Mining visual actions from movies

Adrien Gaidon, Marcin Marszalek, Cordelia Schmid

Mining visual actions from movies

Adrien Gaidon[1]
http://lear.inrialpes.fr/people/gaidon/
Marcin Marszałek[2]
http://www.robots.ox.ac.uk/~marcin/
Cordelia Schmid[1]
http://lear.inrialpes.fr/people/schmid/

[1] LEAR
INRIA, LJK
Grenoble, France

[2] Visual Geometry Group
University of Oxford
Oxford, UK

Abstract

This paper presents an approach for mining visual actions from real-world videos. Given a large number of movies, we want to automatically extract short video sequences corresponding to visual human actions. First, we find commonly occurring actions by mining verbs extracted from movie transcripts. Next, we align the transcripts with the videos using subtitles. We then retrieve video samples for each action of interest. Not all of these samples visually characterize the action. Therefore, we propose to rank the retrieved videos by visual consistency. We first explore two unsupervised outlier detection methods: one-class Support Vector Machines (SVM) and finding the densest component of a similarity graph. As an alternative, we show how to obtain and use weak supervision. We investigate a direct use of binary SVM and propose a novel iterative re-training scheme for Support Vector Regression machines (SVR). Experimental results explore actions in 144 episodes of the TV series *Buffy the Vampire Slayer* and show: (a) the applicability of our approach to a large set of real-world videos, (b) how to use visual consistency for ranking videos retrieved from text, (c) the added value of random non-action samples, *i.e.*, the importance of weak supervision and (d) the ability of our iterative SVR re-training algorithm to handle mistakes in the weak supervision. The quality of the rankings obtained is assessed on manually annotated data for six different action classes.

1 Introduction

In this paper, we present an approach to automatically extract visual action samples from real-world videos. Such a system can visually discover which actions are performed and also permits to collect training data for action recognition. Following recent advances on action recognition in realistic videos [5, 15, 16, 18], we use movies and their transcripts to obtain video samples of visual actions. Related work of Cour *et al*. [5] focuses on temporal segmentation of TV series in a hierarchy of shots, threads and scenes and on character naming, while in [15, 16, 18] the authors address the task of action classification. In this paper, we concentrate on automatic action mining using both text and vision. We perform text-based retrieval of candidate action clips and then rank them visually (see Figure 1 for some examples). Similar approaches were explored to build collections of images for specific object classes [2, 10, 17, 24], and for naming characters in images [3, 20] and videos [9].

Figure 1: Key frames of the top 5 "walk" and "kiss" samples, automatically extracted from *Buffy* and ranked with our *iter-SVR* approach. "FP" is the first false positive (rank 30 for "walk", 37 for "kiss").

Movies and TV series often come with detailed textual information in the form of subtitles and transcripts. Transcripts cover both character dialogs and plot events describing what is happening in the video sequence. Subtitles are time-synchronized with the video stream and represent the dialogs. By mining frequent verbs in video transcripts, one can automatically identify actions that are frequently performed. Furthermore, after synchronizing the transcripts with the videos using subtitles, we can extract video clips for each action.

In practice, text-based retrieval processes often return visually irrelevant results and the main goal of this work is to eliminate such irrelevant action clips. To do so, we propose to rank the retrieval results based on visual consistency. Consistency can here be understood as coherence and is relative to a member of a set of samples. Defining relevance as consistency is valid under the loose assumption that a significant part of the retrieved samples are representative for the queried concept and that these clips share some common characteristics.

To quantify the visual consistency of a video clip in a retrieved list, we first consider two unsupervised outlier detection methods. An outlier can be defined as an observation that appears to "deviate markedly from other members of the sample in which it occurs" [1], or as an observation which appears to be "inconsistent with the remainder of the data" [1]. Therefore, outlier detection methods can be used to rank samples by consistency. We explore two unsupervised ranking algorithms: one-class Support Vector Machines [22] and finding the densest component of a similarity graph [20].

Second, we propose to automatically rank clips using supervised techniques to discriminate between consistent and inconsistent samples. In order to use supervised approaches without requiring expensive manual annotations, we investigate the use of annotated data that can be automatically obtained, but might contain incorrect annotations. Such a weak supervision can generally be easily obtained. In our setup, the subset of documents we want to rank can be considered as positives and randomly sampling the rest of the video collection is an inexpensive and automatic way to model negative samples. We explore two methods that use such weak supervision in order to obtain a ranking function reflecting visual consistency. We first investigate the efficiency of standard binary Support Vector Machines (SVM). We then propose a novel iterative re-labeling and re-training heuristic for Support Vector Regression (SVR [8, 26, 27]), referred to as *iter-SVR*. Our approach is related to the maximum margin clustering paradigm [30, 32]. The efficiency and the strong improvement potential of starting from binary target values and using an iterative re-training of SVR was previously shown in [32]. Differently from previous works, we operate in a supervised setup and explicitly deal with weakly labeled data.

To evaluate our approach, we choose the TV show *Buffy the vampire slayer* and perform large-scale action retrieval. *Buffy* consists of 7 seasons (39 DVDs, 144 episodes), *i.e.*, over

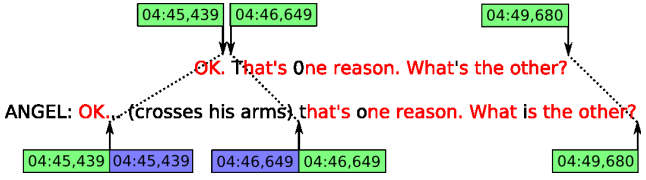


Figure 2: Aligning subtitles (above) and transcript (below). Characters belonging to the longest common subsequence are in red.

100 hours of video, which equals to approximatively ten million frames. The episodes show a large variety of actions, characters, scenes, perspectives and imaging conditions. This TV show has a large community of fans and transcripts for all of the 144 episodes are easy to obtain.

The paper is structured as follows. In section 2, we explain how we mine actions from transcripts and extract video clips with corresponding action labels. In section 3, we detail our video representation and our visual consistency ranking methods: one-class SVM, densest component, ν-SVM and *iter-SVR*. The experimental results on the *Buffy* episodes are presented in section 4. We conclude the paper with section 5.

2 Text-based mining of actions from videos

In the following, we describe how we use the textual information provided by subtitles and transcripts to temporally segment the videos into short (a few hundreds of frames) video sequences and to efficiently extract action samples.

2.1 Text-driven temporal segmentation of movies

Our approach for temporally segmenting movies into meaningful video clips of only a few seconds consists of two steps: subtitle extraction and alignment of subtitles with transcripts.

Subtitle extraction. First, we extract the subtitles from the videos. We obtain a uniform and high quality source of dialog information, which is temporally synchronized with the movies. For many videos, including our *Buffy* episodes, the DVD format encodes subtitles as images. We use an optical character recognition (OCR) algorithm to extract the subtitles, here the Tesseract OCR system [21] developed by Hewlett-Packard and Google. Note that there are still occasional recognition errors, which need to be taken into account by the subsequent alignment with the transcripts.

Aligning subtitles and transcripts. In the second step, we transfer the temporal information associated with subtitles (which is when the dialogs occur) to the transcripts (which do not contain any temporal information). We align the dialog lines contained in the transcripts with the subtitle lines based on text matching. We first find the longest common subsequence [11] in the two character streams obtained from OCRed dialogs and plain-text scripts. We then transfer the timestamps from subtitle characters to matching transcript characters and extend those to sentences, thus obtaining transcripts temporally aligned with the video (*c.f.* figure 2). Note that aligning characters improves over aligning words, as it can handle errors of the OCR algorithm.

We obtain a temporal segmentation of the video by considering each clip delimited by two consecutive dialogs. Non-dialog lines extracted from the temporally aligned transcript form a short textual description attached to each video clip. We will now show how we use this textual information to efficiently mine actions.

2.2 Mining actions from transcripts

Given the text describing the clips, we first parse the sentences using the link grammar [13]. This produces a parse-graph of the sentences, which determines semantic links between words. We use this grammar to extract all verbs present in the transcript extracts and retrieve all clips corresponding to these verbs.

Our temporal segmentation of the *Buffy* episodes results in a set of over 15000 video samples with textual descriptions. From these, we automatically mine a set of commonly occurring verb expressions and retrieve the video sequences corresponding to them. Figure 3 shows the statistics for the most frequent verbs retrieved by our system.

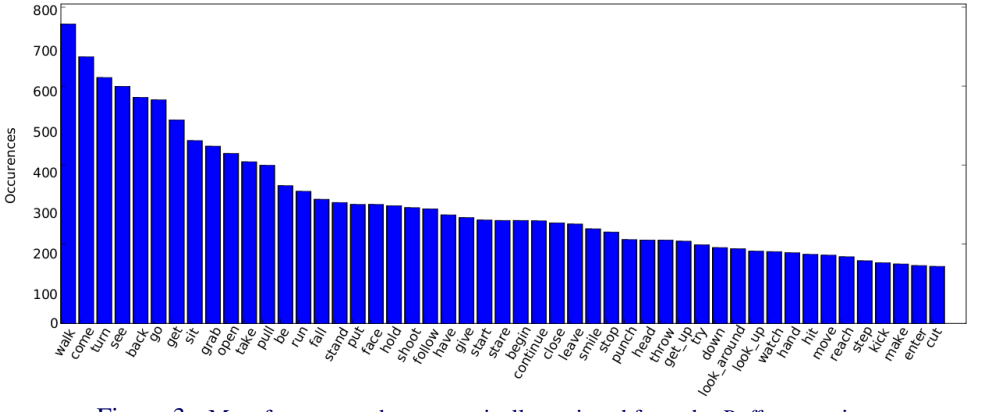


Figure 3: Most frequent verbs automatically retrieved from the *Buffy* transcripts.

3 Ranking action samples by visual consistency

In a set of textually relevant clips, some might not be visually representative for the action of interest. In this section, we explain how we rank video samples based on their visual consistency. First, we describe how we represent the visual content of our video sequences using bags of spatio-temporal features. We then explain how we use the one-class SVM and densest component search to estimate an inconsistency score in an unsupervised way, *i.e.*, using only the retrieved examples. Finally, we exploit weak supervision with "background" samples in order to rank the video clips. We use a binary SVM or iteratively re-train a Support Vector Regression machine (SVR).

3.1 Visual representation of video sequences

We use spatio-temporal bag-of-features [7, 12, 19, 25, 29] to represent video sequences. The first step is to extract sparse space-time features which have recently shown good performance for action description [7, 12, 16, 18, 19, 25]. Local features provide a compact video representation and tolerance to background clutter, occlusions and scale changes. Here we follow [14] and detect interest points using a space-time extension of the Harris operator. However, instead of performing scale selection as in [14], we use a multi-scale approach and extract features at multiple levels of spatio-temporal scales (σ_i^2, τ_j^2) with $\sigma_i = 2^{(1+i)/2}, i = 1,...,6$ and $\tau_j = 2^{j/2}, j = 1,2$.

To describe the extracted interest points, we compute histogram descriptors of space-time volumes in the neighborhood of the detected points. The size of each volume $(\Delta_x, \Delta_y, \Delta_t)$ is

related to the detection scales by $\Delta_x, \Delta_y = 2k\sigma$, $\Delta_t = 2k\tau$. Each volume is subdivided into a (n_x, n_y, n_t) grid of cuboids. For each cuboid we compute a coarse histogram of oriented gradients (*HoG*) [6, 15]. Normalized histograms are concatenated into HoG descriptor vectors and are similar to the well known SIFT descriptor. We use parameter values $k = 9$, $n_x, n_y = 3$, $n_t = 2$.

Given a set of spatio-temporal features, we construct a visual vocabulary. In our experiments we cluster, with the k-means algorithm, a subset of 10^5 features sampled from the videos. The number of clusters is set to $k = 1000$, which has shown empirically to give good results and is consistent with the values used for static image classification. We then assign each feature to the closest vocabulary word (we use Euclidean distance) and compute the histogram of visual word occurrences over a space-time volume corresponding to the video clip. Following recent works on action recognition using bag of spatio-temporal features [16], we use a χ^2 kernel [31] to measure the similarity between bag-of-features representations.

3.2 Unsupervised estimation of inconsistency

In this section, we present two unsupervised consistency ranking methods, based on two outlier detection algorithms: a distance-based approach using one-class SVM, and a density-based approach using densest component search.

One-class SVM. One-class SVM [22, 23] is an outlier detection algorithm considering observations far from the majority as outliers. Contrary to the standard formulation of SVM, one-clas SVM is an unsupervised approach requiring no labeled data. The goal is to estimate the support of the underlying distribution of the data. The main idea is to find a hyper-sphere, in a high-dimensional space induced by a kernel, around the main consistent part of the data. This sphere should be as small as possible, while including most of the data. Given our retrieved videos $\{\mathbf{x_i},\ i = 1..n\}$ and a kernel K (a χ^2 kernel in our case), the confidence value obtained with the one-class SVM for a video $\mathbf{x}$ is: $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x_i}, \mathbf{x}) - b$, where the α_i are the Lagrange multipliers obtained by solving the dual of the one-class SVM optimization problem and b is the offset from the origin [22]. The decision function (the hyper-sphere's boundary) is $sign\big(f(\mathbf{x})\big)$. We consider as a consistency score the opposite of the distance from this boundary of normality, *i.e.*, the opposite of the distance from the margin obtained by the SVM: $-\frac{f(\mathbf{x})}{||\mathbf{w}||}$, where $\frac{2}{||\mathbf{w}||}$ is the width of the margin.

Densest component. The second unsupervised outlier detection method we use is based on the notion of density. Outliers are in this case defined as observations lying in locally sparse regions of the feature space. The idea is to represent visual similarities between the retrieved samples as a graph structure and to find the set of most similar samples by searching for the densest component of this graph [20].

Let $\{\mathbf{x_i},\ i = 1..n\}$ be our retrieved samples and K a similarity measure between video clips (here a χ^2 kernel). First, we construct a complete undirected graph $G = (V, E)$, where the nodes $V = \{1, \ldots, n\}$ represent the retrieved samples and the edges $E = \{(i, j) : i, j \in V\}$ are weighted by the similarity between the concerned nodes: $w_{i,j} = K(\mathbf{x_i}, \mathbf{x_j})$. Let the degree of a vertex v in G be: $deg_G(v) = \sum_{j \in V} w_{v,j}$. The density of a subgraph $G(S)$ of G, induced by a set of nodes $S \subset V$, can be defined as its average degree: $avg\_deg(G(S)) = \frac{\sum_{v \in S} deg_{G(S)}(v)}{card(S)}$, where $card(S)$ is the cardinality of S. The set of the most consistent samples can then be represented by the densest component of the graph G, *i.e.*, the subgraph of G with maximal density.

In practice, we use a greedy 2-approximation algorithm [4] to find the densest subgraph of G. It consists in iteratively deleting the node with minimal degree and updating the graph until all nodes are removed. The densest component is then the subgraph obtained at the iteration where the density is maximal. We rank our video samples by the inverse pruning order in this greedy algorithm. Consequently, the most inconsistent samples are the ones with minimal degree, *i.e.*, those lying in the sparsest regions.

3.3 Ranking with weak supervision

As an alternative to the unsupervised approaches discussed above, we propose to use supervised techniques with weakly annotated data. Compared to the previous unsupervised setup, here we consider the retrieved clips as positives and obtain negative observations by randomly selecting video samples not retrieved for the current action. We call this set of inconsistent clips the "background" (or negative) class. In the following, we explain how we use this weak supervision, either directly with a binary SVM, or by explicitly handling erroneous training data with a novel iterative re-training algorithm for regression.

Binary ν-SVM. SVMs are successful binary classifiers and a reformulation of the standard SVM optimization problem, called ν-SVM [28], is particularly adapted to handle outliers. Basically, it just reformulates the traditional SVM optimization problem by replacing the usual regularization parameter, often denoted C, by another parameter, denoted ν. It belongs to $(0,1]$ and is an upper bound on the fraction of outliers and a lower bound on the fraction of support vectors [28]. Consequently, this parameter can be used to deal with an expected amount of erroneous training data and, therefore, increase the robustness to outliers. Similarly to the one-class SVM case, we use a χ^2 kernel and the distance from margin as a measure of visual inconsistency.

However, the fact that we use weak supervision is not explicitly taken into account with this approach. In the following, we present a ranking heuristic that handles erroneous training data provided by a weak supervision.

Iterative SVR. We propose to explicitly deal with the erroneous training data, by re-labeling inconsistent retrieved samples and iteratively re-training a Support Vector Regression machine (SVR) [8, 27]. We refer to this algorithm as *iter-SVR*. Instead of using the previously defined classification approach, we formulate the ranking problem as a regression one. Our goal is to learn, from the weakly annotated data, a function that assigns high values to consistent samples and low values to the others. There are three main reasons that motivate our choice of using regression instead of classification in this case. First, with the reformulation of our problem as iteratively re-trained regression, there is no need to define a parameter controlling the number of samples we re-label at each iteration. We can re-label all of the retrieved samples at each iteration. Second, regression allows soft re-labeling, as target values for an SVR can be continuous, whereas in the SVM case, the labels are necessarily binary. Finally, as we re-label multiple samples at each iteration, we can quickly converge to a stable solution. The *iter-SVR* algorithm is summarized in table 1.

The first step of our *iter-SVR* algorithm is to assign the target values $+1$ to all of the retrieved samples and -1 to all of the "background" samples. We then train an SVR on this data and compute the obtained regressed values for the retrieved samples. The SVR, not differentiating between good and wrong supervision, will be misguided by the erroneous target values. This will result in regression values inducing a poor ranking. Nevertheless, the generalization capabilities of the SVR should allow some of the most inconsistent target

1.	Initialization:	Retrieved samples $P = \{(\mathbf{x}_i^+, +1)\}$ Random negatives $N = \{(\mathbf{x}_i^-, -1)\}$
2.	Learn regression function f on $P \cup N$ with SVR (normalize outputs)	
3.	Replace target values of retrieved samples $P \leftarrow \{(\mathbf{x}_i^+, f(\mathbf{x}_i^+))\}$	
4.	Go back to 2. until convergence	

Table 1: The *iter-SVR* algorithm

values to clearly shift from the original ones, thus reducing the misleading factors of this supervision. Consequently, we consider the normalized regression outputs of the SVR as new target values for our retrieved samples. This means that we relax the binary constraints on the target values in order to make smoother decisions and interpret the decision values as measures of consistency. We then re-train the SVR with this new improved supervision and repeat these steps until convergence, *i.e.*, until the regression outputs coincide with the target values. At each iteration, the quality of the supervision increases, *i.e.*, the concept of what is consistent improves. Therefore, the ranking induced by the regression values also improves.

4 Experiments

In the following section we present our evaluation protocol and show experimental results for the text-based retrieval and visual ranking.

4.1 Evaluation procedure

We evaluate our ranking algorithms on six different action classes: 'walk', 'fall', 'punch', 'kick', 'kiss' and 'get-up'. We manually selected these classes from the automatically retrieved verbs obtained by our text mining approach described in section 2. These actions were chosen because they occur frequently in the *Buffy* episodes and because they represent different interesting visual events with different characteristics (duration, context, ambiguity, etc.). To perform a quantitative evaluation of the performance of our methods, we considered and manually annotated the 100 shortest text-retrieved samples for each action (except 'kiss' for which only 87 samples were retrieved from the text). Some samples, labeled as unclear, were not considered for performance evaluation. Table 2 reports the ground truth statistics resulting from this annotation. We also randomly sampled 100 "background" samples to constitute the negative class for our weakly supervised approaches. In our experiments, we only used the 587 clips mentioned in table 2 and the 100 "background" samples. We restricted ourselves to the 100 shortest samples per class in order to limit the annotation effort. Furthermore, our experiments show that this amount of data is sufficient to learn our visual models. We assess the quality of the ranking results by the Average Precision (AP) associated with the ranked lists (mean of the precision over all recall rates). We also show curves representing precision at different cut-off ranks (precision of the top 5 results, top 10, ...).

	walk	fall	punch	kick	kiss	get up
true positives	80	59	72	73	71	74
false positives	12	33	23	21	8	19
unclear (unused)	8	8	5	6	8	7

Table 2: Ground truth statistics for the human action samples retrieved from *Buffy*.

	walk	fall	punch	kick	kiss	get up	Mean
Iterative SVR	**96.8**	**81.0**	**91.4**	**95.1**	**97.1**	87.2	**91.4**
ν-SVM	96.4	77.6	89.7	94.5	96.6	86.9	90.3
One class SVM	93.6	76.5	91.0	93.2	95.1	85.8	89.2
Densest component	93.2	73.6	90.7	93.0	94.6	**89.2**	89.1
Text	87.0	64.1	75.8	77.7	89.9	79.6	79.0

Table 3: Average Precisions of the ranking obtained for six human actions chosen from *Buffy*, with the iterative SVR, binary nu-SVM, one-class SVM, densest component and text approaches.

4.2 Experimental results

Table 3 compares our different ranking methods to text-based retrieval. We can observe that all the visual ranking algorithms significantly improve the results compared to the text retrieval results. The gain in mean AP is on average +12.4% for our best method, *iter-SVR* and is maximal for the action 'fall' (+17%). Our experimental evaluation highlights the performance of our *iter-SVR* algorithm. It outperforms all other methods, including the ν-SVM, on average and on five out of six classes. It clearly improves the ranking quality, especially for the top ranks (*c.f.* figure 4). Figure 5(a) illustrates the improvement due to the iterative re-labeling and re-ranking process performed by the *iter-SVR* algorithm, by showing the significant gain in performance between the first iteration and the last one. Furthermore, this method converges quickly to a stable solution, generally in less than 20 iterations (as shown for the class 'fall' in figure 5(b)).

Our experiments also demonstrate that weakly supervised methods outperform the unsupervised ones. Using supervised approaches with imprecise annotations yields better results than fully unsupervised approaches (up to +7.4% for the action 'fall'). This holds even for the ν-SVM (*i.e.*, without re-training): it performs +1.1% better (in mean AP) than the best unsupervised method. Furthermore, the weak supervision we use is obtained automatically and therefore, our weakly supervised algorithms are directly comparable to unsupervised

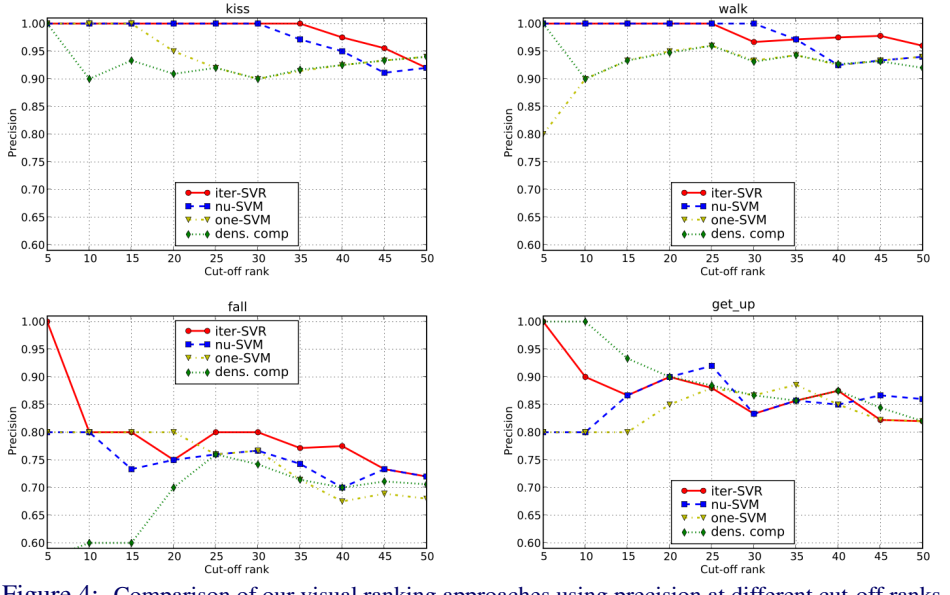
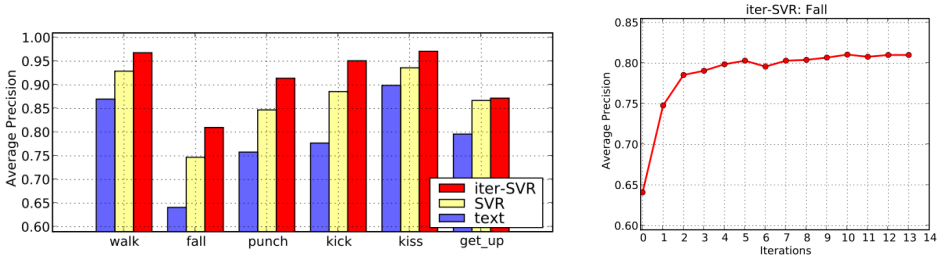


Figure 4: Comparison of our visual ranking approaches using precision at different cut-off ranks.



(a) (b)

Figure 5: (a) Performance (Average Precision) of our iterative SVR approach ("iter-SVR"), the SVR without re-training ("SVR") and the text mining results. (b) AP at each iteration of the *iter-SVR* algorithm, starting from the text results, for the action 'fall'.

methods. Consequently, we show that, in our context, adding random negative samples improves the visual ranking of the retrieved samples. There is only one exception. For the class 'get up', the unsupervised densest component search outperforms all other methods. Unlike for the other actions, the margin between consistent and inconsistent 'get up' clips is not large. Outliers, in this case, are more likely to be found in locally sparse regions of the feature space.

5 Conclusion

In this paper we have presented an approach for mining action clips from large real-world video collections. Our method does not require manually annotated examples and uses both textual and visual information. We have shown that the problem of visually ranking videos retrieved from text can be efficiently addressed using visual consistency. Our method can retrieve action samples which are semantically and visually consistent.

Furthermore, we have demonstrated that weakly supervised algorithms, using random negative samples, are better for ranking than unsupervised methods. In this context, a novel iterative re-labeling and re-training regression algorithm has been proposed. We have shown that this algorithm efficiently benefits from weak supervision and outperforms commonly used approaches.

For the future, we plan to improve our retrieval process by incorporating more information from the text (like actors, objects and locations) and combining density-based and distance-based inconsistency definitions. We also plan to investigate how to use visual consistency to estimate the visual "learnability" of concepts retrieved from text.

Acknowledgements. This work was partially funded by the European research project CLASS and the MSR/INRIA joint project.

References

[1] V. Barnett and T. Lewis. *Outliers in statistical data*. John Wiley and Sons, 1994.

[2] T.L. Berg and D.A. Forsyth. Animals on the web. In *CVPR*, 2006.

[3] T.L. Berg, A.C. Berg, J. Edwards, M. Maire, R. White, Y.W. Teh, E. Learned Miller, and D.A. Forsyth. Names and faces in the news. In *CVPR*, 2004.

[4] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. *LNCS*, pages 84–95, 2000.

[5] T. Cour, C. Jordan, E. Miltsakaki, and B. Taskar. Movie/Script: Alignment and Parsing of Video and Text Transcription. In *ECCV*, 2009.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[7] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, 2005.

[8] H. Drucker, C.J.C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In *NIPS*, 1997.

[9] M. Everingham, J. Sivic, and A. Zisserman. Hello! My name is... Buffy – Automatic Naming of Characters in TV Video. In *BMVC*, 2006.

[10] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from Google's image search. In *ICCV*, 2005.

[11] D. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, 1975.

[12] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *ICCV*, 2007.

[13] J. Lafferty, D. Sleator, and D. Temperley. Grammatical trigrams: A probabilistic model of link grammar. In *AAAI Conf. on Probabilistic Approaches to Natural Language*, 1992.

[14] I. Laptev. On space-time interest points. *IJCV*, 64(2/3):107–123, 2005.

[15] I. Laptev and P. Pérez. Retrieving actions in movies. In *ICCV*, 2007.

[16] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.

[17] L.J. Li, G. Wang, and L. Fei Fei. Optimol: automatic online picture collection via incremental model learning. In *CVPR*, 2007.

[18] M. Marszałek, I. Laptev, and C. Schmid. Actions in context. In *CVPR*, 2009.

[19] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *BMVC*, 2006.

[20] D. Ozkan and P. Duygulu. A graph based approach for naming faces in news photos. In *CVPR*, 2006.

[21] S. Rice, F. Jenkins, and T. Nartker. The fourth annual test of OCR accuracy, 1995.

[22] B. Schölkopf, R. Williamson, A. Smola, and J. Shawe-Taylor. SV estimation of a distribution's support. In *NIPS*, 1999.

[23] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[24] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. In *ICCV*, 2007.

[25] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, 2004.

[26] A.J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[27] V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation, and signal processing. In *NIPS*, 1997.

[28] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2000.

[29] S.-F. Wong, T. K. Kim, and R. Cipolla. Learning motion categories using both semantic and structural information. In *CVPR*, 2007.

[30] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *NIPS*, 2005.

[31] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 73(2): 213–238, 2007.

[32] K. Zhang, I.W. Tsang, and J.T. Kwok. Maximum margin clustering made practical. In *ICML*, 2007.