FAST VISUAL OBJECT COUNTING VIA EXAMPLE-BASED DENSITY ESTIMATION

*Yi Wang, Yuexian Zou**

ADSPLAB/ELIP, School of ECE, Peking University, Shenzhen, 518055, China
*Corresponding author: zouyx@pkusz.edu.cn

ABSTRACT

Density estimation based visual object counting (DE-VOC) methods estimate the counts of an image by integrating over its predicted density map. They perform effectively but inefficiently. This paper proposes a fast DE-VOC method but maintains its effectiveness. Essentially, the feature space of image patches from VOC can be clustered into subspaces, and the examples of each subspace can be collected to learn its embedding. Also, it is assumed that the neighborhood embeddings of image patches and their corresponding density maps generated from training images are similar. With these principles, a closed form DE-VOC algorithm is derived, where the embedding and centroid of each neighborhood are precomputed by the training samples. Consequently, the density map of a given patch is estimated by simple classification and mapping. Experimental results show that our proposed method is comparable with mainstream ones on counting accuracy while running much faster in testing phase.

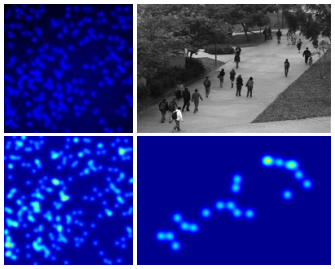
Index Terms— Visual object counting, density estimation, example-based, locally linear embedding, fast implementation

1. INTRODUCTION

The task of visual object counting (VOC) is to estimate the number of objects we are interested in from an image or video. Nowadays, VOC has been attracting tremendous attention as its pervasive application in numerous fields, such as wildlife census, crowd surveillance, etc.

VOC problem is challenging as the result of common overlap between objects, severe occlusion and complicated background environment. To tackle these problems, mainstream methods can be categorized into two types: global regression based and density estimation based. For global regression based methods (GR-VOC) [1-5], they learn the direct mapping between global image features and counterpart count. The performance of such methods depend heavily on the craftiness of the used feature and regression model.

Compared with GR-VOC, density estimation based methods (DE-VOC) are more promising with their ability to estimate the object count in any image region, which can offer the object distribution information. DE-VOC counts the number of objects by predicting an image density whose integral over any image region yields the object counts within that region [6]. This idea is firstly introduced by Lempitsky, and he realized it by learning the pixel-wise linear regression between image dense features and its object density. Following his work, Zhou extended original framework to do VOC for arbitrary objects and scenes [7]; Fiaschi applied regression forests and structural labels for efficient implementation while preserving the effectiveness [8]. These DE-VOC methods are all using regression to estimate the object density. For achieving satisfying regression performance, the dense features they used are sophisticated and time-consuming.



(a) (b)

Fig. 1. Images with objects (first row) and their corresponding generated density maps (second row, displayed in jet colormap). (a) cell image; (b) pedestrian image in dataset UCSD.

In this paper, we propose a fast example-based object density estimation method for VOC (FE-VOC). Instead of learning the mapping between dense features and their counterpart density maps, we exploit relationship between images and their corresponding density maps in two distinguished feature spaces. Specifically, based on the observation of Fig. 1, the images of cells and pedestrians look similar to their counterpart density maps in geometry. Hence, we assume that the patches of object images share the similar local geometry with that of counterpart density maps in two feature spaces. Such geometry can be solved by locally linear embedding (LLE) [9, 10], and then the density map of the image patch can be estimated by preserving the geometry.

However, implementing LLE is time-consuming due to exhaustive search for nearest neighbors. Encouraged by the work in [11], we stick to the same insight that the distinguishable object distribution patterns are finite. It suggests the possible salient neighborhoods of the object image patches are also finite in feature space. Thus, we divide the feature spaces

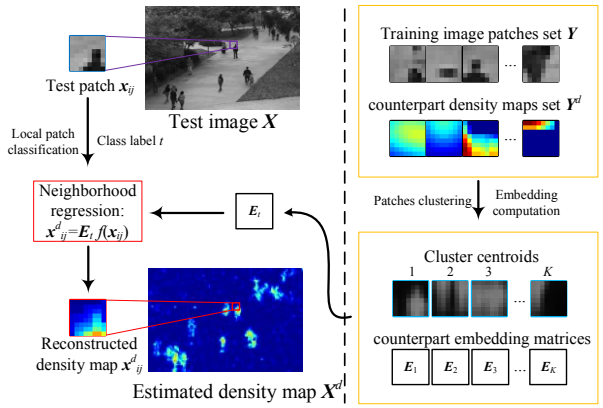


Fig. 2. The framework of our proposed method. Left area of dash line is for testing and the other side is for training.

of image patches and their counterpart density maps into subspaces, and compute the embedding of each subspace formed by image patches. Consequently, the density map of input patch can be estimated by simple classification and mapping with the corresponding embedding matrix. The whole framework of this method is illustrated in Fig. 2.

The rest of paper is organized as follows: section 2 introduces the generation of density maps for training; section 3 presents the DE-VOC problem formulation in LLE way and a fast example-based method for it; section 4 shows extensive experiments and related analysis on cell and pedestrian data; section 5 makes the conclusion of our work.

2. PRELIMINARIES

The core of object density estimation based VOC is to compute the relationship between object images and their counterpart density maps. Usually the ground truth density maps are usually defined as a sum of 2D Gaussian kernels of the distribution of objects, just as described in [6, 7]. In training phase, a set of N images $\boldsymbol{I}_1, \boldsymbol{I}_2, \boldsymbol{I}_3, \dots, \boldsymbol{I}_N$ is pre-allocated. In every $\boldsymbol{I}_i (1 \le i \le N)$, all objects of interest are considered to be annotated with a set of 2D points $\boldsymbol{P}_i$. Thus the ground truth density function of each pixel $p_i \in \boldsymbol{I}_i$ is computed by a sum of 2D Gaussian kernels based on annotated points:

$$F_i^{\mathrm{o}}(p) = \sum_{P \in \boldsymbol{P}_i} \mathcal{N}(p; P, \delta^2) \tag{1}$$

where P is a user-annotated dot and δ is the smoothness parameter. δ is set to 6 for all experiments in Section 3. With the definition in Eqn. (1), the ground truth density map $\boldsymbol{I}_i^d$ of training image $\boldsymbol{I}_i$ is defined as

$$\forall p \in \boldsymbol{I}_i^d, \boldsymbol{I}_i^d(p) = F_i^{\mathrm{o}}(p) \tag{2}$$

Some instances of $\boldsymbol{I}_i^d$ are displayed in Fig. 1.

With the density map $\boldsymbol{I}_i^d$, the object count $c(\boldsymbol{I}_i)$ can be computed by integrating over the density map

$$c(\boldsymbol{I}_i) = \sum_{p \in \boldsymbol{I}_i^d} \boldsymbol{I}_i^d(p) \tag{3}$$

In our method, training data are desired in patch form. Consequently, a set of image patches $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_M\}$ are extracted from the training images $\boldsymbol{I}_i$, $i \in \{1,2,\dots,N\}$, and the density maps set $\boldsymbol{Y}^d = \{\boldsymbol{y}_1^d, \boldsymbol{y}_2^d, \dots, \boldsymbol{y}_M^d\}$ of corresponding patches are extracted from $\boldsymbol{I}_i^d$, $i \in \{1,2,\dots,N\}$. The feature set $\boldsymbol{Y}_f = \{f(\boldsymbol{y}_1), f(\boldsymbol{y}_2), \dots, f(\boldsymbol{y}_M)\}$ is generated by applying feature extractor $f(.)$ to all patches in $\boldsymbol{Y}$. All elements from $\boldsymbol{Y}$, $\boldsymbol{Y}^d$ and $\boldsymbol{Y}_f$ can be seen as feature vectors in their feature spaces, respectively. By using Eqn. (3) with $\boldsymbol{y}_i^d$, the object count $c(\boldsymbol{y}_i)$ of every image patch $\boldsymbol{y}_i$ can be acquired.

3. METHOD

3.1. Example-based object density estimation by LLE

Instead of learning the regression model between image features and their counterpart density maps, we learn to estimate the density of input image patch over the generalization of training patches. As assumed that two manifolds, which are formed by the features of image patches and their counterpart density maps respectively, share the similar local geometry. In LLE, such local geometry of a feature vector can be characterized by how the feature vector can be linearly reconstructed by its neighbors [9, 10]. For a given test image patch $\boldsymbol{x}$ with unknown density, we compute its reconstruction weights of its neighbors from feature space of $\boldsymbol{Y}_f$ by minimizing the reconstruction error. Then the density map $\boldsymbol{x}^d$ will be predicted by using the reconstruction weights to the density maps of neighboring patches from $\boldsymbol{Y}^d$. As this method are implemented via the generalization of examples, it is named as example-based VOC (E-VOC). Similar to the formulation in [9, 10], E-VOC can be modeled as:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} ||f(\boldsymbol{x}) - \widetilde{\boldsymbol{Y}}_f \boldsymbol{w}||_2^2 \tag{4}$$

$$\boldsymbol{x}^d \cong \widetilde{\boldsymbol{Y}}^d \boldsymbol{w}^* \tag{5}$$

where $\widetilde{\boldsymbol{Y}}_f = [f(\widetilde{\boldsymbol{y}}_1), f(\widetilde{\boldsymbol{y}}_2), \dots, f(\widetilde{\boldsymbol{y}}_K)]$ is a training patch subset formed by the K nearest neighbors of $f(\boldsymbol{x})$ from $\boldsymbol{Y}_f$. $\widetilde{\boldsymbol{Y}}^d = [\widetilde{\boldsymbol{y}}_1^d, \widetilde{\boldsymbol{y}}_2^d, \dots, \widetilde{\boldsymbol{y}}_K^d]$ and $\widetilde{\boldsymbol{y}}_i^d$ is the density map of $\widetilde{\boldsymbol{y}}_i$.

In E-VOC, Eqn. (4) achieves the local geometry of $f(\boldsymbol{x})$, and Eqn. (5) reconstructs the target density map $\boldsymbol{x}^d$ by preserving such local geometry.

As its constrained least squares form, Eqn. (4) has an analytic solution and $\boldsymbol{w}$ can be solved efficiently: $\boldsymbol{w} = (\widetilde{\boldsymbol{Y}}_f^{\mathrm{T}} \widetilde{\boldsymbol{Y}}_f + \lambda \boldsymbol{I})^{-1} \widetilde{\boldsymbol{Y}}_f^{\mathrm{T}} f(\boldsymbol{x})$. The implementation of constraints in Eqn. (5) is usually realized by K-nearest-neighbors (KNN) algorithm.

3.2. Fast example-based VOC

Taking the analytic solution of $\boldsymbol{w}$ into Eqn. (5), the solution of $\boldsymbol{x}^d$ can be expanded as:

$$\boldsymbol{x}^d \cong \widetilde{\boldsymbol{Y}}^d \left(\widetilde{\boldsymbol{Y}}_f^{\mathrm{T}} \widetilde{\boldsymbol{Y}}_f + \lambda \boldsymbol{I}\right)^{-1} \widetilde{\boldsymbol{Y}}_f^{\mathrm{T}} f(\boldsymbol{x}) = \boldsymbol{E} f(\boldsymbol{x}) \tag{6}$$

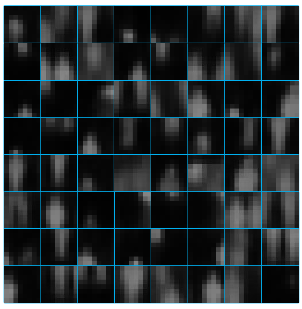


Fig. 3. The partial centroids of clusters on dataset UCSD (displayed in foreground feature). The patch size is 8×8

where $\boldsymbol{E} = \widetilde{\boldsymbol{Y}}^d\left(\widetilde{\boldsymbol{Y}}_f^{\mathrm{T}}\widetilde{\boldsymbol{Y}}_f + \lambda\boldsymbol{I}\right)^{-1}\widetilde{\boldsymbol{Y}}_f^{\mathrm{T}}$ is the embedding matrix computed from the neighborhood of $\boldsymbol{x}$ in two feature spaces.

Inspired by the idea that the possible crowd behaviors are infinite but the space of distinguishable crowd motion patterns may not be all that large [11], we expend this insight further: in VOC, the salient patterns of image patches extracted from images of objects are finite. With this insight, it is believed the primary neighborhoods of image patches in feature space are also finite, hence they can be depicted in training phase instead of testing phase, in which will save huge time.

To describe the possible neighborhoods of specific VOC problem, we cluster the feature space of image patches to approximate the results. Assumed the number of the possible salient neighborhoods of training data $\boldsymbol{Y}_f$ and $\boldsymbol{Y}^d$ is K, so all feature vectors in $\boldsymbol{Y}_f$ will be clustered into groups as $\boldsymbol{C}_i$ ($1 \le i \le K$), and the counterpart density maps cluster $\boldsymbol{C}_i^d$ is produced by putting the density map patches together according to the index set of the corresponding elements in $\boldsymbol{C}_i$. In Fig. 3, some centroids of clusters $\boldsymbol{C}$ on UCSD dataset are visualized. It is obvious that these centroids grasp the prominent crowd patterns, such as the outlines of head, body, or something else. With $\boldsymbol{C}_i$ and $\boldsymbol{C}_i^d$, their embedding matrix can be formulated as

$$\boldsymbol{E}_i = \boldsymbol{C}_i^d\left(\boldsymbol{C}_i{}^{\mathrm{T}}\boldsymbol{C}_i + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{C}_i{}^{\mathrm{T}} \tag{7}$$

where $\lambda = 0.001$ for our method in experiments.

With the all precomputed embedding matrices, we need to figure out which neighborhood the test patch belongs to. In LLE, the correlation is described as the similarity of features vectors in their feature space. Following this idea, we decide the neighborhood of test patch based on the similarity of the centroid of the neighborhood and itself, just as:

$$i^* = \arg\min_{1\le i\le K} D(centroid(\boldsymbol{C}_i), f(\boldsymbol{x})) \tag{8}$$

where i^* is the desired index of the cluster which $\boldsymbol{x}$ most likely lies in. Here $D(.)$ is also Euclidean distance metric, so the classification of input patch is finally formulated as:

$$i^* = \arg\min_{1\le i\le K} ||centroid(\boldsymbol{C}_i) - f(\boldsymbol{x})||_2^2 \tag{9}$$

Thus, the density map of $\boldsymbol{x}$ can be quickly calculated by $\boldsymbol{E}_{i^*}$ and Eqn. (6).

The whole FE-VOC algorithm is summarized as following Algorithm 1. It also has been illustrated in Fig. 2.

Algorithm 1 (FE-VOC)

Input: test image $\boldsymbol{X}$, training examples sets $\boldsymbol{Y}_f$ and $\boldsymbol{Y}^d$
Output: the density map $\boldsymbol{X}^d$, the estimated count $c(\boldsymbol{X})$
1: **for** each input patch $\boldsymbol{x}_{ij}$ extracted by $\boldsymbol{P}_{ij}$ in test image $\boldsymbol{X}$, where $\boldsymbol{P}_{ij}$ is a projection matrix that extracts the (i,j)th patch from $\boldsymbol{X}$
do
2: Find the index t^* of the desired neighborhood based on $\boldsymbol{x}_{ij}$ and Eqn. (9).
3: Compute the density map patch: $\boldsymbol{x}_{ij}^d = \boldsymbol{E}_{t^*} f(\boldsymbol{x})$. Put $\boldsymbol{x}_{ij}^d$ into the $\boldsymbol{X}^d$ based on $\boldsymbol{P}_{ij}$.
4: **end for**
5: Get the estimated density map of $\boldsymbol{X}$: $\boldsymbol{X}^d$, and the estimated count of $\boldsymbol{X}$: $c(\boldsymbol{X}) = \sum_{p\in\boldsymbol{X}^d}\boldsymbol{X}^d(p)$.

4. EXPERIMENT

We evaluate our method on cell and pedestrian data from two public benchmark datasets including bacterial cell dataset [6] and UCSD [3]. The details of these two datasets can be found in Table 1 and example frames are shown in Fig. 1. For comparison of different methods, mean absolute error (MAE) is employed as the evaluation metric. Unless otherwise specified, the patch size used is 4×4, and the patch step is set to 2 for both training and testing in our method.

4.1. Performance on bacterial cell dataset

On this dataset, we adhere to the experimental protocols in [6], hence the performance of comparative methods can be comparable directly. Specifically, the first 100 images are reserved for training and the rest are for validation. Each time 5 different random subsets containing N ($N = 1,2,4,\ldots,32$) samples from training set are generated for calculating the MAE and their standard deviations. In our method, the clustering number K is estimated empirically on the union of the training and testing sets. For features, RR [1], KRR [12] and Density MESA [6] employ dense SIFT coded by bag of words as features, while E-VOC and FE-VOC only use the raw data extracted from blue channel of images as features.

From Fig. 4, it is worth noting that the MAE computed by all methods descent gradually with the rise of training number, and the MAE produced by FE-VOC stays lowest with no more than 8 training images among all. When training size is larger than 8, Density MESA [6] performs best but slightly better than FE-VOC.

4.2. Performance on pedestrian datasets

On UCSD pedestrian dataset, the experimental protocols we use in UCSD are just the same as that in [6]. Specifically, the dataset is divided into 4 different training and testing sets: 1) 'maximal': training on frames 600:5:1400; 2) 'downscale':

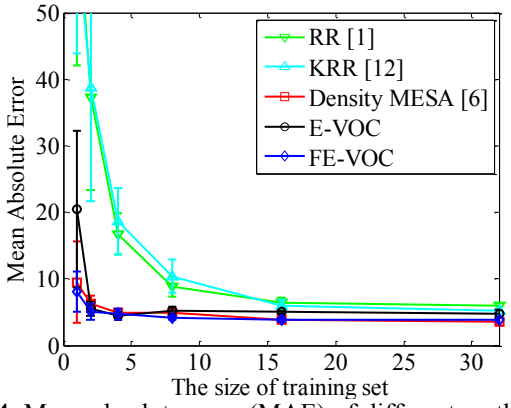


Fig. 4. Mean absolute error (MAE) of different methods on cell dataset.

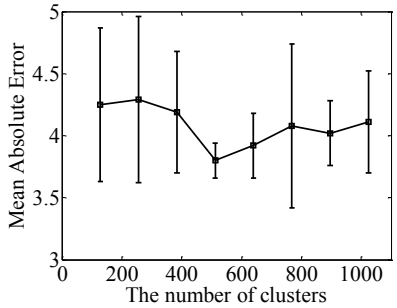


Fig. 5. Mean absolute error (MAE) of FE-VOC with different clustering number on cell dataset.

training on frames 1205:5:1600; 3) 'upscale' : training on frames 805:5:1100; 4) 'minimal': training on frames 640:80:1360. The frames which do not show in training procedure would be tested. For our method, only 32 random images are taken as training set for 'maximal' and 'downscale', and all training data are used for 'upscale' and 'minimal'. The K used is set by trials, and the features used in FE-VOC are simple foreground features of images (just as shown in Fig. 3), while other methods employ fused features [6] and feature selection technique. Here E-VOC is not introduced on this dataset as there exist more than thousands of testing images and E-VOC is very slow on testing. The experimental results are reported in Table 2.

It is clear that Density MESA [6] performs best in all settings, and our method gives fairly good predictions as second only to Density-MESA in setting 'downscale' and 'minimal'.

4.3. The impact of clustering number

Here the diverse number of possible primary neighborhoods in FE-VOC is evaluated on cell dataset. We still adhere to protocols used in subsection 4.1 and set training size $N = 16$. The clustering number K is set from 128 to 1024 with step 128. Fig. 5 indicates the trend of counting accuracy with different K. It is noted that the MAE computed by FE-VOC reaches the minimum ($K = 512$). The MAE decreases smoothly from $K = 128$ to 512, and increases gradually from $K = 512$ to 1024. Thus, for FE-VOC, the smaller MAE would be obtained with a more reasonable setting on K, and vice versa.

Table 1. Statistics of three datasets (T_f: total number of frames; R: the resolution; N_o: the mean number of objects presented in single frame; C: the color channel of frames)

Dataset	T_f	R	N_o	C
Cell	200	256×256	171±64	RGB
UCSD	2000	158×238	29±9	Gray

Table 2. Mean absolute errors (MAE) on UCSD dataset

Method	max	down	up	min
Regression [2]	2.07	2.66	2.78	N/A
Regression [4]	1.8	2.34	2.52	4.46
Density-MESA [6]	**1.7**	**1.28**	**1.59**	**2.02**
Density-RF [8]	**1.7**	2.16	1.61	2.2
FE-VOC	1.98	1.82	2.74	2.10

Table 3. Computational cost on cell dataset

Method	Feature Extraction	Density map reconstruction	Total time
Density [6]	9.499 s	0.006 s	9.505 s
E-VOC	0.084 s	201.242 s	201.326 s
FE-VOC	0.083 s	1.569 s	1.652 s

4.4. Computational efficiency evaluation

In this part, the computational cost among original E-VOC, Density-MESA [6] and FE-VOC is compared on bacterial cell dataset. All experimental settings of these methods are the same, and they all employ the same 16 images for training. The final time consumed is calculated by the mean processing time of 100 test images.

Just as shown in Table 3, FE-VOC is one or two orders of magnitude speed faster than other two. Specifically, FE-VOC and E-VOC spend much less time than Density-MESA on feature extraction as they just use simple features. Owing to the precomputed embedding matrices, FE-VOC runs much faster compared with E-VOC on density map reconstruction.

5. CONCLUSION

In this paper, we propose a fast example-based method for VOC problem. It runs fast while nearly makes no compromise on counting accuracy. This method is developed under the intuition that the distinguishable object distribution patterns are finite, thus all the embedding of counterpart neighborhoods can be computed in training phase instead of testing phase. Extensive experiments validate the effectiveness and efficiency of our method even with simple geometric features. In the future, we will try to estimate the quantity of salient object distribution patterns automatically.

6. ACKNOWLEDGEMENT

This work was partially supported by the Shenzhen Science & Technology Fundamental Research Program (No: JCYJ20150430162332418).

7. REFERENCES

[1] K. Chen, C. C. Loy, S. Gong, and T. Xiang, "Feature Mining for Localised Crowd Counting," in *BMVC*, 2012, p. 3.

[2] D. Kong, D. Gray, and H. Tao, "A viewpoint invariant approach for crowd counting," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, 2006, pp. 1187-1190.

[3] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1-7.

[4] D. Ryan, S. Denman, C. Fookes, and S. Sridharan, "Crowd counting using multiple local features," in *Digital Image Computing: Techniques and Applications, 2009. DICTA'09.*, 2009, pp. 81-88.

[5] K. Chen, S. Gong, T. Xiang, and C. C. Loy, "Cumulative attribute space for age and crowd density estimation," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013, pp. 2467-2474.

[6] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Advances in Neural Information Processing Systems*, 2010, pp. 1324-1332.

[7] Y. Zhou and J. Luo, "A practical method for counting arbitrary target objects in arbitrary scenes," in *Multimedia and Expo (ICME), 2013 IEEE International Conference on*, 2013, pp. 1-6.

[8] L. Fiaschi, R. Nair, U. Koethe, and F. Hamprecht, "Learning to count with regression forest and structured labels," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012, pp. 2685-2688.

[9] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science,* vol. 290, pp. 2323-2326, 2000.

[10] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004, pp. I-I.

[11] M. Rodriguez, J. Sivic, I. Laptev, and J.-Y. Audibert, "Data-driven crowd analysis in videos," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 1235-1242.

[12] S. An, W. Liu, and S. Venkatesh, "Face recognition using kernel ridge regression," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 2007, pp. 1-7.