ACTIVE OBJECT DETECTION ON GRAPHS VIA LOCALLY INFORMATIVE TREES

Daphney–Stavroula Zois, Maxim Raginsky

Coordinated Science Laboratory
University of Illinois at Urbana–Champaign
Urbana, IL 61801
{zois, maxim}@illinois.edu

ABSTRACT

Active object detection refers to the problem of determining the existence and location of objects in an image by actively selecting which regions of the image to explore. Herein, an object detection algorithm is proposed that models image regions as vertices and overlap relationships as edges in a directed weighted graph. Information is propagated from labeled vertices through graph edges that operate as noisy channels via message passing over locally informative trees that are extracted from the original graph using an information-theoretic criterion. Influential vertices are determined by an appropriate centrality index. Our algorithm can be applied on top of any state–of–the–art region proposal method as it treats it as a black box. The effectiveness of the proposed algorithm is illustrated on different scenarios, where in some cases only 0.45% of the total regions is evaluated.

Index Terms— active object detection, graphs, locally informative trees, mutual information, degree centrality

1. INTRODUCTION

Object detection is a fundamental, yet very challenging task in image analysis. A typical object detection algorithm first generates a set of region proposals. These can be either fixed and class–independent [1–5] or generated on the fly while searching for a particular object [6]. Each such proposal is then assigned a class label by running a set of detectors. Evaluating a large number of region proposals will certainly lead to high detection accuracy, but will incur high computational costs if each detector evaluation is computationally expensive.

Active object detection refers to the problem of determining the existence and location of objects in an image by actively selecting which region proposals to explore. In contrast to traditional approaches, where all region proposals are evaluated, we argue that this is not necessary if we appropriately exploit the structure of the generated regions and carefully select which ones to evaluate. We base our intuition on the fact that relationships between region proposals carry important information, i.e., the labels of spatially related proposals are correlated. This fact can significantly reduce the computational effort required to detect objects and give rise to significant speed ups.

In this work, we adopt a graphical model approach for the active object detection problem, where region proposals are modeled as vertices in a graph, edges between them capture overlap relations, and edge weights denote the amount of overlap. We then propose the DE–LITE (DEtection via Locally Informative TrEes) algorithm that selects informative vertices based on weighted degree centrality and propagates information from labeled vertices to unlabeled ones through graph edges that operate as noisy channels. This is achieved by building locally informative trees on top of the original graph and propagating information via message passing. The flow of information is captured by the mutual information between the root and the leaves of the trees. We demonstrate the effectiveness of our algorithm by illustrating its performance on different scenarios. Our method can be a salient part of a successful detection pipeline, since it is agnostic of the intricacies of the region proposal generation process and the operation of the rest parts of the pipeline.

Our approach is inspired by [7], where query strategies are devised for graphs with hidden vertex types that are correlated with the graph topology. In contrast to the above work, where vertices are queried one at a time without information propagation taking place, we select vertices based on weighted degree centrality and exploit the correlation between labels to propagate information. Furthermore, we work with the more challenging case of noisy labels.

Prior art has considered the design of active object detection strategies [4, 5, 8]. A Bayesian approach and two heuristic strategies are proposed in [8]. An active search strategy proposed in [4] sequentially chooses the region proposal to evaluate on the basis of a belief map, which is updated using context and classifier score information. A Markov Decision Process formulation is proposed in [5], which starts with the whole image and proceeds to narrow down the object loca-

This work is supported by the NSF grant CIF-1302438.

We would also like to thank Juan C. Caicedo and Svetlana Lazebnik for the useful discussions and suggestions.

978-1-5090-0746-2/16/$31.00

tion. Unlike the above strategies that either adopt a specific model for the active detection task and/or define subjective cost functions, we adopt a simple approach that estimates key decision statistics from data and exploits information–theoretical principles to achieve efficient object detection. In [4, 5], only one region is evaluated at each time step and a fixed number of regions is evaluated in total. In contrast, in an effort to speed up the detection process, we evaluate multiple regions at each time step and exploit the spatial correlations to label adjacent regions through information propagation. In addition, we stop the search process based on application–specific criteria (e.g., a positive example has been identified or a negative example is inferred).

The remainder of the paper is organized as follows. Section 2 introduces the active object detection problem and our approach. Next, Section 3 describes the tree–growing process and two information bounds used during this process. In Section 4, we present the DE–LITE algorithm, and in Section 5, we illustrate its performance. Finally, we conclude the paper in Section 6.

2. APPROACH

We consider an image represented by a set of region proposals. Our goal is to *quickly* and *accurately* detect any objects present by evaluating only a subset of them. Since region proposals exhibit certain structure, we encode this relation–rich environment in a weighted graph and formulate active object detection as a labeling task on this graph. We devise an algorithm that decides which vertices to query and exploits their labels to infer the labels of other vertices. The main idea is that information "propagates" from already labeled vertices through graph edges that operate as noisy channels. We achieve this by extracting *subtrees* of the original graph and propagating information along these trees via *message passing*, where we have assumed that both the hidden labels and the observable detector outputs behave as random variables on the vertices of the graph.

The joint probability distribution over the graph vertices characterizes the structure of the hidden labels. However, a general graph contains loops that may affect performance of iterative inference strategies, such as belief propagation. To avoid this pitfall, we consider a tree approximation of the joint distribution by extracting locally informative subtrees of the original graph. Tree-structured models are a useful approximation because many inference tasks on them can be carried out exactly using belief propagation. Our approach essentially avoids dealing with a large inference problem by exploiting local graph structure (i.e., vertex–specific content information and link–based dependencies) and local decision making in an effort to achieve a near–optimal global solution.

We need a quantitative measure of influence to pick a root vertex that will initiate the tree–growing process. In graph theory and network analysis, centrality indices are real–valued functions on the graph vertices that provide a ranking and identify the most important nodes. Due to the way region proposal methods operate [9,10], many region proposals, which are related to the objects of interest, overlap in various ways. Weighted degree centrality indices are able to capture this structure and can be computed very efficiently. Thus, we use them for selecting informative vertices.

2.1. Setting

We consider a directed weighted graph $G = (V, E, w)$ with edges (ordered pairs) $e = (u, v) \in E \subseteq V \times V$ and real-valued edge weights $w = \{w_e\}_{e \in E}$. Each $v \in V$ corresponds to a region proposal. The weighted indegree and outdegree centralities of a vertex $v \in V$ are defined as

$$d_v^- = \sum_{u\colon (u,v) \in E} w_{(u,v)} \quad \text{and} \quad d_v^+ = \sum_{u\colon (v,u) \in E} w_{(v,u)}, \quad (1)$$

respectively. The presence of an object in $v \in V$ is described by a random hidden label $X_v \in \{-, +\}$, which can only be observed imperfectly through a detector, with output $Y_v \in \{-, +\}$. The joint distribution of $X = (X_v)$ and $Y = (Y_v)$ is unknown and may be arbitrarily complicated, so we only collect information about the performance of each detector and about pairwise conditional distributions among the Y_v's. Specifically, we make the following assumptions:

1. The detectors are accurate, i.e.,

$$\max_{v \in V} \mathbb{P}[Y_v \neq X_v] \leq \varepsilon \text{ for some } \varepsilon < 1/2. \quad (2)$$

2. For each edge $e = (u, v) \in E$, the conditional distribution of Y_v given Y_u is known and given by a 2×2 stochastic matrix Ψ_e with entries

$$\Psi_e(y, y') = \mathbb{P}[Y_v = y' | Y_u = y], \ y, y' \in \{-, +\}. \quad (3)$$

We will work with rooted subtrees in G. Consider an arbitrary tree $T = (V_T, E_T)$ with root $r \in V$, where $V_T \subset V$ and $E_T \subset E$. We attach to each $v \in V_T$ a binary random variable $Z_v \equiv Y_v$, and let $Z_T \triangleq (Z_v)_{v \in V_T}$, $Z_{T \setminus r} \triangleq (Z_v)_{v \in V_T \setminus \{r\}}$. Then, we can write down the following tree-structured conditional distribution for all internal vertices given the root:

$$\mathbb{P}[Z_{T \setminus r} = z_{T \setminus r} | Z_r = z_r] = \prod_{e=(u,v) \in E_T} \Psi_e(z_u, z_v). \quad (4)$$

Note that if the original graph G is not a tree, then (4) will not capture all of the dependencies between the Z_v's, so we treat it as a tractable local approximation. The amount of information Z_r contains about the hidden label X_r is measured by the *mutual information* $I(X_r; Z_r) = H(X_r) - H(X_r | Z_r)$ [11]; in particular, from the assumption (2) and from Fano's inequality [11, Sec. 2.10] we see that

$$I(X_r; Z_r) \geqslant \log 2 - h_2(\varepsilon), \quad (5)$$

where $h_2(\varepsilon) \triangleq -\varepsilon \log \varepsilon - (1-\varepsilon)\log(1-\varepsilon)$ is the binary entropy function. Now let $L_T \subset V_T$ denote the set of leaves of T. The amount of information about Z_r that will propagate through the tree to the leaves is captured by the *information transmission coefficient* of T:

$$\vartheta(r, T) \triangleq \frac{I(Z_r; Z_{L_T})}{H(Z_r)}, \tag{6}$$

Since $I(A; B) \le \min\{H(A), H(B)\}$, $\vartheta(r, T) \le 1$, and we will say that the tree T is δ-*informative* (about r) if $\vartheta(r, T) \ge 1 - \delta$ for some $\delta \in [0, 1)$. If we attach additional edges to the leaves of T to get a new subtree T' of G rooted at r, then $\vartheta(r, T') \le \vartheta(r, T)$ by the data processing inequality [11]. Therefore, the informativeness of such local subtrees will monotonically decrease with depth.

3. TREE–GROWING PROCESS

In this section, we describe the tree–growing process. The process has two iteration loops. At each iteration t of the outer loop, we have a subgraph $G_t = (V_t, E_t)$ of G, induced by a vertex set $V_t \subset V$. Assume a root $r_t \in V_t$ has been selected to build a rooted tree $T_t = (V_{t,T}, E_{t,T})$. Once the root is picked, we sequentially add edges to the tree, while the tree T_t stays δ–informative.

The addition of edges takes place in the inner loop. At iteration $\tau = 0$ of the inner loop, we start with $T_t^\tau = (\{r_t\}, \varnothing)$. At each iteration $\tau \ge 1$, we add another edge to $T_t^{\tau-1}$ to generate T_t^τ and check whether $\vartheta(r_t, T_t^\tau) \geqslant 1-\delta$. The inner loop terminates once this condition is violated. The computation of ϑ requires the knowledge of the joint probability distributions of Z_{r_t} and $Z_{L_{T_t^\tau}}$, which can be computed exactly using Sum–Product Variable Elimination. Unfortunately, determining the optimal elimination order is $\mathcal{NP}$–hard [12], while at each step of the tree–growing process the number of vertices in the tree increases. To avoid these pitfalls, we will use easily computable lower and upper bounds for the mutual information instead of the exact value.

3.1. Mutual information bounds

Consider a rooted tree $T = (V_T, E_T)$ with root r. It will be convenient to parametrize the entries of the stochastic matrix Φ_e corresponding to each edge $e = (u, v) \in E_T$ as follows:

$$\Psi_e = \begin{pmatrix} 1-\varepsilon_e^- & \varepsilon_e^- \\ \varepsilon_e^+ & 1-\varepsilon_e^+ \end{pmatrix}. \tag{7}$$

The mutual information can be lower-bounded as follows:

Theorem 1. *Let* $\pi \triangleq \mathbb{P}[Z_r = +]$. *Then*

$$I(Z_r; Z_{L_T}) \ge$$
$$2\pi(1-\pi)\log e \left(\frac{1}{|L_T|} \sum_{v \in L_T} \prod_{e \in \mathrm{path}(v)} (1 - (\varepsilon_e^+ + \varepsilon_e^-)) \right)^2, \tag{8}$$

where $\mathrm{path}(v)$ *is the set of* $e \in E_T$ *on the path from* r *to* v.

For the upper bound, we introduce the *strong data processing constant* (or SDP constant, for short) [13] of a noisy channel $P_{U|V}$ with input U and output V:

$$\eta(P_{V|U}) \triangleq \sup_{P_U \ne Q_U} \frac{D(P_V \| Q_V)}{D(P_U \| Q_U)}, \tag{9}$$

where $D(\cdot\|\cdot)$ denotes the relative entropy (Kullback–Leibler divergence), and where P_V (respectively, Q_V) denotes the resulting output distribution when the input distribution is P_U (respectively, Q_U). It can be shown that, for any input distribution P_U, $I(U; V) \le \eta(P_{V|U})H(U)$. Therefore, $\vartheta(r, T) \le \eta(r, T)$, where $\eta(r, T)$ denotes the SDP constant of the channel with input Z_r and output Z_{L_T} induced by the tree-structured conditional distribution (4). Moreover, as we will show next, the quantity $\eta(r, T)$ can be upper-bounded using the entries of the matrices (7) for all $e \in E_T$. For any $v \in V_T$, let T_v denote the rooted subtree of T consisting of v and all of its descendants, and let $\eta(v, T)$ denote the SDP constant of the channel with input Z_v and output $(Z_w)_{w \in L_{T_v}}$. Then an upper-bound on $\eta(r, T)$ can be computed recursively as follows:

Theorem 2. *If T has depth 1, i.e., if $V_T = \{r\} \cup L_T$ and $E_T = \{(r, v) : v \in L_T\}$, then*

$$\eta(r, T) \le \eta^{\mathrm{UB}}(r, T)$$
$$\triangleq \left(1 - \prod_{e \in E_T} \left(\sqrt{(1-\varepsilon_e^-)\varepsilon_e^+} + \sqrt{(1-\varepsilon_e^+)\varepsilon_e^-} \right)^2 \right). \tag{10}$$

If the depth of T is greater than 1, then

$$\eta(r, T) \le \eta^{\mathrm{UB}}(T_t) \triangleq \left(1 - \prod_{v \in \mathrm{ch}(r)} (1 - \eta_{(r,v)}\eta(v, T)) \right), \tag{11}$$

where $ch(r)$ is the set of all children of the root r.

3.2. Evolution of information during tree-growing

There are two possible scenarios of growing the existing tree $T_t^{\tau-1}$ by adding a new vertex v to it:

- a lateral addition, where the new vertex v is a neighbor of a non–leaf of $T_t^{\tau-1}$;
- a vertical addition, where the new vertex v is a neighbor of a leaf $u \in L_{T_t^{\tau-1}}$.

In the first case, we have $L_{T_t^\tau} = L_{T_t^{\tau-1}} \cup \{v\}$, and the chain rule for mutual information gives

$$I(Z_{r_t}; Z_{L_{T_t^\tau}}) = I(Z_{r_t}; Z_{L_{T_t^{\tau-1}}}, Z_v)$$
$$= I(Z_{r_t}; Z_{L_{T_t^{\tau-1}}}) + I(Z_{r_t}; Z_v | Z_{L_{T_t^{\tau-1}}}), \tag{12}$$

so the mutual information increases. In the second case, though, the mutual information will *decrease*. To see this, we need the following lemma.

Lemma 1. *Let X, U, V, W be four discrete random variables, whose joint distribution factorizes as $\mathbb{P}_{XUVW}(x,u,v,w) = \mathbb{P}_{XUV}(x,u,v)\mathbb{P}_{W|V}(w|v)$. Then:*

$$I(X;UW) - I(X;UV) = I(X;W|U) - I(X;V|U) \leqslant 0. \quad (13)$$

Let $u \in L_{T_t^{\tau-1}}$ be the leaf whose neighbor $v \notin V_{T_t^{\tau-1}}$ will be added to $T_t^{\tau-1}$. Then, applying the above lemma to $X = Z_{r_t}$, $U = Z_{L_{T_t^{\tau-1}}\setminus\{u\}}$, $V = Z_u$, and $W = Z_v$, we see that

$$\begin{aligned} I(Z_{r_t}; Z_{L_{T_t^\tau}}) &= I(Z_{r_t}; Z_{L_{T_t^{\tau-1}}\setminus\{u\}}, Z_v) \\ &\leqslant I(Z_{r_t}; Z_{L_{T_t^{\tau-1}}\setminus\{u\}}, Z_u) \\ &\equiv I(Z_{r_t}; Z_{L_{T_t^{\tau-1}}}). \end{aligned} \quad (14)$$

We can upper–bound the amount of mutual information decrease using the following result [14, Cor. 1]:

Lemma 2. *Let X, U, V, W be four discrete random variables, where $V, W \in \{-,+\}$, and $(X,U) \to V \to W$ is a Markov chain. Then*

$$\frac{I(X;W|U)}{I(X;V|U)} \leq \sin^2\theta, \quad (15)$$

where $p = \mathbb{P}[W=+|V=-]$, $q = \mathbb{P}[W=-|V=+]$, and θ is the angle between the vectors $(\sqrt{1-p}, \sqrt{p})$ and $(\sqrt{q}, \sqrt{1-q})$ in $\mathbb{R}^2$.

For each $e \in E$, let $\theta(e)$ be the angle between the vectors

$$\left(\sqrt{1-\varepsilon_e^-}, \sqrt{\varepsilon_e^-}\right) \text{ and } \left(\sqrt{\varepsilon_e^+}, \sqrt{1-\varepsilon_e^+}\right).$$

Let u be the leaf of $T_t^{\tau-1}$, whose neighbor $v \notin V_{T_t^{\tau-1}}$ is to be added as a new leaf. Then, by Lemmas 1 and 2, we get:

$$\begin{aligned} I(Z_{r_t}; Z_{L_{T_t^\tau}}) \leq\ & I(Z_{r_t}; Z_{L_{T_t^{\tau-1}}}) \\ & - \cos^2\theta((u,v)) I(Z_{r_t}; Z_u | Z_{L_{T_t^{\tau-1}}\setminus\{u\}}). \end{aligned} \quad (16)$$

Since lateral additions increase mutual information, while vertical additions decrease it, the tree–growing process is essentially an alternating sequence of lateral and vertical additions. This is because, at some point, there may be no more neighbors in a particular direction, but the termination criterion has not been met yet. As a result, the tree–growing process begins with a vertical addition, continues with lateral additions until there are no neighbors in this direction, then performs a vertical addition, continues with lateral additions, *etc*. The process stops when all possible vertical additions result in $I(Z_{r_t}; Z_{L_{T_t^\tau}})$ becoming less than $(1-\delta)H(Z_{r_t})$. This protocol is followed during the execution of the DE–LITE algorithm, described in the next section. The next two corollaries provide recursive forms of the bounds presented in Section 3 that exploit the structure of the tree–growing process. Specifically, let I_τ^{LB} denote the lower bound on the mutual information at iteration τ of the inner loop, and let η_τ^{UB} denote the upper bound on the SDP constant of the tree.

Corollary 1. *If a lateral addition of a new leaf u is performed at round τ, then*

$$I_{\tau+1}^{\mathrm{LB}} = 2\pi(1-\pi)\log e \left(\frac{|L_{T_t^\tau}|}{|L_{T_t^\tau}|+1} \sqrt{\frac{I_\tau^{\mathrm{LB}}}{2\pi(1-\pi)\log e}} + \frac{1}{|L_{T_t^\tau}|+1} \cdot c(\mathrm{parent}(u)) \cdot (1-(\varepsilon_e^+ + \varepsilon_e^-)) \right)^2, \quad (17)$$

where $e = (\mathrm{parent}(u), u)$ and the function $c(\cdot)$ captures the contribution of the path from the root r to the parent of the newly added vertex u with $c(r) = 1$. For a vertical addition of a new leaf u, the lower bound can be updated as follows:

$$I_{\tau+1}^{\mathrm{LB}} = 2\pi(1-\pi)\log e \left(\sqrt{\frac{I_\tau^{\mathrm{LB}}}{2\pi(1-\pi)\log e}} - \frac{1}{|L_{T_t^\tau}|} \cdot c(\mathrm{parent}(u)) \cdot (\varepsilon_e^+ + \varepsilon_e^-) \right)^2, \quad (18)$$

with the initial condition $I_0^{\mathrm{LB}} = 2\pi(1-\pi)\log e\,(1-(\varepsilon_e^+ + \varepsilon_e^-))^2$ and $e = (r,u)$.

Corollary 2. *For lateral additions, the upper bound can be recursively updated as follows:*

$$\eta_{\tau+1}^{\mathrm{UB}} = \eta_\tau^{\mathrm{UB}} + \eta_{(\mathrm{pa}(v_{\tau+1}), v_{\tau+1})} \times \left(\prod_{e \in \mathrm{path}(\mathrm{pa}(v_{\tau+1}))} \eta_e \right) \left(\prod_{i=0}^{\tau} A_{v_i, v_{i+1}} \right), \quad (19)$$

where $\mathrm{pa}(v_{\tau+1})$ denotes the parent of $v_{\tau+1}$, $\mathrm{path}(\mathrm{pa}(v_{\tau+1}))$ the path from the root to the parent of the newly added vertex $v_{\tau+1}$, $\eta_e \triangleq \sin^2(\theta(e))$, $A_{v_i,v_j} \triangleq \prod_{v \in \mathrm{ch}(v_i)-\{v_j\}} (1 - \eta_{(v_i,v)} \eta(T_{t,v}))$, and $v_0 \triangleq r$. For vertical additions, the upper bound can be recursively updated as follows:

$$\eta_{\tau+1}^{\mathrm{UB}} = \eta_\tau^{\mathrm{UB}} - (1 - \eta_{(\mathrm{pa}(v_{\tau+1}), v_{\tau+1})}) \times \left(\prod_{e \in \mathrm{path}(\mathrm{pa}(v_{\tau+1}))} \eta_e \right) \left(\prod_{i=0}^{\tau-1} A_{v_i, v_{i+1}} \right). \quad (20)$$

The initial condition is $\eta_1^{\mathrm{UB}} = \eta_{(r,v)}$, where v is the first added child of the root r.

4. DE–LITE ALGORITHM

In this section, we describe the DEtection via Locally Informative TrEes (DE–LITE) algorithm, which consists of a *training* and a *testing* phase. In both phases, a graph

$G = (V, E, w)$ is created for each image, where each vertex corresponds to a region proposal, the edges represent overlap/containment relationships and the weights represent the Intersection over Union (IoU) affinity measure

$$\text{IoU}(A, B) \triangleq \frac{\text{area}(A \cap B)}{\text{area}(A \cup B)}, \tag{21}$$

where A and B are region proposals defined by the coordinates of the top–left and bottom–right corners. An edge $e = (u, v) \in E$ is created as follows: 1) if A and B do not overlap, u and v are not connected through an edge, 2) if A and B overlap, u and v are connected with two edges $u \to v$ and $v \to u$, 3) if A is contained in B, u and v are connected with edge $u \to v$, and 4) if B is contained in A, v and u are connected with edge $v \to u$.

During the training phase, the channel transition probabilities in (3) are estimated from training data using the Krichevsky–Trofimov estimator [15]:

$$\Psi_e(y, y') = \frac{N((y, y'), w_e) + \frac{1}{2}}{\sum_x N(y'', y), w_e) + 1}, \tag{22}$$

where $N((y, y'), w_e)$ denotes the number of edges with label assignment (y, y') and weight w_e. The above smoothed estimator alleviates the fact that some of the counts above may be zero or close to zero. We estimate the root pmf as follows:

$$\pi_r = \left[\frac{N_0}{N_0 + N_1}, \frac{N_1}{N_0 + N_1} \right]^T, \tag{23}$$

where N_1 (N_0) denote the number of images in the dataset that (do not) contain the object of interest.

The testing phase of the algorithm consists of two parts. During the first part, a test image is selected from the testing dataset, a set of region proposals is extracted via an appropriate algorithm, and the corresponding graph is generated. During the second part, the actual algorithm is run on the graph G. At each iteration $t = 1, 2, \ldots$ of the algorithm, we have a graph $G_t = (V_t, E_t)$, induced by a vertex set $V_t \subset V$. Note that we initially start with the graph $G = (V, E, w)$. As a first step, we select a vertex $v \in V_t$ to serve as the root. We may accomplish this step by either of two ways. If this is the first iteration of the algorithm or the leaves of the tree generated in the previous iteration have all their neighboring vertices already labeled, then we select as root the vertex that has the largest weighted outdegree (indegree) centrality and has not been labeled yet. If a tree was generated in the previous iteration and the associated leaves have unlabeled neighboring vertices, we select as root the neighbor that has the largest weighted outdegree (indegree) centrality among the unlabeled neighbors. After having selected the root vertex, we use the tree–growing process described in Section 3 to build a rooted tree $T_t = (V_{t,T}, E_{t,T})$, such that T_t is δ–informative. Once the tree is built, we query the detectors at the leaves and observe their outputs $\{Z_{v_t} \equiv Y_{v_t}, v_t \in L_{t,T}\}$. We assign labels to the rest of the vertices $u \in V_{t,T}$ via the Max–Product variable elimination algorithm [12]. Once this is done, we form V_{t+1} by removing $V_{t,T} : V_{t+1} = V_t \backslash V_{t,T}$. This process continues until a certain application–specific condition has been met. During the execution of the DE–LITE algorithm, the root pmf is either estimated from training data as described above or determined based on the channel matrices and the label of the leaf neighbor that will serve as the new root.

Table 1. Performance of DE–LITE algorithm (LB: lower bound, UB: upper bound)

	AP	q_T	q_P
LB (true labels)	1	18.97%	2.50%
UB (true labels)	1	18.97%	2.50%
LB (noisy labels)	0.76	18.95%	0.45%
UB (noisy labels)	0.35	21.54%	18.19%

5. EXPERIMENTS

In this section, we illustrate the performance of the DE–LITE algorithm on the detection of a specific object type. Training and testing were performed on a subset of the training and testing sets of the Pascal 2007 dataset [16]. All included images contain at most one object instance of the aeroplane class. Since we are interested in detecting single object instances, images with multiple instances of aeroplanes are ignored during training and testing. Both the training and testing datasets consist of 200 images. The Selective Search algorithm [9] is used to extract region proposals for these two datasets. In general, the number of generated proposals per image ranges from 259 and 4102. We have also assumed that there is always a tight region proposal that encompasses the object of interest.

The DE–LITE algorithm stops either if a label 1 is assigned to a specific region proposal or a maximum number of queries per image has been met. The latter threshold corresponds to a percentage, which is estimated from the training dataset using the weighted degree centrality value of the ground truth boxes as a discriminator of positive and negative examples. To be considered as a correct detection, the IoU between the recovered region proposal and the ground truth must exceed 0.5. We evaluate performance with respect to: (1) *average Precision (AP)* [16], (2) *total executed queries percentage* q_T, which denotes the percentage of executed queries over the entire testing dataset, and (3) *total executed positive queries* q_P, which denotes the percentage of executed queries over the positive examples in the testing dataset. Results are reported for $\delta = 0.8$, weighted in–degree centrality and the two mutual information bounds. Results for the weighted out–degree centrality are not included since they result in smaller AP.

We consider two cases for evaluation purposes. In the first case, the ground truth labels are used during training and testing. We study this scenario, since we wanted to test the behavior and correctness of the DE–LITE algorithm in a detection error free setting. In the second case, noisy labels are used during training and testing. These are generated by considering the addition of structured noise as follows: random Bernoulli noise is added with probability $p = 0.1$ to the ground truth labels of the region proposals that have IoU greater than or equal to zero with the ground truth region. We study this particular case since these region proposals usually lead to false positive detections that significantly affect AP. We do not consider location–independent noise, because this is not the case in practice. Note that bad results in terms of AP are usually caused by detection errors (e.g., detector misses an object or makes a false detection), which is the case for large values of p or actual detector scores.

Table 1 illustrates AP, q_T and q_P results for the two cases using the two bounds. In the case of true labels and using either of the two bounds, AP is equal to 1, which suggests that all objects are retrieved. This perfect performance is achieved by querying only 18.97% of the total number of region proposals. Furthermore, q_P is only 2.5%, which implies that most effort is spent on recognizing negative examples. *In fact, recognizing positive examples requires minimal effort!* In the case of noisy labels, we observe a decrease in AP due to detection errors. Using the upper bound to build locally informative trees leads to lower AP and larger values of q_T and q_P. This is due to the fact that we may significantly overestimate the true value of mutual information, thus adding more uninformative vertices to the tree. To alleviate this issue, one should use larger values of δ. On the other hand, using the lower bound leads to lower q_P values, since the search process may stop earlier due to a lower quality detection. Note that using either bound leads to better AP versus the brute force approach of evaluating all proposals, which yields AP 0.29 due to the existence of many false positives. At the same time, randomly selecting which region proposals to evaluate results in AP 0.31 with $q_T = 65.82\%$ and $q_P = 45.32\%$. Thus, carefully ranking and actively selecting region proposals can decrease both the number of queries and false positives.

6. CONCLUSION

In this paper, an algorithm was proposed that addresses the active object detection problem as information propagation in graphs. To this end, region proposals are modeled as vertices and overlap relationships as edges in a directed weighted graph. Information is propagated through graph edges that operate as noisy channels. Locally informative trees are built on top of the graph to facilitate this process and vertices are selected based on weighted degree centrality. The performance of the proposed algorithm is illustrated, where in some cases only 0.45% of the total regions is evaluated. In future work, we plan to consider the case where multiple objects of different classes and different types of detectors are present.

7. REFERENCES

[1] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *ArXiv e-prints*, Dec. 2013.

[2] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “Scalable object detection using deep neural networks,” in *Proc. of the IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2155–2162.

[3] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *ArXiv e-prints*, 2015.

[4] A. Gonzalez-Garcia, A. Vezhnevets, and V. Ferrari, “An active search strategy for efficient object detection,” in *Proc. of the IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3022–3031.

[5] J. C. Caicedo and S. Lazebnik, “Active object localization wih deep reinforcement learning,” in *Proceedings of the International Conference on Computer Vision (ICCC)*, 2015.

[6] Y. Lu, T. Javidi, and S. Lazebnik, “Adaptive Object Detection Using Adjacency and Zoom Prediction,” in *Proc. of the IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2351–2359.

[7] X. Yan, Y. Zhu, J.-B. Rouquier, and C. Moore, “Active Learning for Hidden Attributes in Networks,” in *NIPS Workshop on Analyzing Networks and Learning with Graphs*, Dec. 2009.

[8] M. Naghshvar and T. Javidi, “Two–dimensional visual search,” in *IEEE International Symposium on Information Theory (ISIT)*, July 2013, pp. 1262–1266.

[9] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, “Selective Search for Object Recognition,” *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[10] C. L. Zitnick and P. Dollar, “Edge boxes: locating object proposals from edges,” in *Proc. of ECCV*, 2014, pp. 391–405.

[11] T. M. Cover and J. T. Thomas, *Elements of Information Theory*, Wiley, 2nd edition, 2006.

[12] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.

[13] M. Raginsky, “Strong data processing inequalities and Φ-Sobolev inequalities for discrete channels,” *IEEE Trans. on Information Theory*, vol. 62, no. 6, pp. 3355–3389, June 2016.

[14] W. S. Evans and L. J. Schulman, “Signal propagation and noisy circuits,” *IEEE Trans. on Information Theory*, vol. 45, no. 7, pp. 2367–2373, Nov. 1999.

[15] R. Krichevsky and V. Trofimov, “The performance of universal encoding,” *IEEE Trans. on Information Theory*, vol. 27, no. 2, pp. 199–207, March 1981.

[16] M. Everingham, L. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal Visual Object Classes (VOC) Challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2009.