In Proceedings of the European Conference on Computer Vision (ECCV), 2012.

Active Frame Selection for Label Propagation in Videos

Sudheendra Vijayanarasimhan and Kristen Grauman

University of Texas at Austin
{svnaras,grauman}@cs.utexas.edu

Abstract. Manually segmenting and labeling objects in video sequences is quite tedious, yet such annotations are valuable for learning-based approaches to object and activity recognition. While automatic label propagation can help, existing methods simply propagate annotations from arbitrarily selected frames (e.g., the first one) and so may fail to best leverage the human effort invested. We define an *active frame selection problem*: select k frames for manual labeling, such that automatic pixel-level label propagation can proceed with minimal expected error. We propose a solution that directly ties a joint frame selection criterion to the predicted errors of a flow-based random field propagation model. It selects the set of k frames that together minimize the total mislabeling risk over the entire sequence. We derive an efficient dynamic programming solution to optimize the criterion. Further, we show how to automatically determine how many total frames k should be labeled in order to minimize the total manual effort spent labeling and correcting propagation errors. We demonstrate our method's clear advantages over several baselines, saving hours of human effort per video.

1 Introduction

Semantic segmentation of objects in video sequences is important for many high-level applications, such as recognizing human actions, medical imaging, and automated vehicle driving. Gathering useful labeled data appears key for methods to learn to parse videos, but it requires considerable manual effort. In particular, labeling the boundaries of all objects of interest in each frame is tedious and time-consuming. The cost can be mitigated by exploiting interactive segmentation techniques [24, 15, 1, 3] or region tracking and segmentation methods [16, 9]. Researchers have also developed methods to propagate manual annotations across video frames using interfaces with interpolation tools [23, 26] or inference in space-time graphical models [14, 6, 2, 18, 8]. Typically a user annotates some frame (e.g., the first one), then invokes the propagation engine.

While semi-automatic methods are promising, existing techniques have two main limitations. First, they assume that the provided labeled frame(s) are already fixed, and focus only on how to optimize the propagation across the remaining unlabeled frames. However, there is no guarantee an arbitrarily selected frame (or even a human-selected frame) provides sufficient information to optimally propagate to the rest. Second, they assume some fixed number of initial frames, or else that a human labeler will watch the algorithm's intermediate outputs and decide when a new label is necessary to get the method back on track. However, this neglects the fact that there is a direct trade-off between the number of frames initially labeled and the amount of erroneously propagated labels someone will need to fix afterwards—and that trade-off is video dependent.

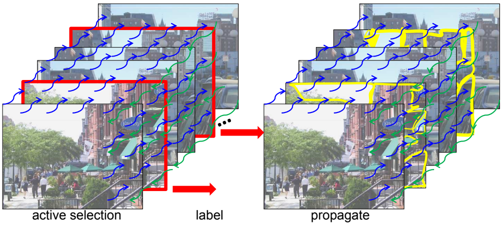


Fig. 1. Goal: actively select k video frames for a human to label, so as to ensure minimal expected error across the entire sequence after automatic propagation. Best viewed in color.

More importantly, requiring a human in the loop to catch propagated errors precludes the possibility of "farming" each frame-level annotation job to multiple people working in parallel, which would be desirable for large-scale annotation efforts.

We instead propose to *actively* select frames for label propagation. The goal is to leverage the required human effort more purposefully, by allowing the propagation algorithm's expected errors to automatically guide which frames are presented to a human for manual labeling.[1] The k most useful frames are jointly chosen according to the expected label error, were they to be propagated via a dense flow-based random field model. Specifically, we compute the predicted mislabeling rate for every frame j should frame i be labeled and propagated to it, based on the expected optical flow error and model uncertainty. We then formulate the best k-selection as an optimization problem to minimize total propagation error, and provide an efficient dynamic programming algorithm to solve it in time polynomial in the number of total frames. After obtaining the selected annotations, we propagate the labels sequentially with that same model. We further show how to optimize over the number of frames that need to be selected.

In this way, our method reduces total manual effort—both by keeping the number of selected frames low, and by ensuring that after propagation minimal human fixing is required. Moreover, by reducing video annotation to k independent image labeling tasks, it has the advantage that one may elect to have them all labeled in parallel (e.g., on Mechanical Turk, if desired). The propagation to unlabeled frames is completely automatic and done offline, so no further user intervention is required.

While our work shares *active learning*'s high-level motivation to minimize human involvement, the active frame selection problem we define is distinct. Traditional active selection methods aim to choose useful instances for category labeling, such that a classifier's uncertainty on unseen instances will be reduced (e.g., [21, 11]). In contrast, the active video frame selection problem aims to *jointly* select those frames in light of their *known temporal ordering* such that the *expected propagation errors on the current sequence* will be minimized. Furthermore, the fact that the k-selection jointly influences many frames in either direction in time means that a naive approach—i.e., one that selects representative keyframes, or one that looks only forward in time to detect abrupt changes—would not meet our goal. Rather, we need to model the "trackability" as part of the selection criterion.

To our knowledge, we are the first to define the active video frame selection problem, where the system determines which subset of frames require labeling. The pro-

[1] Throughout we assume a dense labeling, where the annotator marks the pixel-level boundaries of all objects present in the frame.

posed approach is a novel solution to maximize annotator effort in this important, practical setting. We demonstrate its advantages on challenging videos, as compared to a method that uniformly samples frames for labeling and a clustering-based keyframe selection technique. Our results indicate that active frame selection is crucial to most efficiently use human time for video annotation.

2 Related Work

Interactive segmentation techniques help a user extract objects from videos [1, 24, 15, 3] or groups of related images [4]. Such methods offer novel interfaces to indicate foreground objects in a space-time volume [24, 3], to propagate an initial foreground region while the user corrects any mistakes along the way [1, 15], or to intelligently recommend where a user should scribble [4]. In contrast to our problem, these methods attempt binary labelings and, more importantly, assume a user is closely involved throughout to refine the segmentation at each step. Our goal is to *guide* the user to the frames that most require attention.

Researchers are also developing **novel video annotation tools** amenable to online data collection [26, 23]. LabelMe Video allows users to draw polygons around objects and select a start and end frame; interpolation transfers the polygons to other frames. The crowd-sourcing study in [23] asks a worker to draw a bounding box every T frames, and then interpolates the object path efficiently. Both methods assume the object's motion is either static or uniform during interpolation. As such, our approach can naturally enhance such tools, removing the burden on a user to have insight on which frames are usable for propagation.

Video object segmentation takes an unsupervised approach [9, 16, 10, 19]. Graph-based clustering [10], tracking [9], and random field models [16, 19] have all been explored. Optionally, when labeled frames are available, such methods can perform label transfer using tracks [9] or dense correspondences [12]. A well-known challenge in tracking is "drift", where small errors accumulate over time. Our approach counters this pitfall by requesting more labeled frames where flow errors appear to accumulate, and by using an appearance model for uniform regions with few keypoints.

A few recent methods directly address **label propagation in video** using probabilistic models [14, 6, 2, 18]. The methods typically assume the label field in the first (and/or last) frame of the sequence is given, and then automatically track through the remaining frames based on the objects' color and motion properties.

Active learning methods consider how to select useful instances to refine a classifier, and in particular "batch-mode" selection methods have been explored for training object classifiers [21, 11]. We also want to reduce manual intervention, but our setting differs significantly: our objective is to minimize propagation error rather than build a classifier, and the selection criterion must account for both the information overlap between selected frames as well as the likelihood of successful flow-based transfer to all unlabeled frames. Very recent work considers ways to actively train an "object vs. background" classifier for a given video, iteratively requesting a bounding box or superpixel label in a selected uncertain frame [22, 8]. Like our approach, these methods aim to efficiently use annotator effort for video labeling. However, whereas we jointly solve for a labeling of all objects in the video and explicitly model the “trackability” for active selection, the previous techniques handle a single object of interest independently and base selection on traditional measures of classifier uncertainty. Furthermore, both prior methods assume an annotator remains in the loop for each sequential request following a classifier update, whereas our method computes the set of frames requiring annotation at once. This makes it uniquely amenable to annotators working in parallel (e.g., for crowdsourcing), and in principle enables non-myopic selection.

Finally, **keyframe selection** finds *representative* frames using clustering (e.g., [25]) or by maximizing the dissimilarity between keyframes [7, 13]. While intended for visual summarization—not active annotation—they serve as a natural baseline; we find they underperform our approach, due to their failure to quantify how well labels can be transferred to the unselected frames.

3 Approach

Our goal is to annotate all objects in a video with minimal manual effort. To achieve this, our method first selects a set of informative frames for human labeling, and then propagates those labels to the rest of frames using optical flow and appearance-based models.

We first define the propagation algorithm (Section 3.1), since by design our selection criterion is closely aligned with it. Then in Section 3.2, we define the optimization problem for selection that minimizes the total predicted error on all frames, and finally derive a dynamic programming algorithm to efficiently solve it.

3.1 Video Label Propagation

Let $F = \{f_1, f_2, ..., f_N\}$ be the sequence of N frames from a video that need to be annotated, such that each pixel will be assigned one of $\mathcal{C}$ object labels. Given k frames $S = \{f_{n_1}, \dots, f_{n_k}\}$, with corresponding labels $\{L_{n_1}, \dots, L_{n_k}\}$, where $S \subset F$, we propagate their labels to the rest of the video. Each L_{n_i} is a matrix of labels having the same dimensions as the image frame (height by width) indexed by the 2D pixel coordinates $\boldsymbol{p}$, and each $L_{n_i}(\boldsymbol{p}) \in \{1, \dots, \mathcal{C}\}$. Let (l_t, r_t) be the indices of the closest labeled frames before and after frame t, respectively (“left” and “right” of t).[2] We assume that given labels (L_{l_t}, L_{r_t}), the rest of the frames do not affect the labels of frame t.

In this section, we devise two methods for propagation: a basic flow-based approach and an enhanced variant that uses the flow model within a Markov Random Field. The simpler flow-based model is the core that ties to the selection procedure, while the MRF strengthens it with motion-based data terms and usual smoothness constraints. We test both in experiments.

Pixel Flow Propagation Method The basic propagation method uses dense optical flow to track every pixel in f_t in both the forward and backward directions until it reaches the closest labeled frames on either side. We estimate the expected propagation

[2] Frame indices (l_t, r_t) might not exist if the number of labeled frames is < 2, or if $i < n_1$ or $i > n_k$. For clarity we omit such cases, as they do not affect the method description.

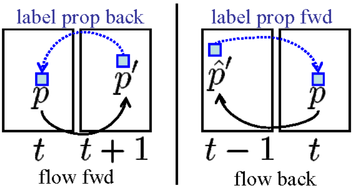


Fig. 2. Relationship between flow and label transfer.

error as the pixel is being tracked, and choose its label from either l_t or r_t, whichever has the smaller error. By tracking the flow in both temporal directions, we account for the fact that object motion in a video can cause one or the other to be more reliable; for example, if an object is moving away from the camera, the earlier frames offer higher resolution on the object and are more reliably propagated, whereas if it is approaching the camera, the opposite is true.

Let $\boldsymbol{w}$ denote a flow field indexed by pixel positions that returns the 2D flow vector at a given point. Given the forward flow field from frame t to $t+1$, $\boldsymbol{w}_t$, and the backward field from t to $t-1$, $\hat{\boldsymbol{w}}_t$, each pixel position $\boldsymbol{p}$ in frame t can be tracked to the next and previous frames:

$$\begin{aligned}\boldsymbol{p}' &= \boldsymbol{p} + \boldsymbol{w}_t(\boldsymbol{p}),\\ \hat{\boldsymbol{p}}' &= \boldsymbol{p} + \hat{\boldsymbol{w}}_t(\boldsymbol{p}).\end{aligned} \tag{1}$$

Defining the expected propagation error. Even with a good dense flow algorithm, inevitably errors occur due to boundaries, occlusions, and when pixels change in appearance, or enter/leave the frame. Thus, we explicitly model the probability that a pixel is mistracked. In the following, we define this propagation error for a later frame $t+j$ *back* to t, i.e., using the forward flow from $\boldsymbol{p}$ to $\boldsymbol{p}'$ (see Figure 2). All terms are analogously defined for propagating from a prior frame $t-j$.

The probability that pixel $\boldsymbol{p}$ in frame t will be mislabeled if we were to obtain its label from frame $t+1$ is:

$$\mathrm{P}(\boldsymbol{p},\ t+1,\ t) = 1 - \exp(-d(\boldsymbol{p},t)), \quad \text{where} \tag{2}$$

$$d(\boldsymbol{p},t) = \beta\left(d_{app}(\boldsymbol{p},t) + d_{mot}(\boldsymbol{p},t) + d_{occ}(\boldsymbol{p},t) + d_{out}(\boldsymbol{p},t)\right),$$

and β is a scaling parameter. $\mathrm{P}(\boldsymbol{p},t-1,t)$ is defined analogously using $\hat{\boldsymbol{w}}$, $\hat{\boldsymbol{p}}'$, and $t-1$.

The component distances reflect the expected forms of tracking error. Specifically,

$$d_{app}(\boldsymbol{p},t) = \|f_t(\boldsymbol{p}) - f_{t+1}(\boldsymbol{p}')\| \tag{3}$$

computes the color difference, and the flow differences are

$$d_{mot}(\boldsymbol{p},t) = \|\boldsymbol{w}_t(\boldsymbol{p}) - \boldsymbol{w}_{t+1}(\boldsymbol{p}')\|. \tag{4}$$

The latter helps identify pixels that drift across object boundaries, thus having the motion of two different objects. We detect occlusions using the consistency of the forward and backward flow:

$$d_{occ}(\boldsymbol{p},t) = \frac{\|\boldsymbol{w}_t(\boldsymbol{p}) + \hat{\boldsymbol{w}}_{t+1}(\boldsymbol{p})\|}{\|\boldsymbol{w}_t(\boldsymbol{p})\| + \|\hat{\boldsymbol{w}}_{t+1}(\boldsymbol{p})\|}. \tag{5}$$

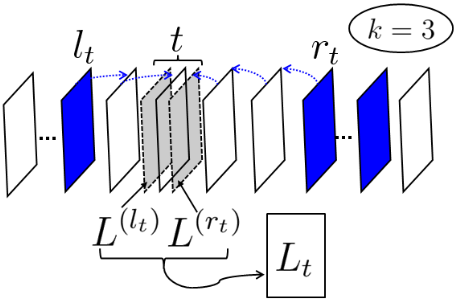


Fig. 3. Schematic for the propagation process to label frame t. Frames l_t and r_t denote t's closest labeled frames before ("left") and after ("right"). Dark blue frames denote the $k = 3$ selected frames. Dotted arrows denote optical flow or MRF-based propagation from adjacent frames, which propagate labels to t from either direction to generate label matrices $L_t^{(l_t)}$ and $L_t^{(r_t)}$ (shown in gray). They combine to form L_t, the final label estimate for t.

Essentially, d_{occ} reflects that if a pixel is not occluded, we expect the two flows to be opposite in direction, making the numerator close to 0. Finally, we set $d_{out}(\boldsymbol{p}, t) = 0$ if $\boldsymbol{p}'$ is within the frame, and a constant R if it has left. Large values for any $d(\cdot)$ term indicates a pixel may have been wrongly mapped to a different object, and hence is likely to cause a propagation error. Note that each term contributes equally to the distance since large values for any one is likely to cause an error.

When there is more than one frame between labeled frame r_t and current frame t, we must predict errors accumulated over successive frames. Defining the error recursively, we have:

$$\begin{aligned} \mathrm{P}(\boldsymbol{p},\ t+j,\ t) = {} & \mathrm{P}(\boldsymbol{p},\ t+j-1,\ t) \\ & + (1-\mathrm{P}(\boldsymbol{p}, t+j-1, t))\mathrm{P}(\boldsymbol{p}, t+j, t+j-1), \end{aligned} \tag{6}$$

for $j > 1$. In other words, pixel $\boldsymbol{p}$ was either mislabeled along some hop from $t+j-1$ back to t, or else those hops were all correct and the wrong label was propagated from the single hop from adjacent frames $t+j$ and $t+j-1$. We will refer to these mislabeling probabilities again in Section 3.2 to define the selection objective.

Minimization and final label map. Thus, to estimate the final label $L_t(\boldsymbol{p})$ for pixel $\boldsymbol{p}$ in frame t with the flow alone, we first compute $\mathrm{P}(\boldsymbol{p}, l_t, t)$ and $\mathrm{P}(\boldsymbol{p}, r_t, t)$ recursively using Eqn. 6 to obtain the two corresponding label estimates $L^{(l_t)}(\boldsymbol{p})$ and $L^{(r_t)}(\boldsymbol{p})$, and then take the best prediction: $L_t(\boldsymbol{p}) = L^{(j^*)}(\boldsymbol{p})$, where $j^* = \mathrm{argmin}_{j=\{l_t, r_t\}} \mathrm{P}(\boldsymbol{p}, j, t)$. See Figure 3. Tracking in both directions helps avoid mistakes made if propagating only one way. If l_t or r_t does not exist, then the labels are simply obtained from the tracked points in the other labeled frame.

Pixel Flow + MRF Propagation Method The previous section defined both the basic flow-based propagation, and (more crucial to our selection approach) a means to estimate propagation errors. Next we explain an enhanced variant that uses flow tracking within a space-time Markov Random Field (MRF) model. The MRF variant helps us (a) infer label maps that are smooth in space and time, and (b) enhance each pixel's label estimate using object appearance models defined by the labeled frames.

The use of a random field for video segmentation is itself not new (e.g., see [16, 15, 18, 19] for variants); however, our formulation specifically allows for the propagated error predictions that are central to active frame selection, as we will see in the following.

Given labels on the subsequent frame $t+1$, we define a random field for t with nodes at every pixel, and hidden nodes corresponding to their unknown labels. To obtain the backward label assignment[3] for t, we minimize the energy:

$$E(L_t) = \sum_{\boldsymbol{p}} A_p(L_t(\boldsymbol{p})) + T_p(L_t(\boldsymbol{p})) \; + \sum_{\boldsymbol{p},\boldsymbol{q} \in \mathfrak{N}} V_{p,q}\big(L_t(\boldsymbol{p}), L_t(\boldsymbol{q})\big),$$

where A_p is a unary potential based on an appearance model defined by frame r_t, T_p is a unary potential based on transferred labels from $t+1$, and $V_{p,q}$ is the pairwise potential computed over $\mathfrak{N}$, the set of neighboring pixels in a $4-$connected grid. They are defined as follows.

Node potentials. The manual segmentation of frame r_t yields object regions taking on (perhaps a subset of) the $\mathcal{C}$ possible object labels. We use its regions to fit $\mathcal{C}$ Gaussian mixture models, one per label. Let $\mathcal{N}(\mu_c, \Sigma_c)$ denote the c-th label's mixture model, defined over a feature $F(\boldsymbol{p})$ consisting of color and entropy-based features (detailed in Sec. 4). We define:

$$A_p(c) = -\log \mathrm{P}(\; F(\boldsymbol{p}) \mid \mathcal{N}(\mu_c, \Sigma_c)). \tag{7}$$

We expect this color model to primarily help fill in background objects at pixels occluded in the previous frame.

The other node potential reflects the cost of transferring a label for $\boldsymbol{p}$ from the next frame $t+1$. We define:

$$T_p(c) = \begin{cases} d(\boldsymbol{p}, t) & \text{if } c = L_{t+1}(\boldsymbol{p}') \\ U & \text{otherwise,} \end{cases} \tag{8}$$

where $d(\boldsymbol{p}, t)$ is defined in Eqn. 2, and U is a constant. This achieves label smoothness in time, where we account for estimated motion by using $\boldsymbol{p}'$.

Pairwise potential. The edge term is based on both the appearance and motion similarity of neighboring pixels $\boldsymbol{p}$ and $\boldsymbol{q}$ in frame t:

$$V_{p,q}(L_t(\boldsymbol{p}), L_t(\boldsymbol{q})) = \delta(L_t(\boldsymbol{p}) \neq L_t(\boldsymbol{q})) S(\boldsymbol{p}, \boldsymbol{q}), \tag{9}$$

where δ denotes the delta function, and

$$S(\boldsymbol{p}, \boldsymbol{q}) = \exp(-\beta_f \|f_t(\boldsymbol{p}) - f_t(\boldsymbol{q})\|) + \exp(-\beta_w \|\boldsymbol{w}_t(\boldsymbol{p}) - \boldsymbol{w}_t(\boldsymbol{q})\|), \tag{10}$$

with scaling parameters β_f, β_w set as the inverse of the mean values of the corresponding terms over the entire frame. This term penalizes assigning different labels to neighboring pixels with similar color and flow.

Minimization and final label map. The total MRF energy can be efficiently minimized using the algorithm of [5] in order to transfer labels from $t+1$ and obtain L_t.

[3] As above, we describe the label transfer in the backward direction, from r_t to t in order to estimate $L^{(r_t)}$; again, analogous equations apply to map from l_t to t in order to estimate $L^{(l_t)}$.

Since this requires that we have already obtained labels on the subsequent frame $t+1$, we start from the nearest labeled frame to the right, r_t, and transfer labels backward sequentially to t in order to obtain $L^{(r_t)}$. Analogously, we transfer from l_t forwards to t to obtain $L^{(l_t)}$. Finally, to smoothly combine the two label maps, we simply minimize a second MRF energy function using the expected propagation errors. See Figure 3.

3.2 Active Selection of a Set of Frames

With the label propagation and error predictions defined, we now explain the novel active selection optimization problem. Recall that existing methods sample manual labels at fixed intervals [26, 23] or simply annotate some manually chosen frame(s) [9, 14, 3, 6, 2, 18]. The pitfall of such an arbitrary selection is that it ignores correlations between frames that can affect interpolation/propagation reliability, which do *not* necessarily vary uniformly over time. In the following we show how to automate this selection.

Selection Criterion To get a well-segmented video, there are two sources of manual effort cost: (1) the cost of fully *labeling* a frame from scratch, denoted C_ℓ, and (2) the cost of *correcting* errors by the automatic propagation, denoted C_c. Both are in units of time. One can obtain realistic estimates of these constants by observing annotators with the label propagation tool. In our experiments we let $C_\ell = 25$ minutes (based on reports from [9]), and $C_c = 1$ minute, the correction time typically needed to achieve 1% pixel error. Alternatively, one could replace the constants with frame-specific segmentation costs when available, e.g., as predicted with a learned model [20].

We now define an optimization problem for the best set of frames from which to propagate. Our aim is to choose $S^* = \{f_{n_1}, f_{n_2} ..., f_{n_{k^*}}\}$ to minimize the total expected effort:

$$S^*, k^* = \operatorname*{argmin}_{S \subset F, k} \; k\, C_\ell + \mathbb{E}(S) C_c, \qquad \text{where} \tag{11}$$

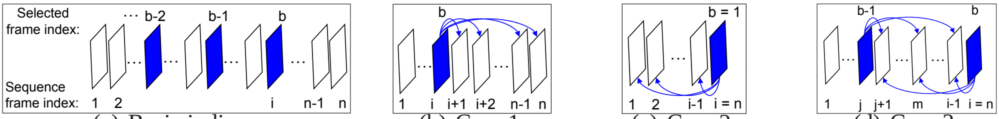
$$\mathbb{E}(S) = \sum_{t=1}^{N} \sum_{\boldsymbol{p} \in t} \min_{j \in \{l_t, r_t\}} \mathrm{P}(\boldsymbol{p}, j, t). \tag{12}$$

$\mathbb{E}(S)$ counts the expected number of erroneous pixels, and is computed using Eqns. 6 and 2. Since choosing which frame to propagate *per pixel* adds a factor of height×width to the computation time, we modify this to select which frame to propagate *per frame*. Thus we can rewrite the cost in terms of an $N \times N$ matrix C, where $C(j,t) = \sum_{\boldsymbol{p} \in t} \mathrm{P}(\boldsymbol{p}, j, t)$:

$$\mathbb{E}(S) = \sum_{t=1}^{N} \min_{j \in \{l_t, r_t\}} C(j, t). \tag{13}$$

In many practical applications, our algorithm would be given a “budget” for k, meaning the total number of frames that one is willing to pay to have labeled. In that case, we target the fixed number of k frames that minimize total propagation error, and the above reduces to:

$$S^* = \operatorname*{argmin}_{S \subset F} \mathbb{E}(S). \tag{14}$$



(a) Basic indices (b) Case 1 (c) Case 2 (d) Case 3

Fig. 4. Sketch to illustrate the index notation (a) and the three cases in the DP solution (b-d).

Without a specified budget, the algorithm chooses both k^* and S^* by solving Eqn. 14 for $k = \{1, \ldots, N\}$ and then selecting the best one. In that case, we automatically determine which frames *and* how many are necessary to minimize the combined labeling and correcting effort.

Note that our approach specifically models the interaction between work done "up front" when a user labels frames and the work done making corrections after propagation. While requesting too many labeled frames overburdens the annotators, so can requesting too few—since correction costs are likely to increase in response. Importantly, our algorithm accounts for this *combined* tradeoff when making its selection.

A naive approach for optimizing Eqn. 14 would take time $O(\binom{N}{k})$, since there are that many subsets of F. However, since the problem exhibits optimal substructure, we next present a much more efficient polynomial time dynamic programming solution.

Dynamic Programming Solution Let $T(i, b, n)$ be the optimal value of $\mathbb{E}(\cdot)$ for selecting b frames from the first n frames, where i denotes the index of the b-th selected frame. See Figure 4(a). Note that this value is valid only when $b \geq 1,\ i \geq b, n \geq i$; otherwise, we set it to ∞. We define the following recurrences for computing all other valid values of T:

Case 1: 1-way → end. $n > i$

$$T(i, b, n) = T(i, b, n-1) + C(i, n).$$

Since i is the last labeled frame, it will propagate its labels to all frames to its right (see Fig. 4(b)). Therefore, the optimal cost of propagating to the first n frames is simply the sum of the optimal cost of propagating to all $n-1$ frames, plus the cost of propagating from frame i to frame n.

Case 2: 1-way → beginning. $b = 1$ and $n = i$

$$T(i, b, n) = \sum_{j=1}^{i-1} C(i, j).$$

Since i is the first frame that is labeled, it propagates its labels to all frames before it (see Fig. 4(c)).

Case 3: Both ways. $b > 1$ and $n = i$

$$\begin{aligned} T(i,b,n) = \min_{j=b-1}^{i-1} \quad & T(j, b-1, n-1) - \sum_{m=j+1}^{i-1} C(j, m) \\ & + \sum_{m=j+1}^{i-1} \min(C(j, m), C(i, m)). \end{aligned} \tag{15}$$

In this case, we need to consider all possible choices $j = \{b-1, \ldots, i-1\}$ for the index of the $(b-1)$-th frame, and select the best in conjunction with frame i. See Fig. 4(d). The last term reflects that every frame m between i and j obtains its label from the frame with the smaller error. We subtract the value $C(j, m)$ in the second term because it was already added in Case 2.

Once T is computed, we obtain the optimal value for a given k as:

$$\mathbb{E}(S^*) = \min_{i \in \{k, \ldots, N\}} T(i, k, N), \tag{16}$$

where i starts at k since the minimum selected index for k total frames is k. We obtain the selected indices by keeping track of which frame j resulted in the smallest value in Eqn. 15 for every i, b, and then backtracking from the minimum index.

The time complexity of the procedure is $O(N^3 k)$, since we need to compute Nk values in Case 3, where in the worst case each value would require N^2 computations. For $N = 1000$ our Matlab implementation takes about 6 seconds. We can reduce this complexity further by keeping the matrix C sparse, by computing values only within a range of frames. In addition, for very long videos, it would be natural to run our algorithm on sub-clips found automatically with shot detection or event segmentation.

4 Results

We now demonstrate our approach is an effective way to select frames for labeling. We consider **three baselines**:

- **Uniform-f**: samples k frames uniformly starting with the first frame, and propagates labels in the forward direction only using our pixel flow method.
- **Uniform**: samples k frames uniformly and transfers labels in both directions. Each frame obtains its labels from the closest labeled frame.
- **Keyframe**: selects k representative frames by performing k-way spectral clustering on global Gist features extracted for each frame. It requests labels for the frame per cluster with highest intra-cluster affinity.

We evaluate three variants of our approach; **DP-PF:** selects k frames using our dynamic programming (DP) algorithm and propagates labels using our pixel flow approach, **DP-MRF:** selects using our DP algorithm and propagates using our MRF-based formulation. **DP2-MRF:** automatically selects the number of frames and their indices by minimizing total annotation cost as defined in Eqn. 11.

Datasets. We use four publicly available datasets[4]: (1) Camseq01: 101 frames of a moving driving scene. (2) Camvid_seq05: first 3000 frames from 0005VD sequence depicting a driving scene. (3) Labelme_8126: (MVI_8126 from ICCV LabelMeVideo [26]) 167 frames depicting a traffic signal, and (4) Segtrack [18], which consists of 6 videos. All four are challenging due to camera ego-motion, color overlap between fg and bg, interframe motion, occluding objects, and deformable shapes.

Ground truth. Both Camseq01 and Camvid_seq05 have labels for each pixel from one of 32 object classes relevant in a driving environment. Camvid_seq05 has ground

[4] Dataset links available at `http://vision.cs.utexas.edu/projects/videoseg`.

		Labelme_8126				Camseq01				Camvid_seq05				
	$k =$	1	5	10	15	1	5	10	15	1	15	30	45	60
Error	DP-MRF (Ours)	**103**	**43**	34	**25**	**305**	**120**	**84**	**75**	**1017**	**342**	**201**	**136**	**92**
	DP-PF (Ours)	109	47	**31**	27	314	129	90	81	1137	420	283	159	107
	Keyframe	119	62	41	30	323	153	113	86	1119	571	390	232	144
	Uniform	166	58	35	31	609	132	101	83	1394	506	298	161	127
	Uniform-f	166	81	51	41	609	180	120	96	1394	463	254	163	127
Time savings over best baseline (mins)		133.6	125.3	33.4	41.8	90.9	60.6	85.9	40.4	504.9	599.0	262.4	123.8	173.3

Table 1. Results on Labelme, Camseq, and Camvid datasets. Values are average number of incorrect pixels (the standard metric in prior work [2, 4, 6, 8, 15, 18]) over all frames **in hundreds of pixels** for our method and the 3 baselines, for varying k values. In all cases, our active approach outperforms the baselines, and yields significant savings in human annotation time (last row).

	birdfall2	girl	cheetah	parachute	penguin	monkeydog
DP-MRF (Ours)	38	491	**466**	**32**	728	723
DP-PF (Ours)	50	**487**	487	45	**612**	592
Keyframe	**36**	557	534	42	706	569
Uniform	37	518	581	52	1172	**472**
Uniform-f	98	2564	802	119	967	787
Time savings over best baseline (secs)	-1.5	19.5	43.0	6.2	59.1	-75.4

Table 2. Results on the Segtrack dataset. Values denote pixel errors when selecting $k = 5$ frames for annotation.

truth for only every 30 frames, so for that data we restrict both selection and evaluation to the labeled frames. This also serves to illustrate how we can reduce selection time for long sequences, since we make C a 100×100 matrix rather than its default 3000×3000 (selection time drops from 750 to 0.06 secs). The Segtrack videos have ground truth for the foreground target object, and thus allow us to demonstrate our method for the case where there is only one main object of interest.

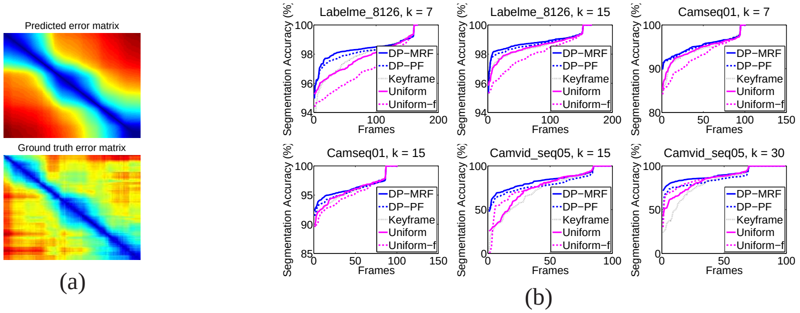
Since Labelme_8126 lacked ground truth, we manually labeled each frame by segmenting the first frame using an interactive toolkit[5]. This took 2-3 minutes *per frame* to correct 2-3% pixel errors, which confirms that even correcting segmentation errors takes significant effort, a major motivator for this work!

Implementation details. We compute optical flow using [17]. We resize all images to 398x530 and choose the 10 most frequent classes for all three videos. All other classes, which occur in $< 0.1\%$ of the pixels, are Background. We use 5 components for the GMMs over 6-dim features per pixel (r, g, b color plus each channel's entropy in a 9×9 patch surrounding the pixel). We set $\beta = 1$, $R = 0.5$, and $U = 10$; we did not try other values. We set C_ℓ to 25 minutes (as reported in [9]) and C_c to 1 minute for 1% pixel error (2000 pixels) based on our labeling experience on LabelMe.

Error prediction model. Figure 5(a) compares the propagation errors *predicted* by our model on LabelMe to the *actual* propagation errors incurred by our pixel flow algorithm if using ground truth segmentations. Each entry in the heat map on top corresponds to $C(i, j) = \sum_{\boldsymbol{p}} \mathrm{P}(\boldsymbol{p}, i, j)$. It shows our error predictions are quite good, hence our selections based on those predicted errors will reflect the true labeling errors well.

The error matrices reveal low risk around the diagonals, which means that every frame has a small range of frames on either side of it to which it can propagate its labels well. Importantly, however, the width of the blue band differs significantly across the frames, confirming our claim that propagation reliability is not always uniform, as assumed by existing techniques.

[5] http://www.robots.ox.ac.uk/∼vgg/software/iseg/index.shtml



(a)

(b)

Fig. 5. (a) Comparison of ground truth label propagation error with the error predicted by our model (C) for Labelme. Our error predictions follow the actual errors fairly closely. (b) Each method's accuracy plotted for all frames, for two values of k per sequence. Accuracy values are sorted from high to low per method for clarity. Our DP approaches (darker blue curves) have higher accuracy, esp. on frames far away from labeled frames. Best viewed on pdf.

Fixed size selections. Table 1 reports the label errors for the first three datasets and all methods, for multiple choices of k (number of labeled frames). As expected, for all methods, error decreases for larger values of k since more effort helps in general. However, our active selection approach (top two rows) outperforms all baselines for all values of k and all videos. Table 2 reports the pixel error on the Segtrack dataset when five frames are selected for annotation. Again our active approach improves over the baselines on a majority of videos. The magnitude of errors and gains on Segtrack are necessarily smaller, since those videos are much shorter than the other datasets and contain only one foreground object. We focus the remaining analysis on the longer multi-object datasets accordingly.

A difference of 20 in Table 1 denotes 2000 incorrect pixels, which would require 1 minute per frame to fix. The last rows of the Tables 1 and 2 shows our method's savings relative to the best baseline per test using this conversion. This clearly shows that active frame selection is crucial to most efficiently use annotator time for video data.

This savings estimate assumes that the cost of correcting errors is proportional to the number of mislabeled pixels. While a simple model of cost, we find it is realistic in practice for these datasets. Most of the errors occur near object boundaries; thus, using the interactive segmentation tool, after a couple initial broad strokes, most time is spent correcting the near-boundary errors. In addition, even when refining the error metric to count only pixels close to the segmentation boundary (up to 20 pixels away), we obtain similar relative outcomes, with our DP-PF and DP-MRF approaches outperforming the baselines.

Figure 5(b) reports accuracy across all frames for the driving datasets. DP-MRF outperforms our basic pixel flow technique, showing the inclusion of an appearance model and smoothness terms reduces propagation errors due to occlusion, drift, and incorrect flow. Keyframe performs poorly compared to our approach, and, surprisingly, is weaker than Uniform for larger k values. This shows that picking *representative* frames does not correctly model how well new labels may influence the rest; our approach specifically models this "trackability" and therefore makes better selections.

Uniform selection with two-way propagation is typically better than Uniform-f, indicating that tracking pixels and transferring labels in both directions is valuable. How-

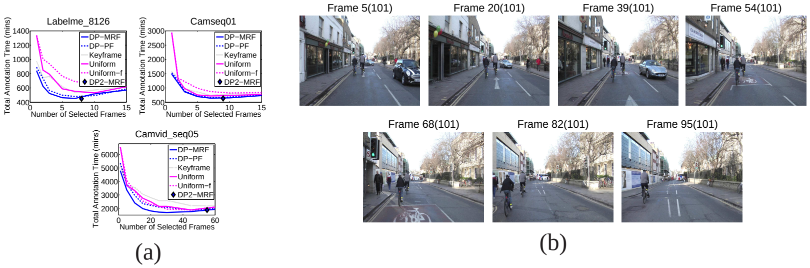


Fig. 6. (a) Total human annotation time required to label each sequence, as a function of selections made per method. Darker lines are ours. Our method reduces effort better than the baselines, and can also predict the optimal number of frames to have labeled (see DP2-MRF diamonds). (b) Frames selected by our approach.

ever, on Camvid_seq05, two-way is worse. This is because the sequence is taken from a car moving forward, and labels are sampled every 30 frames, and so most points tracked in the forward direction move out of the frame. Uniform performs better than our approaches on the monkeydog sequence in Segtrack. This particular sequence is fairly challenging for optical flow computation due to fast movements and indistinctive, low resolution features on the foreground object, which affects our cost matrix.

Figure 6(b) shows the frames selected by our approach for $k = 7$ on the Camseq01 sequence. We see our approach selects non-uniformly spaced frames so that they contain high resolution information of most of the objects that occur in the video (the two cars, bicyclists, pedestrians).

Minimizing total annotation cost. Figure 6(a) shows the total time ($kC_\ell + \mathbb{E}C_c$) each method requires to annotate each video sequence, as a function of k. As k increases, error reduces (decreasing $\mathbb{E}C_c$), but the kC_ℓ term increases. For all methods, the total annotation time has a sweet spot (reflected by the dip and then slow climb in the curves vs. k) where the *combined* effort cost is minimized. Again, our methods require lower total effort on all videos.

This also shows how our DP2-MRF variant can automatically predict the optimal number of frames to get labeled ($k = 8, 9, 55$ for these sequences), which is close to the actual minimum. Labeling all frames would require $4175,\ 2525,\ 2475$ min. for each video, whereas our DP2-MRF's intelligent requests brings that down to $449,\ 633,\ 1880$ min., respectively. This equates to saving up to 90% of annotator effort.

5 Conclusions

We introduced the active multi-frame selection problem. Our approach models expected label propagation errors, and provides an efficient DP solution to make the optimal choice. Results show the real impact of our method in using human time for video labeling most effectively. This line of work has the potential to greatly enhance video labeling tasks, which are increasingly of interest for activity recognition and other applications.

Acknowledgements

This research is supported in part by ONR YIP grant N000141210754.

References

[1] A. Agarwala, A. Hertzmann, D. Salesin, and S. Seitz. Keyframe-Based Tracking for Rotoscoping and Animation. In *SIGGRAPH*, 2004.

[2] V. Badrinarayanan, F. Galasso, and R. Cipolla. Label Propagation in Video Sequences. In *CVPR*, 2010.

[3] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video SnapCut: Robust Video Object Cutout using Localized Classifiers. In *SIGGRAPH*, 2009.

[4] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. Interactive Co-segmentation with Intelligent Scribble Guidance. In *CVPR*, 2010.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast Approximate Energy Minimization via Graph Cuts. *TPAMI*, 2001.

[6] I. Budvytis, V. Badrinarayanan, and R. Cipolla. Label Propagation in Complex Video Sequences using Semi-supervised Learning. In *BMVC*, 2010.

[7] M. Cooper and J. Foote. Discriminative Techniques for Keyframe Selection. *ICME*, 2005.

[8] A. Fathi, M. Balcan, X. Ren, and J. Rehg. Combining Self Training and Active Learning for Video Segmentation. In *BMVC*, 2011.

[9] J. Fauqueur, G. Brostow, and R. Cipolla. Assisted Video Object Labeling by Joint Tracking of Regions and Keypoints. In *Proc. Int. Workshop on Interactive Computer Vision*, 2007.

[10] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient Hierarchical Graph-based Video Segmentation. In *CVPR*, 2010.

[11] S. Hoi, R. Jin, J. Zhu, and M. Lyu. Semi-supervised SVM Batch Mode Active Learning with Applications to Image Retrieval. *ACM Trans. on Info Systems*, 2009.

[12] C. Liu, J. Yuen, and A. Torralba. Nonparametric Scene Parsing: Label Transfer via Dense Scene Alignment. In *CVPR*, 2009.

[13] T. Liu and J. Kender. Optimization Algorithms for the Selection of Key Frame Sequences of Variable Length. In *ECCV*, 2002.

[14] I. Patras, E. Hendriks, and R. Lagendijk. Semi-automatic Object-based Video Segmentation with Labeling of Color Segments. *Signal Processing: Image Communication*, 2003.

[15] B. L. Price, B. S. Morse, and S. Cohen. Livecut: Learning-based Interactive Video Segmentation by Evaluation of Multiple Propagated Cues. In *ICCV*, 2009.

[16] X. Ren and J. Malik. Tracking as Repeated Figure/Ground Segmentation. In *CVPR*, 2007.

[17] N. Sundaram, T. Brox, and K. Keutzer. Dense Point Trajectories by Large Displacement Optical Flow. In *ECCV*, 2010.

[18] D. Tsai, M. Flagg, and J. M.Rehg. Motion Coherent Tracking with Multi-label MRF Optimization. *BMVC*, 2010.

[19] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller. Multiple Hypothesis Video Segmentation from Superpixel Flows. In *ECCV*, 2010.

[20] S. Vijayanarasimhan and K. Grauman. What's It Going to Cost You?: Predicting Effort vs. Informativeness for Multi-Label Image Annotations. In *CVPR*, 2008.

[21] S. Vijayanarasimhan, P. Jain, and K. Grauman. Far-Sighted Active Learning on a Budget for Image and Video Recognition. In *CVPR*, 2010.

[22] C. Vondrick and D. Ramanan. Video Annotation and Tracking with Active Learning. In *NIPS*, 2011.

[23] C. Vondrick, D. Ramanan, and D. Patterson. Scaling Up Video Annotation with Crowdsourced Marketplaces. In *ECCV*, 2010.

[24] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen. Interactive Video Cutout. In *SIGGRAPH*, 2005.

[25] W. Wolf. Key Frame Selection by Motion Analysis. In *ICASSP*, 1996.

[26] J. Yuen, B. Russell, C. Liu, and A. Torralba. Labelme Video: Building a Video Database with Human Annotations. In *ICCV*, 2009.