3-1-2017

Bayesian Optimization for Refining Object Proposals

Anthony D. Rhodes
Portland State University

Jordan Witte
Portland State University

Melanie Mitchell
Portland State University, mm@pdx.edu

Bruno Jedynak
Portland State University, bruno.jedynak@pdx.edu

Citation Details

Rhodes, A. D., Witte, J., Mitchell, M., Jedynak, B., (2017). Bayesian Optimization for Refining Object Proposals arXiv preprint arXiv: arXiv:1703.08653

Bayesian Optimization for Refining Object Proposals

Anthony D. Rhodes[1], Jordan Witte[2], Melanie Mitchell[2], Bruno Jedynak[1]

Portland State University
[1]Department of Mathematics and Statistics, [2]Department of Computer Science

Abstract

We develop a general-purpose algorithm using a Bayesian optimization framework for the efficient refinement of object proposals. While recent research has achieved substantial progress for object localization and related objectives in computer vision, current state-of-the-art object localization procedures are nevertheless encumbered by inefficiency and inaccuracy.

We present a novel, computationally efficient method for refining inaccurate bounding-box proposals for a target object using Bayesian optimization. Offline, image features from a convolutional neural network are used to train a model to predict an object proposal's offset distance from a target object. Online, this model is used in a Bayesian active search to improve inaccurate object proposals.

In experiments, we compare our approach to a state-of-the-art bounding-box regression method for localization refinement of pedestrian object proposals. Our method exhibits a substantial improvement for the task of localization refinement over this baseline regression method.

1. Introduction

Fine-grained object localization is an enduring and critical challenge in computer vision. For example, precise localization of pedestrians in images remains an area of active research due to its rich application potential [27]. Although recent advances in computer vision have achieved impressive results for object detection, these methods commonly employ semi-exhaustive search, requiring a high volume—typically thousands—of potentially expensive function evaluations, such as classifications by a convolutional neural network (CNN). Furthermore, such methods, by virtue of their black-box nature, often lack the kind of interpretability desirable in artificial intelligence applications [12]. In contrast, our approach aims for efficiency, accuracy and intelligibility.

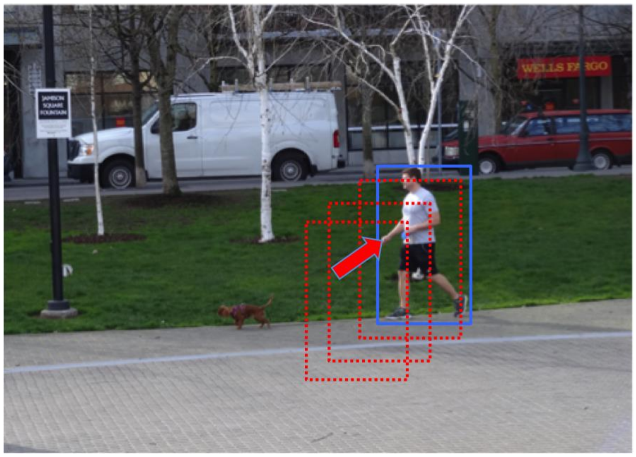

Figure 1. Idealized visualization of refining object proposals for pedestrian image data. (All figures in this paper are best viewed in color.)

Girshick et al. [8][9] achieved state-of-the-art performance on several object detection benchmarks using a "regions with convolutional neural networks" (R-CNN) approach. More recently, faster R-CNN [19] combined the fine-tuned improvements of Fast R-CNN for detection by merging it with a region-proposal network (RPN) that simultaneously predicts object bounds and objectness scores for proposals. R-CNN and its extensions crucially rely on the computation of CNN features, which have been shown to outperform hand-coded features—e.g. HOG features—in difficult vision tasks [5]. To avoid the problem of exhaustively computing CNN features over an entire image, R-CNN utilizes a selective search algorithm [25] that effectively diminishes this computational overhead, with the requirement that the image is segmented first. While the various R-CNN models perform well on general detection tasks, localization error can be a significant weakness of this framework. In particular, Hoiem et al. [11] show that inaccurate or "misaligned" bounding-boxes (i.e., boxes with a small IOU or intersection over union: $0.05 < \text{IOU} < 0.5$) present a significant difficulty, as localization error is not easily handled by object detectors. Indeed, the R-CNN models are critically reliant on high-quality (i.e., $\text{IOU} > 0.5$) initial proposals; when no such proposals are present, R-CNN achieves much weaker results [27]. The various R-CNN models all use category-specific "bounding-box regression" models to refine object proposals made by the system.

More generally, refinement of inaccurate bounding-box proposals for fine-grained localization is a vital task for many real-world applications of computer vision, including autonomous driving [7], object tracking, medical computer vision [1], and robotics [15]. In this paper, we describe an algorithm that improves in several ways on the bounding-box regression method used in R-CNN and other state-of-the-art object-detection architectures.

As part of our method, we use features computed by a pretrained CNN to provide a localization "signal." This is in contrast to using the CNN as a discriminative object detector. We show that this signal (a function of the normalized offset distance of a bounding-box from the target ground-truth object) can be used effectively in a Bayesian optimization setting to quickly improve inaccurate proposals.

Our work provides the following contributions: (1) We demonstrate that CNN features computed from an object-proposal bounding box can be used to predict spatial offset from a target object. (2) We frame the localization method as an active search using Gaussian processes and a dynamic Bayesian optimization procedure requiring very few bounding-box proposals for substantial localization refinement. (3) By rendering an active Bayesian search, our method can provide a principled and interpretable groundwork for more complex vision tasks. We compare our approach with the bounding-box regression method used in R-CNN through experiments that test efficiency and accuracy for the task of localization refinement.

The subsequent sections give some background on related work, the details of our method and algorithm, experimental results, summary remarks, and considerations of future work.

2. Background and Related Work

Object localization is the task of locating an instance of a particular object category in an image, typically by specifying a tightly-cropped bounding box centered on the instance. An object proposal specifies a candidate bounding box, and an object proposal is said to be a correct localization if it sufficiently overlaps a human-labeled "ground truth" bounding box for the given object. In the computer vision literature, overlap is measured via the intersection over union (IOU) of the two bounding boxes, and the threshold for successful localization is typically set to 0.5 [6]. In the literature, the "object localization" task is to locate one instance of an object category, whereas "object detection" focuses on locating all instances of a category in a given image.

For humans, recognizing a visual situation—and localizing its components—is an *active process* that unfolds over time, in which prior knowledge interacts with visual information as it is perceived, in order to guide subsequent eye movements. This interaction enables a human viewer to very quickly locate relevant aspects of the situation [16].

Our method supports this more human-like approach of active object localization (e.g., [4][10][14]), in which a search for objects likewise unfolds over a series of time steps. At each time step the system uses information gained in previous time steps to decide where to search.

In recent years, CNN-based features have become common for detection and localization tasks. Sermanet et al. [22] apply an exhaustive, sliding window approach with CNNs but use convolutions on the entire image for efficiency. Girshick et al. [9] achieved state-of-the-art results with R-CNN by exploiting the richness of CNN features in combination with the efficiency of selective search for object proposals. Several subsequent extensions of R-CNN further improve the region proposal module [8][19]. Of note, the OverFeat method [21] applies deep learning to directly predict box coordinates for localization; Multibox [24] utilizes a saliency-inspired network for proposals, and then applies bounding-box regression for detection.

Finally, the work of Zhang et al. [28] provides an extension of R-CNN that relates most closely to the present work due to its use of Bayesian optimization. Despite this similarity, our work differs significantly in several important ways. Zhang et al., for instance, train their classifier as an object detector, whereas we instead train an offset-prediction signal. Furthermore, where Zhang et al. demonstrate a marginal improvement over baseline R-CNN on localization tasks, our method is fine-tuned for refining object proposals, particularly in the case of very inaccurate initial proposals.

3. Bayesian Optimization for Refining Object Proposals

Bayesian optimization is frequently applied in domains for which it is either difficult or costly to directly evaluate an objective function. In the case of object detection and localization, it is computationally prohibitive to extract CNN features for a large number of bounding-box proposals (this is why, for instance, R-CNN utilizes selective search). There consequently exists a fundamental tension at the heart of any object localization paradigm: with each bounding box for which we extract CNN (or some such robust set of learned) features, we gain useful knowledge that can be directly leveraged in the localization process, but each such piece of information comes at a price.

A Bayesian approach is well-suited for solving the problem of function optimization under these challenging circumstances. In the case of object-proposal refinement, we are attempting to minimize the spatial offset from a ground-truth bounding box (Figure 1). To do this, we train a model, y (described in Section 3.1), to predict spatial offset of a proposal using CNN features extracted from the proposal. Once trained, the model's output can be used to minimize the predicted offset. Ideally, the model's output is minimum when it is given the features of the actual ground-truth bounding box of the target object.

Because we wish to minimize the number of calls to a potentially expensive prediction model, we choose instead to optimize a cheap approximation—the *surrogate* (also called the response surface) to the offset prediction—over the image space. We give details of the realization of the surrogate function as a Gaussian process in Section 3.2.

Finally, after rendering this approximation, we decide where to sample next according to the principle of *maximum expected utility*, which is itself a secondary optimization problem. We identify utility using a dynamically defined *acquisition function* that strikes a balance between minimizing uncertainty and greedy optimization. This method is described in more detail in Section 3.3.

3.1 Training an Offset-Prediction Model

We train a model that, given an inaccurate object proposal, can predict the proposal's normalized offset distance[1] from a target ground-truth object. The output of this model is the predicted distance of a proposal's center from the center of the target object, and the inverse of the output is the predicted proximity. We call the latter the "response signal." The higher the response signal, the closer the proposal is predicted to be to the target.

For each image in the training set, we generate a large number of image crops that are offset from the ground-truth pedestrian by a random amount. These randomized offset crops cover a wide range of IOU values (with respect to the ground-truth bounding box). Furthermore, these offset crops are also randomly scaled, so that the offset-prediction model can learn scale-invariance (with regard to bounding box size) for approximating offset distance. For each of the offset crops, we extracted CNN features using the pre-trained imagenet-vgg-f network in MatConvNet [29].

Using these features, we trained a ridge regression model mapping features to normalized offset distance from the ground-truth bounding box center. Next, we transformed this mapping in two steps using: (1) a scale transformation so that our feature-mapping scale corresponds to the bandwidth parameter used in the Gaussian process (see Section 3.2); and (2) a Gaussian-like transformation so that our prediction model renders an appropriate basin of attraction around the center of a target object that coheres with basic Gaussian process model assumptions. Note that in our regime, small offsets from the center of the target ground will yield (ideally) a maximum response signal. To improve the accuracy of our offset predictor, we average an ensemble of model outputs ranging over five different bounding-box scales.

The performance results of the offset-prediction model are plotted in Figure 2.

3.2 Gaussian Processes

We use a Gaussian Process (GP) to compute a surrogate function f using observations $\{y\}$ of response signals from our prediction model: $y(x) = f_0(x) + \varepsilon$. (Recall that the signal y is high when the input proposal is predicted to be close to the target object.) The surrogate function approximates f_0, the objective signal value for coordinates x in the image space, with ε connoting the irreducible error for the model.

GPs offer significant advantages over other general-purpose approaches in supervised learning settings due in part to their non-parametric structure, relative ease of computation and the extent to which they pair well with a Bayesian modeling regime. GPs have been applied recently with success in a rich variety of statistical inference domains, including [3][26].

More formally, we let $x_i \in \mathbb{R}^2$ be the ith observation from a dataset $D_{1:T} = \{x_{1:T}, y(x_{1:T})\}$ consisting of T total pairs of object-proposal coordinates x in the image space and response signals y, respectively. We wish to estimate the posterior distribution $p(f|D_{1:T})$ of the objective function given these data: $p(f|D_{1:T}) \propto p(D_{1:T}|f)p(f)$. This simple formula allows us to iteratively update the posterior over the signal as we acquire new data.

A GP for regression defines a distribution over functions with a joint Normality assumption. We denote f, the realization of the Gaussian process:

$$f \sim GP(m, k) \qquad (1)$$

Here the GP is fully specified by the mean m and covariance k. A common kernel function that obeys suitable continuity characteristics for the GP realization is the squared-exponential kernel, which we use here:

$$k(x, x') = \sigma_f^2 exp\left[-\frac{1}{2l^2}\|x - x'\|^2\right] + \sigma_\varepsilon^2 \delta_{xx'} \qquad (2)$$

where σ_f^2 is the variance of the GP realization, which we set heuristically; σ_ε^2 is the variance of the ε parameter that we estimate empirically; and $\delta_{xx'}$ is the *Kroenecker delta function* which is equal to 1 if and only if $x = x'$ and is equal to zero otherwise. GPs are particularly sensitive to the choice of the length-scale/bandwidth parameter l, which we

---

[1] We use the Euclidean distance between the centers of two bounding boxes, scaled by the square root of the area of the image for the measure of "normalized offset distance."

optimize with grid search for the reduced log marginal likelihood (see [18] for additional details).

The posterior predictive of the surrogate function for a new datum x_* is given by [2]:

$$p(f_*|x_*, X, y) = N(f_*|k_*^T K_\sigma^{-1} y, k_{**} - k_*^T K_\sigma^{-1} k_*)\,, \qquad (3)$$

where X is the data matrix (all prior observations x), $k_* = [k(x_*, x_1), \dots, k(x_*, x_T)], k_{**} = k(x_*, x_*)$ and $K_\sigma = K + \sigma_y^2 I_T$, where $K = k(x_i, x_j), 1 \le i, j \le T$.

For our algorithm, we compute posterior predictive updates using equation (3) in batch iterations (see Section 4.2). At each iteration, the realization of the GP is calculated over a grid of size M corresponding with the image space domain of the object localization process. This grid size can be chosen to match a desired granularity/computational overhead tradeoff.

Considering equation (3) further, we note that posterior predictive updates entail a one-time (per iteration) inversion of the matrix K_σ, requiring $O(T^3)$ operations, where T is the number of calls to the offset-prediction model. Naturally, choosing information-rich bounding-box proposals (see Section 3.3) will improve the efficiency of the localization process and thus keep T reasonably small in general. To this end, we furthermore incorporate a "short memory" mechanism in our algorithm so that older proposal query values, which convey less information pertinent to the current localization search, are "forgotten" (see Section 4). For improved numerical stability, we apply a Cholesky decomposition prior to matrix inversion [18].

3.3 Bayesian Optimization for Active Search

In the framework of Bayesian optimization, acquisition functions are used to guide the search for the optimum of the surrogate (which approximates the true objective function). Intuitively, acquisition functions are defined in such a way that *high acquisition* indicates greater likelihood of an objective function optimum. Most commonly, acquisition functions encapsulate a data query experimental design that favors either regions of large signal response, large uncertainty, or a combination of both.

One can formally express the *utility* of a Bayesian optimization procedure with GP parameter θ, observations $\{y\}$, and acquisition function instantiated by $a(\xi)$ with design parameter $\xi \geq 0$, as the information gained when we update our prior belief $p(\theta|a(\xi))$ to the posterior, $p(\theta|y, a(\xi))$, after having acquired a new observation [2].

At each iteration of our algorithm the acquisition function, defined below, is maximized to determine where to sample from the objective function (i.e., the response signal value) next. The acquisition function incorporates the mean and variance of the predictions over the image space to model the utility of sampling [2]. We then evaluate the objective function at these maximal points and the Gaussian process is updated appropriately. This procedure is iterated until the stopping condition is achieved.

A standard acquisition function used in applications of Bayesian optimization is the *Expected Improvement* (EI) function [23]. We define a dynamic variant of EI that we call *Confidence-EI* (CEI) that better accommodates our problem setting:

$$a_{CEI}(x, \xi) \triangleq \begin{cases} (\mu(x) - f(x^+) - \xi)\Phi(Z) + \sigma(x)\varphi(Z) \\ Z = \dfrac{\mu(x) - f(x^+) - \xi}{\sigma(x)} \end{cases} \qquad (4)$$

In equation (4), $f(x^+)$ represents the incumbent maximum of the surrogate function, $\mu(x)$ is the mean of the surrogate at the input point x in the image space, $\sigma(x) > 0$ is the standard deviation of the surrogate at the input; $\varphi(\cdot)$ and $\Phi(\cdot)$ are the *pdf* and *cdf* of the Gaussian distribution, respectively; and ξ is the dynamically-assigned design parameter. The design parameter controls the exploration/exploitation tradeoff for the Bayesian optimization procedure; if, for instance, we set $\xi = 0$, then EI performs greedily.

For our algorithm, we let ξ vary over the course of localization run by defining it as a function of a per-iteration *total confidence score*. Lizotte [13] showed that varying the design parameter can improve performance for Bayesian optimization. With each iteration of localization, we set the current total confidence value equal to the median of the response signal for the current batch of bounding-box proposals. In this way, high confidence disposes the search to be greedy and conversely low confidence encourages exploration.

4. Algorithm and Experimental Results

4.1 Dataset

Following [17] and [20], in the current study we use a subset consisting of single pedestrian instances from the Portland State Dog-Walking Images for our proof of concept and comparative experiments [30]. This subset contains 460 high-resolution annotated photographs, taken in a variety of locations. Each image is an instance of a "Dog-Walking" visual situation in a natural setting containing visible pedestrians. Quinn et al. [17] used this dataset to demonstrate the utility of applying prior situation knowledge and active, context-directed search in a structured visual situation for efficient object localization. This dataset represents a challenging benchmark for pedestrian localization refinement, due to its high degree of variability.

4.2 GPLR Algorithm

Below we present details of the Gaussian Process Localization Refinement (GPLR) algorithm. To begin, we randomly set aside 400 images from our dataset for training and 60 for testing. We train the prediction model, *y*, using features computed by the pre-trained imagenet-vgg-f network in MatConvNet [29]. The features we use are from the last fully-connected layer, which yields feature vectors of dimension 4096. For training, we generated 100k offset crops of pedestrians from the training images. In addition, we fit a log-Normal distribution $p(\cdot)_{w,h}$ for width and height parameters of the pedestrian object-proposal bounding boxes over the training set, to serve as a general prior for target bounding box size. We optimize the hyperparameter θ for the Gaussian process using grid search. The design parameter ξ is set as a function of the per-step total. Lastly, we set the size of the GP realization, $M = 500^2$ (i.e., the realization occurs over a 500x500 grid). We found that this size achieved a suitable balance between localization precision and computational overhead.

In order to simulate bounding boxes generated by a detection algorithm (e.g. R-CNN), we begin by randomly generating a set of misaligned bounding-box proposals of size $n_0 = 10$. We then use our trained model to compute response signal values for this proposal set, yielding $D^{(0)}_{proposal}$. At each subsequent step of the GPLR algorithm we generate a GP realization using the proposal set (step 3). To find the next batch (*n* = 5) of proposals, we use the top-*n* ranked points in the space, ranked using the CEI acquisition function defined in Section 3.3. We then augment the proposal set with this new batch of points and the previous generations of proposals specified by the GP_{mem} parameter, which indicates the number of batches contained in the algorithm "memory" (steps 9 and 10). For our experiments, we set GP_{mem}= 3 with *T* = 10, for a total of 50 proposals per execution of GPLR.

Algorithm: Gaussian Process Localization Refinement (GPLR)

Input: Image *I*, trained model *y* giving response signals, a set of n_0 initial, misaligned bounding-box proposals and response signal values: $D_{n_0} = \left\{\left(x_i, y(x_i)\right)_{i=1}^{n_0}\right\}$, GP hyperparameters θ, size of GP realization space *M*, dynamic design parameter for Bayesian active search ξ, learned prior distribution for bounding-box size parameters $(w_i, h_i) \sim p(\cdot)_{w,h}$, size of GP memory GP_{mem} (as number of generations used), batch size *n*, number of iterations *T*, current set of bounding-box proposals and response signals $D^{(t)}_{proposal}$.

```
1: $D^{(0)}_{proposal} \longleftarrow D_{n_0}$
2: for $t = 1$ to $T$ do
3:   Compute $\mu(x)^{(t)}$ and $\sigma(x)^{(t)}$ for the GP realization
       $f_M^{(t)}$ of $D^{(t-1)}_{proposal}$ over grid of $M$ points (Equation 3)
4:   for $i = 1$ to $n$ do
5:     $z_i = \operatorname{argmax}_x a_{CEI}\left(f_M^{(t)} \setminus \{z_j\}_{j=1}^{j=i-1}, \xi\right)$ (Equation 4)
6:     $sample: (w_i, h_i) \sim p(\cdot)_{w,h}$
7:     $p_i = (z_i, w_i, h_i)$
8:   end for
9:   $D^{(t)} \longleftarrow \left\{\left(x_i, y(x_i)\right)_{i=1}^{n}\right\}$
10:  $D^{(t)}_{proposal} \longleftarrow \bigcup_{j=t-GP_{mem}}^{t} D^{(j)}$
11: end for
12: Return $\operatorname{argmax}_x \mu(x)^{(T)}$
```

4.3 Experimental Results

We evaluate the GPLR algorithm described in Section 4.1 in comparison with the benchmark bounding-box regression model used in Faster R-CNN [19] for the task of refining object proposals. Both the GP and bounding-box regression models were trained with 100k offset image crops taken from the test image set. For the bounding-box regression trials, the algorithm receives a randomized offset crop in the IOU range [0, .4], and then outputs a refined bounding box. In the case of GPLR, the algorithm is initialized with a small set ($n_0 = 10$) of inaccurate proposals in the same range; the median IOU of this initial set of proposal bounding-boxes was .12 for the experimental trials. The output of the GPLR algorithm is a single refined bounding box, as in the case of the regression model. In each case, we compare the final refined bounding-box with the ground truth for the target object. In total, we tested each method for 400 experimental trials, including multiple runs with different random initializations on test images.

Girshick et al. [9] thresholded their training regime for localization with bounding-box regression at large bounding-box overlap (IOU ≥ 0.6). To comprehensively test our method against bounding-box regression (BB-R), we trained two distinct regression models: one with IOU thresholded for training at 0.6, as used with R-CNN, and one with IOU thresholded at 0.1.

Results for our experiments are summarized in Table 1 and Figure 4. We report the median and standard error (SE) for IOU difference (final – initial), the median relative IOU improvement (final – initial) / initial, the total percentage of the test data for which the method yielded an IOU improvement, in addition to the total percentage of test data for which the target was successfully localized (i.e., final IOU ≥ 0.5).

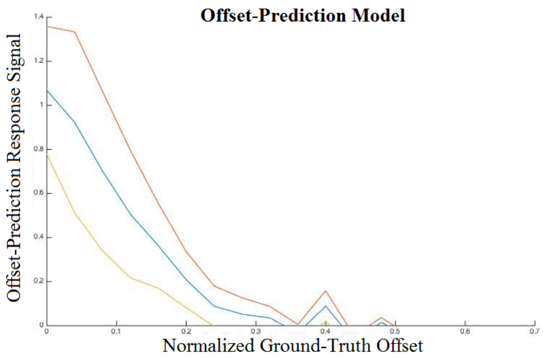

Figure 2. Performance of the offset-prediction model on test data (n = 1000 offset image crops). The mean (center curve) and +/−1 standard deviations (outer curves) are shown. As desired, the response signal yields a Gaussian-like peak around the center of the target object bounding-box (i.e., zero ground-truth offset). The bumps present in the range of values above 0.35 offset from the ground truth is indicative of noisy model outputs when offset crops contain no overlap with the target object. (Figure is best viewed in color.)

Method	IOU Difference Median (SE)	Median Relative IOU Improvement	% of Test Set with IOU Improvement	% of Test Set Localized
BB-R (0.6)	.0614 (.0035)	34.62%	**90.1%**	12.3%
BB-R (0.1)	.1866 (.0077)	92.91%	90.0%	33.2%
GPLR	**.4742** (.012)	**194.02%**	89.3%	**75.2%**

Table 1. Summary statistics for the pedestrian localization task. BB-R (0.6) indicates the bounding-box regression model with training thresholded at initial IOU 0.6 and above; BB-R (0.1) denotes the bounding-box regression model with training thresholded at initial IOU 0.1 and above; GPLR denotes Gaussian Process Localization Refinement.

4.4 Discussion

Our experimental results are strongly favorable for the GPLR algorithm. Using only a small number of total bounding box proposals (50) per trial, GPLR performed comparably with BB-R for percentage of test images for which the IOU improved. In addition, GPLR significantly outperformed BB-R for all other localization metrics, including the percentage of test set images achieving successful localization and the median relative IOU improvement.

During our experimental trials, we discovered a substantial disparity in performance for BB-R depending on the training regime. In general, BB-R (0.6), as used in R-CNN, yielded inferior localization results in general when compared to BB-R (0.1) (see Table 1). In particular, BB-R (0.1) was much stronger for low initial IOU values than BB-R (0.6). However, as initial IOU increased, localization results deteriorated starkly with BB-R (0.1) due to overfitting. For larger initial IOU values (e.g., IOU > 0.4), BB-R (0.1) yielded IOU improvement on only 22.1% of the experimental trials; when the IOU threshold was increased to 0.5 this IOU improvement percentage dropped even further to 13.0%. In contrast, GPLR indicated no signs of deterioration in localization performance when given initial offset proposals with a large IOU. For separate test runs of 100 trials each, GPLR achieved an IOU improvement on 98% of the trials (for median initial IOU > 0.4) and an IOU improvement on 100% of the trials (for median initial IOU > 0.5).

In addition to this strong experimental performance, GPLR provides several broad methodological advantages over previous techniques, particularly in applications requiring fast and precise object localization. Most importantly, by working within a Bayesian framework, GPLR is able to perform an efficient, active search by "learning" continuously from its response signal at each step of the algorithm. Because GPLR renders both the mean and standard deviation for the predictive posterior, the GPLR model maintains a measure of uncertainty that can be applied in systems as a potential (early) stopping condition when real-world resources are limited (e.g. robotics, video tracking using Kalman filters).

5. Conclusion and Future Work

We have presented a novel technique for the challenging task of the efficient refinement of object proposals. Our method trains a predicted-offset model, demonstrating successfully the ability of CNN-based features to serve as the input for an object localization method. Using Bayesian optimization, we surpass the state-of-the-art regression method employed in R-CNN (and its extensions) for the localization refinement of pedestrian object proposals with computational efficiency.

With future research, we plan to extend our approach to massively scalable GPs, so that our model can directly incorporate bounding-box size parameters, leverage visual context for localization and search for multiple target objects simultaneously.

Our work indicates the strong promise of applying the Bayesian paradigm to the outstanding goal of computer vision: real-time object detection.

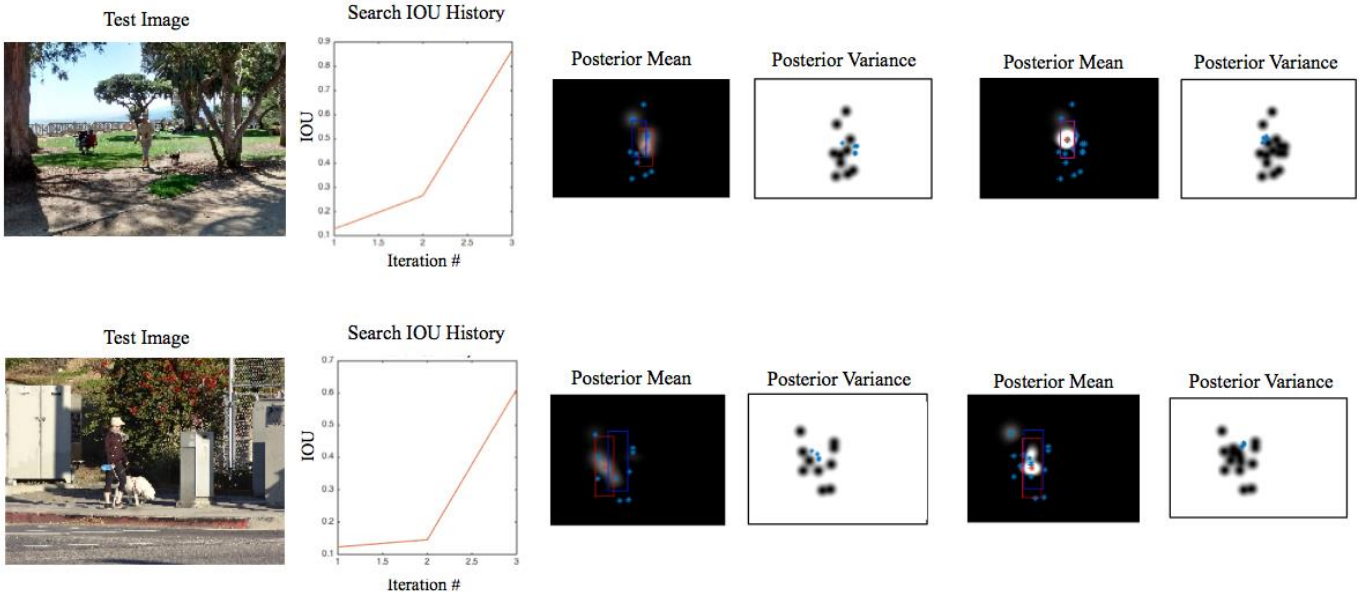

Figure 3. Examples of runs on two test images with the GPLR algorithm. In each row the test image is shown on the far-left; the "search IOU history" is displayed in the second column, with the algorithm iteration number on the horizontal axis and IOU with the ground-truth target bounding box on the vertical axis. The remaining columns present the GPLR response surface for the posterior mean and variance; the first pair of boxes reflect the second iteration of the algorithm and the last pair show the third iteration of the algorithm. In each case localization occurs rapidly thus requiring a very small number of proposals.

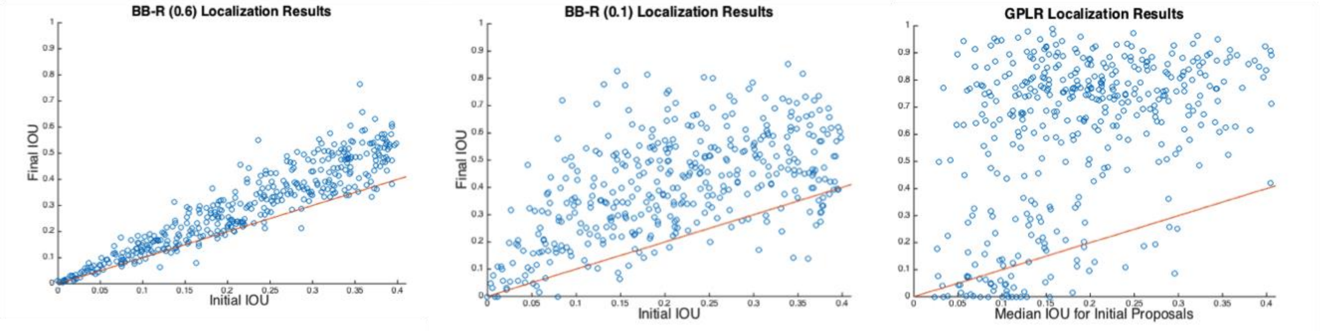

Figure 4. Graph of BB-R (0.6), BB-R (0.1) and GPLR localization results for test images. The horizontal axis indicates the median IOU for the initial proposal bounding boxes, while the vertical axis designates the final IOU with the target object ground truth. The line depicted indicates "break-even" results.

References

[1] D. Bhattacharjee, A. Paul, J.H. Kim, M. Kim, An object localization optimization technique in medical images using plant growth simulation algorithm., Springerplus. 5 2016 1784.

[2] E. Brochu, V.M. Cora, N. De Freitas, A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning, ArXiv:1012.2599. (2010).

[3] M. Carpin, S. Rosati, M.E. Khan, B. Rimoldi, UAVs using Bayesian Optimization to Locate WiFi Devices, ArXiv. (2006) 1–5.

[4] G.C.H.E. de Croon, E.O. Postma, H.J. van den Herik, Adaptive gaze control for object detection., Cognit. Comput. 3 (2011) 264–278.

[5] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: 2005 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., IEEE, n.d.: pp. 886–893.

[6] M. Everingham, L. Gool, C.K.I. Williams, J. Winn, A. Zisserman, The Pascal visual object classes (VOC) challenge, Int. J. Comput. Vis. 88 (2010) 303–338.

[7] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The KITTI vision benchmark suite, in: 2012 IEEE Conf. Comput. Vis. Pattern Recognit., IEEE, 2012: pp. 3354–3361.

[8] R. Girshick, Fast R-CNN, in: Int. Conf. Comput. Vis., IEEE, 2015: pp. 1440–1448.

[9] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Conf. Comput. Vis. Pattern Recognit., IEEE, 2014: pp. 580–587.

[10] A. Gonzalez-Garcia, A. Vezhnevets, V. Ferrari, An active search strategy for efficient object class detection, in: Conf. Comput. Vis. Pattern Recognit., IEEE, 2015: pp. 3022–3031.

[11] D. Hoiem, Y. Chodpathumwan, Q. Dai, Diagnosing Error in Object Detectors, in: Springer, Berlin, Heidelberg, 2012: pp. 340–353.

[12] P. Langley, Artificial intelligence and cognitive systems, AISB Q. (2011).

[13] D.J. Lizotte, D. James, Practical Bayesian optimization., University of Alberta, 2009.

[14] Y. Lu, T. Javidi, S. Lazebnik, Adaptive object detection using adjacency and zoom prediction, arXiv:1512.07711. (2015).

[15] A. Morales, T. Asfour, P. Azad, S. Knoop, R. Dillmann, Integrated Grasp Planning and Visual Object Localization For a Humanoid Robot with Five-Fingered Hands, in: 2006 IEEE/RSJ Int. Conf. Intell. Robot. Syst., IEEE, 2006: pp. 5663–5668.

[16] M. Neider, G. Zelinsky, Scene context guides eye movements during visual search, Vision Res. 46 (2006) 614–621.

[17] M. Quinn, M. H., Rhodes, A. D., Mitchell, Active object localization in visual situations, arXiv:1607.00548. (2016).

[18] C.E. Rasmussen, Gaussian processes in machine learning, Adv. Lect. Mach. Learn. 3176 (2004) 63–71.

[19] J. Ren, S., He, K., Girshic, R., Sun, Faster R-CNN: Towards real-time object detection with region proposals, in: Adv. Neural Inf. Process. Syst. 28 (NIPS 2015), 2015.

[20] A.D. Rhodes, M.H. Quinn, M. Mitchell, Fast On-Line Kernel Density Estimation for Active Object Localization, 2016.

[21] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks, (2013).

[22] P. Sermanet, K. Kavukcuoglu, S. Chintala, Y. Lecun, Pedestrian Detection with Unsupervised Multi-stage Feature Learning, (2013) 3626–3633.

[23] J. Snoek, H. Larochelle, R.P. Adams, Practical Bayesian Optimization of Machine Learning Algorithms, (2012) 2951–2959.

[24] C. Szegedy, S. Reed, D. Erhan, D. Anguelov, Scalable, high-quality object detection, arXiv:1412.1441. (2014).

[25] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, A.W.M. Smeulders, Selective Search for Object Recognition, Int. J. Comput. Vis. 104 (2013) 154–171.

[26] J. Vanhatalo, J.V. Fi, A. Vehtari, N. Lawrence, A. Schwaighofer, J. Quiñonero-Candela, Sparse Log Gaussian Processes via MCMC for Spatial Epidemiology, JMLR Work. Conf. Proc. 1 (n.d.) 73–89.

[27] S. Zhang, R. Benenson, M. Omran, J. Hosang, B. Schiele, How Far are We from Solving Pedestrian Detection?, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016: pp. 1259–1267.

[28] Y. Zhang, K. Sohn, R. Villegas, G. Pan, H. Lee, Improving Object Detection With Deep Convolutional Networks via Bayesian Optimization and Structured Prediction, (2015) 249–258.

[29] Imagenet-VGG-f, http://www.v1feat.org/matconvnet/pretrained/

[30] Portland Dog-Walking Dataset, http://web.cecs.pdx.edu/~mm/PortlandStateDogWalkingImages.html