2014 IEEE Conference on Computer Vision and Pattern Recognition

# Visual Semantic Search: Retrieving Videos via Complex Textual Queries

Dahua Lin[1] Sanja Fidler[1,2] Chen Kong[3] Raquel Urtasun[1,2]
TTI Chicago[1] University of Toronto[2] Tsinghua University[3]
dhlin@ttic.edu, kc10@mails.tsinghua.edu.cn, {fidler,urtasun}@cs.toronto.edu

## Abstract

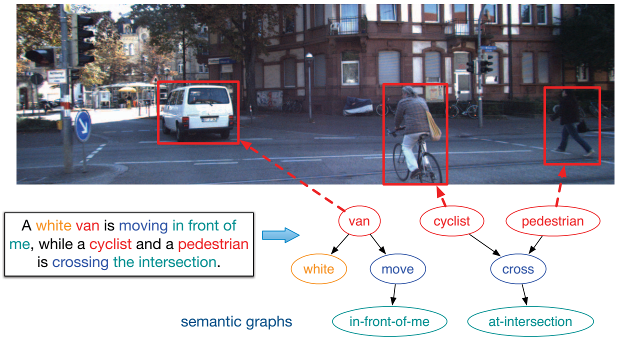
*In this paper, we tackle the problem of retrieving videos using complex natural language queries. Towards this goal, we first parse the sentential descriptions into a semantic graph, which is then matched to visual concepts using a generalized bipartite matching algorithm. Our approach exploits object appearance, motion and spatial relations, and learns the importance of each term using structure prediction. We demonstrate the effectiveness of our approach on a new dataset designed for semantic search in the context of autonomous driving, which exhibits complex and highly dynamic scenes with many objects. We show that our approach is able to locate a major portion of the objects described in the query with high accuracy, and improve the relevance in video retrieval.*

## 1. Introduction

One of the fundamental challenges in video search is to be able to perform retrieval given a semantic query. Consider the following example, where a user wants to retrieve a movie she has seen but she remembers neither the title nor the authors. She has, however, a vivid memory of a particular scene. Thus she enters the following description in a search engine: *"A man is sitting on the staircase. Suddenly a car from the twenties rushes by and picks him up. That is the night he meets Ernest Hemingway."* The top queries returned are pictures of Chiang Mai and Steve Jobs. But, what she really wanted is a scene from *"Midnight in Paris"*.

Understanding images semantically is key in order to retrieve relevant candidates for these complex queries. However, semantic parsing of images is an extremely difficult task. Despite decades of research, the performance of visual recognition algorithms is still rather low. To improve recognition systems, additional information in the form of depth data [18], contextual models [14] or video can be used.

In this paper, we are interested in performing semantic retrieval of videos in the context of autonomous driving. Retrieval of relevant events is particularly beneficial in this setting due to the abundance of available data, *e.g.* after a day of capture. In particular, semantic retrieval in this context has applications in improving driver safety, studying traffic congestion as well as building autonomous systems.



A white van is moving in front of me, while a cyclist and a pedestrian is crossing the intersection.

Figure 1. This figure shows a video frame and the associated description. Our approach constructs semantic graphs to capture the semantic structure of the description, and infers the matching between the nouns and objects detected in the video. The action and relative positions of the objects will also be taken into account and matched with the verbs and adverbs in the sentence.

Towards this goal, we developed an approach that first parses the videos semantically by means of object detection, tracking, and ego-motion estimation, all in 3D. As illustrated in Fig. 1, given a complex query, we first parse the description into a semantic graph, which is then matched to the visual concepts using a generalized bipartite matching algorithm. This enables exact inference via a linear program. Our approach takes advantage of object appearance, motion, and spatial relations, and learns the importance of each energy term using structure prediction.

We demonstrate the effectiveness of our approach in the context of videos captured from an autonomous driving platform. These are particularly interesting as semantic queries can contain temporal as well as spatial information about multiple objects and "stuff" (*e.g.*, trees, buildings) present in the scene. Towards this goal, we asked annotators to partition the videos from the KITTI tracking benchmark [9] into possibly overlapping segments containing interesting activities, and provide natural sentential descriptions of those activities. These serve as queries for our semantic search. In a long video sequence, our approach is able to locate a major portion of the objects described in the query with remarkably high accuracy (60% vs below 20% for the baseline). It also substantially improves the relevance of the retrieved segments in our challenging setting.

The contributions of this work consists of three aspects.

1063-6919/14 $31.00 © 2014 IEEE
DOI 10.1109/CVPR.2014.340

First, we consider a new problem that is fundamentally different from traditional video retrieval tasks. Most previous work on video retrieval find relevant videos given a query, while we focus on matching individual words in the query to specific objects (*i.e*. a tracklet of 2D boxes) participating in sub-events in highly dynamic and complex scenes. Second, we collect a new dataset for semantic retrieval, which provides lingual descriptions for video segments and includes detailed time-stamped associations between nouns and visual objects. This distinguishes it from those used in traditional video retrieval applications (*e.g*. TRECVID). Third, we develop a new framework, where we construct semantic graphs from textual descriptions, formulate the problem as graph matching allowing efficient exact inference, and explicitly exploit both spatial and semantic relations.

## 2. Related Work

Text that appears jointly with images usually contains useful information for visual analysis. In the past decade, utilizing textual description to help tasks such as image/video retrieval, has become an active research area. Relevant work generally falls into several categories, which we roughly classify into video retrieval and joint modeling.

**Video Retrieval:** With the explosive growth of online videos, video search and retrieval has become a hot topic in computer vision. In their seminal work, Sivic and Zisserman described *Video Google* [19], a system that retrieves videos from a database via bag-of-words matching. Lew *et al*. [13] reviews earlier efforts in video retrieval, which, similar to content-based image retrieval, mostly rely on feature-based relevance feedback or similar methods.

Recently, concept-based methods have emerged as a popular approach to video retrieval. Snoek *et al*. [20] proposed a method based on a set of concept detectors, with the aim to bridge the semantic gap between visual features and high level concepts. The importance of concepts have also been recognized by other researchers [25]. Since then, a series of methods have been developed to improve video concept detection [1, 21, 5]. It is important to note that most current work on concept-based retrieval focuses on mining characteristic concepts or improving the performance of concept detectors. The spatial and semantic relations between concepts have rarely been explored in practice.

**Joint modeling of Images and Text:** Several generative models that couple modeling of text and images were studied in [3], including a multi-modal extension of Latent Dirichlet Allocation. Iyengar *et al*. [11] proposed a probabilistic model that relates words and image parts through an intermediate layer that captures common concepts. Models in this category usually rely on strong assumptions, *e.g*. words in a document are exchangeable, and thus neglect important relations between words.

Recently, Matuszek *et al*. [15] proposed a joint model of language and perception that exploits semantic parsing of a sentence to align objects to classifiers. The primary goal of this work is to classify attributes of RGBD objects. Fidler *et al*. [8] developed a CRF model that incorporates parsed sentences for holistic scene understanding, and extended it to aligning text-to-images [12]. In recent years, many approaches have been developed to generate image [6, 27, 17] or video descriptions [2]. For instance, Rohrbach *et al*. [17] presented a system, which uses a CRF to capture relations between image components and generates the description through statistical machine translation.

## 3. A New Video Dataset for Semantic Search

In this paper, we are interested in semantic search in the domain of autonomous driving, where rich visual and contextual information (e.g., video, stereo, ego-motion, road information, traffic patterns) can be exploited. Towards this goal, we adopted the KITTI benchmark [9] and collected natural lingual descriptions for the training subset of the tracking benchmark, which comprises 21 videos of length 381 frames on average, for a total of 8008 frames.

We used in-house annotators to ensure quality. We asked them to describe whatever (sub)events they felt were relevant for someone driving a car. This could include picking single static frames, a video segment or the video as a whole. Towards this goal, we created an annotation tool that allowed the annotators to watch a video and select a particular time chunk by choosing a beginning and an end frame. For each time chunk they wrote a description, and were asked to link the objects/stuff they were talking about to the image by either placing a bounding box around the object or creating a segmentation mask. This link could be done in any of the frames in the described time chunk. All objects that were annotated in this way and are part of KITTI classes (*cars*, *vans*, *trucks*, *pedestrians*, *cyclists*) were then matched to KITTI ground-truth, which provided us with 3D GT trajectories across each video segment.

Six annotators were employed to label the videos, three annotators per video in order to capture a wide variety of events. Our new dataset comprises of 443 descriptions, which contain 520 sentences and 3520 words in total. These sentences talk about 1068 objects in the videos.

It is worth emphasizing this database is very challenging: Video segments are visually similar and difficult to distinguish; descriptions provided by the annotators are generally quite concise, and sometimes ambiguous. This may be seen from the examples we provide in the supplemental material. Small training set size and difficulties in tracking also limits the performance improvement.

## 4. Visual Semantic Search

In order to be able to perform visual semantic search, we need to establish correspondences between entities in the

|  | DC | HM | MCF | SSP | HM+MCF | HM+SSP | MCF+SSP |
|---|---|---|---|---|---|---|---|
| recall (%) | 22.4 | 32.3 | 30.0 | 27.4 | 35.0 | 36.1 | 34.2 |
| # tracklets | 3444 | 2678 | 1326 | 1084 | 3514 | 3503 | 2030 |

Table 1. Recall of tracking methods for our domain vs number of tracklets (counted over all the video segments).

text and objects/stuff in the visual scene. Towards this goal, we exploit a variety of cues including object appearance, motion, and relations/interactions between objects and the scene. Our approach proceeds as follows: we first detect and track candidate objects in each input video fragment, and characterize each object track with a variety of cues (*e.g.* appearance, motion). We then construct a graph representation to capture the semantic structure of the textual description given as query. Finally, we develop a matching algorithm, which takes into account both visual cues and spatial/semantic relations, to infer correspondence between the entities described in the query and the objects detected in the video. It is important that, in this process, we allow some entities to not match anything as we might have false negatives and false positives in our visual set of candidates.

## 4.1. Extracting Candidates from Video Segments

To generate tracks in our videos we utilize the MCF [28] and SSP [16] trackers (which were provided to us by [9]), as they have relatively high recall. We say that a ground-truth (GT) trajectory is *recalled* by a tracklet if they overlap more than $50\%$ in space and time. That is, overlap in a frame is said to be correct if the two bounding boxes overlap more than $50\%$ IOU, and a tracklet is correct if the number of correct frames divided by the number of frames spanned by both the GT and the candidate is higher than $50\%$. Note that this is a rather strict criteria, but reasonable in our setting as our system depends on having longer and quality tracks with enough discriminative information to classify actions.

Both MCF and SSP are tracking-by-detection approaches, which first run DPM [7] and then find an optimal path through the detections. In the KITTI benchmark [9], the detection performance (AP) for *cars*, *pedestrians*, and *cyclists* are $56.5\%$, $39.4\%$, and $29.9\%$, respectively. Note that the benchmark for cars measures performance at 70% IOU. This low accuracy reflects the difficulty of the dataset, and limits the success of the tracking methods. We took trajectories from both methods, and performed non-maxima suppression to remove redundant tracks.

## 4.2. Constructing Semantic Graph from Text

We use a graph representation to capture the semantic structure of a descriptive sentence, which we refer to as the *semantic graph*. As illustrated in Fig. 2, each node of a semantic graph corresponds to a word with specific meaning (*e.g. car*, *pedestrian*, and *walk*), which may be a noun, a verb, an adjective, or an adverb. Nodes are connected by different types of edges expressing the semantic relations between them. The procedure for constructing a semantic graph from a text description comprises three steps: parsing, transforming and distilling, which we describe next.

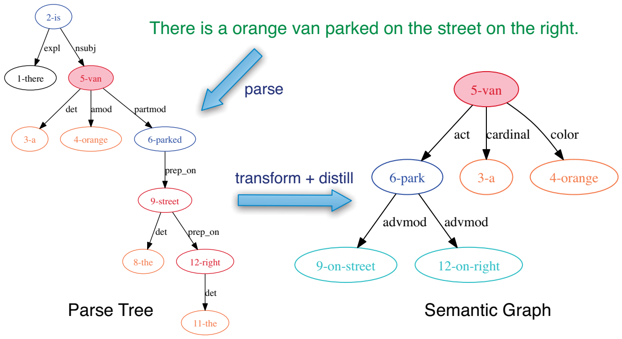


Figure 2. This figure shows a parse tree of a sentence (on the left) and the constructed semantic graph (on the right). We can see that the semantic graph captures the key semantic structure of the sentence: a van is parking, whose color is orange. The action of parking is modified by two adverb phrases: on-street and on-right.

**Parsing:** We use the Stanford parser [22] to obtain a parse tree of the sentence, in which each word is attached with a part-of-speech tag that specifies its syntactic role.

**Transforming:** The parse tree of a sentence in its original form is difficult to manipulate. Many grammatical structures in the trees are not pertinent to visual analysis. For example, in a sentence *"there is a car in front of me"*, the part *"there is"* is not relevant to our purpose, while the phrase "in front of" needs to be compressed into a single compound preposition *"in-front-of"* to convey a certain spatial relation. Specifically, we devise a series of transformation rules by carefully examining the descriptions in the training set, and apply them to the parse trees in order to obtain simplified *syntactic graphs*. We refer the reader to the supplementary material for more details.

**Distilling:** This step generates a semantic graph by selecting relevant nodes and edges from the transformed syntactic graph. More specifically, we classify relevant words into four categories: *entities*, *actions*, *cardinalities*, *colors*, and *action modifiers*. Each category has several classes: we use five entity classes (*cars*, *vans*, *trucks*, *pedestrians*, and *cyclists*), fifteen action classes (such as *move*, *turn*, *park*, *walk*, etc), as well as other classes for modifiers. We refer the reader to the supplemental material for a full list. We use synonyms obtained from WordNet to match the words to our classes. Nodes that match our classes are selected for insertion into the semantic graph, where they are associated with a class tag. Edges that connect the selected nodes are also preserved in the distilled graph.

Generally, the algorithm to construct semantic graphs, which involves both *transforming* and *distilling*, is based on a series of manually derived rules. Due to page limits, we are unable to present all these rules in the paper. A detailed documentation will be provided as we release the codes.

### 4.3. Matching Text and Video Segments

The primary task in this work is to match the entities in a semantic graph to the object tracks extracted from a video clip. We formulate this as a *generalized bipartite matching* problem. In particular, our approach tries to match entities in a textual description, which we refer to as *sources*, to the object tracks, which we refer to as *sinks*. To account for plural forms, *e.g.* *"two cars"* and *"several people"*, we allow each source to be matched to one or multiple sinks, depending on the associated cardinality modifier. It is also possible that some entities mentioned in text have been missed by our visual detection/tracking algorithm, and thus no match exists. We thus introduce a *no-obj sink*, which allows semantic entities to be matched to none of the visual objects.

Suppose we have $m$ sources and $n$ sinks, which are respectively indexed using $u$ and $v$. We use $y_{uv} \in \{0,1\}$ to denote whether the source $u$ is matched to the sink $v$, and $h_{uv}$ to denote the score of matching $u$ and $v$. We formulate the matching problem as a LP as follows:

$$\begin{aligned} \max_{\mathbf{y}} \quad & \sum_{uv} h_{uv} y_{uv} && (1) \\ \text{s.t.} \quad & \sum_{v} y_{uv} = s_u, \quad \forall u = 1, \ldots, m \\ & \sum_{u} y_{uv} \leq t_v, \quad \forall v = 1, \ldots, n \\ & 0 \leq y_{uv} \leq 1, \quad \forall u = 1, \ldots, m, \;\; v = 1, \ldots, n-1 \end{aligned}$$

where $\mathbf{y} = (y_{11}, \cdots, y_{mn})$ denotes the set of all matching indicators that we are optimizing over, and $s_u$, $t_v$ are the capacity for the source $u$ and sink $v$ respectively. Particularly, the different $s_u$ are set according to the cardinality modifier associated with the corresponding noun, while $t_v$ is generally set to 1, as each object in a video clip is very rarely referenced by more than one instance in a sentence (the so called *coreference resolution*), except that when $v$ refers to the *no-obj sink*, *i.e.* $v = n$, we set $t_v = \sum_u s_u$, thus allowing all sources to be matched to it.

This problem has at least one feasible solution ($y_{uv} \equiv 0$) when all $s_u$ and $t_v$ values are non-negative. Furthermore, it can be shown that all basic solutions (*i.e.* vertices of the domain) for this problem are integer-valued when $s_u$ and $t_v$ are all integers. Hence, in our specific setting, the optimum for this LP must be an integer-valued solution.

We would like the scoring function $h_{uv}$ to utilize different sources of information, e.g., object appearance, motion, prior knowledge. We thus define the score of an edge to be a linear combination of scores coming from different cues:

$$h_{uv} = \sum_{k=1}^{K} w_k f_{uv}^{(k)} = \mathbf{w}^T \mathbf{f}_{uv}. \qquad (2)$$

Here, $K$ is the number of all scoring channels (*e.g.* appearance, motion, spatial relations), and $f_{uv}^{(k)}$ is the $k$-th matching score between $u$ and $v$. Note that we learn the weights $\mathbf{w} = (w_1, \cdots, w_K)$ using structure prediction from training data (see Section 4.5). We now describe the different scores in more detail.

### 4.4. Visual Scores

**Appearance:** Our object appearance scores are based on the deformable part-based model (DPM) [7]. Specifically, for each object track we take a bounding box in every other frame and run a DPM for each object class of interest in order to predict how likely the patch belongs to particular class (*e.g.* cars, pedestrians). Here, we allow the root filter to be adjusted in position and scale, under the constraint that the placement overlaps with the original bounding box at least 60% IOU. For each class we take the score corresponding to the highest scoring placement of the root and parts. We transform the scores into positive numbers via a logistic function with unit scale. The final score for the full track w.r.t. each object class is obtained by averaging the class scores over all selected frames.

**Motion:** Consider a description *"a car turns left"*, where the "car" is associated with an action *"turn"*, which is modified by an adverb "left". To match this car to a trajectory, it is useful to test whether the trajectory contains a left turn. Further, to cope with descriptions like *"a car is parking"* or *"a speeding car"*, we also need to reason about the absolute speed of the objects in the world. To this end, we exploit 2D and 3D motion information. As 2D features we use: (1) *Scale factor*: defined between bounding boxes in consecutive frames as $area(box_i)/area(box_{i-1})$. (2) *Difference in foot position*: $\big(foot(box_i) - foot(box_{i-1})\big)/height(box_i)$, where *foot* denotes the middle point of the lower side of the box, representing the contact point between the object and the ground. We also use absolute position of the foot normalized by the image size as a feature. (3) *Box dimensions*: width and height of the box normalized by image size.

Note that all the features defined between consecutive frames are in fact computed between the current and a few past frames (we used 4 frames), averaged to smooth out the noise in the tracking and depth estimation algorithms. We also use a simple duration feature, where the duration of the track is divided by the total length of the video segment. This helps us score lower shorter and non-salient tracks.

To extract 3D motion features, we project each 2D bounding box to 3D using depth information. In particular, we take a smaller centroid region within the box and find a median of the (non-zero) depth values, computed using StereoSLIC [26]. We also compute visual odometry [10] to transform the 3D frame's local coordinates into a global coordinate system across the full video. We define the following 3D motion features in bird's eye perspective. (1) *Velocity*: We use the displacement vector in 3D between consecutive frames, its magnitude and angle. Note that this displacement compensates for the ego-motion and thus models object's *absolute* movement. (2) *Curvature*: We use the curvature of the trajectory, computed by fitting a third-order polynomial to the curve. (3) *Shape-context* [4]: We form a shape context descriptor over the full trajectory to capture particular shapes of the object's movement.

**Relative motion:** Positions of objects are often described relative to the observer, *e.g. "in front of me"*, *"on my left"*, *"is overtaking me"*. Note that by "me" the annotators were typically referring to the ego-car. To exploit such phrases, we use the following features to characterize the motion and position of an object relative to the ego-motion. (1) *Velocity*: We use the difference between the object's and ego velocities, forming a 3-dim vector, as well as the relative angle between both moving directions. (2) *Depth*: We use the difference in depth between the object and the ego-car. (3) *Position*: We use the difference in $X$ values corresponding to the horizontal location of the object and ego-car in 3D, helping us distinguish, *e.g. "to-right-of-me"*.

Note that each of the features (appearance, motion and relative motion) is defined per frame. To turn them into a descriptor for the entire track (irrespective of its length) we split the trajectory in $K$ non-overlapping segments (we used $K = 3$), average each feature in each segment, and pool them together to form a $K$-dimensional feature. All features are then concatenated to form the final descriptor.

## 4.5. Learning

We learn the weights $\mathbf{w}$ for score combination using a structural SVM [24, 23]. In what follows, we first present the learning problem, and then derive a simplified learning algorithm by exploiting the *conciseness* of our model.

### 4.5.1 The Learning Problem

We formalize the learning problem as follows:

$$\begin{aligned} \min_{\xi, \mathbf{w}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \qquad (3)\\ \text{s.t.} \quad & \xi_i \geq \mathbf{w}^T(\boldsymbol{\phi}_i(\mathbf{y}) - \boldsymbol{\phi}_i(\mathbf{y}^{(i)})) + \Delta(\mathbf{y}, \mathbf{y}^{(i)}), \ \ \forall \mathbf{y} \in \mathcal{Y}^{(i)}.\\ & \xi_i \geq 0, \ \ \forall i = 1, \ldots, N. \end{aligned}$$

Here, $\mathbf{y}^{(i)}$ is the ground-truth matching for the $i$-th instance, $\boldsymbol{\phi}_i(\mathbf{y})$ a vector of matching scores of $\mathbf{y}$, $\Delta(\mathbf{y}, \mathbf{y}^{(i)})$ the loss function, and $N$ the total number of training examples. In particular, $\boldsymbol{\phi}_i(\mathbf{y})$ can be expressed as

$$\boldsymbol{\phi}_i(\mathbf{y}) = [\phi_i^{(1)}(\mathbf{y}), \ldots, \phi_i^{(K)}(\mathbf{y})], \text{ with } \phi_i^{(k)} = \sum_{uv} f_{uv}^{(ik)} y_{uv}.$$

Note that the domain $\mathcal{Y}^{(i)}$ encodes the constraints that $\mathbf{y}$ has to satisfy, and is different for each example, with

$$\mathcal{Y}^{(i)} = \left\{ \mathbf{y} : \ \sum_v y_{uv} = s_u^{(i)}, \ \sum_u y_{uv} \leq t_v^{(i)}, \ 0 \leq y_{uv} \leq c_{uv}^{(i)} \right\}.$$

Here, $c_{uv}$ equals $s_u$ when $v = n$ (the *no-obj* sink), or 1 otherwise. These constraints are the same as those in Eq.(1).

We use the Hamming loss as the loss function, which is decomposable as

$$\Delta(\mathbf{y}, \mathbf{y}^{(i)}) = \sum_{uv} \mathbb{1}(y_{uv} \neq y_{uv}^{(i)}) = a^{(i)} - \sum_{uv} y_{uv} y_{uv}^{(i)},$$

where $a^{(i)} \triangleq \sum_u c_u^{(i)}$ is the total number of matching edges, which is a constant.

### 4.5.2 Simplified Learning by Leveraging Conciseness

We take advantage of the decomposable property of the constraints to derived a simplified learning algorithm. The first set of constraints in Eq.(3) can be rewritten as:

$$\mathbf{w}^T \boldsymbol{\phi}_i(\mathbf{y}^{(i)}) \geq \max_{\mathbf{y} \in \mathcal{Y}^{(i)}} \left( \mathbf{w}^T \boldsymbol{\phi}_i(\mathbf{y}) + \Delta(\mathbf{y}, \mathbf{y}^{(i)}) \right) - \xi_i,$$

This model is called *concise* [23] if there exists a function $\tilde{f}_i$ that is concave in $\boldsymbol{\mu}$ and a convex set $\mathcal{U}^{(i)}$ such that

$$\max_{\mathbf{y} \in \mathcal{Y}^{(i)}} \left( \mathbf{w}^T \phi_i(\mathbf{y}) + \Delta(\mathbf{y}, \mathbf{y}^{(i)}) \right) = \max_{\boldsymbol{\mu} \in \mathcal{U}^{(i)}} \tilde{f}_i(\mathbf{w}, \boldsymbol{\mu}) \quad (4)$$

**Proposition 1** *Our model is concise with*

$$\tilde{f}_i(\mathbf{w}, \boldsymbol{\mu}) = a^{(i)} + \sum_{uv} \left( \mathbf{w}^T \mathbf{f}_{uv}^{(i)} - y_{uv}^{(i)} \right) \mu_{uv}. \quad (5)$$

*and the constraint $\boldsymbol{\mu} \in \mathcal{U}^{(i)}$ can be written as*

$$\sum_v \mu_{uv} = s_u^{(i)} \ \ \forall u, \sum_u \mu_{uv} \leq t_v^{(i)} \ \ \forall v,$$
$$0 \leq \mu_{uv} \leq c_{uv}^{(i)} \ \ \forall u, v.$$

Note that proofs of propositions are provided in the supplementary material. With Eq.(5), the Lagrangian dual of $\max_{\boldsymbol{\nu} \in \mathcal{U}^{(i)}} \tilde{f}_i(\mathbf{w}, \boldsymbol{\mu})$ is given by

$$\rho_i = a^{(i)} + \sum_u \lambda_u s_u^{(i)} + \sum_v \eta_v t_v^{(i)} + \sum_{uv} \nu_{uv} c_{uv}^{(i)}, s \quad (6)$$

with constraints

$$\mathbf{w}^T \mathbf{f}_{uv}^{(i)} \leq y_{uv}^{(i)} + \lambda_u + \eta_v + \nu_{uv}, \ \ \eta_v \geq 0, \ \ \nu_{uv} \geq 0 \ \forall u, v.$$

Combining the optimization over $\mathbf{w}$ and that over $\boldsymbol{\lambda}$, $\boldsymbol{\eta}$, and $\boldsymbol{\nu}$, we finally get the following learning problem:

$$\begin{aligned} \min_{\mathbf{w}, \xi, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\nu}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \qquad (7)\\ \text{s.t.} \quad & \mathbf{w}^T \mathbf{z}^{(i)} \geq \rho_i(\boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\nu}) - \xi_i, \quad \forall i = 1, \ldots, N,\\ & \mathbf{w}^T \mathbf{f}_{uv}^{(i)} \leq y_{uv}^{(i)} + \lambda_u + \eta_v + \nu_{uv}, \quad \forall u, v, i\\ & \eta_v \geq 0, \ \ \nu_{uv} \geq 0 \ \ \forall u, v, i. \end{aligned}$$

|  | K | rand | noun | verb | adv | n.+v. | v.+a. | all |
|---|---|---|---|---|---|---|---|---|
| GT | 1 | .0397 | .0613 | .0873 | .0967 | .1061 | .1274 | .1486 |
|  | 2 | .0794 | .1250 | .1533 | .1651 | .1910 | .2288 | .2335 |
|  | 3 | .1191 | .1840 | .2052 | .2217 | .2712 | .3160 | .3467 |
|  | 5 | .1985 | .3042 | .3443 | .3514 | .4057 | .4481 | .4693 |
| real | 1 | .0425 | .0755 | .0566 | .0889 | .0836 | .1078 | .0943 |
|  | 2 | .0849 | .1375 | .1132 | .1321 | .1429 | .1698 | .1779 |
|  | 3 | .1274 | .1914 | .1752 | .1698 | .2022 | .2264 | .2399 |
|  | 5 | .2123 | .2722 | .2857 | .2722 | .3181 | .3342 | .3208 |

Table 3. Average hit rates of video segment retrieval.

|  | K | rand | noun | verb | adv | n.+v. | v.+a. | all |
|---|---|---|---|---|---|---|---|---|
| GT | 1 | .1673 | .2571 | .3029 | .2800 | .3286 | .3429 | .3629 |
|  | 2 | .1673 | .2686 | .2771 | .2600 | .3400 | .3386 | .3557 |
|  | 3 | .1673 | .2790 | .2714 | .2610 | .3410 | .3267 | .3533 |
|  | 5 | .1673 | .2749 | .2640 | .2589 | .3280 | .3109 | .3383 |
| real | 1 | .1673 | .2680 | .2484 | .2876 | .2810 | .2941 | .2941 |
|  | 2 | .1673 | .2647 | .2304 | .2484 | .2843 | .2680 | .2908 |
|  | 3 | .1673 | .2702 | .2462 | .2495 | .2898 | .2800 | .3017 |
|  | 5 | .1673 | .2686 | .2444 | .2477 | .2784 | .2758 | .2869 |

Table 4. Average relevance of video segment retrieval.

Here, $\mathbf{z}^{(i)} = [z_1^{(i)}, \ldots, z_K^{(i)}]$ with $z_k^{(i)} = \sum_{uv} s_{uv}^{(ik)} y_{uv}^{(i)}$, and $\rho_i(\boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\nu})$ is given by Eq.(6).

We optimize this function using the Gurobi solver. Note that this optimization is much more efficient than the standard cutting plane algorithm of [24], as we only have to solve this QP once to get $\mathbf{w}$, without iteratively adding new constraints and solving the problem again.

**Run-time Complexity:** The learning involves solving a QP problem with $K + \sum_{i=1}^{N}(m_i + 1)(n_i + 1)$ and $N + \sum_{i=1}^{N} m_i n_i$ constraints, where $m_i$ and $n_i$ are the number of noun entities and the number of object tracks for the $i$-th video segment, respectively, $K$ is the dimension of weight vector, and $N$ the number of video segments. In a typical video segment, $m_i$ and $n_i$ are relatively small (less than 10 for most videos). Hence, for large datasets, the problem size scales linearly as $N$ increases. It takes less than 1 second to learn $\mathbf{w}$ on our training set (using Gurobi). The inference is an LP problem with $m_i n_i$ variables and $(m_i + 1)(n_i + 1)$ constraints, and can be solved in the matter of milliseconds on a standard laptop. Hence, the proposed algorithm is well suited for real-time applications.

## 5. Experimental Evaluation

We tested the proposed approach on the dataset introduced in Sec. 3. In particular, we performed experiments in two applications: (1) find the objects described in a query, and (2) retrieve the video segment relevant to a query. These applications demonstrate our method's ability in both semantic analysis and video retrieval.

### 5.1. Finding Objects of Interest

In the first application, we are interested in locating the objects, described by a query, in a video. Specifically, we partitioned the 21 videos into disjoint training and test subsets. The training set comprises of 13 videos with 296 queries, in which 698 objects are described; while the test set comprises of 8 videos with 147 queries, describing 370 objects. We measured performance with recall, precision, and F1-scores. We used the proposed method to infer the object-id of each tracklet found in the video, and compared it with the ground-truth trajectory associated with the query object. A track is regarded as a *correct match* if its bounding boxes overlap with ground-truth by more than 50% IOU for at least half of its time span. Then, *recall*[1] is the fraction of mentioned objects that are correctly matched, *precision* the proportion of correct matches in all detected tracks, and *F1-score* the harmonic average of precision and recall.

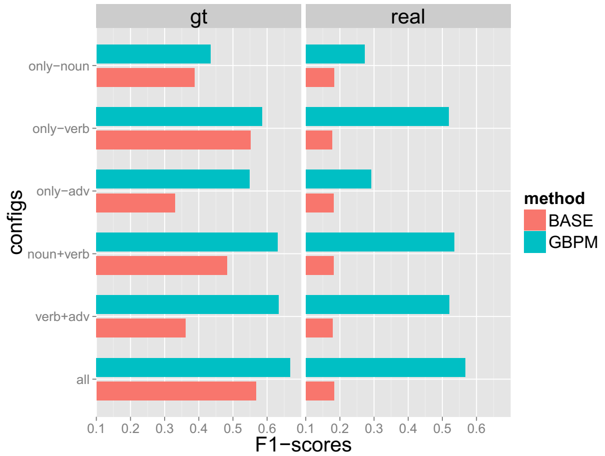


Figure 4. The bar charts that compare the F1-scores obtained using different methods under various configurations.

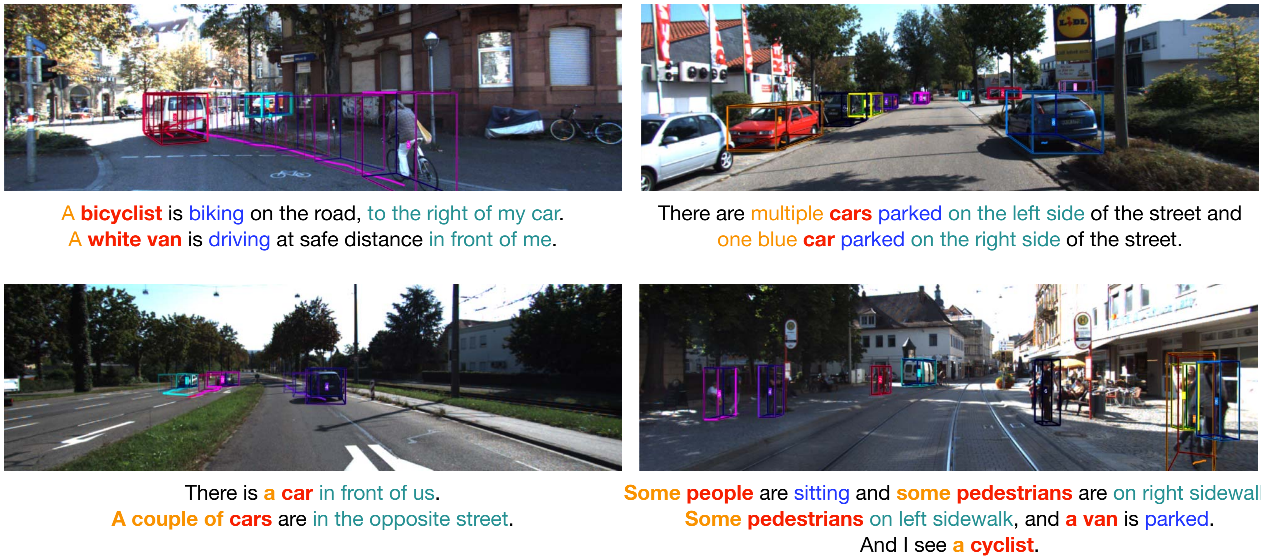
As baseline we implemented an algorithm which instead of solving the LP problem, uses feature-based classification scores to associate each detected object track to the highest scoring noun-entity if the score is above a certain threshold. This threshold is determined empirically via cross-validation, and is chosen as the one achieving highest overall F1-score. We call this method *BASE* and we refer to our proposed method as *GBPM (Generalized Bipartite Matching)*. Fig. 3 shows the several results obtained using GBPM.

We tested each method on two scenarios: when employing ground-truth trajectories, which we refer to as *GT*, and those computed using DPM and visual trackers, which we refer to as *Real*. Comparison of performance between these two settings shows how detection and tracking errors influence the matching performance. Moreover, to identify the contributions of different components, we compared different combinations of features for both methods. Specifically, we tested six different configurations: *only-noun*, *only-verb*, *only-adv*, *noun+verb*, *verb+adv*, and *noun+verb+adv*.

[1] **Node:** this recall is computed w.r.t. the only those objects actually mentioned in the queries. This recall value is different from the one that we used to evaluate trackers in Sec 4.1, which is computed w.r.t. all trajectories in the KITTI dataset.

|  |  | BASE |  |  |  |  |  | REAL |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  | noun | verb | adv | n.+v. | v.+a. | all | noun | verb | adv | n.+v. | v.+a. | all |
| GT | recall | .8777 | .5897 | .2170 | .6884 | .2485 | .6726 | .4379 | .5700 | .5562 | .6391 | .6430 | .6765 |
|  | prec. | .2483 | .5182 | .7006 | .3721 | .6632 | .4906 | .4302 | .6021 | .5434 | .6243 | .6257 | .6583 |
|  | F1 | .3871 | .5517 | .3313 | .4830 | .3615 | .5674 | .4340 | .5856 | .5497 | .6316 | .6342 | .6673 |
| real | recall | .5301 | .5137 | .5246 | .5246 | .5191 | .5301 | .3251 | .4563 | .3497 | .5328 | .4754 | .5710 |
|  | prec. | .1102 | .1068 | .1091 | .1091 | .1080 | .1102 | .2333 | .6007 | .2485 | .5357 | .5743 | .5633 |
|  | F1 | .1825 | .1769 | .1806 | .1806 | .1787 | .1825 | .2717 | .5186 | .2906 | .5342 | .5202 | .5672 |

Table 2. This table lists the performance in terms of recall, precision, and F1-scores, obtained using both BASE and GBPM methods.



A **bicyclist** is biking on the road, to the right of my car.
A **white van** is driving at safe distance in front of me.

There are multiple **cars** parked on the left side of the street and
one blue **car** parked on the right side of the street.

There is a **car** in front of us.
**A couple of cars** are in the opposite street.

**Some people** are sitting and **some pedestrians** are on right sidewalk.
**Some pedestrians** on left sidewalk, and **a van** is parked.
And I see **a cyclist**.

Figure 3. Results on several scenes obtained using GBPM. Here, each video segment is shown together with the descriptions, where words are highlighted, with different colors to indicate their different roles (*e.g.* nouns, verbs, etc). Trajectories that match to the nouns in the descriptions are shown using a sequence of boxes in gradually varying colors.

Table 2 shows several important observations: **(1)** GBPM, which seeks the optimal matching under the capacity constraints, consistently outperforms BASE. This can be seen more clearly in Fig. 4, which shows the F1-scores of GBPM and BASE in juxtaposition. **(2)** The performance of BASE degrades severely when tested on the real trajectories, where the F1-scores are below $0.2$. This suggests that the BASE method is very sensitive to feature noise, and relies on high quality features. In contrast, the performance of GBPM degrades much more gracefully in the presence of noisy features. Such resilience to noise is partly ascribed to the capacity constrains that we enforce in the LP problem. **(3)** For both methods, the configurations that combine multiple features generally work better than those that only use one feature type. For example, on real trajectories, the configuration *noun* + *verb* achieves an F1-score of $0.534$, higher than those obtained with *nouns* or *verbs* alone (which are $0.272$ and $0.519$). Additional incorporation of *adverbs* further pushes the F1-score to $0.567$. These results clearly indicate that the different features are complementary.

## 5.2. Retrieving Relevant Video Segments

Next, we consider the application of retrieving the relevant video segment given a text query. This is an even more challenging problem, as we do not have knowledge about the correspondence between descriptions and video segments. Our approach to this problem is to evaluate the total matching score between the given description and each video segment as the measurement of the relevance, i.e., $\sum_{uv} h_{uv}\hat{y}_{uv}$, where $\hat{y}_{uv}$ is the optimal solution to Eq.(1). Then, we sort the video segments in descending order of the total matching scores. A good ranking algorithm should place the most relevant segments to the top of the list.

We measured the performance using two different metrics. The first metric is the *average hit rate*, which is defined to be the relative frequency that the ground-truth segment is placed at one of the top-$K$ positions of the sorted list. Table 3 shows the results obtained on both GT trajectories and real ones, under all six configurations. We also compare them with random guesses. It is evident that our method yields substantially higher performance. Particularly, it raises the hit-rate by three times when working with GT trajectories, and two times with real trajectories, across different settings of $K$. Also, combining multiple types of features works better than using individual features alone.

We note that in KITTI videos, a query can be a good match to multiple segments. For example, a sentence *"a car is moving forward"* can match pretty well to many segments in a typical traffic video. Taking this into account, we consider another metric for performance assessment, namely the *average relevance*, which is defined as the average proportion of the top-$K$ segments that are truly relevant. To provide objective evaluation of the relevance of the results, we presented pairs of queries and video segments to independent annotators, and asked them to judge whether they are relevant or not. Table 4 shows the average relevance obtained on both GT and real trajectories with different configurations. Again, we observe that the proposed method considerably increases the relevance of the retrieved video, which clearly indicates that the total matching scores, which we used as the criteria in retrieval, are positively correlated with semantic relevance. The results in this table also corroborate our previous observation that combining complementary features improves the retrieval accuracy.

## 6. Conclusions

We have tackled the problem of semantic retrieval of videos using complex natural language queries, and demonstrated that our approach is able to locate objects in the video with high accuracy by parsing the lingual descriptions into a semantic graph that is then matched to the visual concepts by solving a linear program. In the future, we plan to further improve the performance by incorporating additional features and relations for both text and videos. We also consider connecting descriptions for different video segments to provide a coherent interpretation of an entire video and support context-aware retrieval.

**Acknowledgments:** We thank Kaustav Kundu, Wenjie Luo, Abhishek Sen, Liang-Chieh Chen, Botao Wang, and Jia Xu for helping us annotate the KITTI videos with descriptions and text-to-video alignment.

## References

[1] Y. Aytar, M. Shah, and J. Luo. Utilizing semantic word similarity measures for video retrieval. In *CVPR*, 2008. 2

[2] A. Barbu, A. Bridge, Z. Burchill, D. Coroian, S. Dickinson, S. Fidler, A. Michaux, S. Mussman, S. Narayanaswamy, D. Salvi, L. Schmidt, J. Shangguan, J. Siskind, J. Waggoner, S. Wang, J. Wei, Y. Yin, and Z. Zhang. Video-in-sentences out. In *UAI*, 2012. 2

[3] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan. Matching words and pictures. In *JMLR*, 2003. 2

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transaction on PAMI*, 24(24), 2002. 5

[5] J. Dalton, J. Allan, and P. Mirajkar. Zero-shot video retrieval using content and concepts. In *CIKM*, 2013. 2

[6] A. Farhadi, M. Hejrati, M. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences for images. In *ECCV*, 2010. 2

[7] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9), 2010. 3, 4

[8] S. Fidler, A. Sharma, and R. Urtasun. A sentence is worth a thousand pixels. In *CVPR*, 2013. 2

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1, 2, 3

[10] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *IVS (IV)*, 2011. 4

[11] G. Iyengar, P. Duygulu, S. Feng, P. Ircing, S. Khudanpur, D. Klakow, M. Krause, R. Manmatha, and H. Nock. Joint visual-text modeling for automatic retrieval of multimedia documents. In *Proc. of ACM Multimedia*, 2005. 2

[12] C. Kong, D. Lin, M. Bansal, R. Urtasun, and S. Fidler. What are you talking about? text-to-image coreference. In *CVPR*, 2014. 2

[13] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval. *ACM Trans. on Multimedia Computing, Comm., and Applications*, 2006. 2

[14] L. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding:classification, annotation and segmentation in an automatic framework. In *CVPR*, 2009. 1

[15] C. Matuszek, N. FitzGerald, L. Zettlemoyer, L. Bo, and D. Fox. A joint model of language and perception for grounded attribute learning. In *ICML*, 2013. 2

[16] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011. 3

[17] M. Rohrbach, W. Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele. Translating video content to natural language descriptions. In *ICCV*, 2013. 2

[18] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 1

[19] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2

[20] C. G. M. Snoek, B. Huurnink, L. Hollink, M. de Rijke, G. Schreiber, and M. Worring. Adding semantics to detectors for video retrieval. *IEEE Transaction of Multimedia*, 9(5):975–986, 2007. 2

[21] C. G. M. Snoek and M. Worring. Concept-based video retrieval. *Foundations and Trends in Information Retrieval*, 2(4):215–232, 2008. 2

[22] R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng. Parsing with compositional vector grammars. In *ACL*, 2013. 3

[23] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediciton models: A large margin approach. In *Proc. of ICML*, 2005. 5

[24] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, 2005. 5, 6

[25] D. Wang, X. Li, J. Li, and B. Zhang. The importance of query-concept-mapping for automatic video retrieval. In *Proc. of ACM Multimedia*, 2007. 2

[26] K. Yamaguchi, D. McAllester, and R. Urtasun. Robust monocular epipolar flow estimation. In *CVPR*, 2013. 4

[27] B. Yao, X. Yang, M. L. L. Lin, and S. Zhu. I2t: Image parsing to text description. In *PAMI*, 2010. 2

[28] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR'08*. 3