

Sentence Directed Video Object Codetection

Haonan Yu, *Student Member, IEEE* and Jeffrey Mark Siskind, *Senior Member, IEEE*

Abstract—We tackle the problem of video object codetection by leveraging weak semantic constraints implied by sentences that describe the video content. Unlike most existing work that focuses on codetecting large objects which are usually salient both in size and appearance, we can codetect objects that are small or medium. Also our method assumes no human pose or depth information which is exploited by the most recent state-of-the-art method. We employ weak semantic constraints on the codetection process by pairing the video with sentences. Although the semantic information is usually simple and weak, it can greatly boost the performance of our codetection framework by reducing the search space of the hypothesized object detections. Our experiment demonstrates an average IoU score of 0.43 on a new challenging dataset which contains 15 object classes and 150 videos with 12,509 frames in total.

Index Terms—video, object codetection, sentences

1 INTRODUCTION

In this paper, we address the problem of codetecting objects with bounding boxes from a set of videos, *without* any pretrained object detectors. The codetection problem is typically approached by selecting one out of many object proposals per image or frame that maximizes a combination of the confidence scores associated with the selected proposals and the similarity scores between proposal pairs. While much prior work focuses on codetecting objects in still images (*e.g.*, [7, 21, 29, 33]), little prior work [20, 27, 32] attempts to codetect objects in video. In both lines of work, most [7, 20, 21, 27, 29, 33] assume that the objects to be codetected are salient, both in size and appearance, and located in the center of the field of view. Thus they easily “pop out.” As a result, prior methods succeed with a small number of object proposals in each image or frame. Tang et al. [33] and Joulin et al. [20] used approximately 10 to 20 proposals per image, while Lee and Grauman [21] used 50 proposals per image. Limiting codetection to objects in the center of the field of view allowed Prest et al. [27] to prune the search space by penalizing proposals in contact with the image perimeter. Moreover, under these constraints, the confidence score associated with proposals is a reliable measure of salience and a good indicator of which image regions constitute potential objects [29]. In prior work, the proposal confidence dominates the overall scoring process and the similarity measure only serves to refine the confidence. In contrast, Srikantha and Gall [32] attempt to codetect small to medium sized objects in video, without the above simplifying assumptions. However, in order to search through the larger resulting object proposal space, they avail themselves of human pose and depth information to prune the search space. It should also be noted that all these codetection methods, whether for images or video, codetect only one common object at a time: different object classes are codetected independently.

The confidence score of a proposal can be a poor indicator of whether a proposal denotes a salient object, especially

when objects are occluded, the lighting is poor, or motion blur exists (*e.g.*, see Figure 1). Salient objects can have low confidence scores while nonsalient objects or image regions that do not correspond to objects can have high confidence scores. Thus our scoring function does not use the confidence scores produced by the proposal generation mechanism. Moreover, our method does not rely on human pose and depth information, which is not always available. Human pose can be difficult to estimate reliably when a person is only partially visible or is self-occluded [3], as is the case with most of our videos.

We avail ourselves of a different source of constraint on the codetection problem. In videos depicting human interaction with objects to be codetected, descriptions of such activity can impart *weak* spatial or motion constraint either on a single object or among multiple objects of interest. For example, if the video depicts a “pick up” event, some object should have an upward displacement during this process, which should be detectable even if it is small. This motion constraint will reliably differentiate the object which is being picked up from other stationary background objects. It is weak because it might not totally resolve the ambiguity; other image regions might satisfy this constraint, perhaps due to noise. Similarly, if we know object *A* is on the left of object *B*, then the detection search for object *A* will weakly affect the detection search for object *B*, and vice versa. To this end, we extract spatio-temporal constraints from *sentences* that describe the videos and then impose these constraints on the codetection process to find the most salient collections of objects that satisfy these constraints. Even though the constraints implied by a single sentence are usually weak, when accumulated across a set of videos and sentences, they together will greatly prune the detection search space. We call this process *sentence directed* video object codetection. It can be viewed as the *inverse* of video captioning/description [5, 13, 15] where object evidence (detections or other visual features) is first produced by pretrained detectors and then sentences are generated given the object appearance and movement.

Generally speaking, we extract a set of predicates from each sentence and formulate each predicate around a

Both H. Yu and J. M. Siskind are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA.
E-mail: haonan@haonanyu.com, qobi@purdue.edu
Manuscript received

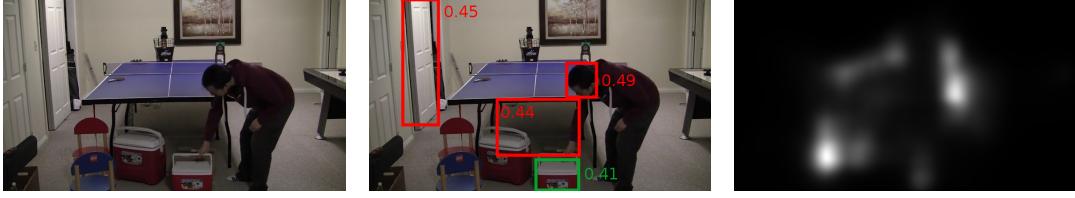


Fig. 1. Object proposal confidence scores and saliency scores for a sample frame from our dataset. Left: the original input video frame. Middle: several proposals and associated confidence scores produced by the method of Arbelaez et al. [4]. Note that the red boxes, which do not correspond to objects, let alone salient ones, all have higher scores than the green box, which does denote a salient object. Right: the saliency map output by the saliency detection method of Jiang et al. [19], currently highest ranking method on the MIT saliency benchmark [10]. Note that the cooler is not highlighted as salient. Using these scores as part of the scoring function can drive the codetection process to produce undesired results.

set of primitive *functions*. The predicates may be verbs (e.g., CARRIED and ROTATED), spatial-relation prepositions (e.g., TOTHELEFTOF and ABOVE), motion prepositions (e.g., AWAYFROM and TOWARDS), or adverbs (e.g., QUICKLY and SLOWLY). The sentential predicates are applied to the candidate object proposals as arguments, allowing an overall predicate score to be computed that indicates how well these candidate object proposals satisfy the sentence semantics. We add this predicate score into the codetection framework, on top of the original similarity score, to guide the optimization. To the best of our knowledge, this is the first work that uses sentences to guide generic video object codetection. To summarize, our approach differs from the indicated prior work in the following ways:

- (a) Our method can codetect small or medium sized non-salient objects which can be located anywhere in the field of view.
- (b) Our method does not require assume human pose or depth information.
- (c) Our method can codetect multiple objects simultaneously. These objects can be either moving in the foreground or stationary in the background.
- (d) Our method allows fast object movement and motion blur. Such is not exhibited in prior work.
- (e) Our method leverages sentence semantics to help codetection.

We evaluate our approach on a new dataset that contains 15 distinct object classes and 150 video clips with a total of 12,509 frames. Our approach achieves an IoU (Intersection-over-Union) score of 0.43 and a detection accuracy of 0.7 to 0.8 (when the IoU threshold is 0.3 to 0.4).

2 RELATED WORK

Corecognition is a simpler variant of codetection [34], where the objects of interest are sufficiently prominent in the field of view that the problem does not require object localization. Thus corecognition operates like unsupervised clustering, using feature extraction and the similarity measure. Codetection [7, 21, 33] additionally requires localization, often by putting bounding boxes around the objects. This can require combinatorial search over a large space of possible object locations. One way to remedy this is to limit the space of possible object locations to those produced by an object proposal method [1, 4, 11, 35]. These methods typically associate a confidence score with each proposal which can be used to prune or prioritize the search. Codetection is typically formulated as the process of selecting one proposal per

image or frame, out of the many produced by the proposal mechanism, that maximizes the collective confidence of and similarity between the selected proposals. This optimization is usually performed with Belief Propagation [26] or with nonlinear programming. Recently, the codetection problem has been extended to video [20, 27, 28, 30, 32]. Like Srikantha and Gall [32], we codetect small and medium objects, but do so without using human pose or a depth map. Like Schulter et al. [30], we codetect both moving and stationary objects, but do so with a larger set of object classes and a larger video corpus. Also like Ramanathan et al. [28], we use sentences to guide video codetection, but do so for a vocabulary that goes beyond pronouns, nominals, and names that are used to codetect only human face tracks.

Another line of work learns visual structures or models from image captions [6, 16–18, 24, 25], treating the input as a parallel image-text dataset. Since this work focuses on images and not video, the sentential captions only contain static concepts such as the names of people or the spatial relations between objects in the images.

3 SENTENCE DIRECTED CODETECTION

Our sentence-directed codetection approach is illustrated in Figure 2. The input is a set of videos paired with human-elicited sentences, one sentence per video. A collection of object-candidate generators and video-tracking methods are applied to each video to obtain a pool of object proposals.¹ Object instances and predicates are extracted from the paired sentence. Given multiple such video-sentence pairs, a graph is formed where object instances serve as vertices and similarities between object instances and predicates linking object instances in a sentence serve as edges. Finally, Belief Propagation is applied to this graph to jointly infer object codetections.

3.1 Sentence Semantics

Our main contribution is exploiting sentence semantics to help the codetection process. We use a conjunction of predicates to represent (a portion of) the semantics of a sentence. Object instances in a sentence fill the arguments of the predicates in that sentence. An object instance that fills the arguments of multiple predicates is said to be *coreferenced*. For a coreferenced object instance, only one track is

¹ For clarity, in the remainder of this paper, we refer to object proposals for a single frame as object candidates, while we refer to object tubes or tracks across a video as object proposals.

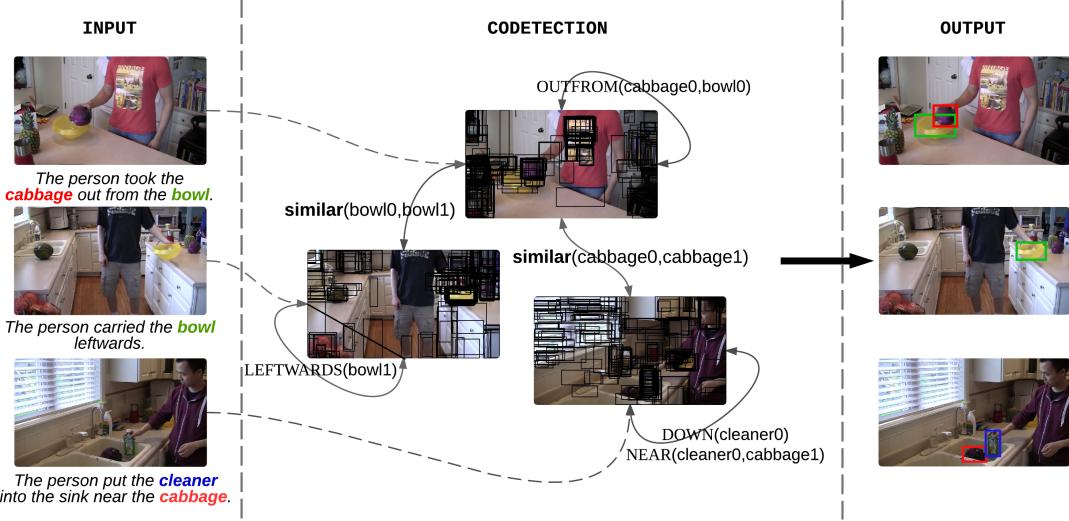


Fig. 2. An overview of our codetection process. Left: input a set of videos paired with sentences. Middle: sentence directed codetection, where black bounding boxes represent object proposals. Right: output original videos with objects codetected. Note that no pretrained object detectors are used in this whole process. Also note how sentence semantics plays an important role in this process: it provides both unary scores, e.g., LEFTWARDS(*bowl*) and DOWN(*cleaner*), for proposal confidence, and *n*-ary scores, e.g., OUTFROM(*cabbage,bowl*) and NEAR(*cleaner,cabbage*), for relating multiple objects in the same video. (Best viewed in color.)

codetected. For example, a sentence like “*the person put the cleaner into the sink near the cabbage*” implies the following conjunction of predicates:

$$\text{DOWN}(\text{cleaner}) \wedge \text{NEAR}(\text{cleaner}, \text{cabbage})$$

In this case, *cleaner* is coreferenced by the predicates DOWN (fills the sole argument) and NEAR (fills the first argument). Thus only one *cleaner* track will be produced, simultaneously constrained by the two predicates (Figure 2, blue track).

Following Lin et al. [22], our procedure for extracting predicates from a sentence consists of two steps: parsing and transforming/distilling. We first use the Stanford parser [31] to obtain a parse tree of the sentence. We then employ a set of rules to transform the parsed results to ones that are better suited to visual analysis, removing those that are not relevant to our purposes. For example, for the phrase “*into the sink*” in the above sentence, it is difficult (and also beyond our interest) to actually detect the object *sink* and impose the constraint INTO(*cleaner,sink*). Thus we replace this predicate with just DOWN(*cleaner*) by our predefined rules. Also, although the semantic representation for this sentence will include PUT(*person,cleaner*), we simplify this two-argument predicate to a one-argument predicate MOVE(*cleaner*) since we do not attempt to codetect people. To ensure that we do not introduce surplus semantics, a transformed/distilled predicate always implies a *weaker* constraint than the original one.

Each predicate is formulated around a set of primitive functions on the arguments of the predicate. The primitive functions produce scores indicating how well the arguments satisfy the constraint. The aggregated score over the functions constitutes the predicate score. Table 1 shows the complete list of our 24 predicates and the scores they compute. The function medFIMg(p) computes the median of the averaged optical flow magnitudes within the detections for the proposal p . The functions $x(p^{(t)})$

and $y(p^{(t)})$ return the x - and y -coordinates of the center of $p^{(t)}$, normalized by the frame width and height, respectively. The function distLessThan(x, a) is defined as log sigmoid $\log[1/(1 + \exp(-b(x - a)))]$, where we set $b = -20$ in the experiment. Similarly, the function distGreaterThan(x, a) is defined as distLessThan($-x, -a$). The function dist($p_1^{(t)}, p_2^{(t)}$) computes the distance between the centers of $p_1^{(t)}$ and $p_2^{(t)}$, also normalized by the frame size. The function smaller($p_1^{(t)}, p_2^{(t)}$) returns 0 if the size of $p_1^{(t)}$ is smaller than that of $p_2^{(t)}$, and $-\infty$ otherwise. The function tempCoher(p) evaluates whether the position of proposal p changes during the video, by checking the position offsets between every two frames. A higher tempCoher score indicates that p is more likely to be stationary in the video. The function rotAngle($p^{(t)}$) computes the current rotated angle of the object inside $p^{(t)}$ by looking back 1 second (30 frames). We extract SIFT features [23] for both $p^{(t)}$ and $p^{(t-30)}$ and match them to estimate the similarity transformation matrix, from which the angle can be computed. Finally the function hasRotation(α, β) computes the rotation log-likelihood given angle α through the von Mises distribution for which we set the location $\mu = \beta$ and the concentration $\kappa = 4$.

3.2 Generating Object Proposals

We first generate N object candidates for each video frame. We use EdgeBoxes [35] to obtain the $\frac{N}{2}$ top-ranking object candidates and MCG [4] to obtain the other half, filtering out candidates larger than $\frac{1}{20}$ of the video-frame size to focus on small and medium sized objects. This yields NT object candidates for a video with T frames. We then generate K object proposals from these NT candidates. To obtain object proposals with object candidates of consistent appearance and spatial location, one would nominally require that $K \ll NT$. To circumvent this, we first randomly sample a frame t from the video with probability proportional to the

$\Delta \text{DISTLARGE} \triangleq 0.25$
$\Delta \text{DISTSMALL} \triangleq 0.05$
$\Delta \text{ANGLE} \triangleq \pi/2$
$\text{MOVE}(p) \triangleq \text{medFIMG}(p)$
$\text{MOVEUP}(p) \triangleq \text{MOVE}(p) + \text{distLessThan}\left(y(p^{(T)}) - y(p^{(1)}), -\Delta \text{DISTLARGE}\right)$
$\text{MOVEDOWN}(p) \triangleq \text{MOVE}(p) + \text{distGreaterThan}\left(y(p^{(T)}) - y(p^{(1)}), \Delta \text{DISTLARGE}\right)$
$\text{MOVEHORIZONTAL}(p) \triangleq \text{MOVE}(p) + \text{distGreaterThan}\left(x(p^{(T)}) - x(p^{(1)}) , \Delta \text{DISTLARGE}\right)$
$\text{MOVELEFTWARDS}(p) \triangleq \text{MOVE}(p) + \text{distLessThan}\left(x(p^{(T)}) - x(p^{(1)}), -\Delta \text{DISTLARGE}\right)$
$\text{Moverightwards}(p) \triangleq \text{MOVE}(p) + \text{distGreaterThan}\left(x(p^{(T)}) - x(p^{(1)}), \Delta \text{DISTLARGE}\right)$
$\text{ROTATE}(p) \triangleq \text{MOVE}(p) + \max_i \text{hasRotation}\left(\text{rotAngle}(p_i^{(t)}), \Delta \text{ANGLE}\right)$
$\text{TOWARDS}(p_1, p_2) \triangleq \text{MOVE}(p_1) + \text{distLessThan}\left(\text{dist}(p_1^{(T)}, p_2^{(T)}) - \text{dist}(p_1^{(1)}, p_2^{(1)}), -\Delta \text{DISTLARGE}\right)$
$\text{AWAYFROM}(p_1, p_2) \triangleq \text{MOVE}(p_1) + \text{distGreaterThan}\left(\text{dist}(p_1^{(T)}, p_2^{(T)}) - \text{dist}(p_1^{(1)}, p_2^{(1)}), \Delta \text{DISTLARGE}\right)$
$\text{LEFTOFSTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distLessThan}\left(x(p_1^{(1)}) - x(p_2^{(1)}), -\Delta \text{DISTSMALL}\right)$
$\text{LEFTOFEEND}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distLessThan}\left(x(p_1^{(T)}) - x(p_2^{(T)}), -\Delta \text{DISTSMALL}\right)$
$\text{RIGHTOFSTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distGreaterThan}\left(x(p_1^{(1)}) - x(p_2^{(1)}), \Delta \text{DISTSMALL}\right)$
$\text{RIGHTOFEEND}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distGreaterThan}\left(x(p_1^{(T)}) - x(p_2^{(T)}), \Delta \text{DISTSMALL}\right)$
$\text{ONTOPOFSTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2)$
$+ \text{distGreaterThan}\left(y(p_1^{(1)}) - y(p_2^{(1)}), -2\Delta \text{DISTLARGE}\right)$
$+ \text{distLessThan}\left(y(p_1^{(1)}) - y(p_2^{(1)}), 0\right)$
$+ \text{distLessThan}\left(x(p_1^{(1)}) - x(p_2^{(1)}) , 2\Delta \text{DISTSMALL}\right)$
$\text{ONTOPOFEEND}(p_1, p_2) \triangleq \text{tempCoher}(p_2)$
$+ \text{distGreaterThan}\left(y(p_1^{(T)}) - y(p_2^{(T)}), -2\Delta \text{DISTLARGE}\right)$
$+ \text{distLessThan}\left(y(p_1^{(T)}) - y(p_2^{(T)}), 0\right)$
$+ \text{distLessThan}\left(x(p_1^{(T)}) - x(p_2^{(T)}) , 2\Delta \text{DISTSMALL}\right)$
$\text{NEARSTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distLessThan}\left(\text{dist}(p_1^{(1)}, p_2^{(1)}), 2\Delta \text{DISTSMALL}\right)$
$\text{NEAREND}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distLessThan}\left(\text{dist}(p_1^{(T)}, p_2^{(T)}), 2\Delta \text{DISTSMALL}\right)$
$\text{INSTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{NEARSTART}(p_1, p_2) + \text{smaller}(p_1^{(1)}, p_2^{(1)})$
$\text{INEND}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{NEAREND}(p_1, p_2) + \text{smaller}(p_1^{(T)}, p_2^{(T)})$
$\text{BELOWSTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distGreaterThan}\left(y(p_1^{(1)}) - y(p_2^{(1)}), \Delta \text{DISTSMALL}\right)$
$\text{BELOWEND}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distGreaterThan}\left(y(p_1^{(T)}) - y(p_2^{(T)}), \Delta \text{DISTSMALL}\right)$
$\text{ABOVESTART}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distLessThan}\left(y(p_1^{(1)}) - y(p_2^{(1)}), -\Delta \text{DISTSMALL}\right)$
$\text{ABOVEEND}(p_1, p_2) \triangleq \text{tempCoher}(p_2) + \text{distLessThan}\left(y(p_1^{(T)}) - y(p_2^{(T)}), -\Delta \text{DISTSMALL}\right)$
$\text{OVER}(p_1, p_2) \triangleq \text{tempCoher}(p_2)$
$+ \max_i \left(\begin{array}{l} \text{distLessThan}(y(p_1^{(t)}) - y(p_2^{(t)}), -\Delta \text{DISTSMALL}) \\ \text{distLessThan}(x(p_1^{(t)}) - x(p_2^{(t)}) , \Delta \text{DISTLARGE}) \end{array} \right)$

TABLE 1

Our predicates and their semantics. For simplicity, we show the computation on only a single first frame $p^{(1)}$ or last frame $p^{(T)}$ of a proposal. In practice, to reduce noise, all of the scores are averaged over the first or last L frames.

averaged magnitude of optical flow [14] within that frame. Then we sample an object candidate from the N candidates in frame t . To decide whether the object is moving or not, we sample from $\{\text{MOVING}, \text{STATIONARY}\}$ with distribution $\{\frac{1}{3}, \frac{2}{3}\}$. We sample a MOVING object candidate with probability proportional to the averaged flow magnitude within the candidate. Similarly, we sample a STATIONARY object candidate with probability reciprocal to the averaged flow magnitude within the candidate. The sampled candidate is then propagated (tracked) bidirectionally to the start and the end of the video. We use the CamShift algorithm [9] to track both MOVING and STATIONARY objects, allowing their size changes during the process. We do not use optical-flow-based tracking methods since these methods suffer from drift when objects move quickly. We repeat this sampling and propagation process K times to obtain K object proposals $\{p_k\}$ for each video. Examples of the sampled proposals ($K = 240$) are shown in the middle column of Figure 2.

3.3 Similarity between Object Proposals

We compute the appearance similarity of two object proposals as follows. We first uniformly sample M detections $\{b^m\}$ from each proposal along its temporal extent. For each sampled detection, we extract PHOW [8] and HOG [12]

features to represent its appearance and shape. We also do so after we rotate this detection by 90° , 180° , and 270° . Then, we measure the similarity g between a pair of detections b_1^m and b_2^m with:

$$g(b_1^m, b_2^m) = \max_{i,j} \frac{1}{2} \left(g_{\chi^2}(b_{1,i}^m, b_{2,j}^m) + g_{L_2}(b_{1,i}^m, b_{2,j}^m) \right),$$

$$i, j \in \{0, 1, 2, 3\}$$

where $0, 1, 2, 3$ represents the rotation by 0° , 90° , 180° , and 270° , respectively. We use g_{χ^2} to compute the χ^2 distance between the PHOW features and g_{L_2} to compute the Euclidean distance between the HOG features, after which the distances are minmax normalized and converted to log similarity scores. Finally the similarity between two proposals p_1 and p_2 is taken to be:

$$g(p_1, p_2) = \text{median}_m g(b_1^m, b_2^m)$$

3.4 Joint Inference

We extract object instances (see all 15 classes in Section 4) from the sentences and model them as vertices in a graph. Each vertex v can be assigned one of the K proposals in the video that is paired with the sentence in which the vertex occurs. The score of assigning a proposal k_v to a vertex v is taken to be the unary predicate score $h_v(k_v)$ computed from the sentence (if such exists, or otherwise 0). We put an edge between every two vertices v, u that belong to the same object class. We denote this class membership relation as $(v, u) \in \mathcal{C}$. The edge score for two proposals k_v, k_u assigned to the vertices on this edge is taken to be the similarity score $g_{v,u}(k_v, k_u)$ between the two proposals, as described in Section 3.3. Similarly, we also put an edge between vertices v, u that are arguments of the same predicate. We denote this predicate membership relation as $(v, u) \in \mathcal{P}$. The edge score for two proposals k_v, k_u assigned to the vertices on this edge is taken to be the predicate score $h_{v,u}(k_v, k_u)$ between the proposals, as described in Section 3.1. Our problem then is to select a proposal for each vertex that maximizes the joint score on this graph, i.e., solving the following objective function:

$$\max_{\mathbf{k}} \sum_v h_v(k_v) + \sum_{(v,u) \in \mathcal{C}} g_{v,u}(k_v, k_u) + \sum_{(v,u) \in \mathcal{P}} h_{v,u}(k_v, k_u)$$

where \mathbf{k} is the collection of the selected proposals for all the vertices. Note that we simply weigh the unary scores and the binary scores equally in the above function. This discrete inference problem on graphical models can be approximately solved by Belief Propagation [26]. In the experiment, we use the OpenGM [2] implementation to find the approximate solution.

4 EXPERIMENT

We collected a new dataset to demonstrate the power that sentential information adds to the video object codetection process. This dataset was filmed in 6 different scenes (four in the KITCHEN, one in the BASEMENT, and one outside the GARAGE) of a house. The lighting conditions vary greatly across the different scenes, with the BASEMENT scene the darkest, the KITCHEN scene exhibiting modest lighting, and the GARAGE scene the brightest. Within each scene,

Scene	Fold #									
	1 K1	2 K2	3 K2,3	4 K4	5 B	6 B	7 G	8 K1,2,3	9 B&G	10 K1,2,3,4
Objects	box cabbage cleaner coffee pineapple squash	bowl cabbage cleaner coffee pineapple squash	bowl cabbage pineapple squash	cup juice ketchup milk	box cooler	box cooler	bucket gas-can watering-pot	bowl cabbage pineapple squash	cooler box bucket gas-can watering-pot	bowl cabbage cleaner cup juice ketchup pineapple milk
# of videos	26	27	17	21	19	17	23	17	25	24

TABLE 2
The experimental setup of the 10 folds.

the lighting often varies across different video regions. We assigned 5 actors (four adults and one child) with 15 distinct everyday objects (*bowl*, *box*, *bucket*, *cabbage*, *cleaner*, *coffee*, *cooler*, *cup*, *gas-can*, *juice*, *ketchup*, *milk*, *pineapple*, *squash*, and *watering-pot*), and had them perform different actions which involve interaction with these objects. No special instructions were given requiring that the actors move slowly or the that objects not be occluded. The actors often are partially outside the field of view. Note that the dataset used by Srikantha and Gall [32] does not exhibit this property. Indeed, their method employs human pose which requires that the human be sufficiently visible to estimate such. The filming was performed using a normal consumer camera that introduces motion blur on the objects when the actors move quickly.

We downsampled the filmed videos to 768×432 and divided them into 150 short video clips, each clip depicting a specific event lasting between 2 and 6 seconds at 30 fps. The 150 video clips constitute a total of 12,509 frames. Each video clip was then paired with a human-elicited sentence that describes the event in the video, resulting in 150 sentences for the entire dataset. The sentence semantics were formulated around the 24 distinct predicates in Table 1 and include a total of 246 object occurrences in the sentences, with an average of $\frac{246}{150} = 1.64$ object occurrences per sentence.

We compared the resulting codetections against human annotation obtained with Amazon Mechanical Turk (AMT).² We obtained five bounding box annotations for each target object in each video frame. We asked annotators to annotate the referent of a specific highlighted word in the sentence associated with the video containing that frame. Thus the annotation reflects the semantic constraint implied by the sentences. This resulted in $5 \times 246 = 1,230$ human annotated tracks. To measure how well codetections match human annotation, we use the *IoU*, namely the ratio of the area of the intersection of two bounding boxes to the area of their union. The object codetection problem exhibits inherent ambiguity: different annotators tend to annotate different parts of an object or make different decisions whether to include surrounding background regions when the object is partially occluded. To quantify such ambiguity, we computed intercoder agreement between the human annotators. We computed $\frac{5 \times 4}{2} = 10$ IoU scores for all box pairs produced by the 5 annotators in every frame and averaged them over the entire dataset, obtaining an overall

human-human IoU of 0.72.³

4.1 Experimental Setup

We split our dataset into 10 folds, each fold containing a selected set of video-sentence pairs. The 10 folds contained 26, 27, 17, 21, 19, 17, 23, 17, 25, and 24 video-sentence pairs respectively, totaling 216 pairs. (Some pairs are reused among different folds.) The scenes and objects in each fold are summarized in Table 2, where K, B, and G denote KITCHEN, BASEMENT, and GARAGE, respectively. To rule out the possibility of codetecting objects by simple background modeling (e.g., background subtraction), we intentionally mixed several scenes as in folds 3, 8, 9, and 10.

We found no publicly available implementations of existing video object codetection methods [20, 27, 30, 32], thus for comparison we employ four variants of our method that alternatively disable different scores in our codetection framework. These variants help one understand the relative importance of different components of the framework. Together with our full method, they are summarized below:

	SIM (variant 1)	FLOW (variant 2)	SIM+FLOW (variant 3)	SENT (variant 4)	SIM+SENT (our full method)
Flow score?	no	yes	yes	yes	yes
Similarity score?	yes	no	yes	no	yes
Sentence score?	no	partial	partial	yes	yes

Note that **SIM** uses the similarity measure but no sentential information. This method is similar to prior video codetection methods that employ similarity and the proposal confidence score output by proposal generation methods to perform codetection. When the proposal confidence score is not discriminative, as is the case with our dataset, the prior methods degrade to **SIM**. **FLOW** exploits only binary movement information from the sentence indicating which objects are probably moving and which are probably not (*i.e.*, using only the functions `medFIMg` and `tempCoher` in Table 1), without similarity or any other sentence semantics (thus “partial” in the table). **SIM+FLOW** adds the similarity score on top of **FLOW**. **SENT** uses all possible sentence semantics but no similarity measure. **SIM+SENT** is our full methods that employs all scores. Except for the changes indicated in the above table, all other parameters were kept unchanged, thus resulting in an apples-to-apples comparisons of the results.

Our codetection method and the four variants were then applied to each codetection fold. During the experiment, we empirically set $N = 500$, $K = 240$, $M = 20$, and $L = 15$ (see Section 3 for details), and fixed them for all 10 folds.

3. The dataset, including videos, sentences, and bounding-box annotations, is available at <http://upplysingaofun.ecn.purdue.edu/~qobi/cctp/sentence-codetection.html>

2. <https://www.mturk.com/mturk/>

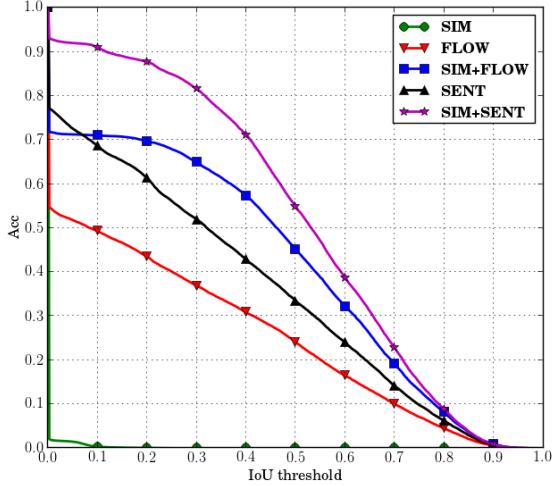


Fig. 3. The codetection accuracy curves of all five methods on our dataset.

4.2 Results

We quantitatively evaluate our method and several variants by computing $\text{IoU}_{\text{frame}}$, $\text{IoU}_{\text{object}}$, and IoU_{fold} as follows. Given an output box for an object in a video frame, and the corresponding set of annotated bounding boxes (five boxes in our case), we compute IoU scores between the output box and the annotated ones, and take the averaged IoU score as $\text{IoU}_{\text{frame}}$. Then $\text{IoU}_{\text{object}}$ is computed as the average of $\text{IoU}_{\text{frame}}$ over the output object track. Finally, IoU_{fold} is computed as the average of $\text{IoU}_{\text{object}}$ over all the object instances in a fold.

We compute IoU_{fold} for each method on each fold as shown in Table 3, where **Human-Human** scores are calculated as described earlier. Our full method outperforms all of the variants and is $\frac{0.4262}{0.7294} = 58.43\%$ of the way towards human performance on this dataset. The first variant **SIM**, using only the similarity measure, completely fails on this task as expected. The binary movement information from sentences is important for reducing the object proposal search space, but without the similarity measure, the performance is still quite poor (**FLOW**). The importance of the similarity measure can be seen from the result of **SENT**, where there is a huge performance drop of 0.13 compared to **SIM+SENT**, *i.e.*, our full method. It is interesting to see that one can get moderate results just by combining similarity and the binary movement information (**SIM+FLOW**), but to further boost performance, more sentence semantics is needed. However, over-constrained semantics can, at times, hinder the codetection process rather than help, especially given the generality of our dataset. This is exhibited on fold 9 where **SIM+FLOW** outperforms **SIM+SENT**. Thus it is important to only impose *weak* semantics on the codetection process.

To evaluate the performance of our method in simply finding objects, we define codetection accuracy $\text{Acc}_{\text{frame}}$, $\text{Acc}_{\text{object}}$, Acc_{fold} , and Acc as follows. Given an IoU threshold, we compute IoU scores between an output box and the corresponding annotated boxes, and binarize the scores according to a specified threshold. Then $\text{Acc}_{\text{frame}}$ is set to the maximum of the binarized scores, $\text{Acc}_{\text{object}}$ is computed

as the average of $\text{Acc}_{\text{frame}}$ over the output object track, and Acc_{fold} is computed as the average of $\text{Acc}_{\text{object}}$ over all the object instances in a fold. Finally, we average Acc_{fold} scores over the 10 folds to get Acc . By adjusting the IoU threshold from 0 to 1, we get an Acc -vs-threshold curve for each of the methods (Figure 3). It can be seen that the codetection accuracies of our full method under different IoU thresholds consistently outperform those of the variants. Finally we demonstrate some codetected object examples in Figure 4. For more examples, we refer the readers to our project page: <http://upplysingaoflun.ecn.purdue.edu/~qobi/cccp/sentence-codetection.html>.

5 CONCLUSION

We have developed a new framework for object codetection in video, namely, using natural language to guide the codetection. Our results on video object codetection indicate that weak sentential information can significantly improve video object codetection. This demonstrates that natural language, when combined with typical computer vision problems, could provide the capability of high-level reasoning that yields better solutions to these problems.

ACKNOWLEDGMENTS

This research was sponsored, in part, by the Army Research Laboratory, accomplished under Cooperative Agreement Number W911NF-10-2-0060, and by the National Science Foundation under Grant No. 1522954-IIS. The views, opinions, findings, conclusions, and recommendations contained in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the Army Research Laboratory, the National Science Foundation, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

REFERENCES

- [1] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 73–80, 2010.
- [2] B. Andres, T. Beier, and J. H. Kappes. OpenGM: A C++ library for discrete graphical models. *CoRR*, abs/1206.0111, 2012.
- [3] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693, 2014.
- [4] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marqués, and J. Malik. Multiscale combinatorial grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 328–335, 2014.
- [5] A. Barbu, A. Bridge, Z. Burchill, D. Coroian, S. Dickinson, S. Fidler, A. Michaux, S. Mussman, N. Siddharth, D. Salvi, L. Schmidt, J. Shangguan, J. M. Siskind, J. Waggoner, S. Wang, J. Wei, Y. Yin, and Z. Zhang. Video in sentences out. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 102–12, 2012.

Method	IoU _{fold}										Average
	1	2	3	4	5	6	7	8	9	10	
SIM	0.0011	0.0000	0.0003	0.0000	0.0005	0.0027	0.0000	0.0036	0.0003	0.0013	0.0010
FLOW	0.1881	0.2396	0.2252	0.1003	0.2151	0.3176	0.2431	0.1853	0.2308	0.1528	0.2098
SIM+FLOW	0.3074	0.3740	0.2828	0.1915	0.3773	0.4356	0.5018	0.2859	0.4633	0.2309	0.3450
SENT	0.2500	0.2885	0.3613	0.2486	0.2268	0.3745	0.3058	0.3238	0.2874	0.2654	0.2932
SIM+SENT	0.3921	0.4738	0.3600	0.3499	0.4033	0.5240	0.5027	0.3733	0.4553	0.4281	0.4262
Human-Human	0.6939	0.7024	0.7820	0.6546	0.6916	0.7861	0.7757	0.7590	0.7295	0.7187	0.7294

TABLE 3
IoU scores for different methods on different codetection folds.

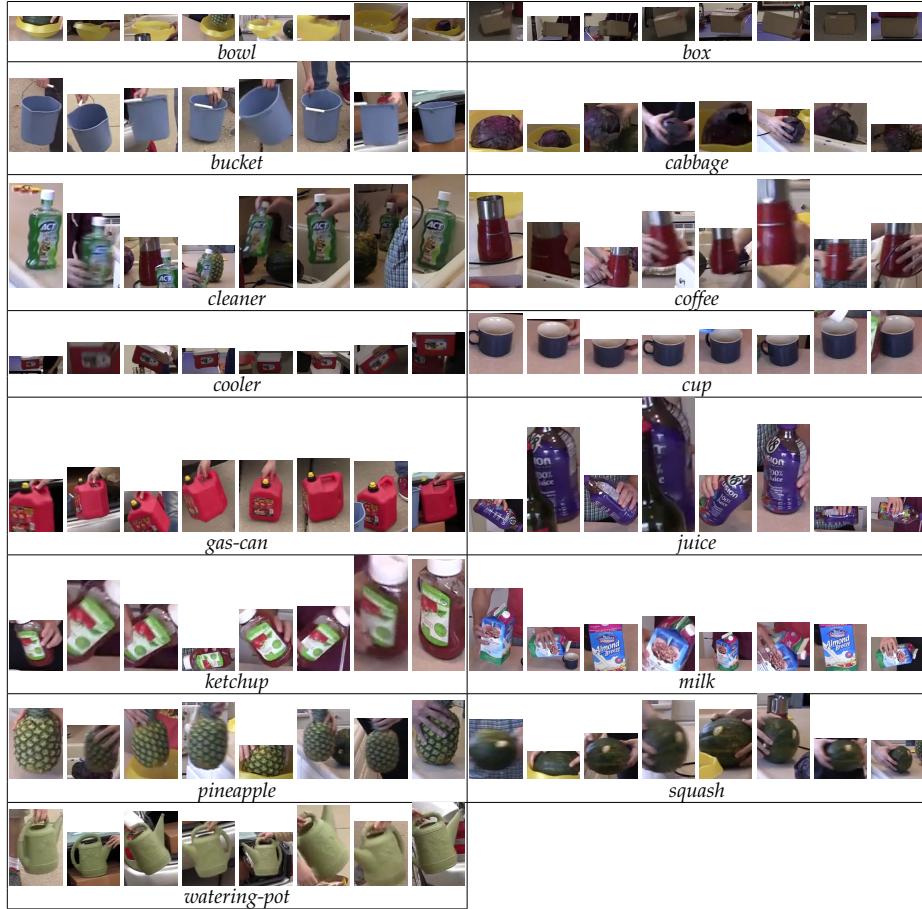


Fig. 4. Examples of the 15 codetected object classes. Note that in some examples the objects are occluded, rotated, poorly lit, or blurred due to motion, but they are still successfully codetected. (For demonstration purposes, the original output detections are slightly enlarged to include the surrounding context; zoom in on the screen for the best view).

- [6] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, E. G. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 848–854, 2004.
- [7] M. Blaschko, A. Vedaldi, and A. Zisserman. Simultaneous object detection and ranking with weak supervision. In *Advances in Neural Information Processing Systems*, pages 235–243, 2010.
- [8] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–8, 2007.
- [9] G. R. Bradski. Computer vision face tracking for use in a perceptual user interface, 1998.
- [10] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba. MIT saliency benchmark, 2012.
- [11] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3286–3293, 2014.
- [12] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*,

- pages 886–893, 2005.
- [13] P. Das, C. Xu, R. F. Doell, and J. J. Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2634–2641, 2013.
- [14] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the Scandinavian Conference on Image Analysis*, pages 363–370, 2003.
- [15] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2712–2719, 2013.
- [16] A. Gupta and L. S. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *Proceedings of the European Conference on Computer Vision*, pages 16–29, 2008.
- [17] M. Jamieson, Y. Eskin, A. Fazly, S. Stevenson, and S. Dickinson. Discovering multipart appearance models from captioned images. In *Proceedings of the European Conference on Computer Vision*, pages 183–196, 2010.
- [18] M. Jamieson, A. Fazly, S. Stevenson, S. J. Dickinson, and S. Wachsmuth. Using language to learn structured appearance models for image annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):148–164, 2010.
- [19] M. Jiang, S. Huang, J. Duan, and Q. Zhao. SALICON: Saliency in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
- [20] A. Joulin, K. Tang, and L. Fei-Fei. Efficient image and video co-localization with Frank-Wolfe algorithm. In *Proceedings of the European Conference on Computer Vision*, pages 253–268, 2014.
- [21] Y. J. Lee and K. Grauman. Learning the easy things first: Self-paced visual category discovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1721–1728, 2011.
- [22] D. Lin, S. Fidler, C. Kong, and R. Urtasun. Visual semantic search: Retrieving videos via complex textual queries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2657–2664, 2014.
- [23] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [24] J. Luo, B. Caputo, and V. Ferrari. Whos doing what: Joint modeling of names and verbs for simultaneous face and pose annotation. In *Advances in Neural Information Processing Systems*, pages 1168–1176, 2009.
- [25] J. Moringen, S. Wachsmuth, S. Dickinson, and S. Stevenson. Learning visual compound models from parallel image-text datasets. *Pattern Recognition*, 50:486–496, 2008.
- [26] J. Pearl. Reverend Bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the Conference on Artificial Intelligence*, pages 133–136, 1982.
- [27] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3282–3289, 2012.
- [28] V. Ramanathan, A. Joulin, P. Liang, and L. Fei-Fei. Linking people with “their” names using coreference resolution. In *Proceedings of the European Conference on Computer Vision*, pages 95–110, 2014.
- [29] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1939–1946, 2013.
- [30] S. Schulter, C. Leistner, P. M. Roth, and H. Bischof. Unsupervised object discovery and segmentation in videos. In *Proceedings of the British Machine Vision Conference*, pages 53.1–53.12, 2013.
- [31] R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng. Parsing with compositional vector grammars. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 455–465, 2013.
- [32] A. Srikantha and J. Gall. Discovering object classes from activities. In *Proceedings of the European Conference on Computer Vision*, pages 415–430, 2014.
- [33] K. Tang, A. Joulin, J. Li, and L. Fei-Fei. Co-localization in real-world images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1464–1471, 2014.
- [34] T. Tuytelaars, C. H. Lampert, M. B. Blaschko, and W. L. Buntine. Unsupervised object discovery: A comparison. *International Journal of Computer Vision*, 88(2):284–302, 2010.
- [35] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *Proceedings of the European Conference on Computer Vision*, pages 391–405, 2014.



Haonan Yu is currently a PhD student in the school of Electrical and Computer Engineering at Purdue University. He received the B.S. degree in Computer Science from Peking University, China, in 2011. His research interests are Computer Vision and Natural-Language Processing, especially combining video and language. He was the recipient of the Best Paper Award of ACL 2013.



Jeffrey Mark Siskind received the B.A. degree in computer science from the Technion, Israel Institute of Technology in 1979, the S.M. degree in computer science from MIT in 1989, and the Ph.D. degree in computer science from MIT in 1992. He did a postdoctoral fellowship at the University of Pennsylvania Institute for Research in Cognitive Science from 1992 to 1993. He was an assistant professor at the University of Toronto Department of Computer Science from 1993 to 1995, a senior lecturer at the Technion Department of Electrical Engineering in 1996, a visiting assistant professor at the University of Vermont Department of Computer Science and Electrical Engineering from 1996 to 1997, and a research scientist at NEC Research Institute, Inc. from 1997 to 2001. He joined the Purdue University School of Electrical and Computer Engineering in 2002 where he is currently an associate professor.