (3)	Pixar	191
V2	Free Birds (2011)	91	11 (4)	Reel FX Creative	55
V3	Frozen (2013)	102	9 (4)	Walt Disney	400
V4	How to Train your Dragon 2 (2014)	102	12 (4)	DreamWorks	177
V5	Shrek Forever After (2010)	93	9 (5)	DreamWorks	238
V6	Tangled (2010)	100	9 (4)	Walt Disney	200
V7	The Lego Movie (2014)	101	12 (3)	Warner Animation	257
V8	Toy Story 3 (2010)	103	18 (9)	Pixar	415

† ( ) indicates number of minor characters.

TABLE II
PERCENTAGE OF INITIAL NUMBER OF OBJECTS AFTER EACH STEP OF PRUNING ON THE EVALUATION DATASET

Movie ID	N_{ini}	$N_{obj}(\%)$	N_{sal}**(%)***	N_{trk}**(%)**$^{+}$
V1	15395	19.88	16.99	5.61
V2	13102	14.08	12.50	5.01
V3	14676	9.36	6.83	2.56
V4	14676	9.25	6.32	3.17
V5	13406	10.61	8.06	3.32
V6	14372	9.42	8.22	3.05
V7	14460	9.37	6.96	2.92
V8	14748	11.80	9.79	3.79

*relative saliency threshold $= 10\%$. $+$ track duration threshold $= 1$ frame.

a lower clustering error compared to other clustering methods such as k-means [39] and does not require a predetermined number of clusters.

We used the *ImageNet* model proposed in [40] to extract features to cluster the character candidates. Several previous works (e.g., [41]) have shown that feature representations from fully-connected layers in a CNN generalize well for various image recognition tasks. Specifically, we use a 4096-dimensional feature from the second fully connected layer, "FC7" from the ImageNet model which was trained with ILSVRC-2012 [29] competition data.

Because the FC7 features are sparse, we use cosine distance to compute a pairwise similarity matrix, $\mathbf{S_{ij}}$ between the feature vectors, $\{\mathbf{v_i}\}$

$$\mathbf{S_{ij}} = \frac{\mathbf{v_i}\mathbf{v_j}^T}{\|\mathbf{v_i}\|\|\mathbf{v_j}\|} \forall i, j \in [1, N_{trk}]. \quad (2)$$

The appearance of most characters is somewhat homogeneous (except for variations in pose and deformation) throughout a movie in terms of shape, color or attire of the character. Leveraging this observation, we also used GIST descriptors [42] for clustering. GIST features provide a low dimensional representation that describes the prominent spatial structure in an image. GIST features have been used for clustering tasks such as scene clustering (e.g., [43]) with some success. We obtained a 960-dimensional GIST descriptor for the character candidates using *pyleargist*[4] package in Python. We then computed negative Euclidean distance between all the candidates from a movie to form a similarity matrix for clustering. Additionally, we also evaluated the clustering performance of GIST and FC7 features.

We used the AP algorithm proposed in [44] to cluster the similarity matrices obtained from the character candidates. The goal of AP clustering is to choose a character candidate j to be the exemplar of the $i$thcandidate. Define *responsibility* $r(i, j)$: degree of support that the candidate j should be the exemplar of i and *availability* $a(i, j)$: degree of support by which the candidate i should choose j to be its exemplar. Initialize $r(i, j), a(i, j) = 0; \forall i, j$ and update responsibility and availability as below:

$$r(i,j) \leftarrow \mathbf{S_{ij}} - \max_{k:k\neq j}(a(k,i) + \mathbf{S_{ik}}) \quad (3)$$

$$a(j,j) \leftarrow \sum_{k:k\neq j} \max[0, r(k,j)] \quad (4)$$

$$a(j,i) \leftarrow \min\big(0, r(j,j) + \sum_{k:k\notin(j,i)} \max[0, r(k,j)]\big). \quad (5)$$

Introduce a damping factor, $\lambda \in [0, 1)$ to account for numerical oscillations over iterations in time t

$$r(j,i)_t \leftarrow (1-\lambda) r(j,i)_t + \lambda r(j,i)_{t-1} \quad (6)$$

$$a(j,i)_t \leftarrow (1-\lambda) a(j,i)_t + \lambda a(j,i)_{t-1}. \quad (7)$$

Pick j to be an exemplar of i if

$$\arg\max_j (r(i,j) + a(j,i)). \quad (8)$$

We set the damping factor, λ which controls the update of $r(i, j)$ and $a(i, j)$ in each step to 0.5. Changing this parameter had no effect on the exemplars we obtain. Let N_{xmp} be the total number of exemplars returned.

AP clustering works well with animation movies since the appearance of most characters (e.g., attire) is consistent within a given movie and the features we used for clustering can capture these attributes. An additional benefit of using AP clustering is that the number of exemplars (i.e., the size of character dictionary) need not be pre-specified. On the other hand, we risk *over-clustering*, i.e., a single character may be represented by multiple exemplars since the features we use are generic and not designed to capture variation in scale, orientation or view-point of a character. This was evident when we performed a second pass of AP clustering on the exemplars obtained here and failed to cluster the *perceptually identical* characters together. In

[4]https://pypi.python.org/pypi/pyleargist

order to penalize for over-clustering, we define an *over-clustering index* in our performance evaluation measures as described in Section III-C.

III. Experiments

The problem of identifying character dictionaries for animation movies addressed in this paper is unique. Due to the lack of existing performance evaluation frameworks for this task, we first created a *reference character dictionary* (movie-cast) for each movie in our database. We then used these reference character dictionaries as ground truth to evaluate the character dictionaries output by the proposed method. These reference character dictionaries have been made publicly available as a part of the SAIL-AMDb[5] along with outputs used for our system evaluation.

A. Evaluation Database

Our animation movie database consisted of a total of forty-six movies produced between 2010–2014. Of the forty-six movies available, we chose eight top-grossing movies to evaluate the performance of our method in greater detail and to determine the best parameter choices for *relative saliency threshold* and the *track duration threshold*. The year of release, duration, production company and size of the reference character dictionary are shown in Table I. For brevity, we refer to these movies as V1–V8.

These eight movies were chosen to test the generalizability of the proposed system. They represent a diverse set of characters in terms of design and composition produced by prominent animation studios. These movies include instances of human or human-like characters (V3, V5, V6), non-human but anthropomorphic (V3, V5), toy-like (V7, V8) and animals (V2, V4, V5). All movies (except V6) include at least one instance of a character which is abstract in design. The dataset includes movies with varying degrees of illumination, background/environment and motion of the characters. For example, V1, V6 and V8 have overall higher illumination compared to V3, V4 and V5. The movies V1 and V4 have faster moving characters (e.g., dragons and cars) compared to the others. Quantitative analyses to evaluate the diversity of this dataset (e.g., variation in color, illumination or other characteristics) are beyond the scope of this paper (and an objective of our future work).

As described in Section II-C, the character dictionary output by the proposed system for each movie are the exemplars identified by AP clustering. The character candidates on which the clustering is performed are obtained by optimizing two system parameters using a grid search: relative saliency threshold and track duration threshold. The settings used for the two parameters are $R_s(X) = \{0, 10, 20, 50, 80, 90\}$ and $\tau = \{1, 12, 24, 48, 120\}$. The values for τ (in frames) correspond to the least possible value (one frame), and approximately 0.5 s, 1 s, 2 s and 5 s of the movie duration respectively.[6]

B. Reference Character Dictionaries

We borrow the same definitions for a character as described in [45] and [46] to create a movie-specific *reference character dictionary*. All named characters (speaking and non-speaking) displayed on-screen were included. Similar to [46], we first used the set of prominent characters as listed by a leading online box-office reporting service.[7] The designation of a *minor character* available in this resource was retained. This list however, does not include non-speaking characters (e.g., dragons). Hence, if a character was given a specific name in the movie (as opposed to generic names such as *a Spanish ambassador*), we included them in the reference. For each of these characters, we obtained a representative full-body image from the movie posters or DVD covers available online. If the said character was absent in these sources, a representative image was manually obtained from the internet. The number of prominent characters including the number of minor characters are listed in Table I. For annotation purposes, all characters in the reference dictionaries are assigned a unique ID to preserve character anonymity.

We use annotations from Mechanical Turk workers (MTurk; a crowdsourcing platform by Amazon Web Services) to compare the *proposed* and *reference* character dictionaries. As discussed in Section II-C, the exemplars in the proposed dictionaries may vary from the representative image used to construct the reference. Hence, by using MTurk, we leverage the human perceptual ability to match the exemplars with the items in the reference. The annotators are instructed to consider an exemplar to be a match if 1) it is identifiable regardless to variation in scale, illumination, orientation or viewpoint or 2) an identifiable segment of the reference character is present in the exemplar or 3) if the exemplar consists of the said reference character. The annotators indicate a match with a unique ID available for every character in the *reference*. Furthermore, if an exemplar consists of multiple reference characters, the annotators are instructed to list all the relevant IDs. Three different annotations were acquired for each of the exemplars from unique annotators. In order to check for possible confounding factors, additional information on whether the annotator had watched the movie prior to annotating was also collected.

We performed an inter-rater reliability analysis to ensure that the MTurk annotations were reliable. Since we obtained more than two annotations, inter-rater agreement (more specifically, inter-annotation agreement) was quantified using Krippendorf's alpha [47] for each movie. The categorical values that were used to compute this measure were the unique IDs assigned to each character from the reference. Krippendorf's alpha was high for the eight movies used in our system evaluation with mean/standard deviation of $\alpha = 0.81 \pm 0.05$ indicating strong agreement. Across all forty-six movies, Krippendorf's Alpha was similarly high (0.82). Furthermore, no difference in agreement was observed between the set of annotations performed by workers who had watched the movie and those who had not. Following high agreement, we obtained a single annotation per exemplar by performing simple majority voting on the

[5]https://goo.gl/WbESbz

[6]Frame rate for all movies in the dataset was 23.98 fps

[7]www.boxofficemojo.com

three annotations. Three-way ties were resolved with random assignment.

C. Performance Evaluation

The performance of our method for different experiments was quantified by comparing the reference character dictionaries with the output dictionaries from the proposed method. We refer to the set of exemplars in the proposed dictionary that were successfully matched to a character in the reference as the *relevant exemplars* and the remaining as, the *noisy exemplars*. As described earlier, multiple exemplars can represent a single character. Therefore, we examine the unique set of character IDs in the proposed dictionary (*matched characters*) and those never identified (*missed characters*). Following this, we compute three measures; precision, P, recall, R and F1 score, F_1 as follows:

$$P = \frac{|\{relevant\ exemplars\}|}{|\{relevant\ exemplars\} \cup \{noisy\ exemplars\}|} \quad (9)$$

$$R = \frac{|\{matched\ characters\}|}{|\{matched\ characters\} \cup \{missed\ characters\}|} \quad (10)$$

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}. \quad (11)$$

Additionally, we define *over-clustering index* as a measure to quantify the extent to which multiple exemplars per character appear in our character dictionaries. In other words, the extent to which we *over-cluster* the relevant characters. Over-clustering index for a movie is computed as the median of number of exemplars per character in the set of the relevant exemplars. Since this metric is defined only over the set of relevant exemplars, it is independent of precision. It is bounded below by 1 (one exemplar per character) and bounded above by N_{xmp} (all exemplars represent just one character).

In order to compare the clustering performance of GIST and FC7 features, we measure the *purity* of clustering as described in [48]. We assign each cluster to the most frequently occurring character in that cluster. Then, the we measure purity by counting the total number of correctly assigned characters, across all clusters and dividing by the total number of candidates clustered (N_{trk}) as below:

$$purity = \frac{1}{N_{trk}} \sum_{k} \max_{j} |\omega_k \cap c_j| \quad (12)$$

where *purity* $\in [0, 1]$, $\Omega = \{\omega_1, \omega_2, ..., \omega_k\}$ is the set of all clusters and $\mathcal{C} = \{c_1, c_2, ..., c_j\}$ is the set of all relevant exemplars.

By our definition of precision (9), a lower value would indicate that character candidates which are not listed in the reference were identified as exemplars. These *noisy exemplars* could either be a result of minor characters not being listed in the reference or background objects being identified as exemplars. Similarly, a high recall (10) would reflect the ability to identify all the prominent characters at least once. Ideally, recall $= 1.0$ and over-clustering index $= 1$ would indicate that every character in the reference was detected by exactly one relevant exemplar. Higher values of the over-clustering index reflect on the failure to cluster similar character candidates. This is likely a consequence of the features not being invariant to the orientation, view-point or scale of the character candidates. Complementary to precision, recall and F1 score which measure the performance of clustering with respect to a reference, purity (12) measures the extent to which clusters belonged to a single character, thus evaluating the features (FC7 versus GIST) used for clustering.

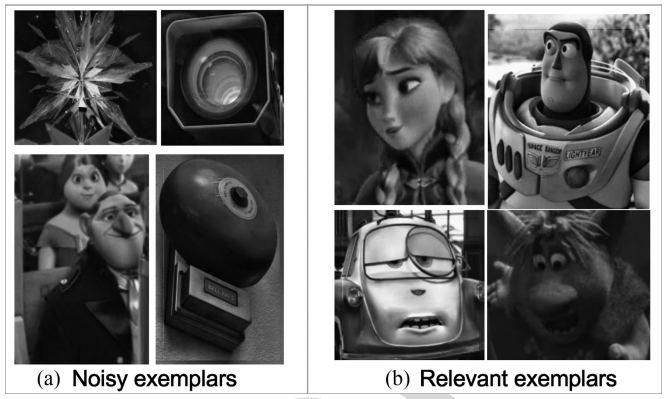

Fig. 5. Examples of noisy and relevant exemplars.

The F1 score, precision and recall measures for all eight movies are averaged for each experiment to determine the best choice of relative saliency threshold and track duration threshold. These optimal parameters were used to obtain character dictionaries for the remaining thirty-eight movies in our evaluation dataset.

IV. Results and Discussion

A few examples of the *relevant* and *noisy* exemplars from the proposed character dictionaries are shown in Fig. 5. As described earlier, exemplars are categorized as relevant or noisy based on a reference dictionary constructed for each movie. One source of noisy exemplars is how we construct these reference dictionaries. Since the reference consists of only the prominent characters, it may result in some minor characters being categorized as noisy (See bottom-left image in Fig. 5(a)).

The second source of noisy exemplars is the training data used for the *MultiBox* object detector which comprised only of natural images. Characters which belong to object classes that the DNN was trained on tend to get detected more often and consistently (e.g., traffic lights, bell). The subsequent steps in our method that use relative saliency score and local-tracking attempt to eliminate some of these noisy exemplars. However, depending on the frequency of occurrence or saliency of the character candidates, they may not always be successfully pruned. Table II shows the percentage of the input frames pruned at each step. The proposed character dictionaries for three movies; V1, V2 and V3 are shown in Fig. 10–12 in Appendix B.

The precision, recall and F1 score measures that we used to quantify the performance of our method are shown in Fig. 6. The relative saliency threshold and track duration threshold were chosen corresponding to the best F1 score (highlighted in Fig. 6(a)). These measures were averaged across the eight movies for each setting of two parameters, relative saliency

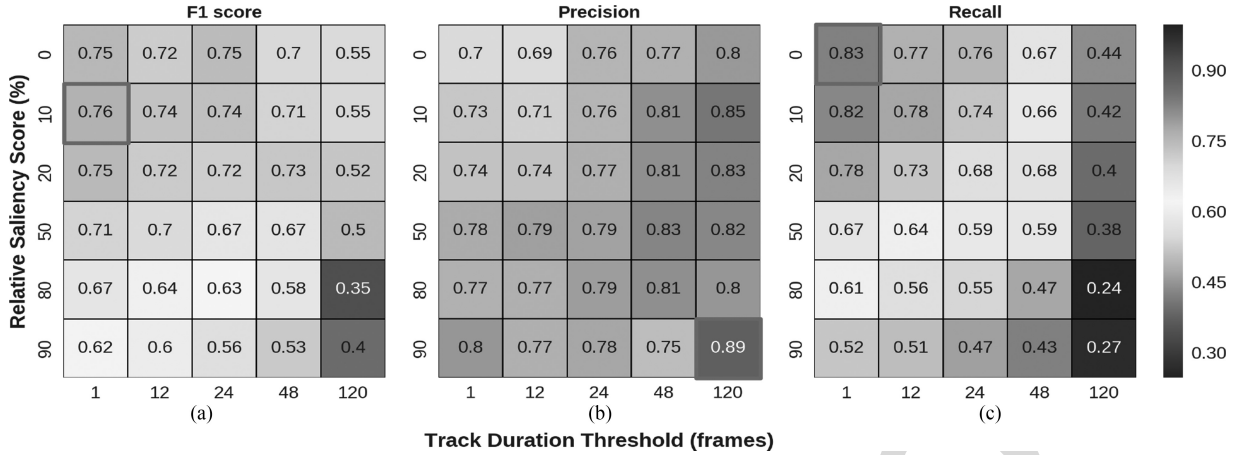

Fig. 6. Average (a) F1 score, (b) Precision and (c) Recall for all experiments.

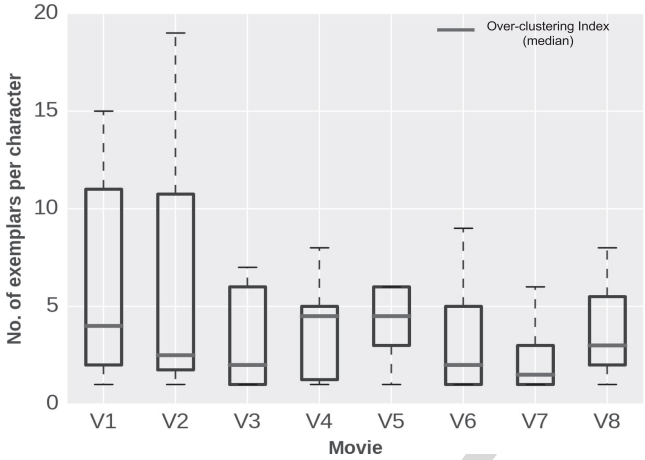

Fig. 7. Distribution of number of exemplars per character in each movie for $R_s(X) = 10\%$ and $\tau = 1$.

TABLE III
F1 SCORE AND PURITY FOR FC7 AND GIST FEATURES USED IN CLUSTERING

Movie ID	FC7 features		GIST descriptors	
	F1 score	Purity	F1 score	Purity
V1	0.691	0.708	0.713	0.414
V2	0.773	0.651	0.769	0.345
V3	0.825	0.842	0.821	0.304
V4	0.764	0.598	0.693	0.322
V5	0.532	0.712	0.653	0.408
V6	0.740	0.677	0.732	0.398
V7	0.732	0.693	0.743	0.438
V8	0.752	0.745	0.799	0.392
Average:	**0.726**	**0.703**	**0.740**	**0.378**

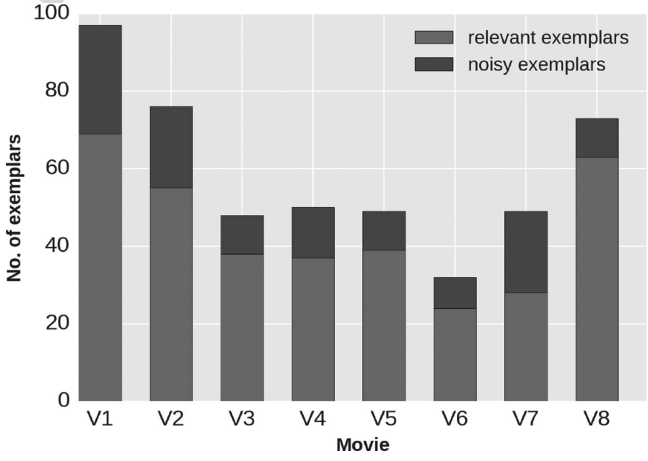

Fig. 8. Number of relevant and noisy exemplars for each movie with $R_s(X) = 10\%$ and $\tau = 1$.

threshold, $R_s(X)$ and track duration threshold, τ. Overall, recall is high (over 80% for $\tau = 1$ and $R_s(X) = 10\%$) which indicates that our proposed character dictionaries were able to identify most of the characters in the reference at least once. Precision ranges between 70% and 90% indicating that less than one-third of exemplars in our proposed dictionaries are noisy.

We note that the recall measure defined here has to be interpreted alongside over-clustering index; a metric that captures the extent to which multiple exemplars represent a single reference character. The distribution of number of relevant exemplars per character for the eight movies is shown in Fig. 7. The median number of exemplars per character, i.e., the over-clustering index is less than 5 for all the eight movies. As described in Section III-C, this measure lies between 1 and the number of exemplars. Here, the number of exemplars range between 35 and 95 (with $R_s(X) = 10\%; \tau = 1$) but the over-clustering index is less than 5 which reflects on the effective performance of the affinity propagation (AP) algorithm used for clustering.

Additionally, we compared the F1 score and purity of clustering for the eight movies, in order to evaluate the features used in clustering, as shown in Table III. Although the F1 scores (computed by comparing the exemplars to the reference) were similar between the two descriptors, the clustering purity using FC7 features was significantly higher (paired t-test, $p \ll 0.01$ to reject $H_0 : \mu_0 \leq \mu_1$) than that of GIST descriptors. This indicates that FC7 features yield less noisy and more homogeneous clusters from AP clustering. Furthermore, FC7 features perform better for clustering than GIST features, perhaps because *ImageNet* was trained to classify objects robust to variation in the the background or view-point and occlusions, whereas GIST descriptors capture the holistic shape information in an image.

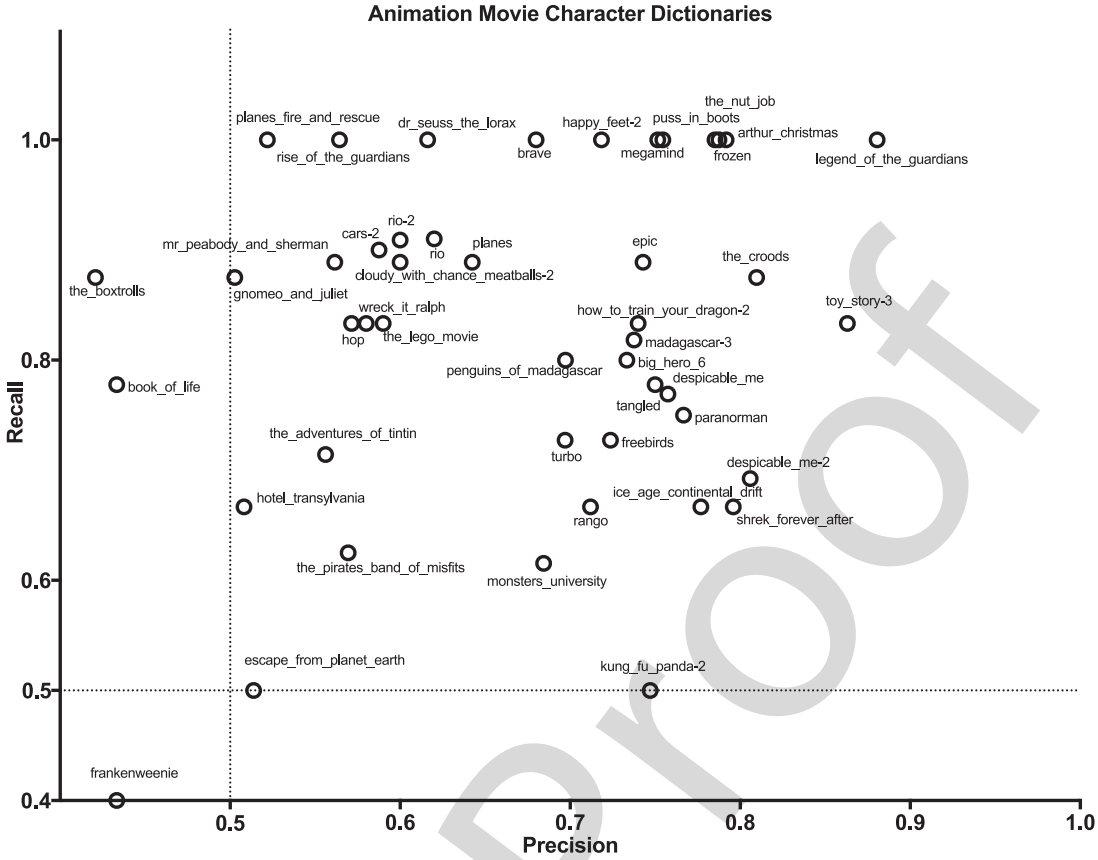

Fig. 9. Precision versus recall for forty-six movies.

As shown in Fig. 6(c), recall drops with an increase in τ as expected. Since, by increasing τ we retain only those character candidates which remain longer on-screen and do not always co-occur with other salient objects. This results in excluding some relevant exemplars. In contrast, an increase in precision (See Fig. 6(b)) is noticed on increasing τ since a few noisy character candidates that are infrequent get pruned successfully. Relative saliency threshold had the desired effect on the system output i.e., increasing $R_s(X)$ results in an increase in precision. However, these gains in precision by increasing $R_s(X)$ beyond 10% were not substantial.

In order to determine a good choice of the system parameters, we examine F1 score for different combinations of $R_s(X)$ and τ as shown in Fig. 6(a). $R_s(X) = 10\%$ and $\tau = 1$ would be the best choice of settings. For these settings, the number of relevant and noisy exemplars for the eight movies are shown in Fig. 8. It is interesting to note that movies V1 and V2 have relatively larger character dictionaries and a higher range of number of exemplars per character (See Appendix Fig. 10–11). All the characters in the movies, V1 and V2 are similar to cars and birds in appearance. The results at a glance show that all instances of these characters in different scenes were detected in these movies (which include the minor characters and different appearances of the same character with respect to view-point). This is likely because both cars and birds are among the object classes in ILSVRC-2014 data used to train *MultiBox*.

Character dictionaries for the remaining thirty-eight movies were obtained with the choice of $R_s(X)$ and τ determined above. The range of precision was 0.45–0.89 (mean/standard deviation: 0.66 ± 0.12) and recall: 0.42–1.0 (0.83 ± 0.16). The range of over-clustering index was 1.5–6.0 (3.5 ± 1.5). See Fig. 9 (Appendix A) for precision and recall measures of all the forty six movies in our dataset. Further error analysis considering different aspects of all the movies (e.g., character design, color, illumination) is warranted and will be a part of our future work.

We note that the movie Frankenweenie (2013) which was produced in black and white has the lowest precision and recall in our dataset. This indicates that color rendering is an important factor since the DNNs we employ were trained with RGB images. The movies, Boxtrolls (2014) and The Book of Life (2014) both have a low precision and high recall indicating a larger number of noisy exemplars. On analyzing the errors in these samples, we observed that the local tracking method pruned approximately 42% of the initial character candidates (c.f. the average percentage of candidates pruned by local tracking for the rest of the movies was 62.23%). This is likely because these movies, unlike the others in the dataset use a *rapid-fire* film editing style which includes fast-action scene cuts and rapidly changing backgrounds which are not ideally suited for visual object tracking.

On the other hand, movies like Kung Fu Panda 2 (2011) and Escape from Planet Earth (2013) yield high precision and low

Fig. 10. Character dictionary of the movie Frozen: precision = 0.81, recall = 1.0; over-clustering index=2.

recall. This is likely because these movies feature only a small number of prominent characters and a larger number of unnamed characters which are not included in the reference dictionaries that we created. As expected, movies that feature distinct lifelike animals or humans, generally performed the best. For example, the movie Legend of Guardians (2010) featured only birds and The Nut Job (2014) featured animals – both animals and birds are included in the set of object categories of the ILSVRC datasets.

V. Conclusions and Future Work

In this paper, we proposed an unsupervised method to automatically create a dictionary of characters from an animation movie. We evaluated our method on a set of eight movies with diverse character styles and demonstrated high precision and recall on a dataset of forty-six movies. We also showed that the proposed method generalizes for animation movies at scale. These character dictionaries can serve as a powerful tool for character labeling to delineate aspects of *who appeared*, *when* and for *how long* in a movie (video diarization). We believe that our efforts can lay a foundation to provide an impetus for multimedia research endeavors specifically involving animated media content.

One of the drawbacks of the proposed method is that we use an object detector that was trained with natural images. We plan to address this issue using transfer learning to adapt the existing models to specialize the network for detecting characters from animation movies. The relevant and noisy exemplars that we annotated for the system evaluation can potentially be used for these methods. Our future work would also include using the relevant exemplars and associated cluster members as a single unit to facilitate robust video diarization of animation movies.

Fig. 11. Character dictionary of the movie Free Birds: precision = 0.72, recall = 0.72; over-clustering index = 2.5.

Appendix A
Precision and Recall of our Proposed Method for the Forty-Six Movies

The Fig. 9 plots precision vs. recall for all the movies in our dataset. The relative saliency threshold and track duration threshold was set to 10% and one frame respectively (tuned on a subset of 8 movies as described in Section IV).

Appendix B
Examples

Fig. 10–12 illustrate the proposed character dictionaries for three movies with the settings of relative saliency threshold =

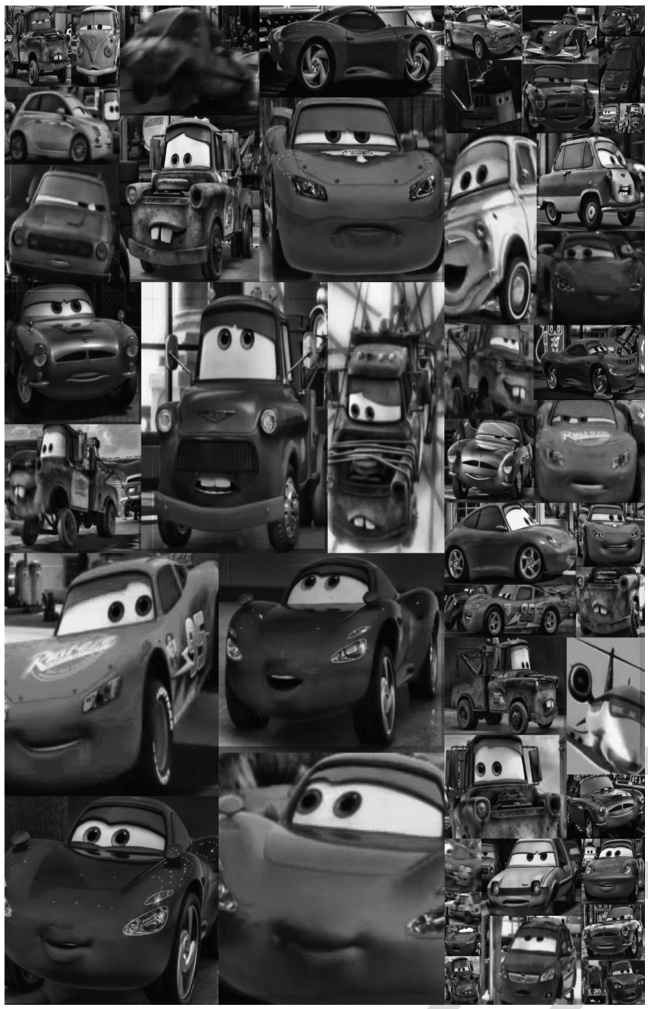

Fig. 12. Character dictionary of the movie Cars-2: precision=0.61, recall = 0.9; over-clustering index = 4.

10% and track duration threshold = 1 frame. The exemplars here are arranged in no particular order to maintain their aspect ratios.

REFERENCES

[1] P. P. Mohanta, S. K. Saha, and B. Chanda, "A model-based shot boundary detection technique using frame transition parameters," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 223–233, Feb. 2012.

[2] C. Liu, D. Wang, J. Zhu, and B. Zhang, "Learning a contextual multithread model for movie/TV scene segmentation," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 884–897, Jun. 2013.

[3] B. W. Chen, J. C. Wang, and J. F. Wang, "A novel video summarization based on mining the story-structure and semantic relations among concept entities," *IEEE Trans. Multimedia*, vol. 11, no. 2, pp. 295–312, Feb. 2009.

[4] Y. Li, S.-H. Lee, C.-H. Yeh, and C. C. J. Kuo, "Techniques for movie content analysis and skimming: tutorial and overview on video abstraction techniques," *IEEE Signal Process. Mag.*, vol. 23, no. 2, pp. 79–89, Mar. 2006.

[5] K. Kurzhals, M. John, F. Heimerl, P. Kuznecov, and D. Weiskopf, "Visual movie analytics," *IEEE Trans. Multimedia*, vol. 18, no. 11, pp. 2149–2160, Nov. 2016.

[6] C. Y. Weng, W. T. Chu, and J. L. Wu, "Rolenet: Movie analysis from the perspective of social networks," *IEEE Trans. Multimedia*, vol. 11, no. 2, pp. 256–271, Feb. 2009.

[7] T. Guha, C.-W. Huang, N. Kumar, Y. Zhu, and S. S. Narayanan, "Gender representation in cinematic content: A multimodal approach," in *Proc. ACM Int. Conf. Multimodal Interaction*, 2015, pp. 31–34.

[8] A. Fitzgibbon and A. Zisserman, "On affine invariant clustering and automatic cast listing in movies," *Proc. 7th Eur. Conf. Comput. Vis.*, 2002, pp. 304–320.

[9] F. Vallet, S. Essid, and J. Carrive, "A multimodal approach to speaker diarization on TV talk-shows," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 509–520, Apr. 2013.

[10] J. Sivic, M. Everingham, and A. Zisserman, "Who are you?"—Learning person specific classifiers from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2009, pp. 1145–1152.

[11] M. Everingham, J. Sivic, and A. Zisserman, "Hello! My name is... Buffy"—automatic naming of characters in TV video," in *Proc. Brit. Mach. Vis. Conf.*, 2006.

[12] M. Everingham, J. Sivic, and A. Zisserman, "Taking the bite out of automated naming of characters in TV video," *Image Vis. Comput.*, vol. 27, no. 5, pp. 545–559, Apr. 2009.

[13] "Box office history for digital animation" [Online]. Available: http://www.the-numbers.com/market/production-method/digital-animation

[14] Z. Aghbari, K. Kaneko, and A. Makinouchi, "Content-trajectory approach for searching video databases," *IEEE Trans. Multimedia*, vol. 5, no. 4, pp. 516–531, Dec. 2003.

[15] B. Ionescu, V. Buzuloiu, P. Lambert, and D. Coquin, "Improved cut detection for the segmentation of animation movies," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2006, vol. 2, pp. II–II.

[16] B. Ionescu, D. Coquin, P. Lambert, and V. Buzuloiu, "Fuzzy color-based approach for understanding animated movies content in the indexing task," *J. Image Video Process.*, vol. 2008, pp. 8:1–8:17, Jan. 2008.

[17] L. Ott, P. Lambert, B. Ionescu, and D. Coquin, "Animation movie abstraction: Key frame adaptative selection based on color histogram filtering," in *Proc. 14th Int. Conf. Image Anal. Process.*, 2007, pp. 206–211.

[18] B. Ionescu, P. Lambert, D. Coquin, L. Ott, and V. Buzuloiu, "Animation movies trailer computation," in *Proc. 14th ACM Int. Conf. Multimedia*, 2006, pp. 631–634.

[19] K. Takayama, H. Johan, and T. Nishita, "Face detection and face recognition of cartoon characters using feature extraction," in *Proc. Image, Electron. Visual Comput. Workshop*, 2012, p. 48.

[20] S. Wang, J. Zhang, T. X. Han, and Z. Miao, "Sketch-based image retrieval through hypothesis-driven object boundary selection with HLR descriptor," *IEEE Trans. Multimedia*, vol. 17, no. 7, pp. 1045–1057, Jul. 2015.

[21] *The Illusion of Life: Disney Animation*. New York, NY, USA: Abbeville Press, 1981.

[22] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[23] Y. Tang, X. Wang, E. Dellandrea, and L. Chen, "Weakly supervised learning of deformable part-based models for object detection via region proposals," *IEEE Trans. Multimedia*, vol. 19, no. 2, pp. 393–407, Feb. 2017.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2016, pp. 779–788.

[25] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2014, pp. 2155–2162.

[26] C. Szegedy, S. E. Reed, D. Erhan, and D. Anguelov, "Scalable, high-quality object detection," arXiv:1412.1441, 2014.

[27] K. H. Lee and J. N. Hwang, "On-road pedestrian tracking across multiple driving recorders," *IEEE Trans. Multimedia*, vol. 17, no. 9, pp. 1429–1438, Sep. 2015.

[28] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Jun. 2015.

[29] O. Russakovsky *et al.*, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[30] P. Jacard, "The distribution of the flora in the alpine zone," *New Phytologist*, vol. 11, pp. 37–50, 1912.

[31] E. Rahtu and J. Heikkilä, "A simple and efficient saliency detector for background subtraction," in *Proc. 12th Int. Conf. Comput. Vis. Workshops*, Sep. 2009, pp. 1137–1144.

[32] Y. Hu, X. Xie, W.-Y. Ma, L.-T. Chia, and D. Rajan, "Salient region detection using weighted feature maps based on the human visual attention model," in *Proc. 5th Pacific Rim Conf. Multimedia Adv. Multimedia Inf. Process.*, 2005, pp. 993–1000.

[33] V. Mahadevan and N. Vasconcelos, "Background subtraction in highly dynamic scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Jun. 2008, pp. 1–6.
[34] E. Rahtu, J. Kannala, M. Salo, and J. Heikkilä, "Segmenting salient objects from images and videos," *Proc. 11th Eur. Conf. Comput. Vis. Comput. Vis.,* 2010, pp. 366–379.
[35] M. Kristan *et al.*, "The visual object tracking VOT2015 challenge results," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2015.
[36] G. Nebehay and R. Pflugfelder, "Clustering of static-adaptive correspondences for deformable object tracking," *Comput. Vis. Pattern Recogn.*, Jun. 2015, pp. 2784–2791.
[37] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2548–2555.
[38] D. Dueck and B. J. Frey, "Non-metric affinity propagation for unsupervised image categorization," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.
[39] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.
[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[41] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn. Workshops*, 2014, pp. 512–519.
[42] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, May 2001.
[43] M. Merler, B. Huang, L. Xie, G. Hua, and A. Natsev, "Semantic model vectors for complex video event recognition," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 88–101, Feb. 2012.
[44] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, pp. 972–977, 2007.
[45] "Comprehensive Annenberg Report on Diversity in Entertainment," 2016.
[46] S. L. Smith *et al.*, "Media, diversity, & social change initiative," 2016.
[47] K. Krippendorff, "Reliability in content analysis," *Human Commun. Res.*, vol. 30, no. 3, pp. 411–433, 2004.
[48] D. M. Christopher, R. Prabhakar, and S. Hinrich, "Introduction to information retrieval," *Introduction Inf. Retrieval*, vol. 151, p. 177, 2008.

Krishna Somandepalli received the Master's degree from University of California at Santa Barbara, CA, USA, in electrical and computer engineering. He received the Bachelor's degree in electronics and communication engineering from University Visvesvaraya College of Engineering, Bangalore, India. Following his Master's degree, he worked as an Assistant Research Scientist at NYU Langone Medical Center, New York, NY, USA. His research interests include multimodal analysis with image and signal data. He is currently working toward the Ph.D. degree in the Signal Analysis and Interpretation Laboratory Group, Department of Electrical Engineering, University of Southern California, Los Angeles, CA, USA.

Naveen Kumar received the Ph.D. degree in electrical engineering from the USC Viterbi School of Engineering, where he was a member of the Media Informatics and Content Analysis Group, Signal Analysis and Interpretation Lab. He received the B.Tech. degree in instrumentation engineering from the Indian Institute of Technology, Kharagpur, India, in 2009. He currently works at the Sony PlayStation R&D in San Mateo, CA, USA. His broad research interests include machine learning and signal processing for speech, multimedia and multimodal applications.

Tanaya Guha received the Ph.D. degree in electrical and computer engineering from the University of British Columbia (UBC), Vancouver, BC, Canada. She is an Assistant Professor in the Department of Electrical Engineering, Indian Institute of Technology (IIT) Kanpur. Prior to joining IIT Kanpur, she was a Postdoctoral Fellow at the Signal Analysis and Interpretation Lab (SAIL), University of Southern California.

Her current research interests include social and affective computing, multimedia analysis, and multimodal signal processing. Dr. Guha received Mensa Canada Woodhams Memorial Scholarship, Google Anita Borg Scholarship, and Amazon Grace Hopper celebration scholarship.

Shrikanth S. Narayanan (SM'88–M'95–SM'02–F'09) is the Niki & C. L. Max Nikias Chair in Engineering at the University of Southern California (USC), Los Angeles, CA, USA, and holds appointments as a Professor of Electrical Engineering, Computer Science, Linguistics, Psychology, Neuroscience and Pediatrics, the Research Director of the Information Science Institute, and as the Founding Director of the Ming Hsieh Institute. Prior to USC, he was with AT&T Bell Labs and AT&T Research from 1995–2000. At USC, he directs the Signal Analysis and Interpretation Laboratory (SAIL). His research focuses on human-centered signal and information processing and systems modeling with an interdisciplinary emphasis on speech, audio, language, multimodal and biomedical problems and applications with direct societal relevance. Prof. Narayanan is a Fellow of the National Academy of Inventors, the Acoustical Society of America, the International Speech Communication Association (ISCA) and the American Association for the Advancement of Science (AAAS), and a member of Tau Beta Pi, Phi Kappa Phi, and Eta Kappa Nu. He is the Editor-in-Chief for IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, an Editor for the *Computer Speech and Language Journal* and an Associate Editor for the APSIPA *Transactions On Signal And Information Processing*. He was also previously an Associate Editor of the IEEE TRANSACTIONS OF SPEECH AND AUDIO PROCESSING (2000–2004), IEEE SIGNAL PROCESSING MAGAZINE (2005–2008), IEEE TRANSACTIONS ON MULTIMEDIA, (2008–2011), the IEEE TRANSACTIONS ON SIGNAL AND INFORMATION PROCESSING OVER NETWORKS (2014–2015), IEEE TRANSACTIONS ON AFFECTIVE COMPUTING (2010–2016), and the *Journal of the Acoustical Society of America* (2009–2017). He received several honors including Best Transactions Paper awards from the IEEE Signal Processing Society in 2005 (with A. Potamianos) and in 2009 (with C. M. Lee) and selection as an IEEE Signal Processing Society Distinguished Lecturer for 2010–2011 and ISCA Distinguished Lecturer for 2015–2016. Papers coauthored with his students have received awards including the 2014 Ten-year Technical Impact Award from ACM ICMI and at several conferences. He has published more than 750 papers and has been granted 17 U.S. patents.

993 **QUERIES**

994 Q1. Author: Please check whether the affiliation of authors is okay as set.
995 Q2. Author: Please provide year information in Ref. [13].
996 Q3. Author: Please update Ref. [26], if already published in prints.
997 Q4. Author: Please provide page range in Refs. [19], [28], and [35].
998 Q5. Author: Please provide full bibliographic details in Ref. [46].