Factorizing Appearance Using Epitomic Flobject Analysis

Patrick S. Li
University of California, Berkeley
patrickli.2001@gmail.com

Brendan J. Frey
University of Toronto
frey@psi.utoronto.ca

Abstract

Previously, 'flobject analysis' was introduced as a method for using motion or stereo disparity information to train better models of static images. During training, but not during testing, optic flow is used as a cue for factorizing appearance-based image features into those belonging to different flow-defined objects, or flobjects. Here, we describe how the image epitome can be extended to model flobjects and introduce a suitable learning algorithm. Using the CityCars and CityPedestrians datasets, we study the tasks of object classification and localization. Our method performs significantly better than the original LDA-based flobject analysis technique, SIFT-based methods with and without spatial pyramid matching, and gist descriptors.

1. Introduction

Humans excel at classifying static test images, but they benefit hugely by incorporating motion and stereo disparity information during training [10]. In contrast, most artificial vision systems do not follow this approach. This motivated us to ask "Can motion or stereo disparity information be used to train better methods for extracting representations from static images?" The idea is that clustering a flow field often elucidates objects with coherent flow, or 'flobjects'. We used this pop-out effect in a bag-of-SIFT-words LDA model (FLDA) to factorize the appearance features from different objects. Because of noise in the flow and the aperture problem, it was found that it is important to jointly learn to factorize appearance-based features and cluster flow in a probabilistic framework [7].

Here, to model appearance features and link regions with similar flow, we propose using an image epitome [4] instead of the LDA [1] model used in the original flobject analyzer. Each location in a 'flobject epitome' is associated with an appearance feature and a distribution over flobject labels (Fig. 1). The flobject epitome (epitome and label distribution map) is learnt using an unsupervised EM algorithm that iteratively labels patches and flow vectors using the current flobject epitome and then updates the flobject epitome. The resulting flobject epitome can be used without flow to label patches in test images, as shown in Fig. 4.

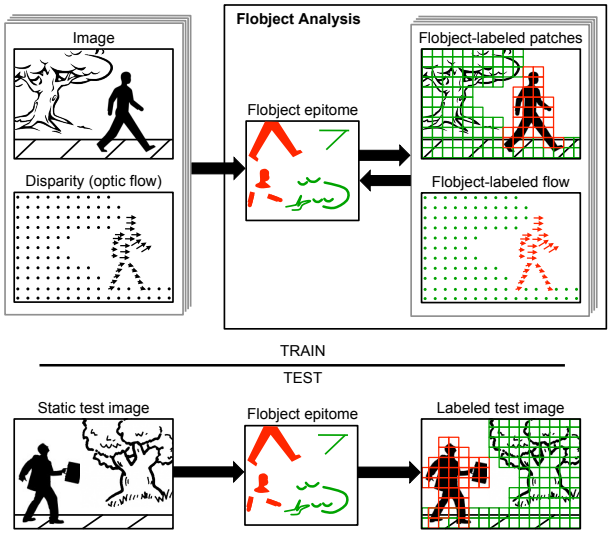

Figure 1: Epitomic flobject analysis takes image-flow field pairs and iteratively estimates a flobject epitome that has a distribution over flobject labels (red and green) at each position, and labels each patch and flow vector. Labels are *not* provided during training.

The previous flobject analyzer (FLDA) made use of a latent Dirichlet allocation (LDA) [1] model to train bags-of-SIFT-words [8] models for objects given pairs of consecutive video frames, similar to the setup in [15]. At test time, FLDA factorizes an image's SIFT features into individual object SIFT histograms. The FLDA descriptors created by concatenating these individual histograms were shown to outperform other state-of-the-art classifiers on the CityCars and CityPedestrians datasets [7].

Despite its good performance, FLDA suffers from two drawbacks. First, FLDA does not account for the spatial coherence of appearance features. This

severely limits the utility of the model. Flagging image locations whose features are associated with each object results in spatially incoherent labelings. As shown below, by accounting for spatial coherence using an epitome, we are able to significantly improve performance on classification and localization. Second, FLDA requires running an expensive Markov Chain Monte Carlo (MCMC) chain to perform inference. We find that a much faster EM algorithm can be used to train flobject epitomes.

2. Related Work

While we focus on the task of object classification, the core idea of training a model using data with motion cues to improve test performance on static images has been explored extensively in the field of image segmentation. The Segmentation According to Natural Examples (SANE) algorithm developed by Ross and Kaebling [11] uses background subtraction to learn to segment static images from video training data. From a high level our approach is also similar to the techniques used by Russel et al. [12] and Lee and Grauman [6] for unsupervised object discovery from a collection of static images. They use NCuts [13] to partition each image into segments, and discover objects by grouping segments that are similar across frames. Grouping is done using LDA in the case of Russel et al., or context-aware clustering, in the case of Lee and Grauman. Analogously, our method partitions images using optical flow and groups partitions using an epitome model. One benefit to using a robust statistical model to handle the noisy flow vectors is the ability of the model to trade off trusting the optical flow vectors with trusting the appearance model. This allows the image partitioning to influence the appearance model and vice versa. Lastly, Sivic et al. [14] discovers objects by identifying co-occurrences of HOG features across images and was an inspiration for both the original FLDA model and this work.

3. Epitomes

There are different ways to account for spatial coherence of appearance features, but one of the simplest and quite effective ways is to use an image epitome [4]. Image epitomes keep track of features that tend to occur adjacently in training images, by representing the features in a spatial feature map. Also, since epitomes can directly model pixel intensities, it is possible to visually interpret the features that are modeled by the epitome.

In the standard epitome model, a training set of N images of size $H \times W$ is used to find a representative $E \times E$ feature map and corresponding variance map, called the image epitome, in such a way that any $D \times D$ patch from the image collection can be well explained by some $D \times D$ patch from the epitome.

The parameters in an epitome consist of the mean η_j and std. dev. ν_j of the appearance feature (*e.g.*, pixel intensity, color value, or texture feature) at coordinates $j = (j_1, j_2)$ in the $E \times E$ epitome. x_i^p is the observed appearance feature for the pixel at coordinates $i = (i_1, i_2)$ in image patch p. For each image patch, the hidden variable $r^p = (r_1^p, r_2^p) \in \{0, \ldots, E-1\}^2$ indicates the coordinates of the top-left pixel in the epitome patch from which it was generated. Given r^p, x_i^p is modeled using a Gaussian distribution:

$$p(x_i^p | r^p, \eta, \nu) = \mathcal{N}(x_i^p | \eta_{r^p+i}, \nu^2_{r^p+i}), \quad (1)$$

where $\eta = \{\eta_j\}$ and $\nu = \{\nu_j\}$ and $r^p + i = (r_1^p + i_1, r_2^p + i_2)$.

Epitomes can be viewed as Gaussian mixture models where the means and variances of different clusters share a subset of their parameters. For example, the 10×10 epitome patch with top-left corner at $(1, 1)$ shares 180 parameters with the patch at $(1, 2)$. So even though images are modeled as unordered bags of independent $D \times D$ patches, spatial relationships are still captured by shared parameters.

4. Flobject Epitomes

A flobject epitome (FE) consists of an epitome that is used to model the appearance of patches drawn from an input image, plus a flobject label distribution map that is the same size as the epitome. Under this model, the patches and flow vectors in an image are generated by first generating a prototype flow vector and its variance for each possible flobject. Then, each patch of pixel intensities and corresponding patch of flow vectors are generated by randomly picking a location in the flobject epitome, generating the patch pixel intensities from the corresponding distributions over pixel intensities prescribed at that location, generating a patch of flobject labels from the corresponding distributions over flobject labels prescribed at that location, and then generating the patch of flow vectors using the corresponding prototype flow vectors.

The FE model extends the epitome model for flobject analysis by introducing a soft partitioning of the epitome into T flobject types using a parameter π. The probability that the epitome feature at coordinate r has flobject label $t \in \{1, \ldots, T\}$ is $\pi_{r,t}$, where $\sum_{t=1}^{T} \pi_{r,t} = 1$. For each image, the flobject label specifies a distribution over the optic flow for pixels belonging to that flobject. Here, we use Gaussian distributions, where $\vec{\mu}_t^n$ and $\vec{\sigma}_t^n$ are the mean and std. dev. of

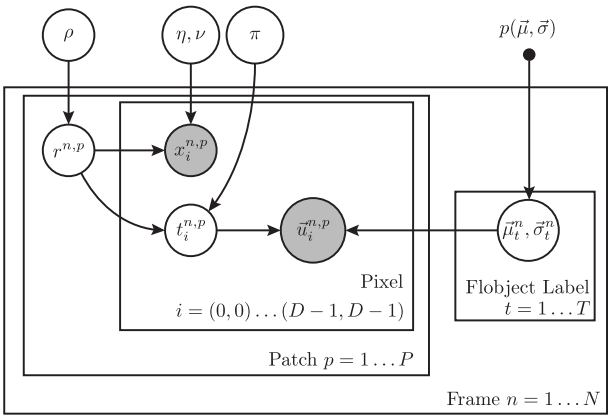

Figure 2: The flobject epitome (FE) graphical model.

the flow vectors for flobject label t in image n. For the p^{th} patch in the n^{th} image, $r^{n,p} \in \{0, \ldots, E-1\}^2$ are the coordinates in the epitome from which the patch is generated. $x_i^{n,p}$ and $\vec{u}_i^{n,p}$ are the appearance feature and flow vector at coordinates i in patch p of image n. $\mathbf{x}^{n,p} = (x_{1,1}^{n,p}, \ldots, x_{D,D}^{n,p})$ is a patch of appearance features and $\vec{\mathbf{u}}^{n,p}$ is a patch of flow vectors.

Since an image patch may lie on boundaries between objects, we allow different pixels within a single patch to belong to different flobjects. $t_i^{n,p}$ is the flobject label of the pixel at coordinates i in patch p of image n and $\mathbf{t}^{n,p}$ denotes the entire patch of flobject labels. The graphical model for the FE model is shown in Fig. 2.

To describe an image, the FE model first generates the mean $\vec{\mu}_t$ and std. dev. $\vec{\sigma}_t$ of the optic flow for each flobject $t \in \{1, \ldots, T\}$ from a uniform prior $p(\vec{\mu}, \vec{\sigma})$. Each patch is generated as follows, where we drop the indices n and p for clarity. First, the epitome coordinates r used to explain the patch are drawn from a multinomial distribution with parameter ρ:

$$p(r|\rho) = \rho_r, \tag{2}$$

where $\sum_r \rho_r = 1$. Given r, a flobject label t is generated for each pixel according to the multinomial distribution π:

$$p(\mathbf{t}|r, \pi) = \prod_{i=(1,1)}^{(D,D)} p(t_i|r, \pi) = \prod_i \pi_{r+i, t_i}. \tag{3}$$

Next, a flow vector $\vec{u}_i$ for each pixel is drawn from a Gaussian with mean $\vec{\mu}_t$ and std. dev. $\vec{\sigma}_t$:

$$p(\vec{\mathbf{u}}|\mathbf{t}, \vec{\mu}, \vec{\sigma}) = \prod_i p(\vec{u}_i|t_i, \vec{\mu}, \vec{\sigma}) = \prod_i \mathcal{N}(\vec{u}_i|\vec{\mu}_{t_i}, \vec{\sigma}_{t_i}^2). \tag{4}$$

As before, the pixel feature (*e.g.*, grey level) x_i is generated by sampling from a Gaussian with mean η_{r+i} and std. dev. ν_{r+i}:

$$p(\mathbf{x}|r, \eta, \nu) = \prod_i \mathcal{N}(x_i|\eta_{r+i}, \nu_{r+i}^2). \tag{5}$$

The probability of $r, \mathbf{t}, \vec{\mathbf{u}}, \mathbf{x}$ for a patch p is

$$p(\mathbf{x}^p, \vec{\mathbf{u}}^p, \mathbf{t}^p, r^p|\vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho) = p(r^p|\rho)p(\mathbf{t}^p|r^p, \pi)p(\vec{\mathbf{u}}^p|\mathbf{t}^p, \vec{\mu}, \vec{\sigma})p(\mathbf{x}^p|r^p, \eta, \nu). \tag{6}$$

To write the joint distribution for all patches in an image, we assume that patches are generated independently, given the image-specific flow model, $\vec{\mu}, \vec{\sigma}$. This assumption avoids the combinatorics of accounting for patch overlap during inference.

We use $\{\mathbf{x}^{n,p}, \vec{\mathbf{u}}^{n,p}, \mathbf{t}^{n,p}, r^{n,p}\}$ to denote the collection of epitome variables for all patches, p, in each image, n, in an entire training set of images. $\{\vec{\mu}^n, \vec{\sigma}^n\}$ denotes the corresponding flow models for each image. The joint probability of the flow models and epitome variables is

$$\begin{aligned} &p(\{\mathbf{x}^{n,p}, \vec{\mathbf{u}}^{n,p}, \mathbf{t}^{n,p}, r^{n,p}\}, \{\vec{\mu}^n, \vec{\sigma}^n\}|\eta, \nu, \pi, \rho) \\ &= \prod_{n=1}^{N} \Big(p(\vec{\mu}^n, \vec{\sigma}^n) \prod_{p=1}^{P} \Big(p(r^{n,p}|\rho)p(\mathbf{t}^{n,p}|r^{n,p}, \pi) \cdot \\ &\quad p(\vec{\mathbf{u}}^{n,p}|\mathbf{t}^{n,p}, \vec{\mu}^n, \vec{\sigma}^n)p(\mathbf{x}^{n,p}|r^{n,p}, \eta, \nu) \Big) \Big). \end{aligned} \tag{7}$$

Note that whereas the parameters ρ, π, η and ν are shared across images, the flow model $\vec{\mu}^n, \vec{\sigma}^n$ is different for different images. This enables the FE model to account for the same object having different motion patterns in different images.

A distinct advantage of the FE model over FLDA is the ability to model *per-pixel* flow vectors instead of modeling the single average flow vector of an entire patch. This feature is important, as many patches, such as the ones lying on the boundaries of moving objects, contains multiple flobject labels. Accounting for per-pixel flow vectors enables the accurate partitioning of the epitome along the boundaries of such objects.

4.1. Marginalizing over flobject labels

Probabilistic inference involves summing over hidden variables, including the flobject labels. One concern might be that the number of configurations of $\mathbf{t}$ in a patch is T^{D^2} and that this will lead to intractable summations. However, we designed the flobject epitome described here so that the two terms involving t's in the above expressions factorize across pixels. For a single patch, we have

$$p(\mathbf{t}|r, \pi)p(\vec{\mathbf{u}}|\mathbf{t}, \vec{\mu}, \vec{\sigma}) = \prod_i p(t_i|r, \pi)p(\vec{u}_i|t_i, \vec{\mu}, \vec{\sigma}). \tag{8}$$

Summations of t's can be moved inside the product over pixels. In particular, we can reduce the model of r, $\mathbf{t}$, $\vec{\mathbf{u}}$ and $\mathbf{x}$ to one where all t's are marginalized out:

$$p(\mathbf{x}, \vec{\mathbf{u}}, r|\vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho) = \sum_{\mathbf{t}} p(\mathbf{x}, \vec{\mathbf{u}}, \mathbf{t}, r|\vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho)$$
$$= p(r|\rho)p(\mathbf{x}|r, \eta, \nu) \prod_i p(\vec{u}_i|r, \pi, \vec{\mu}, \vec{\sigma}), \quad (9)$$

where

$$p(\vec{u}_i|r, \pi, \vec{\mu}, \vec{\sigma}) = \sum_{t_i} p(t_i|r, \pi)p(\vec{u}_i|t_i, \vec{\mu}, \vec{\sigma}). \quad (10)$$

This expression can be used for all patches and for all images to obtain a model with the t's marginalized out.

4.2. Inferring flobject labels in the presence of flow

It is useful to infer the posterior probability of each flobject type at each input pixel, given the image and flow field. This involves summing over r and all t's but one:

$$p(t_i|r, \mathbf{x}, \vec{\mathbf{u}}, \vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho)$$
$$\propto \sum_r \sum_{\mathbf{t}\setminus t_i} p(\mathbf{x}, \vec{\mathbf{u}}, \mathbf{t}, r|\vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho), \quad (11)$$
$$= \sum_r \Big(p(r|\rho)p(\mathbf{x}|r, \eta, \nu)p(t_i|r, \pi)p(\vec{u}_i|t_i, \vec{\mu}, \vec{\sigma})$$
$$\cdot \prod_{i' \neq i} p(\vec{u}_{i'}|r, \pi, \vec{\mu}, \vec{\sigma}) \Big). \quad (12)$$

It is also useful to infer the label given r:

$$p(t_i|r, \mathbf{x}, \vec{\mathbf{u}}, \vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho) \propto p(\mathbf{x}|r, \eta, \nu)$$
$$\cdot p(t_i|r, \pi)p(\vec{u}_i|t_i, \vec{\mu}, \vec{\sigma}) \prod_{i' \neq i} p(\vec{u}_{i'}|r, \pi, \vec{\mu}, \vec{\sigma}). \quad (13)$$

4.3. Inference for static images

The FE model can be used to analyze a static image, *i.e.* when the flow vectors are not observed. This is achieved by analytically marginalizing over the flow flow vectors $\vec{\mathbf{u}}$ and the flow model $\vec{\mu}, \vec{\sigma}$, which gives

$$p(\{\mathbf{x}^p, \mathbf{t}^p, r^p\}|\eta, \nu, \pi, \rho) = \prod_{p=1}^{P} p(\mathbf{x}^p, \mathbf{t}^p, r^p|\eta, \nu, \pi, \rho), \quad (14)$$

where

$$p(\mathbf{x}^p, \mathbf{t}^p, r^p|\eta, \nu, \pi, \rho) = p(r^p|\rho)p(\mathbf{t}^p|r^p, \pi)p(\mathbf{x}^p|r^p, \eta, \nu).$$

This expression can be used to infer the label of each pixel in each patch of a static test image:

$$p(t_i|\mathbf{x}, \eta, \nu, \pi, \rho) \propto \sum_r p(r|\rho)p(\mathbf{x}|r, \eta, \nu)p(t_i|r, \pi). \quad (15)$$

Thus, the prediction for the flobject label is obtained by matching the patch to the epitome (summation over r) and looking up the corresponding distribution over the flobject label.

5. Learning

To learn a flobject epitome from an observed set of patches of appearance $\{\mathbf{x}^{n,p}\}$ and flow $\{\vec{\mathbf{u}}^{n,p}\}$, we use an expectation maximization (EM) algorithm that infers point estimates of the flobject epitome ρ, π, η, ν and the flow models $\{\vec{\mu}^n, \vec{\sigma}^n\}$, while summing over the other variables $\{r^{n,p}, \mathbf{t}^{n,p}\}$.

For the E-step, given the current point estimates of the flobject epitome parameters ρ, π, η, ν and the flow models $\{\vec{\mu}^n, \vec{\sigma}^n\}$, for each observed patch, we compute the posterior probability over epitome positions given the patch pixel intensities and flows:

$$\gamma_j^{n,p} = p(r^{n,p} = j|\mathbf{x}^{n,p}, \vec{\mathbf{u}}^{n,p}, \vec{\mu}^n, \vec{\sigma}^n, \eta, \nu, \pi, \rho)$$
$$= \frac{p(\mathbf{x}^{n,p}, \vec{\mathbf{u}}^{n,p}, r^{n,p} = j|\vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho)}{\sum_{j'} p(\mathbf{x}^{n,p}, \vec{\mathbf{u}}^{n,p}, r^{n,p} = j'|\vec{\mu}, \vec{\sigma}, \eta, \nu, \pi, \rho)} \quad (16)$$

$\gamma_j^{n,p}$ can be thought of as partial assignments of image patches to epitome positions, analogous to the responsibilities computed during EM inference for Gaussian mixture models, and will be used in the M step to update point estimates for other unknowns.

For each possible epitome position, we also compute the pos.terior distribution over flobject labels for each pixel within the patch,

$$\alpha_{j,i,t}^{n,p} = p(t_i^{n,p}{=}t|r^{n,p}{=}j, \mathbf{x}^{n,p}, \vec{\mathbf{u}}^{n,p}, \vec{\mu}^n, \vec{\sigma}^n, \eta, \nu, \pi, \rho), \quad (17)$$

as described in detail above.

In the maximization step, we use the posterior probabilities calculated in the E-step to obtain point estimates of the flobject epitome parameters π, η, ν and the image-specific flow model parameters $\vec{\mu}^n, \vec{\sigma}^n$. The point estimates are obtained by taking the derivative of the expected complete likelihood where the expectation is taken w.r.t. the posterior distribution as computed in the E-Step. Using $\mathcal{R}_r = \{(r_1 - D + 1, r_2 - D + 1), \ldots, (r_1, r_2)\}$ to denote the epitome coordinates of pixels in the patch up and to the left of r, the update equations are as follows:

$$\pi_{r,t} = \frac{\sum_{n=1}^{N} \sum_{j \in \mathcal{R}_r} \sum_{p=1}^{P} \gamma_j^{n,p} \alpha_{r,r-j,t}^{n,p}}{\sum_{t'=1}^{T} \sum_{n=1}^{N} \sum_{j \in \mathcal{R}_r} \sum_{p=1}^{P} \gamma_j^{n,p} \alpha_{r,r-j,t'}^{n,p}}, \quad (18)$$

$$\eta_r = \frac{\sum_{n=1}^{N} \sum_{j \in \mathcal{R}_r} \sum_{p=1}^{P} \gamma_j^{n,p} x_{r-j}^{n,p}}{\sum_{n=1}^{N} \sum_{j \in \mathcal{R}_r} \sum_{p=1}^{P} \gamma_j^{n,p}}, \quad (19)$$

(a) (b)

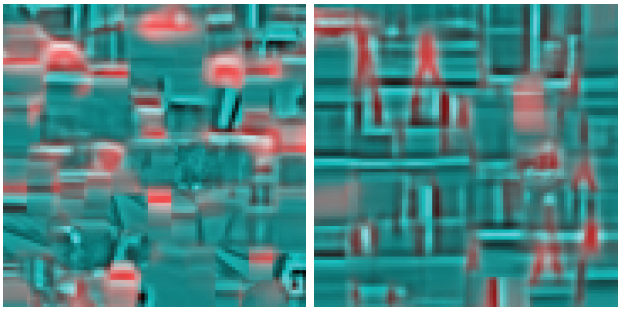

Figure 3: Visualization of the flobject epitomes learnt from unlabeled images and flow fields in the CityCars (a) and CityPedestrians (b) datasets. Pixels are shaded from green to red proportionally to the probabilities of flobject types 1 and 2 in each case.

$$\nu_r^2 = \frac{\sum_{n=1}^{N}\sum_{j\in\mathcal{R}_r}\sum_{p=1}^{P}\gamma_j^{n,p}(x_{r-j}^{n,p}-\eta_r)^2}{\sum_{n=1}^{N}\sum_{j\in\mathcal{R}_r}\sum_{p=1}^{P}\gamma_j^{n,p}}, \quad (20)$$

$$\vec{\mu}_t^n = \frac{\sum_{j\in\mathcal{R}_r}\sum_{p=1}^{P}\gamma_j^{n,p}\alpha_{r,r-j,t}^{n,p}\vec{u}_{r-j}^{n,p}}{\sum_{j\in\mathcal{R}_r}\sum_{p=1}^{P}\gamma_j^{n,p}\alpha_{r,r-j,t}^{n,p}}, \quad (21)$$

$$(\vec{\sigma}_t^n)^2 = \frac{\sum_{j\in\mathcal{R}_r}\sum_{p=1}^{P}\gamma_j^{n,p}\alpha_{r,r-j,t}^{n,p}(\vec{u}_{r-j}^{n,p}-\vec{\mu}_t^n)^2}{\sum_{j\in\mathcal{R}_r}\sum_{p=1}^{P}\gamma_j^{n,p}\alpha_{r,r-j,t}^{n,p}}. \quad (22)$$

6. Experiments

We trained flobject epitomes using the CityCars and CityPedestrians datasets originally described in [7]. The CityCars dataset consists of 160 frames of side views of cars driving in an urban setting, accompanied by flow fields, 155 static images containing cars (positives) and 338 images without cars (negatives). The CityPedestrians dataset consists of 160 frames of side views of pedestrians walking in an urban setting, accompanied by flow fields, 469 static images containing pedestrians (positives) and 338 images without pedestrians (negatives). In both datasets, each frame consists of a 216×384 grayscale image, and a $2\times216\times384$ array containing per-pixel flow vectors computed using a standard optical flow algorithm [2]. The images were contrast normalized by subtracting from each pixel the mean of the 7×7 window of its neighboring pixels.

For the flobject epitomes, we used an epitome size of $E=90$, a patch size of $D=7$, and $T=2$ flobjects. The flobject epitomes were initialized using the standard epitome learning algorithm with 60 iterations of EM. The full FE model was then obtained by applying 60 iterations of EM using the images and flow fields.

6.1. Epitome visualization

Fig. 3 shows the flobject epitomes learnt from the CityCars and CityPedestrian datasets, where, for each position in the epitomes, the distribution over the flobject label is color-coded. Pixel (r,c) is shaded from green to red proportionally to the mixing proportions of flobject types $t\in\{1,2\}$ as specified by $\pi_{r,c,t}$. The distinction between the background regions and the car and pedestrian regions in the epitome are captured by the soft partitioning of the epitomes. Because each pixel in the epitome has its own mixing proportions over the flobject types, the EM algorithm is free to allocate as much or as little of the epitome to modeling each flobject type as necessary.

For both datasets, we see that more of the epitome is allocated towards modeling the background than the cars and pedestrians due to the higher variability of backgrounds. A more rigid model, such as one that trained a separate epitome for each object type, would not be flexible in this way. Notice also that different sizes and orientations of relevant features are captured in the epitomes, *e.g.* both large and small wheels and large and small legs.

6.2. Inferring flobject labels from static images

We used the flobject epitome with the flow component integrated out to analyze static test images as described above. Fig. 4a shows the posterior over flobject labels computed for several test images from the CityCars dataset. Pixels are shaded from green to red proportionally to the probability of them being classified as flobject type 1 or 2.

Keeping in mind that the only input during training is images and flow fields and that the only input during testing is static images, these results indicate that the flobject epitome has successfully learned appearance patterns that separate classes of motion, in this case cars versus streets, buildings, trees, *etc.* In addition to successfully labeling cars with a wide range in scale without using a multi-resolution search, the flobject epitome can correctly label multiple cars appearing in the same image. False positives include curved structures and long horizontal ribbons of dark above light, which correspond to car roofs and shadows cast by cars, respectively. Also, pedestrians are sometimes detected, possibly because some moving pedestrians were included in the training data.

We also examined labelings produced by the CityPedestrians flobject epitome, which are shown in Fig. 4b. Considering the wide variation in pose, articulation, reflectance and scale of pedestrians, the flobject epitome is quite successful at labeling them. The legs are particularly well-labeled, whereas the torsos are less

(a) (b)

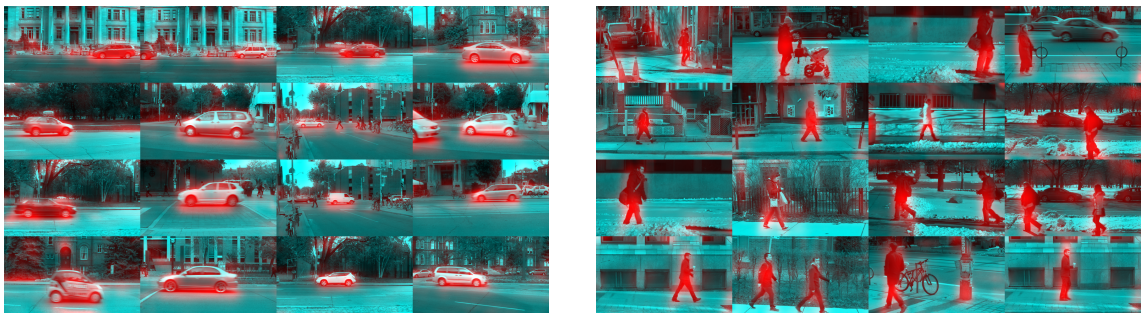

Figure 4: Static test images were colored using the flobject label posteriors obtained from a flobject epitome trained on (a) CityCars and (b) CityPedestrians data that included images and flow, but not labels. Pixels are shaded from green to red proportionally to the probability of flobject types 1 and 2.

well-labeled. Notable false positives include vertical structures such as parking posts, trees and pylons, but these detections are significantly weaker than those of the pedestrians.

In the next two sections, we use the posterior probabilities of flobject labels for object classification and localization.

6.3. Image classification

Here, we study the use of flobject epitomes for classification using the CityCars and CityPedestrians datasets, which were shown to be difficult for classification because the backgrounds are quite similar in all images [7].

To classify static images, we computed HOG [3] descriptors from image regions whose pixels exceeded a threshold applied to the flobject label posterior probabilities. HOG features were computed for every 16×16 patch in a 6×6 grid and each was mapped to a codeword ranging from 1 to V using a codebook that was obtained using k-means clustering applied to training data. To determine which flobject type each HOG patch is associated with, the posterior flobject label probabilities for all pixels in the HOG patch were averaged together and compared to a threshold τ. We created one normalized histogram of HOG codewords for each flobject type, which we refer to as a flobject descriptor. Since we set $T = 2$ for both datasets, there are two flobject descriptors for each image. The best one can be selected using cross validation, but here we report results for both descriptors so they can be compared.

For both the CityCars and CityPedestrians dataset, half the static images are reserved for supervised training for binary classification, and the other half is reserved for testing. As in [7], we ensure that the same car or pedestrian is never present in both the training and test sets. Training and testing was repeated 20 times using the bootstrap to estimate confidence intervals. We ensure an equal number of positive and negative testing images and thus random guessing achieves 50%. We used $\tau = 0.35$ for the CityCars dataset, and $\tau = 0.4$ for the CityPedestrians dataset.

		Dataset	
		CityCars	CityPedestrians
Descriptor	HOG	65% (5%)	55% (1%)
	SPHOG	65% (7%)	53% (1%)
	Gist	70% (5%)	51% (3%)
	LDA	64% (5%)	55% (1%)
	FLDA	82% (4%)	55% (1%)
	FE Type 1	89% (2%)	50% (3%)
	FE Type 2	**97% (1%)**	**69% (2%)**

Table 1: Comparison of classification accuracy on the CityCars and CityPedestrians datasets. Std. dev. computed using the bootstrap.

Descriptor	Classification Accuracy
HOG	68% (2%)
FE Type 2	77% (2%)

Table 2: Classification performance on manually labeled bounding boxes from CityPedestrians.

Table 1 shows the classification results using the two different flobject descriptors as input to an L1 nearest neighbor classifier. We compare our classification results with the original flobject analysis (FLDA) method [7], a bag-of-words classifier with and without a spatial pyramid [5], and a Gist-based [9] classifier.

For both datasets, our method achieves a significantly higher classification rate than other methods. For both CityCars and CityPedestrians, flobject label 1 corresponds to the background while flobject label 2 corresponds to cars or pedestrians. Therefore, it

is perhaps not surprising that the type 2 descriptors achieve higher accuracy, since those descriptors were constructed using HOG features extracted from car and pedestrian regions. However, it is interesting the for CityCars, the type 1 descriptor, comprised from background HOG features, still achieves a better accuracy than other methods. Interestingly, the FE descriptors work well for both datasets, whereas FLDA performed similarly to other methods on the CityPedestrians dataset.

One point we emphasize is that the flow epitome model is trained on *whole* images with motion cues, and we classify whole static images. This is in contrast to sliding window techniques which require bounding box labels at training time. However, if provided with bounding boxes at training time, it is meaningful to ask whether the FE descriptors can still improve upon classification rates. Table 2 compares classification accuracies of HOG and FE descriptors extracted from hand labeled bounding boxes from a subset of the CityPedestrians dataset (430 positive and 430 negative). We see that the additional bounding box labels improve the performance of both descriptors but the FE descriptors still perform better than the HOG descriptors (77% vs. 68%). These results make us hopeful that FE descriptors can be combined with sliding window techniques for object detection applications.

6.4. Object localization

Here, we demonstrate that our method can also be used to localize objects by generating crude object-based segmentations. State of the art methods are optimized to use a sliding window approach to consider every possible scale, aspect ratio and location of a bounding box. Our goal here is not to compete with those methods, but to demonstrate that without explicitly optimizing our method for this task, it nonetheless provides quite sensible localization results. Our approach makes use of the flobject label posteriors discussed above and shown in Fig. 4, but we use a graph labeling method to detect regions with regularized boundaries. We represent an image using a pixel grid $(\mathbb{V}, \mathbb{E})$ and an edge between every pixel and its four immediate neighbours. The goal is to assign to each node an integer labeling, $l_i \in \{1, \ldots, T\}$, that minimizes the sum of unary and pairwise costs, $C(\boldsymbol{l}) = \alpha \sum_{i \in \mathbb{V}} C(l_i) + (1 - \alpha) \sum_{\{(i,i') \in \mathbb{E}} C(l_i, l_{i'})$. We use the unary costs, $C(l_i)$, to persuade the segmentation to heed the object type type posteriors: $C(l_i = t) = \beta_t(1 - p(t_i|\mathbf{x})$, where β_t is a flobject type-specific weighting and $\sum_{t=1}^{T} \beta_t = 1$. The pair-wise costs, $C(l_i, l_{i'})$, are used to persuade the segmentation to follow the intensity contours in the image: $C(l_i, l_{i'}) = e^{-\theta(x_i - x_{i'})^2}$ if $l_i \neq l_{i'}$ and 0 otherwise. $C(l_i, l_{i'})$ can be interpreted as the likelihood that pixels x_i and $x_{i'}$ are of the same object type. If they are of the same type, then there is a cost associated with labeling them differently. θ controls the soft threshold at which pixels are considered to be different object types. Large θ indicates that two pixels are of the same type only if they have exactly the same pixel intensities. A small θ indicates that two pixels are of the same type if they have similar pixel intensities. $\alpha \in [0, 1]$ is the weighting that controls the influence from the intensity contours relative to the type posteriors.

Fig. 5 shows the localization results for test images from the CityCars dataset ($\boldsymbol{\beta} = [0.6 \quad 0.4], \alpha = 0.03, \theta = 0.01$) and CityPedestrians dataset ($\boldsymbol{\beta} = [0.68 \quad 0.32], \alpha = 0.003, \theta = 0.01$). For the most part, the segmentations hug the desired objects, and errors are mostly accounted for by the types of errors in the flobject label posteriors discussed above. Also shown in Fig. 5 are the localization results obtained using the same procedure applied to the original flobject analysis (FLDA) method [7]. Because that method does not account for spatial coherence, it does not produce coherent segmentations. These results demonstrate that the use of an epitome significantly helps our method produce spatially coherent analyses. For reference, we also show results obtained by normalized cuts, which is not expected to perform well because it is not trained to identify objects.

7. Conclusions

We introduced a way of performing flobject analysis using an epitome that is accompanied by a epitome-sized distribution over object labels and image-specific flow models. The flobject epitome (FE) model has distinct advantages over the original flobject analysis model, including that it accounts for spatial coherence in appearance features, it is easily interpretable, and its learning algorithm does not require computationally expensive MCMC techniques for inference. Results on the CityCars and CityPedestrians datasets demonstrate that our method performs better than other techniques on the task of image classification, and that our method can be used to localize objects in images without applying a sliding window approach. While we showed that the FE model can perform well on a small number of object classes, the ultimate goal of flobject analysis is to operate at the scale of many object classes. We are currently exploring techniques that would enable scaling the model to many object classes. This is not trivial because of the problem of the combinatorics involved in unsupervised learning of many object classes and corresponding subclasses corresponding to object parts.

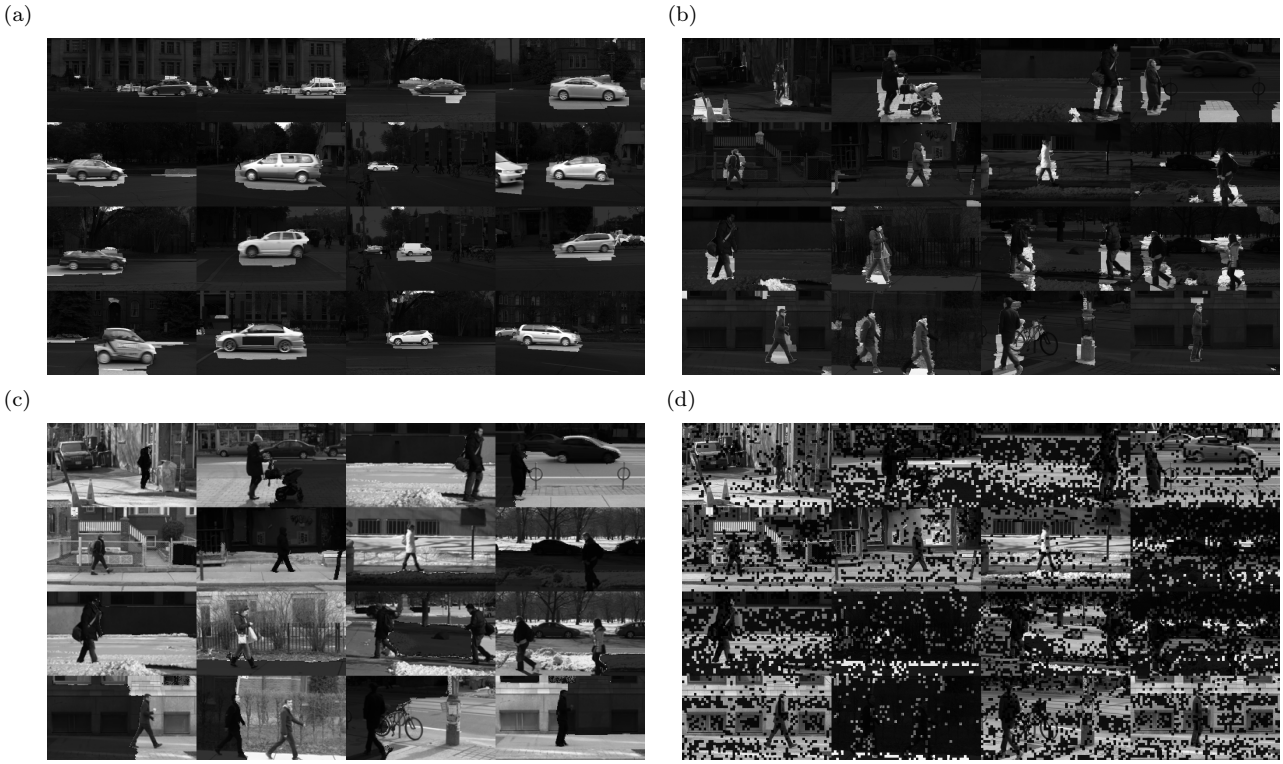

Figure 5: Object localization results for the flobject epitome applied to the (a) CityCars and (b) CityPedestrians datasets. For comparison, segmentations obtained using (c) normalized cuts and (d) the original flobject analysis (FLDA) method are shown. Unlike our method, FLDA does not account for spatial coherence.

8. Acknowledgements

The authors thank Inmar Givoni, Rob Fergus, Nebojsa Jojic, Yann LeCun and Jitendra Malik for helpful discussions. Funding to BJF was provided by the Canadian Institute for Advanced Research and the Natural Sciences and Engineering Research Council of Canada.

References

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 2003. 1

[2] A. Bruhn, J. Weickert, and C. Schnörr. Lucas and Kanade meets Horn and Schunck: Combining local and global optic flow methods. *IJCV*, 61, 2005. 5

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 6

[4] N. Jojic, B. J. Frey, and A. Kannan. Epitomic analysis of appearance and shape. In *ICCV*, 2003. 1, 2

[5] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 6

[6] Y. Lee and K. Grauman. Collect-cut: Segmentation with top-down cues discovered in multi-object images. In *CVPR*, 2010. 2

[7] P. S. Li, I. E. Givoni, and B. J. Frey. Learning better image representations using 'flobject analysis'. In *CVPR*, 2011. 1, 5, 6, 7

[8] D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999. 1

[9] A. Oliva and A. Torralba. Building the Gist of a Scene: The Role of Global Image Features in Recognition. *Progress in Brain Research*, 155, 2006. 6

[10] Y. Ostrovsky, E. Meyers, S. Ganesh, U. Mathur, and P. Sinha. Visual parsing after recovery from blindness. *Psychological Science*, 20, 2009. 1

[11] M. Ross and L. Kaelbling. Segmentation according to natural examples: Learning static segmentation from motion segmentation. *Pattern Analysis and Machine Intelligence*, 2009. 2

[12] B. Russell, W. Freeman, A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006. 2

[13] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAML*, 2000. 2

[14] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering objects and their location in images. In *ICCV*, 2005. 2

[15] G. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *ECCV*, 2010. 1