Fixation Bank: Learning to Reweight Fixation Candidates

Jiaping Zhao, Christian Siagian, Laurent Itti
University of Southern California
{jiapingz, siagian, itti}@usc.edu

Abstract

Predicting where humans will fixate in a scene has many practical applications. Biologically-inspired saliency models decompose visual stimuli into feature maps across multiple scales, and then integrate different feature channels, e.g., in a linear, MAX, or MAP. However, to date there is no universally accepted feature integration mechanism. Here, we propose a new a data-driven solution: We first build a "fixation bank" by mining training samples, which maintains the association between local patterns of activation, in 4 feature channels (color, intensity, orientation, motion) around a given location, and corresponding human fixation density at that location. During testing, we decompose feature maps into blobs, extract local activation patterns around each blob, match those patterns against the fixation bank by group lasso, and determine weights of blobs based on reconstruction errors. Our final saliency map is the weighted sum of all blobs. Our system thus incorporates some amount of spatial and featural context information into the location-dependent weighting mechanism. Tested on two standard data sets (DIEM for training and test, and CRCNS for test only; total of 23,670 training and 15,793 + 4,505 test frames), our model slightly but significantly outperforms 7 state-of-the-art saliency models.

1. Introduction

Saliency models have been developed to measure the likelihood of a location to attract human attention [14]. Typical computational saliency models employ the following paradigm: (1) compute individual activation maps in several feature channels, (2) combine activation maps into a master saliency map [14, 8, 33]. Several bottom-up features including color contrast, intensity, orientation, spatial frequency, and motion, are typically extracted, and they are then integrated linearly [14], in a MAX [17], or MAP [29] manner into a master saliency map.

Yet, to date, there is no universally accepted feature integration mechanism for visual attention allocation; some physiological and psychophysical studies support linear feature integration [28, 21], while others [17] have raised psychophysical arguments against linear summation strategies. Since neural mechanisms underlying visual attention need further exploration, here we propose a data-driven approach using human eye-tracking data to learn associations between patterns of feature responses and human fixations.

Several computational saliency models learn from biological or behavioral data. Itti and Koch [13] learned feature map weights that would render some specific objects more salient, from manual ground-truth annotations around training exemplars of these objects. Zhao and Koch [33] used adaboost to learn visual saliency by taking into account feature selection, weight assignments, and integration in a unified framework. Kienzle et al [16] directly learned a fixation model from fixated and non-fixated image patches using SVM. Judd et al [15] extracted low, middle and high-level features from images, and used SVM to learn the mapping from feature to fixations. All these works learn saliency in a pixel-wise fashion and ignore neighboring-pixel dependencies. Moreover, none of them take into account contextual information, although Judd et al [15] took advantage of semantic faces. While bottom-up, low-level properties of a visual scene play a significant role in visual attention, undoubtedly high-level factors such as scene context [9] and search goals [23] are not negligible. Torralba [27] explicitly model global scene contexts in a visual search task. While their method improves gaze prediction, the drawback is the need for sufficient amounts of labelled data to build different contextual models for different objects. A more recent work [24] studied saliency in dynamic scenes and proposed to use Gaussian blobs on the GBVS [8] saliency map as fixation candidates. They used random forest regression to learn transition probabilities between fixation candidates on contiguous frames to do fixation candidate selection.

Inspired by object bank [19] and action bank [25] for image and video activity representation, we present fixation bank to model fixation allocations. We use primitive features, including color ($\mathcal{C}$), intensity ($\mathcal{I}$), orientation ($\mathcal{O}$) and motion ($\mathcal{M}$), extracted from visual stimuli, along with associated inter-observer ($\mathcal{IO}$) fixation maps obtained from human eye-tracking data. During training, the feature maps

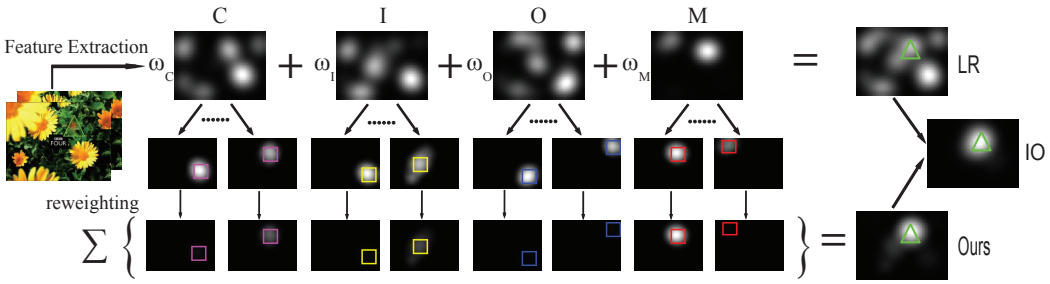

Figure 1. Location-dependent weighting: in the case of Linear Regression (LR), every pixel in a feature map receives the same weight ($\omega_{\mathcal{C}}, \omega_{\mathcal{I}}, \omega_{\mathcal{O}}, \omega_{\mathcal{M}}$). In contrast, our algorithm decomposes each map into up to $\mathcal{N}$ blobs (see ..... markings) and weights the contribution of each blob in a location-dependent manner according to the fixation bank. Our final output is a weighted sum of all blobs. Green triangle indicates the peak location in the human $\mathcal{IO}$ map.

are decomposed into local feature patterns and used jointly with the observer fixation maps to create a fixation bank $\mathscr{B}$, which associates patterns of feature responses around a blob with probability of human fixation at that blob's location. The fixation bank is comprised of both positive and negative pattern configurations (exemplars of feature patterns corresponding to high and low fixation probabilities, respectively). The fixation bank is used to measure how likely a future test feature pattern is to attract human fixations. For a test frame, we decompose its feature maps $\mathcal{CIOM}$ into blobs, extract local feature patterns around each blob, match these patterns agaist the fixation bank by leveraging group lasso [31], and determine weights of blobs based on the ratio of reconstruction errors from the positive and negative exemplars in the fixation bank. The final saliency map for that frame is the sum of weighted blobs. Our method bears resemblance with Rudoy et al [24] work; however, we don't use any high-level features while our fixation bank built from primitive features still incorporates some spatial and featural contexts. Our method differs from work in [14, 15, 33, 16] in the way that we process in units of blobs, which automatically take neighboring pixel continuity into account.

Our method is applicable to fixation prediction on both static images and dynamic video frames. Widely used image saliency data sets include Bruce and Tsotsos[3], Kootstra et al[18] and Judd et al[15], which contain 120, 101 and 1003 images respectively. Data set in[15] is the largest static data set available to date. Since our method is data-driven, while all static data sets are relatively too small and do not cover a rich viariety of local feature patterns, in the paper, we only experiment on dynamic video data sets, which are much larger than existing static data sets. Specifically, we use 23,670 video frames to build the fixation bank.

We have several-fold contributions: (1) we propose the concept of fixation bank to model human fixations. (2) we introduce a location-dependent weighting strategy, which outperforms previous location-independent uniform weighting. (3) we model saliency in units of blobs instead of pixels, which is more semantically sensible. (4) we extensively test and validate our new approach on standard data sets.

2. Methods

Our method contains two major steps: fixation bank construction and fixation candidates reweighting. Fixation bank is constructed on training data, maintaining the association between local feature patterns and probabilities of human fixations, and is used to reweight fixation candidates on testing frames. The bank contains 4 dictionaries, $\mathcal{D}_{\mathcal{F}}$, $\mathcal{F} \in \{\mathcal{CIOM}\}$. Each $\mathcal{D}_{\mathcal{F}}$ is associated with the corresponding feature channel $\mathcal{F}$, and used to reweight fixation candidates only from $\mathcal{F}$. Dictionary $\mathcal{D}_{\mathcal{F}}$ consists of 2 blocks, i.e. $\mathcal{D}_{\mathcal{F}} = [\mathcal{D}_{\mathcal{F}}^{\mathcal{P}} | \mathcal{D}_{\mathcal{F}}^{\mathcal{N}}]$. The 1st block $\mathcal{D}_{\mathcal{F}}^{\mathcal{P}}$ contains feature patterns contributing to fixations, while the 2nd one $\mathcal{D}_{\mathcal{F}}^{\mathcal{N}}$ contains feature patterns less likely to attract attention. To predict fixation density map of a test frame, we extract $\mathcal{CIOM}$ activation maps, decompose each map into blobs, extract local feature patterns on each blob, and formulate reweighting of blobs from $\mathcal{F}$ as a group lasso problem with $\mathcal{D}_{\mathcal{F}}$ as its design matrix. The master conspicuity map is sum of reweighted fixation candidates from 4 channels.

2.1. Feature Extraction and Inter-Observer ($\mathcal{IO}$) map Generation

Four feature channels are extracted on each frame including color ($\mathcal{C}$), intensity ($\mathcal{I}$), orientation ($\mathcal{O}$), and motion ($\mathcal{M}$). They reflect local center-surround contrasts in each feature dimension. All of these features are known to contribute to bottom-up saliency. $\mathcal{CIO}$ were computed as in [14] and $\mathcal{M}$ as in [12]. To generate the inter-observer map for each frame, we convolve an isotropic Gaussian kernel with standard deviation σ, set approximately to the size of the human fovea, at fixation positions of human subjects on that frame, and then linearly combine these fixation maps. This combined fixation map is termed as inter-observer ($\mathcal{IO}$) map (the 2nd row in Fig.3 shows some $\mathcal{IO}$

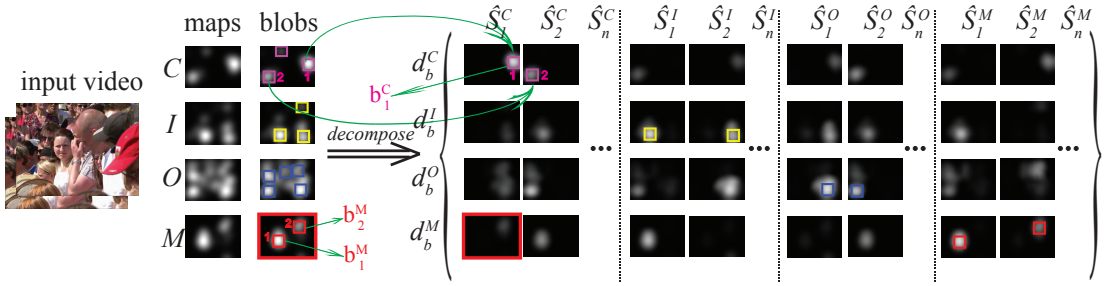

Figure 2. Feature map Decomposition: 4 maps in the 1st column are extracted feature maps of a sample frame, the 2nd column shows fitted Gaussian blobs. In this case, by running decomposition on each Gaussian blob, we end up 13 (3+3+5+2) decomposed feature maps, 8 of which are shown in the bracket. Each block in the bracket shows decomposed feature maps of Gaussian blobs from the same channel $\mathcal{F}$, $\mathcal{F} \in \{\mathcal{CIOM}\}$, e.g. $\hat{S}_1^{\mathcal{C}}$ and $\hat{S}_2^{\mathcal{C}}$ in the 1st block are decomposed feature maps of blobs 1 and 2 from channel $\mathcal{C}$. Each row shows decayed maps $d_b^{\mathcal{F}}$ of channel $\mathcal{F}$ on different blobs. Take motion channel $\mathcal{M}$ decaying w.r.t. blob $b_1^{\mathcal{C}}$ from channel C as an example: the original $\mathcal{M}$ and its decayed map $d_b^{\mathcal{M}}$ are highlighted by red rectangular boxes. $d_b^{\mathcal{M}}$ is computed following Eq.1. Since $b_1^{\mathcal{M}}$ is spatially further to reference blob $b_1^{\mathcal{C}}$ than $b_2^{\mathcal{M}}$, resulting that $b_1^{\mathcal{M}}$is weakened much more than $b_2^{\mathcal{M}}$, which is seen from $d_b^{\mathcal{M}}$that location around $b_1^{\mathcal{M}}$is blacked-out.

maps). $\mathcal{IO}$ maps are computed on the raw-sized frames, afterwards they are subsampled to 20×15 pixels in our experiments and used as groud-truth.

2.2. Fixation Bank Construction

We build a fixation bank $\mathscr{B}$ consisting of 4 feature-channel associated dictionaries, $\mathscr{B} = \{\mathcal{D}_{\mathcal{C}}, \mathcal{D}_{\mathcal{I}}, \mathcal{D}_{\mathcal{O}}, \mathcal{D}_{\mathcal{M}}\}$. Each dictionary contains positive and negative feature patterns, which capture gaze allocation probabilities. Feature patterns in the dictionary $\mathcal{D}_{\mathcal{F}}$ are decomposed feature maps of fixation candidates from feature channel $\mathcal{F}$. We have to emphasize that: although dictionary $\mathcal{D}_{\mathcal{F}}$ is associated with feature-channel $\mathcal{F}$, and only used to reweight fixation candidates from $\mathcal{F}$, its feature patterns do contain information from all 4 channels. Technical details of fixation candidates and their corresponding decomposed feature maps are given in the following.

2.2.1 Fixation Candidates Generation

We treat each ℓ_1-normalized feature map $\mathcal{F}$ as a gaze probability distribution $\mathcal{P}_{\mathcal{F}}$. By sampling sufficient random points from $\mathcal{P}_{\mathcal{F}}$ and clustering them using mean-shift, we obtain $\mathcal{K}_{\mathcal{F}}$ clusters. Each cluster is approximated by a Gaussian blob with cluster center as mean and points covariance matrix as variance. Finally each blob on feature map $\mathcal{F}$ is treated as a fixation candidate. In the following text, Gaussian blobs and fixation candidates will be used interchangeably. Gaussian blobs extraction is shown in Fig.1. The 1st row shows 4 feature maps $\mathcal{CIOM}$, and the 2nd row shows fitted blobs from each map. Decomposition of a feature map into blobs makes location-dependent weighting possible:

By running linear regression between feature and $\mathcal{IO}$ maps, i.e., $(\mathcal{C}, \mathcal{I}, \mathcal{O}, \mathcal{M}, \mathbf{1}) \cdot (\omega_{\mathcal{C}}, \omega_{\mathcal{I}}, \omega_{\mathcal{O}}, \omega_{\mathcal{M}}, \omega_{\mathbf{1}})^T = \mathcal{IO}$ ($\omega_{\mathbf{1}}$accounts for bias), we obtain one weight $\omega_{\mathcal{F}}$ for one feature channel $\mathcal{F}$, i.e. all pixels within $\mathcal{F}$ receive the same weight. This location-invariant weighting mechanism is shown to be suboptimal in case of object search in real scenes [27]. In our work, rather than explicitly building the object-specific contextual models, we adopt a similar location-dependent weighting mechanism by learning weights for each Gaussian blob.

2.2.2 Feature Map Decomposition

After extracting fixation candidates $b_k, k \in \{1, 2, ..., \mathcal{K}_{\mathcal{F}}\}$ from feature map $\mathcal{F}, \mathcal{F} \in \{\mathcal{CIOM}\}$, we decompose raw feature maps $\mathcal{CIOM}$ according to each blob b on $\mathcal{F}$: let $\hat{S}_b^{\mathcal{F}}$ be the decomposed feature map of blob b, which is a concatenation of 4 decayed feature maps, i.e., $\hat{S}_b^{\mathcal{F}} = [d_b^{\mathcal{C}}\ d_b^{\mathcal{I}}\ d_b^{\mathcal{O}}\ d_b^{\mathcal{M}}]^T$ with

$$d_b^f = \sum_{k=1}^{\mathcal{K}_f} \omega_{bk} \cdot g_k,\ f \in \{\mathcal{CIOM}\} \tag{1}$$

d_b^f is the a decayed map of channel f, w.r.t. reference blob b from channel $\mathcal{F}$, which is sum of $\mathcal{K}_f$ decayed blobs from f. In Eq.1, g_k is the k^{th} blob from channel f and $\omega_{bk} = exp\{-\frac{1}{2\sigma^2}((x_b - x_k)^2 + (y_b - y_k)^2)\}$ is its weight w.r.t. reference blob b, where (x_k, y_k) and (x_b, y_b) are image plane coordinates of target blob g_k and reference blob b respectively, and σ controls decaying rate. Weight ω_{bk} is reversely proportional to the spatial proximity of two blob centers. The decomposed feature map $\hat{S}_b^{\mathcal{F}}$ of reference blob b is termed as signature of b, which describes local feature pattern around b, and is used to construct fixation bank during training and reweight blob b during testing. At the end, the raw feature maps $\mathcal{CIOM}$ are decomposed into $\sum_{\mathcal{F} \in \{\mathcal{CIOM}\}} \mathcal{K}_{\mathcal{F}}$ signatures $\hat{S}_b^{\mathcal{F}}$, each of which associates

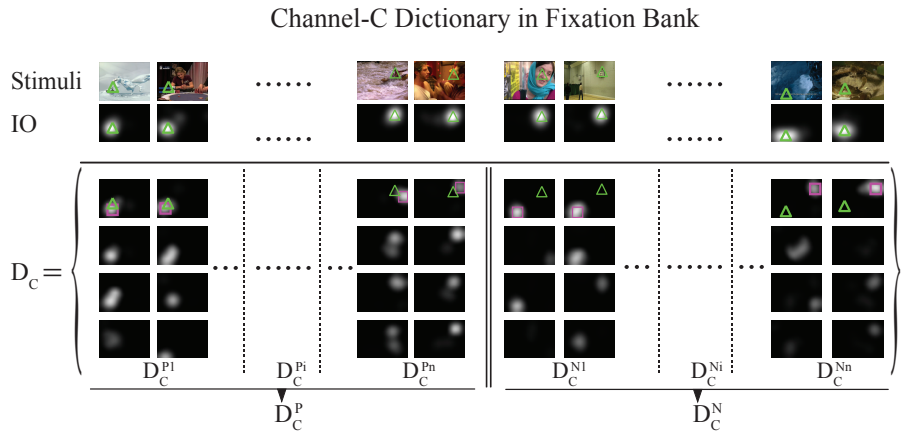

Figure 3. Fixation Bank: This figure shows the constructed channel-$\mathcal{C}$ dictionary $\mathcal{D}_\mathcal{C}$ from bank $\mathscr{B}$, each column in $\mathcal{D}_\mathcal{C}$ is a decomposed feature map $\hat{S}_b^\mathcal{C}$ of some blob b from channel $\mathcal{C}$. $\mathcal{D}_\mathcal{C}$ contains positive and negative blocks $\mathcal{D}_\mathcal{C}^P$ and $\mathcal{D}_\mathcal{C}^N$, which are further divided into sub-blocks based on spatial positions of Gaussian blobs. In the figure, magenta rectangles and green triangles indicate peak locations of blobs and $\mathcal{IO}$ maps. As seen, two peaks are close for exemplars from the positive block $\mathcal{D}_\mathcal{C}^P$ while they are far apart in $\mathcal{D}_\mathcal{C}^N$.

with blob b from channel $\mathcal{F}$. All the decomposed feature maps share the same $\mathcal{IO}$ map of that frame. An example of feature map decomposition is shown in Fig.2.

Feature decomposition decomposes complicated, less likely repeatable raw feature patterns into basic and more repeatable elementary feature patterns. This makes the matching of two raw feature patterns insensitive to their extra non-overlapping parts. Feature decomposition bears some similarity with pyramid match kernel, which is robust to background cluttering [6]. For two frames f_1 and f_2 with different raw feature maps but some common decomposed feature maps, we may infer possible fixation locations on f_2 based on known fixations on f_1.

2.2.3 Blocked Dictionary Construction

First we define the peak of blob b to be the peak of its decomposed feature map $\hat{S}_b^\mathcal{F}$. The blocked dictionary is built under the assumption that only when the peak of a decomposed feature map is close to ($\leq \xi$) the peak of $\mathcal{IO}$ this decomposed feature map is a positive configural pattern that attracts attention, otherwise it is a negative exemplar.

Blocked dictionary has two blocks, keeping positive configural feature patterns and negative ones respectively. By designing such blocked dictionary, we could formulate Gaussian blobs reweighting as a two-class classification problem by group lasso. A classification assumption is proposed by Wright [30] in face recognition that if sufficient training samples are available for a particular class, then it is possible to sparsely represent any test sample by only using training samples from the same class. We generalize this assumption to our setting of Gaussian blobs reweighting. If the decomposed feature map of a blob candidate is reconstructed well from the positive block, then we believe that blob attracts attention, and it will be assigned a high weight; otherwise, it is suppressed in the final conspicuity map.

One dictionary $\mathcal{D}_\mathcal{F}$ is built for each feature channel $\mathcal{F}$ ($\mathcal{F} \in \{\mathcal{CIOM}\}$), without loss of generality, we take channel-$\mathcal{C}$-associated dictionary $\mathcal{D}_\mathcal{C}$ construction as an example.

For a training frame, suppose there are $\mathcal{K}_\mathcal{C}$ Gaussian blobs $b_i, i = \{1, 2, ...\mathcal{K}_\mathcal{C}\}$ on channel $\mathcal{C}$, let be $\hat{S}_{b_i}$ the decomposed feature map of b_i. If the peak location $\mathcal{P}_{b_i}$ of b_i is within some distance ξ to the peak location $\mathcal{P}_\mathcal{IO}$ of $\mathcal{IO}$ map, i.e., $\|\mathcal{P}_{b_i} - \mathcal{P}_\mathcal{IO}\|_2 \leq \xi$, then $\hat{S}_{b_i}$ is treated as a positive exemplar and assigned to the 1st block $\mathcal{D}_\mathcal{C}^P$; while when $\|\mathcal{P}_{b_i} - \mathcal{P}_\mathcal{IO}\|_2 \geq \tau, \tau > 0 \wedge \tau > \xi$, it is a negative exemplar and assigned to the 2nd block $\mathcal{D}_\mathcal{C}^N$. This assignment process iterates over all training frames. Finally, we build a channel-$\mathcal{C}$-associated dictionary $\mathcal{D}_\mathcal{C}$, $\mathcal{D}_\mathcal{C} = [\mathcal{D}_\mathcal{C}^P \mid \mathcal{D}_\mathcal{C}^N]$. τ is a threshold much larger than ξ, in our case, $\tau = 5$, $\xi = 2$. In practice, ξ, τ are determined such that we have approximately equal number of negative and positive exemplars in $\mathcal{D}_\mathcal{C}$. Blocked dictionary construction for the other three feature channels follows the same way, and at last we get four blocked dictionaries,$\mathcal{D}_\mathcal{C}, \mathcal{D}_\mathcal{I}, \mathcal{D}_\mathcal{O}, \mathcal{D}_\mathcal{M}$, constituting a fixation bank $\mathscr{B}$.

Each blocked dictionary $\mathcal{D}_\mathcal{F}$ has two big blocks, and we further divide training exemplars in each big block into smaller blocks by their peak locations. In our case, the image plane is cut into $M \times N$ non-overlapping cells, each with size $s \times s$. When the peak location of an exemplar falls into cell i, then it is assigned to sub-block i, $i = \{1, 2, ..., M \times N\}$. Finally, each blocked dictionary $\mathcal{D}_\mathcal{F}$ has $2 \times M \times N$ blocks, half of them belonging to the 1st positive block and the left belonging to the 2nd negative block, i.e., $\mathcal{D}_\mathcal{F} = [\mathcal{D}_\mathcal{F}^{P_1} \mid \mathcal{D}_\mathcal{F}^{P_2} \cdots \mid \cdots \mid \mathcal{D}_\mathcal{F}^{P_{M\times N}} \parallel \mathcal{D}_\mathcal{F}^{N_1} \mid \mathcal{D}_\mathcal{F}^{N_2} \cdots \mid \cdots \mid \mathcal{D}_\mathcal{F}^{N_{M\times N}}]$. Bottom row in Fig.3 shows the structure of $\mathcal{D}_c$.

2.3. Gaussian Blob Reweighting

For a test frame, after extracting decomposed feature maps of Gaussian blobs, we formulate reweighting of each blob as a group lasso problem. The final gaze density map

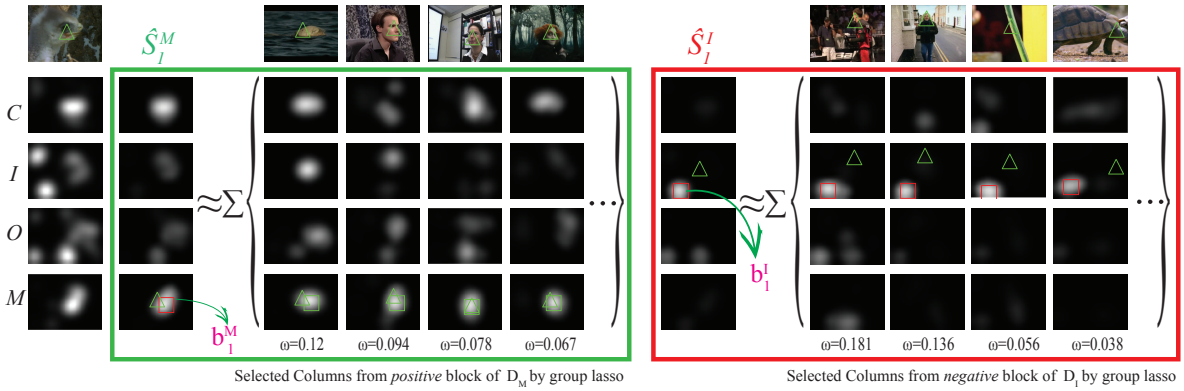

Figure 4. Blob Reweighting by Group lasso. We show reweighting of two blobs $b_1^{\mathcal{M}}$ and $b_1^{\mathcal{I}}$ from channel $\mathcal{M}$ and $\mathcal{I}$ respectively. The green triangles on frames indicate peak locations of $\mathcal{ICO}$. The first column shows a test frame and its extracted feature maps $\mathcal{CICOM}$. In the green box, the 1st column is the decomposed feature map $\hat{S}_1^{\mathcal{M}}$ of blob $b_1^{\mathcal{M}}$ from motion channel $\mathcal{M}$, by reconstructing it from blocked motion dictionary $\mathcal{D}_{\mathcal{M}}$using group sparsity constraints, i.e. by solving problem 2, we get a couple of auto-selected exemplars with varying weights, and we list 4 examplars from the positive block. As we see, the selected positive exemplars have similar featural configurations to $\hat{S}_1^{\mathcal{M}}$, and experiments showed that most of the selected exemplars came from the postive block, making blob $b_1^{\mathcal{M}}$ have a high weight when evaluating 3. Similarly, in the red box, we show how blob $b_1^{\mathcal{I}}$ from intensity channel $\mathcal{I}$ is suppressed. Now $\hat{S}_1^{\mathcal{I}}$ is reconstructed from intensity dictionary $\mathcal{D}_I$ by solving 2, and we chose to list 4 selected negative exemplars. Since it's reconstructed well by negative exemplars, it's assigned a small weight and suppressed therefore. Intuitively blob $b_1^{\mathcal{I}}$ is far from $\mathcal{ICO}$, not attention-attracting, and should be suppressed, while $b_1^{\mathcal{M}}$ overlaps with $\mathcal{ICO}$, catches attentions and should be enhanced.

is a weighted sum of all blobs.

A paradigm for multi-class classification using sparsity-inducing linear regression is introduced in [26]. The ℓ_1 regularized lasso tends to select a single representative sample from a group of correlated samples and does not promote the representation of the test sample in terms of all correlated samples. However decomposed feature maps from the same sub-block (even from different sub-blocks, as will be seen) are correlated and to promote the representation of the query sample with all correlated atoms, we utilize block-norm regularization on the grouped coefficients, specifically, sum-of-ℓ_2-norm of grouped coefficients, and each group corresponds to one sub-block in the dictionary. In practice, we use sparse group lasso [4], which promotes sparsity both at the group level and within the group and favors selection of correlated samples together.

For each Gaussian blob on feature maps of a test frame, to calculate its contributing weight to the final saliency map, we first solve a group lasso problem and then define its weight as a function of reconstruction errors from the positive and negative groups. Given a blob b from channel $\mathcal{F}$ with decomposed feature map $\hat{S}_b^{\mathcal{F}}$, we solve the problem:

$$\min_{\beta}\{\frac{1}{2}\parallel \mathcal{D}_{\mathcal{F}}\cdot\beta-\hat{S}_b^F\parallel_2^2+\lambda_1\sum_{g=1}^{G}L_g\parallel\beta_g\parallel_2+\lambda_2\parallel\beta\parallel_1\} \tag{2}$$

Where β_g is the coefficients of g^{th} group, $\beta = (\beta_1, \beta_2, ..., \beta_G)$, $G = 2 \times M \times N$ is the entire coefficient vector, $L_g = \sqrt{|\beta_g|}$accounts for varying group sizes, and λ_1 and λ_2 are controlling parameters making balance between reconstruction and sparsity. The above problem is a convex problem with non-smooth regularizer, FISTA algorithm proposed by [1] can efficiently solve it. Intuitively, if blob b is the one attracting attention, the non-zero coefficients should mostly come from the the first positive block and its reconstruction error from the 1st block should be lower, on the contrary, a blob less likely to attract attention should be easier to be reconstructed by samples from the 2nd block. We define weight of blob b as the ratio between negative and positive reconstruction errors:

$$\omega_b^{\mathcal{F}} = \varepsilon^N(\hat{S}_b^{\mathcal{F}})/\left(\varepsilon^P(\hat{S}_b^{\mathcal{F}})+\epsilon\right) \tag{3}$$

where $\varepsilon^N(\hat{S}_b^{\mathcal{F}}) = \parallel \mathcal{D}_{\mathcal{F}}^N\cdot\beta^N-\hat{S}_b^{\mathcal{F}}\parallel_2$, $\varepsilon^P(\hat{S}_b^{\mathcal{F}}) = \parallel \mathcal{D}_{\mathcal{F}}^P\cdot\beta^P-\hat{S}_b^{\mathcal{F}}\parallel_2$ and ϵ is a small constant to avoid singularity. β^P and β^N are coefficients of positive and negative groups respectively.

The finally gaze density map $\mathbf{S}_f$ of frame f is a weighted sum of all Gaussian blobs:

$$\mathbf{S}_f = \sum_{\mathcal{F}\in\{\mathcal{CICOM}\}}\sum_{k=1}^{\mathcal{K}_{\mathcal{F}}}\omega_{b_k}^{\mathcal{F}}\cdot g_{b_k}^{\mathcal{F}} \tag{4}$$

Where $g_{b_k}^{\mathcal{F}}$is the k^{th} Gaussian blob b_k from channel $\mathcal{F}$, $\omega_{b_k}^{\mathcal{F}}$ is its weight defined in Eq.3, and $\mathcal{K}_{\mathcal{F}}$ is the number of Gaussian blobs on channel $\mathcal{F}$. In Fig.4, we use a test frame to show how Gaussian blobs get reweighted, making some be enhanced and others be suppressed.

2.4. Data Augmentation and Sharing Atoms

Fixation bank is not translation-invariant yet, since we have limited training frames. Thus, we augmented the blocked dictionaries by shifting each decomposed feature map in 4-directions by at most ζ pixels and also horizontally flipping it, in our case $\zeta = 3$. This makes our algorithm invariant to slight translations. Another problem is that atoms (decomposed feature maps) from different blocks are not necessarily uncorrelated, i.e., some atoms from the positive block are correlated with atoms from the negative block, although they belong to different classes (positive & negative). Intuitively, some local feature patterns are ambiguous, and, sometimes, they contribute to fixation allocation, while in other occasions, they do not draw attention. Sparse group lasso tends to choose these atoms together, making reconstruction errors similar, and in this case the discriminating power of the system is weakened. To improve prediction power, we inspect the correlation between selected atoms from different blocks, if their inner products are greater than $\tau(\tau = 0.95)$, their coefficients are ignored.

3. Experimental Validation

3.1. Data sets

We use two public available dynamic saliency data sets: DIEM [20] and CRCNS [11]. DIEM includes 84 high-resolution videos from different scene types, such as movie trailers, TV clips, sports, etc, and each clip lasts 2 mins on average. It collects eye tracking data from over 250 participants, and each clip has on average over 50 participants. All clips are cropped to a fixed 4/3 width/height ratio from the center, and then resized into resolution 640×480, in order to keep consistent with the clip resolution from CRCNS data set.

The CRCNS data set includes 50 video clips, each lasting from 6 to 90 seconds. All clips had the same resolution 640×480. The videos contained a mix of indoor and outdoor scenes including park scenes, crowds, rooftop bars, TV news, sports, commercials, and video game footage. The eye tracking data was collected from 3 females and 5 males.

3.2. Data Preparation

The frequency of each clip in both data sets is 30Hz, and there is much information redundancy between adjacent frames. Instead of using all frames for training and testing, we sample 1 out of 5 sequentially. We use randomly sampled video clips from the DIEM data set as training data to construct a fixation bank, and test on the remaining DIEM clips. To validate generalization, we test on some randomly chosen CRCNS clips as well. In the end we use 47 clips from DIEM as training data and test on 27/17 clips from DIEM/CRCNS. Concretely, there are in total 23,670 training and 15,793/4,505 test frames. For each frame, extracted feature maps $\mathcal{CIOM}$ and kernel smoothed $\mathcal{IO}$ map are down-sampled to 20×15 pixels.

3.3. Evaluation Metrics

We utilized three universally used saliency metrics: Area Under ROC Curve (AUC), Normalized scanpath salience (NSS) [22] and χ^2 distances. AUC measures the reliability that a saliency model can predict locations of interest. Here we used a shuffled version of it [32], which discounts huge center-biased models. NSS is the response value at eye fixations on the 0-mean-1-variance normalized estimated gaze density map. χ^2 measures distances between two distributions, ℓ_1 normalized $\mathcal{IO}$ map and predicted gaze map. For AUC and NSS scores, the higher the better, while for χ^2 distances, the lower the better.

3.4. Comparison with Linear Regression

Our model assigns location-dependent weights to Gaussian blobs, while linear regression computes one weight for each feature map. In our case, ideally, Gaussian blobs closer to $\mathcal{IO}$ receive higher weights, while those far apart receive weaker weights. Our experimental results verify this. Each test frame has $\sum_{\mathcal{F}\in\{\mathcal{CIOM}\}} \mathcal{K}_{\mathcal{F}}$ Gaussian blobs, where $\mathcal{K}_{\mathcal{F}}$ is the number of blobs on channel-$\mathcal{F}$. We sort these blobs according to their Euclidean distances to $\mathcal{IO}$ of that frame in a ascending way, and record weights of the first 8 blobs and put them into a row of the weight matrix $\boldsymbol{\omega}$, in the end we get a $N \times 8$ matrix $\boldsymbol{\omega}$ with N being the number of test frames. Element $\boldsymbol{\omega}(i, j)$ denotes the weight of the j^{th} closest blob w.r.t. $\mathcal{IO}$ on the i^{th} frame. The mean of weight matrix $\boldsymbol{\omega}$ along columns is a 8D weight vector $\boldsymbol{\varpi}$, with the j^{th} entry being the average weight of the j^{th} closest blobs w.r.t. $\mathcal{IO}$, and $\boldsymbol{\varpi}$ is plotted in the 1st column of Fig.5. As we can see, the mean weights descend from the closest blobs to the 8th closest ones in both data sets, which shows efficacy of our method in reweighting: enhance the attention-attracting blob candidates while suppress attention-irrelevant ones. The middle column of Fig.5 shows number of frames in term of their χ^2 distances to $\mathcal{IO}$ maps. The distributions of χ^2 distances shift toward 0 after applying our reweighting algorithm. The percentages of frames getting smaller χ^2 distances are 70%/68% in DIEM/CRCNS. The 3rd column in Fig.5 shows clip-wise AUC comparison. As observed, our algorithm is almost consistently better than LR. In some clips, LR yields slightly better scores than ours, this is because none of the Gaussian blobs from all 4 channels hit $\mathcal{IO}$, in this case, LR usually yields spreading gaze density maps while $ours$ produces peakier ones. In case of non-hits, shuffled AUC prefers spreading maps to peaky ones.

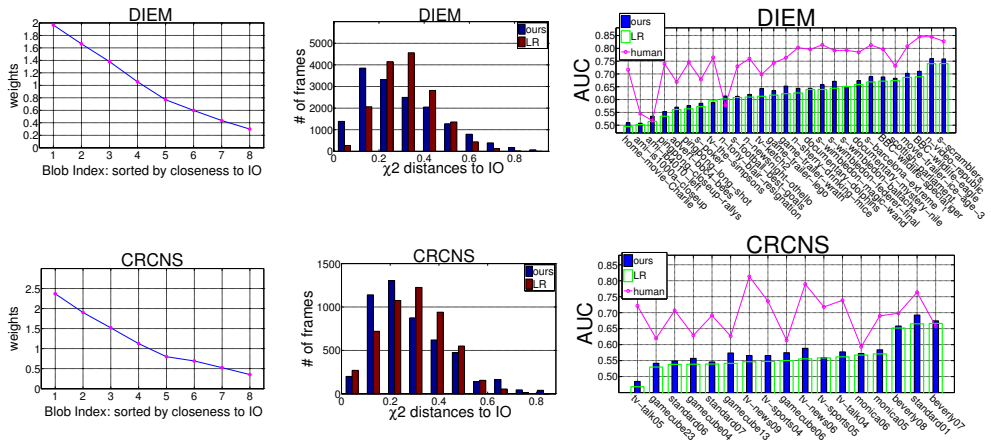

Figure 5. Ours Vs. Linear Regression: the 1st column shows after blob reweighting, closer blobs w.r.t. $\mathcal{IO}$ got higher weights, while far apart ones got smaller weights. The 2nd column shows the distribution of χ^2 distances between predicted conspicuity maps and $\mathcal{IO}$, as seen, our algorithm makes χ^2 distances shift toward 0. The 3rd column shows clip-wise AUC scores. Our algorithm is almost consistently better than LR.

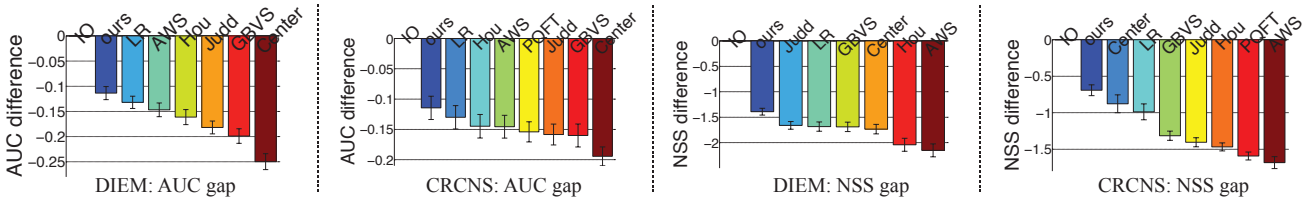

Figure 6. Comparison with other algorithms: our algorithm outperforms others under both metrics, AUC and NSS, in a significant way. See the text for the significance levels under Wilcoxon signed rank test.

3.5. Comparison with other Saliency Models

We compared our algorithm against three static image saliency models, which rank at the top on previous benchmarks [2], including: Adaptive Whitening Saliency model(AWS) [5], Graph-based Visual Saliency(GBVS) [8] and Judd et al [15], and two video saliency models PQFT [7] and Hou and Zhang's model(Hou) [10]. Because of the usual center bias effect in the data set, we also compared to the center-placed Gaussian blob model(center), with sigma of Gaussian blob being $\sigma = 3$.

AWS, GBVS, PQFT, Judd and Hou models are applied on the raw frames with resolution 640×480 under their default parameter settings, and estimated saliency maps are down-sampled into resolution 20×15 for convenience of comparison. On the DIEM data set, $ours$ performs the best, obtaining average AUC/NSS scores 0.62/1.38, and on CRCNS data set, $ours$ outperforms others as well, with average AUC/NSS scores be 0.56/1.29 respectively. We plot AUC and NSS gaps of different algorithms against human performance in Fig.6. The height of each bar is the mean gap of that algorithm from human performance, with error bar showing variance across different clips. The first two plots show AUC gaps under two data sets, as seen, two supervised methods, $ours$ and LR, outperform other unsupervised methods. By running 'Wilcoxon signed rank test' between AUC scores of $ours$ and LR, we get test statistics $W = 38/74$ and p-values $p = 0.0025/0.0059$ on DIEM/CRCNS data sets. The last two plots show NSS gaps. On DIEM data set, running 'Wilcoxon test' between $ours$ and Judd, whose performance is the closest to $ours$, gives $p = 8.8e^{-4}$; and on CRCNS, the significance level between ours and Center is $p = 7.2e^{-4}$. The hypothesis test shows our algorithm outperforms others under both metrics in a significant way.

In Fig.6, we observed that shuffled AUC scores of GBVS are low, indicating GBVS model is center-biased. We noticed as well that AWS's performance under metric AUC on two data sets are among the top, consistent with results reported in Borji and Itti's work [2]. In Fig.7, we visually show saliency results by different algorithms on DIEM data set. As observed, other algorithms tend to produce spreaded gaze density maps while ours generates more peaky results, making them less suitable for fixation prediction in dynamic videos.

4. Conclusion

We proposed a new fixation bank approach to predict attention allocations in dynamic scenes. We built this bank from primitive low-level features, and obtained better performance than other algorithms, which sometimes used more complex features, on two public data sets. This was achieved by leveraging the spatial and featural contextual information implicitly embedded in the bank. Our method is nonparametric, making it less likely to be affected by de-

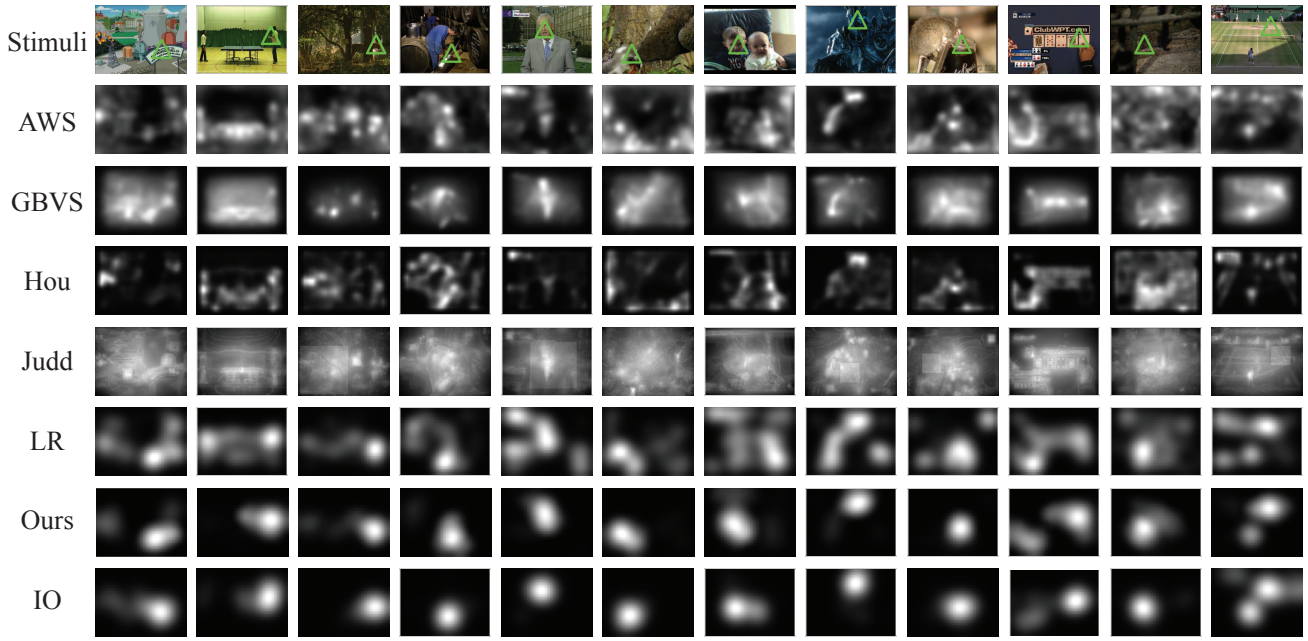

Figure 7. Saliency results on DIEM data set by different algorithms. Frames are chosen from 27 testing clips, with varying scene types, including sports, news, movies trailers, TV documentaries etc. Green triangles on frames show the peak locations of $\mathcal{IO}$. As we see from the results, in dynamic videos, each frame usually has one or two fixation locations, while on traditional static images under free viewing, it's common for more than 2 locations to pop out as attention-catching. Our algorithm automatically reweights each blob, making one or two stand out while suppressing others, and finally generating a peaky saliency map. Other static saliency algorithms or dynamic ones (Hou) tend to produce a spreaded saliency map, making them less suitable for fixation prediction in dynamic videos.

sign parameters. We have shown that our approach outperforms the state of the art and transfers well to a completely different data set.

Acknowledgements: This work was supported by the National Science Foundation (grant numbers CCF-1317433 and CMMI-1235539), the Army Research Office (W911NF-11-1-0046 and W911NF-12-1-0433), and the Office of Naval Research (N00014-13-1-0563). The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

References

[1] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[2] A. Borji, D. N. Sihite, and L. Itti. Quantitative analysis of human-model agreement in visual saliency modeling: A comparative study. *TIP*, 22(1):55–69, 2013.

[3] N. Bruce and J. Tsotsos. Saliency based on information maximization. In *Advances in neural information processing systems*, pages 155–162, 2005.

[4] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*, 2010.

[5] A. Garcia-Diaz, V. Leborán, X. R. Fdez-Vidal, and X. M. Pardo. On the relationship between optical variability, visual saliency, and eye fixations: A computational approach. *J vision*, 12(6):17, 2012.

[6] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, volume 2, pages 1458–1465. IEEE, 2005.

[7] C. Guo, Q. Ma, and L. Zhang. Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform. In *CVPR*, pages 1–8. IEEE, 2008.

[8] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. *NIPS*, 19:545, 2007.

[9] J. M. Henderson, J. R. Brockmole, M. S. Castelhano, and M. Mack. Visual saliency does not account for eye movements during visual search in real-world scenes. *Eye movements: A window on mind and brain*, pages 537–562, 2007.

[10] X. Hou and L. Zhang. Dynamic visual attention: searching for coding length increments. In *NIPS*, volume 5, page 7, 2008.

[11] L. Itti. Crcns-orig video and eye tracking database.

[12] L. Itti, N. Dhavale, and F. Pighin. Realistic avatar eye and head animation using a neurobiological model of visual attention. In *SPIE*, pages 64–78, 2004.

[13] L. Itti and C. Koch. Feature combination strategies for saliency-based visual attention systems. *Journal of Electronic Imaging*, 10(1):161–169, Jan 2001.

[14] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *PAMI*, 20(11):1254–1259, 1998.

[15] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *CVPR*, pages 2106–2113. IEEE, 2009.

[16] W. Kienzle, F. A. Wichmann, B. Schölkopf, and M. O. Franz. A nonparametric approach to bottom-up visual saliency. *NIPS*, 19:689, 2007.

[17] A. R. Koene and Z. Li. Feature-specific interactions in salience from combined feature contrasts: Evidence for a bottom–up saliency map in v1. *Journal of Vision*, 7(7):6, 2007.

[18] G. Kootstra, B. de Boer, and L. R. Schomaker. Predicting eye fixations on complex visual stimuli using local symmetry. *Cognitive computation*, 3(1):223–240, 2011.

[19] L. Li, H. Su, F. Li, and E. Xing. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *NIPS*, pages 1378–1386, 2010.

[20] P. K. Mital, T. J. Smith, R. L. Hill, and J. M. Henderson. Clustering of gaze during dynamic scene viewing is predicted by motion. *Cognitive Computation*, 3(1):5–24, 2011.

[21] H.-C. Nothdurft. Salience from feature contrast: additivity across dimensions. *Vision research*, 40(10):1183–1201, 2000.

[22] R. Peters, A. Iyer, L. Itti, and C. Koch. Components of bottom-up gaze allocation in natural images. *Vision research*, 45(18):2397–2416, 2005.

[23] R. J. Peters and L. Itti. Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention. In *CVPR*, pages 1–8. IEEE, 2007.

[24] D. Rudoy, D. B. Goldman, E. Shechtman, and L. Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In *CVPR*, pages 1147–1154. IEEE, 2013.

[25] S. Sadanand and J. J. Corso. Action bank: A high-level representation of activity in video. In *CVPR*, pages 1234–1241. IEEE, 2012.

[26] S. Shekhar, V. Patel, N. Nasrabadi, and R. Chellappa. Joint sparse representation for robust multimodal biometrics recognition. 2014.

[27] A. Torralba, A. Oliva, M. S. Castelhano, and J. M. Henderson. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological review*, 113(4):766, 2006.

[28] A. M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

[29] B. T. Vincent, R. J. Baddeley, T. Troscianko, and I. D. Gilchrist. Optimal feature integration in visual search. *Journal of Vision*, 9(5):15, 2009.

[30] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *PAMI*, 31(2):210–227, 2009.

[31] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society*, 68(1):49–67, 2006.

[32] L. Zhang, M. Tong, T. Marks, H. Shan, and G. Cottrell. Sun: A bayesian framework for saliency using natural statistics. *Journal of vision*, 8(7):32, 2008.

[33] Q. Zhao and C. Koch. Learning visual saliency by combining feature maps in a nonlinear manner using adaboost. *Journal of Vision*, 12(6):22, 2012.