IEICE TRANS. INF. & SYST., VOL.E98–D, NO.4 APRIL 2015

LETTER

Robust Superpixel Tracking with Weighted Multiple-Instance Learning

Xu CHENG[†a)], **Nijun LI**[†], **Tongchi ZHOU**[†], ***Nonmembers***, **Lin ZHOU**[†], ***Member***, ***and*** **Zhenyang WU**[†], ***Nonmember***

SUMMARY This paper proposes a robust superpixel-based tracker via multiple-instance learning, which exploits the importance of instances and mid-level features captured by superpixels for object tracking. We first present a superpixels-based appearance model, which is able to compute the confidences of the object and background. Most importantly, we introduce the sample importance into multiple-instance learning (MIL) procedure to improve the performance of tracking. The importance for each instance in the positive bag is defined by accumulating the confidence of all the pixels within the corresponding instance. Furthermore, our tracker can help recover the object from the drifting scene using the appearance model based on superpixels when the drift occurs. We retain the first $(k-1)$ frames' information during the updating process to alleviate drift to some extent. To evaluate the effectiveness of the proposed tracker, six video sequences of different challenging situations are tested. The comparison results demonstrate that the proposed tracker has more robust and accurate performance than six ones representing the state-of-the-art.

key words: *visual tracking, multiple instance learning, appearance model, superpixel*

1. Introduction

As important issues in computer vision, object tracking has a wide range of applications in many areas, such as action recognition, object classification and human-computer interaction. The past year has witnessed advances in object tracking, and various attempts have been made to address some challenging problems.

Current Tracking techniques can be classified into either generative or discriminative approaches. Generative approaches formulate the tracking as searching for regions most similar to the learned appearance model. Adam et al. [1] divided the object state into multiple patches to handle the partial occlusions of the object. But it ignored the problem of template updating. Visual tracking decomposition (VTD) [2] could cover a wide range of object changes, which decomposed the observation and motion models into multiple basic corresponding models. David et al. [3] proposed an efficient method which learned a linear subspace online to model the variations of object appearance. In [4], an efficient tracker with the SIFT feature correspondence and multiple fragments was used to track the object. Recently, sparse representation has attracted considerable interest in object tracking [5], [6]. The reason is that the trackers based on sparse representation are robust to occlusion. However, they are less effective for the tracking in background clutter.

Discriminative methods treat object tracking as a binary classification problem. Wang et al. [7] presented a discriminative appearance model based on superpixels to distinguish the object and background. An appropriate combination of generative and discriminative models [8] can better help alleviate the drifting problems. In [9], an ensemble tracking method that combined a set of trained weak classifiers into a strong one to separate the object from the background was presented. A later effort [10] utilized an online boosting scheme to update the discriminative features. However, they only used one positive sample to update the classifier. The entire tracking performance will degrade if the positive sample is imprecise. To overcome this issue, Babenko et al. [11] proposed an online multiple-instance learning (MIL) technique. The method can achieve a better performance to some extent in that it assures that the tracking result is the most correct positive sample. However, the assumption does not always hold when large motion changes occur.

In this paper, our approach takes the sample importance into account for the MIL learning procedure. We exploit the mid-level cue (superpixel) to model the appearance representation. During the tracking, the importance of instances (samples) is evaluated by the learned appearance model. Then these weighted instances are used to update the classifier. In addition, learned appearance model can also help correct the object's drifting. The experimental results demonstrate that the proposed approach performs favorably against other state-of-the-art trackers.

This paper is structured as follows. In Sect. 2, we introduce our tracking scheme, the principle of our tracking algorithm and its advantages over popular algorithms in details. Experimental results are presented in Sect. 3. Section 4 gives our conclusion.

2. The Proposed Tracking Method

2.1 System Overview

The main flow of our tracker is shown in Fig. 1. When a new frame is coming, particles around the object position

Manuscript received September 3, 2014.
Manuscript revised November 16, 2014.
Manuscript publicized January 15, 2015.

[†]The authors are with the School of Information Science and Engineering, Southeast University, Nanjing, Jiangsu, 210096, PR China.
a) E-mail: xcheng@seu.edu.cn (Corresponding author)

DOI: 10.1587/transinf.2014EDL8176

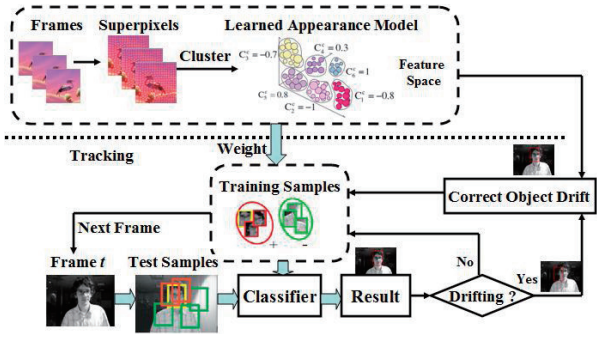

Fig. 1 The workflow of the proposed tracking algorithm.

of last frame are randomly sampled. Then we score the confidence of each particle with trained classifier (Sect. 2.3), and the particle with the maximum confidence is regarded as tracking result. Once the object drifts, superpixels-based the learned appearance model (Sect. 2.2) can help recover the object's position (Sect. 2.4). Different from the MIL [11], we consider the importance for each instance in MIL scheme and the weight of instance is obtained by the sum of pixels responses within the corresponding instance. In addition, it is necessary to update the appearance model in a fixed length frame interval (Sect. 2.5).

2.2 Superpixel Based Appearance Model

In this paper, we utilize the mid-level cue with structural information captured in superpixels. To model the object appearance, we first segment the surrounding region of the object into N superpixels in the first ten frames of a video. In fact, the size of the target region does not have a direct impact on the number of superpixels. But computational time will grow accordingly as the surrounding region of target increases. Therefore, we apply superpixel segmentation to the surrounding region of the target (rather than the entire image) for efficient and effective object tracking. Each superpixel s_i $(i = 1, \ldots, N)$ is represented by a feature vector $\mathbf{f}_i$ which is a Haar-like feature. The locations of some small rectangles are randomly generated in each superpixel, and these rectangles consist of a set of feature templates which are used to widely cover a specific appearance of the object. The number of rectangles in each superpixel ranges from 2 to 4. The heights and the weights are randomly generated. The pixels in the same rectangle have the same weight that is randomly generated from the range (0, 1]. Each Haar-like feature is computed by the sum of weighted pixels. Then, we exploit the mean shift clustering method [12] that can automatically cluster r classes based on the size of feature vectors of superpixels. Superpixel members of the ith cluster cover the image regions to indicate how probable its superpixel members belong to the object or background. Therefore, we define a confidence measure for the ith cluster as follows.

$$Con_i^0 = \frac{S^+ - S^-}{S^+ + S^-}, \quad i = 1, \ldots, r \tag{1}$$

where S^+ denotes the size of cluster area overlapping the object area, and S^- denotes the size of the cluster area outside the object area. Therefore, a prior knowledge Con_i^0 of the object from the first ten frames is regarded as the initial appearance model.

In test frame, a surrounding region of the object state is segmented into M superpixels. The confidence measure for the kth superpixel is evaluated by

$$Con_k = \exp(\eta \times \|f_k - f_{c,i}\|_2) \times Con_i^0, \quad k = 1, \ldots, M \tag{2}$$

where f_k and Con_k denote the feature vector of the kth superpixel and the corresponding confidence value, respectively; $f_{c,i}$ indicates the feature center of the ith cluster that f_k belongs to; η is a normalization term (2 in this paper). We will utilize the superpixel confidence to determine the importance of positive samples.

2.3 Online Weighted MIL

Each pixel on the surrounding region of the object is assigned a response based on the superpixel confidence, and pixels outside the surrounding region with -1.

Our scheme integrates the sample importance into the learning process using weighted sum of instance probability. The weight of each instance can be obtained by accumulating the responses of all the pixels within the corresponding instance.

$$w_l = \sum_{(i,j) \in \mathrm{S}_l} v_l(i, j) \tag{3}$$

where $v_l(i, j)$ denotes the response value at location (i, j) within the lth instance S_l. Then, we need to retrain the classifier with these weighted samples in each frame.

We assure that there are n positive samples and m negative samples at current frame. Positive bag is drawn around the tracking result ($\mathbf{I_1}$) of last frame, which satisfies $\|\mathbf{I}_{pos} - \mathbf{I}_1\| < \alpha$ $(\alpha = 5)$. Negative samples in an annular region specified by $\alpha < \|\mathbf{I}_{neg} - \mathbf{I}_1\| < \beta$, where α and $\beta = 2\alpha$ denote inner and outer radii, respectively.

The positive bag probability is defined as follows.

$$p(y = 1 \mid \mathrm{X}^+) = \sum_{j=0}^{n-1} w_j p(y = 1 \mid x_j) \tag{4}$$

where w_j indexes the weight corresponding to the jth sample in the positive bag X^+; the samples with the higher confidences (weights) at current frame contribute more to the bag probability than those with the lower confidences. $p(y = 1|x_j)$ denotes the posterior probability of sample x_j to be positive; y is a binary label. Sample x_j can be represented by a feature vector $\mathbf{f}(x_j)$. So the posterior probability of x_j to be positive is computed by

$$p(y = 1 \mid x_j) = \sigma\left(\ln\left(\frac{p(\mathbf{f}(x_j) \mid y = 1)p(y = 1)}{p(\mathbf{f}(x_j) \mid y = 0)p(y = 0)}\right)\right) \tag{5}$$

where σ is a sigmoid function.

All of the instances in the negative bag are from the background region. Therefore, the negative instances contribute equally to the negative bag.

$$p(y=0\,|\,\mathrm{X}^-) = \sum_{j=n}^{n+m-1} (1 - p(y=1\,|\,x_j)) \tag{6}$$

Similar to MIL [11], a strong classifier $H_K()$ is defined as

$$H_K(x_j) = \ln\left(\frac{p(\mathbf{f}(x_j)\,|\,y=1)p(y=1)}{p(\mathbf{f}(x_j)\,|\,y=0)p(y=0)}\right) \tag{7}$$

We assure that uniform prior $p(\mathrm{y}=0) = p(y=1)$ and the features in $\mathbf{f}(x_j) = [f_1(x_j), \ldots, f_K(x_j)]^{\mathrm{T}}$ are independently distributed. So Eq. (7) is further written as

$$H_K(x_j) = \sum_{k=1}^{K} \ln\left(\frac{p(f_k(x_j)\,|\,y=1)}{p(f_k(x_j)\,|\,y=0)}\right) = \sum_{k=1}^{K} h_k(x_j) \tag{8}$$

where $h_k()$ is the kth weak classifier that is composed of a Haar-like feature. The conditional distributions are modeled as a Gaussian function, $p(f_k(x)|y=1) \sim N(u_1, \sigma_1)$ and $p(f_k(x)|y=0) \sim N(u_0, \sigma_0)$. The parameters (u_1, σ_1) are updated by the following Eq. (9) and Eq. (10).

$$u_1 = \gamma u_1 + (1-\gamma)\bar{u} \tag{9}$$

$$\sigma_1 = \gamma\sigma_1 + (1-\gamma)\sqrt{\frac{1}{n}\sum_{j|y_i=1}(f_k(x_{ij}) - u_1)^2} \tag{10}$$

where n is the number of positive samples and γ is a learning rate parameter ($\gamma = 0.8$). We can update u_0 and σ_0 with the similar rules.

Finally, we can greedily select the most discriminative weak classifier by maximizing the bag log-likelihood function $L(H)$ in a weak classifier pool $\phi = \{h_1, \ldots, h_M\}$.

$$L(H) = \sum_{s=0}^{1}\left(y_s \log\left(\sum_{j=0}^{n-1} w_j p(y=1\,|\,x_j)\right) + (1-y_s)\log\left(\sum_{j=n}^{n+m-1}(1 - p(y=1\,|\,x_j))\right)\right) \tag{11}$$

The selected K weak classifiers construct a strong classifier $H_K(x)$ to discriminate the object location.

2.4 Object's Recovery from the Drifts

Learned appearance model and the confidence map of superpixel are used to recover the object from the drifts. If the classification score of the object state in Eq. (8) is much larger than the predefined threshold (0.6 in our article) at current frame, it is means that the object location estimation is of high possibility to be object area. On the contrary, a drift is deemed to occur if the classification score for the current state is much lower than the empirical threshold and the distance between the current object location and the result of the last frame is too large (35 pixels in our paper). Once the object is far away from the location of last frame, the surrounding region of the object from the last frame is further expanded to segmenting more superpixels. Then we compute the confidence map for each superpixel based on the learned appearance model. The computed confidence map provides strong evidence where the object will appear; thereby correcting potential drifts from the inaccurate result at current frame. Figure 2 presents how our scheme recovers from the drifts with the superpixels information.

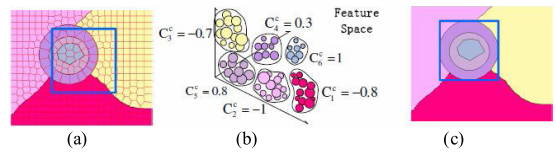

Fig. 2 Recovering from the drifts. (a) the drifted object and its surrounding region are segmented into superpixels. (b) clustering superpixels and computing the confidences of superpixel. (c) the estimation of the object area is corrected by clustering results (b).

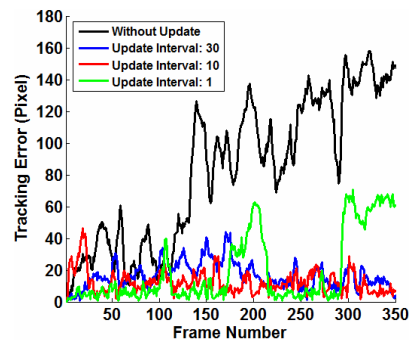

Fig. 3 The relationship of tracking errors and update interval.

Table 1 Average per-frame run time with the different update interval.

Update Interval U	Average Run Time/Frame
$U=0$	0.19s
$U=30$	0.26s
$U=10$	0.28s
$U=1$	0.46s

2.5 Appearance Model Update

The length of the retained sequence is set to L ($L = 12$). We update the object appearance with the retained sequence every U frames. In this paper, U is set to 10 to reach a compromise between the tracking errors and computational cost from Fig. 3 and Table 1. We put the result of every frame into the end of the retained sequence and delete the kth information in L ($k < L$, $k = 4$). All the superpixels are re-clustered after every U frames, which is the same as the training process in Sect. 2.2.

In the updating process, we need to retain the first $k-1$ frames information of L to avoid the drifting problem.

However, during the drift period, the process of appearance model updating is paused, because the inaccurate results may be introduced into the previous appearance model and lead to tracking failure.

For a long-term drift, object appearance cannot be well

matched with previous appearance model. In this case, search region of the object is expanded to segmenting more superpixels for recovering the lost object. Then we compute the confidence map for each superpixels using previous appearance model. The superpixels with high confidence map values are more likely to belong to the object, thereby recovering object location at current frame. Finally, we put the correct result from current frame into the end of retained sequences and delete the $(k+1)$th information. These superpixels are re-clustered and the confidences of all clusters are recalculated. Therefore, the appearance model change is updated.

3. Experiments

We carry out our approach on MATLAB platform with Intel Core 2 Duo 2.93GHz CPU and 2.96GB RAM. We run our tracker on all challenging videos and the average run time for each frame is 0.28s. Most of the computation is spent on superpixel generation. The SLIC (Simple Linear Iterative Clustering) algorithm [13] is used to segment the image into superpixels. On average, drift detection and correction take 0.01s and 0.1s, respectively. Our method is compared against six other popular trackers including IVT [3], VTD [2], FragTrack [1], L1 [6], MIL [11] and SPT [7]. All source codes are provided by the authors' websites for fair comparison.

There are three reasons why distance threshold in Sect. 2.4 is fixed empirically. First, the object change between the consecutive frames is usually gradual due to the mechanical movement. The change of object isn't too large. So object size isn't a main factor to decide the threshold. Second, it is difficult to find a good criterion to adaptively decide the threshold based on the size of the object. Too small or too large value has a bad impact on tracking performance. Third, during the tracking, the change of object size may be inaccurate due to occlusion and illumination change. If we use the inaccurate results to decide the threshold, the tracking performance will be unstable.

Faceocc2 sequence shows that the object experiences the partial occlusion and in-plane rotation. In Fig. 4 (a), MIL can not recover the lost object due to the inaccurate appearance model. Our approach achieves the promising results. The main difficulty of the DavidIndoor video is the illumination and poses variations in Fig. 4 (b). In Fig. 4 (c), woman sequence involves the partial occlusions and similar appearance background clutter. The improvement of our tracker can better track objects, where other trackers are less reliable.

Figure 5 (a)-(c) show some representative results under the circumstance of the illumination and abrupt motion changes. Singer1 clip contains illumination and scale variations as well as camera motion, which lead to most of the conventional trackers drift. Shaking sequence presents the light and the pose of the object is drastically varied due to the head shaking. VTD and MIL can track the object quite well except for some errors in some frames; while other popular trackers perform below par. In Deer sequence, the tracked object undergoes drastically appearance variations. Our tracker is superior to the traditional trackers which are less reliable.

Fig. 4 Object undergoes in plane rotation and partial occlusions.

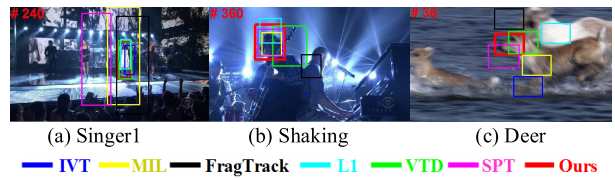

Fig. 5 Sampled tracking results in three sequences.

Table 2 Center location errors. The red bold fonts indicate the best performance; the blue bold fonts indicate the second best.

Video	IVT	VTD	FragTrack	MIL	L1	SPT	Ours
Faceocc2	31.9	**10.4**	16.7	14.1	11.1	18.7	**7.8**
Woman	167.5	136.6	113.6	122.4	28.9	**17.4**	**15.3**
David	**3.1**	13.6	105.5	16.1	7.6	11.2	**6.2**
Singer1	12.8	**4.1**	22.1	15.2	4.6	79.3	**3.2**
Shaking	124.4	**6.1**	85.7	11.2	103.7	66.9	**8.5**
Deer	130.8	**12.3**	94.4	66.5	140.9	108.8	**11.2**

Table 3 Tracking success rates. The red bold fonts indicate the best performance; the blue bold fonts indicate the second best.

Video	IVT	VTD	FragTrack	MIL	L1	SPT	Ours
Faceocc2	0.39	0.59	0.60	0.62	**0.67**	0.57	**0.81**
Woman	0.19	0.15	0.20	0.19	0.18	**0.75**	**0.78**
David	**0.74**	0.53	0.08	0.50	0.62	0.58	**0.75**
Singer1	0.51	**0.79**	0.34	0.34	0.71	0.12	**0.82**
Shaking	0.02	**0.73**	0.13	0.58	0.03	0.09	**0.75**
Deer	0.09	**0.51**	0.08	0.14	0.07	0.11	**0.55**

Table 2 shows the center location error which is defined as the average of Euclidean distance between the tracking result and the ground truth. Our tracker achieves the better or similar performance with other trackers.

Another criterion is the success rate of tracker which is defined by the PASCAL VOC [14] score $= \frac{\text{area}(R_T \cap R_G)}{\text{area}(R_T \cup R_G)}$. Given the tracking results R_T and the corresponding ground truth R_G, an object is successfully tracked when the score is above 0.5. In Table 3, we can see that our approach performs favorably against other trackers.

For feature selection, we only need to select $K = 20$ features with $M = 150$, which is much less computational burden than MIL which selects $K = 50$ from the $M = 250$.

Figure 6 and Table 4 demonstrate the efficiency of random selection. Similar methods using random selection scheme are VTD and MIL. Shaking sequence is more representative than other sequences due to its complex circumstance. In the sequence, we exploit other features to

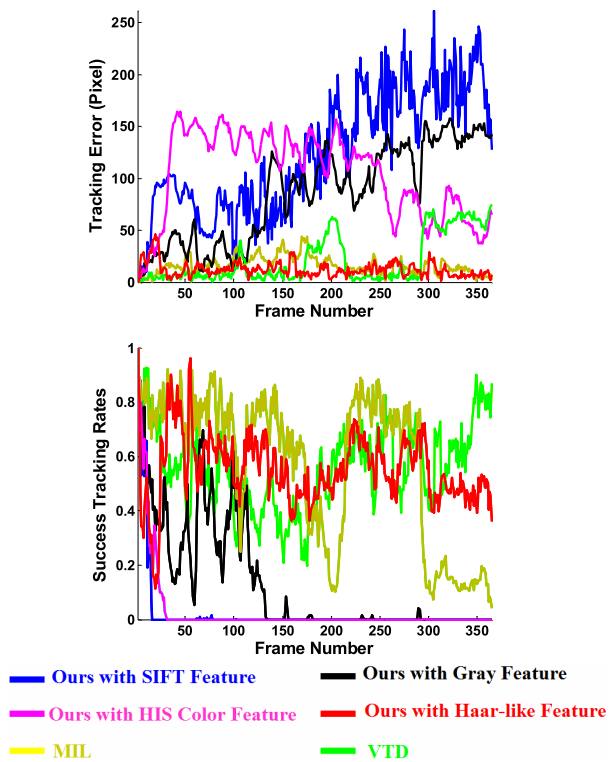

Fig. 6 A comparison of tracking performance between random selection feature and three deterministic features in Shaking sequence.

Table 4 Extraction time for different features.

Methods	Extraction Time(s)
Ours with SIFT Feature	0.27s
Ours with Gray Feature	0.006s
Ours with HSI Color Histograms	0.01s
Ours with Haar-like Feature	0.01s
MIL	0.01s
VTD	0.03s

replace random selection. First, we employ HSI color histograms [7] to represent the object. But color histograms are very sensitive to illumination changes. It is easy to fail when the object undergoes the drastically illumination and pose changes. Second, when the object is represented using SIFT feature points, it has a high computational cost for feature extraction. The proposed algorithm will fail if the target object cannot detect the feature points due to motion blur and lacking the local feature points. Third, gray feature is susceptible to noise leads to the tracking performance degraded.

4. Conclusion

In this paper, a robust superpixels-based tracking approach via weighted multiple-instance learning is proposed. We employ the more promising superpixels to model the object's appearance model. On one hand, the appearance model based on superpixels can determine the weight of each instance in positive bag. On the other hand, it can help recover from drifts. The comparison results indicate that our tracking approach is more robust.

Acknowledgments

This work is supported by National Nature Science Foundation of China (NSFC) under Grant (No. 60971098, 61302152, 61201345) and the Beijing Key Laboratory of Advanced Information Science and Network Technology (No. XDXX1308).

References

[1] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," CVPR, pp.798–805, 2006.
[2] J. Kwon and K.M. Lee, "Visual tracking decomposition," CVPR, pp.1269–1276, 2010.
[3] D. Ross, J. Lim, R.S. Lin, and M.H. Yang, "Incremental learning for robust visual tracking," Int. J. Comput. Vis., vol.77, no.1, pp.125–141, Jan. 2008.
[4] X. Cheng, N. Li, S. Zhang, and Z. Wu, "Robust visual tracking with SIFT features and fragments based on particle swarm optimization," Circuits Syst. Signal Process., vol.33, no.5, pp.1507–1526, May 2014.
[5] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," IEEE Trans. Pattern Anal. Mach. Intell, vol.33, no.11, pp.2259–2272, Nov. 2011.
[6] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust L1 tracker using accelerated proximal gradient approach," CVPR, pp.1830–1837, 2012.
[7] S. Wang, H. Lu, and F. Yang, et al., "Superpixel tracking," ICCV, pp.1323–1330, 2011.
[8] W. Zhong, H. Lu, and M. Yang, "Robust object tracking via sparsity-based collaborative model," CVPR, pp.1–8, 2012.
[9] S. Avidan, "Ensemble tracking," IEEE Trans. Pattern Anal. Mach. Intell, vol.29, no.2, pp.261–271, Feb. 2007.
[10] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," BMVC, pp.47–56, 2006.
[11] B. Babenko, M. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," IEEE Trans. Pattern Anal. Mach. Intell, vol.33, no.8, pp.1619–1632, Aug. 2011.
[12] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," IEEE Trans. Pattern Anal. Mach. Intell, vol.24, no.5, pp.603–619, May 2002.
[13] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," Technical Report 149300, Zurich, EPFL, 2010.
[14] M. Everingham, L.V. Gool, C.K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," Int. J. Comput. Vis., vol.88, no.2, pp.303–338, Feb. 2010.