Recognizing Manipulation Actions in Arts and Crafts Shows using Domain-Specific Visual and Textual Cues

Benjamin Sapp
University of Pennsylvania
Philadelphia, PA, USA
bensapp@cis.upenn.edu

Rizwan Chaudhry
Johns Hopkins University
Baltimore, MD, USA
rizwanch@cis.jhu.edu

Xiaodong Yu
Comcast Corp.
Washington, DC, USA
xiaodong_yu@cable.comcast.com

Gautam Singh
George Mason University
Fairfax, VA, USA
gsinghc@cs.gmu.edu

Ian Perera
University of Rochester
Rochester, NY, USA
iperera@cs.rochester.edu

Francis Ferraro
Johns Hopkins University
Baltimore, MD, USA
ferraro@cs.jhu.edu

Evelyne Tzoukermann
The MITRE Corp.
McLean, VA, USA
tzoukermann@mitre.org

Jana Kosecka
George Mason University
Fairfax, VA, USA
kosecka@cs.gmu.edu

Jan Neumann
Comcast Corp.
Washington, DC, USA
jan_neumann@cable.comcast.com

Abstract

We present an approach for automatic annotation of commercial videos from an arts-and-crafts domain with the aid of textual descriptions. The main focus is on recognizing both manipulation actions (e.g. cut, draw, glue) and the tools that are used to perform these actions (e.g. markers, brushes, glue bottle). We demonstrate how multiple visual cues such as motion descriptors, object presence, and hand poses can be combined with the help of contextual priors that are automatically extracted from associated transcripts or online instructions. Using these diverse features and linguistic information we propose several increasingly complex computational models for recognizing elementary manipulation actions and composite activities, as well as their temporal order. The approach is evaluated on a novel dataset of comprised of 27 episodes of PBS Sprout TV, each containing on average 8 manipulation actions.

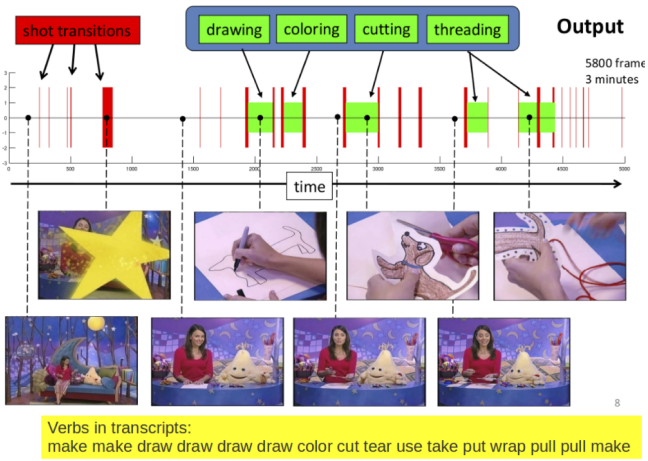

Figure 1. Example video with the desired annotations in green, example screen shots for different scene categories below the time line, and some of the action verbs contained the transcript in yellow.

1. Introduction

In this paper we seek to develop techniques for automated annotation and labeling of video data with the aid of textual descriptions. With the large amounts of video being generated every day, it becomes essential to develop techniques for indexing and organizing this data so that it is easy to search and browse. While current video browsing methodologies are time-based, there are many more intuitive ways to organize the video content, e.g. based on the natural semantic content of scenes, actions, people and events present. The type of annotations which one seeks and the techniques developed to acquire them vary greatly with the domain and the types of videos available.

We focus on the arts-and-crafts domain and the annotation of manipulation actions commonly encountered in

2011 IEEE International Conference on Computer Vision Workshops
978-1-4673-0063-6/11/$26.00

them. This domain has several novel and interesting characteristics: each activity is naturally composed of basic action units necessitating reasoning over different time scales; each action involves characteristic manipulation motions, hand poses, tools and objects; and the activities have accompanying text instructions and transcripts which one can use for contextual grounding. The goal is annotations like the ones shown in Fig. 1.

Manipulation actions are difficult to recognize and often ambiguous because they are defined by rapid changes in finger and hand poses, as well as the movement and appearance change of the manipulated objects. This makes it necessary to include contextual constraints and domain priors into the recognition process. There is a tight interaction between the type of motion that hands or the human body undergo, the shape of the hand and tool being held, as well as the object being manipulated. We study and model these interactions and cues explicitly and exploit them in an action recognition framework. In the first stage we use state-of-the-art object detectors [3], motion descriptors [10] and hand pose estimators to gather some evidence about presence of individual cues in the keyframes and also propagate this evidence via tracking and detection across the sequence. We then model the interactions between these video-derived cues and text-derived contextual and domain constraints in a Conditional Random Field [8] and formulate video annotation as a problem of most likely sequence assignment given the available evidence.

Along with the development of computational models for recognition of complex manipulation actions using contextual cues, we explore how one can apply natural language processing techniques to external textual descriptions such as transcripts, plot summaries, cooking recipes or craft instructions to automatically mine both the semantic and temporal information required for the annotation of the actions in the video.

Since there are no existing benchmark datasets focusing on the recognition of complex manipulation actions, as part of our efforts we created our new baseline dataset (see Fig. 2 for some examples) for research in this challenging area, which we hope will be used as a benchmark dataset for future research. We further propose an end-to-end system that automatically annotates real-world broadcast videos with the presence of actions and objects. Both the dataset and the code will be publicly available, thereby reducing the barrier of entry for further research.

2. Related work

There is a long history of human motion analysis in computer vision. The surveys by Gavrila ([4]) and more recently by Moeslund et al., ([13]), provide a broad overview of more than three hundred papers and numerous approaches for analyzing human motion in videos, including human motion capture, tracking, segmentation and recognition. Most of the work in activity recognition can be divided into two classes: 1) collections of local models and 2) global models. Local models compute a collection of spatio-temporal interest points such as the ones defined in [2, 9] and compute a descriptor based on intensity, optical flow and their gradients in a spatio-temporal cuboid centered at each interest point. On the other hand, global models for human actions compute statistics of motion and intensity over the whole frame or an extracted human skeleton or silhouette(e.g. [16]).

Joint modeling of actions and objects has recently also become a topic of interest. Early work of [14] looked at recognition of actions and objects in an HMM framework, using video feeds from ceiling-mounted cameras by observing typical office and kitchen activities. More recent work of [5, 7] focused on interactions between objects and manipulation actions and better classification and localization of these in a Bayesian framework, using a relatively small number of sequences recorded in a laboratory. Action recognition in movies using scene context has been demonstrated for head/whole body actions [12] using scripts aligned to videos as level of supervision. Authors in [6] used Multiple Instance Learning using action, object and scene context features for human action recognition from YouTube videos, focusing again on whole body actions relying on good human detectors. Recent work by [18] use jointly object detectors and human pose detectors to classify the object and pose (action performed) of the human in the single image setting. With the exception of few, most of the work in the area of automatic action recognition considers whole body actions such as walking, running, jumping or whole body manipulation action sequences captured in the laboratory. The use of commercial videos and unstructured textual descriptions used in our work creates new challenges and opens new avenues for combining natural language processing and computer vision techniques for human action recognition.

Our work is also related to the problem of multi-media information retrieval, although our work differs significantly from standard approaches in that field that combine video and text information (e.g. see [17] for a recent review of the field). Typically when text is used in this type of multimedia applications, it is simply treated as a feature like any other, and the grammatical structure and relationship between actions/tools and verbs/objects is not analyzed. We see our work more as an extension of recent work that uses probabilistic models to combine text and visual features for activity recognition (e.g. [5, 12, 15]).

3. Dataset

Our dataset consists of commercially available broadcast-quality videos and their associated transcripts

Figure 2. Examples of manipulation actions in the PBS dataset.

(closed captions) from a TV show aimed at children, demonstrating a variety of arts and crafts projects[1]. The commercial videos of arts and craft shows typically have available transcripts and/or textual descriptions of tasks to be accomplished, and have clear segmentation boundaries so that they can be partitioned into shots. Objects do not have large scale variation and activities are typically observed from a limited number of viewpoints. These constraints make commercial broadcast videos more applicable to the task of action annotation compared to unconstrained consumer-generated content as found on YouTube for which much less contextual information is available. Overall the dataset is made up of 27 episodes, each of them 3-5 minutes long.

As described later in the implementation Sec. 6, we automatically split the video into different segments according to the viewpoint of the camera (e.g. *Zoomed-in* or *Zoomed-out*). The *Zoomed-out* shots typically include conversations of the host and do not contain any visual evidence of manipulation actions. Hence we focus our recognition and annotation task on the subset of *Zoomed-in* shots. These typically show human hands in motion, holding tools and manipulating and transforming objects. Examples of representative shots for different shot categories and desired annotations that we want our system to create are shown in Fig. 1. An episode contains on average 8 such manipulation actions, resulting in 220 total action shots, and 43K total frames of action. Each shot was annotated with one action and one tool class label. Example actions with their corresponding labels can be seen in Fig. 2.

4. Single Shot Action and Tool Recognition

The focus of this work lies in the combination of video cues with external information such as contextual constraints derived from text and domain knowledge. Due to the complex nature of the data, we propose to use several low and mid-level features extracted from videos to aid the classification. These features will include action descriptors based on local spatial temporal interest point signatures $f_{STIP}(x)$ [9], the absence and presence of specific object categories $f_{tool}(x)$, hand poses $f_{hand}(x)$ and domain-specific contextual priors extracted from textual descriptions.

In this section we describe models of increasing complexity to simultaneously classify actions *and* tools in a single shot. We treat each zoomed-in shot in our dataset as a single example, which we wish to map to a single action category (e.g., Cut) and a single tool category (e.g., $Scissors$). We assume a fixed finite list of actions and tools (e.g. automatically extracted from the text based on a domain specific dictionary), and model the rest of the possibilities with action category $Other$ and tool category $Other/None$. We first discuss a straightforward combination via supervised multi-class machine learning methods. Next, we propose a joint model for inferring actions and tools, which can explicitly model the co-occurrence relations between different actions and tools. Finally, we show that we can incorporate prior domain knowledge into our joint model, which allows our system to scale to different and larger domains with minimal human supervision.

4.1. Independent modeling

Let $f_{hand}(x)$, $f_{tool}(x)$, and $f_{STIP}(x)$ be our three sources of features, described in detail in Section 6, for an example shot x. Let A be a set of action labels we are interested in applying. In our setting, $A = \{ Color, Cut, Draw, Glue, Paint, Other\}$. A standard way to model a multi-label classification task is with a linear function of the features for each class $a \in A$ as $g^a(x) = w^a \cdot f(x)$, where $g^a(x)$ is a score for example x having label a, and $f(x)$ is a vector of features for example x, which can be some or all of $[f_{hand}(x); f_{tool}(x); f_{STIP}(x)]$[2]. Using a labeled dataset, we can learn a set of parameters w^a for each class $a \in A$ using a one-class-versus-rest type loss function. At test-time, the most likely action label $a^\star$ can be obtained by $a^\star = \arg\max_{a \in A} g^a(x)$. Similarly for example x we wish to determine which tool $t \in T$ is being used where T is the set of labels $\{Brush, GlueBottle, WritingTool, Scissors, None\}$. We can learn linear parameters w^t for each tool, and classify with $t^\star = \arg\max_{t \in T} g^t(x) = \arg\max_{t \in T} w^t \cdot f(x)$.

4.2. Joint action-tool modeling

Clearly, different actions are highly correlated with the use of different tools. For example, a strong signal for a particular action (e.g., Cut) may help with an ambiguous signal for what tool is present (e.g., $Scissors$), and vice

[1] www.sproutonline.com

[2] We use the convention that vectors in d dimensions are $d \times 1$ (vertical), and use notation $[x; y]$ to mean the concatenation of vectors x and y.

versa. In light of this, we propose to model the joint probability distribution over possible actions and tools for each example: $p(A = a,\ T = t \mid x)$. We decompose this distribution into factors for how likely each action and tool are independently, as well as a term which explicitly encodes the likelihood of each possible (action,tool) pair. One of the important things this probabilistic model allows for is the incorporation of explicit, prior domain knowledge about action and tool co-occurrences. In Section 4.3 we will show that this knowledge can be obtained at little or no cost—as is the case when automatically extracting it from web text—which significantly reduces the amount of human work labeling data. This is of critical importance when scaling up to larger or more varied domains and when dealing with sparse annotations that can provide only unreliable estimates of action-tool co-occurrences.

We model $p(A,\ T \mid x)$ as a log-linear conditional random field ([8]):

$$p(A = a,\ T = t \mid x) = \tag{1}$$
$$\frac{1}{Z(x)} \exp\Big(w_A \cdot f_A(a, x) + w_T \cdot f_T(t, x) + w_{A,T} \cdot f_{A,T}(a, t)\Big),$$

where w_A, f_A and w_T, f_T correspond to action (respectively tool) parameters and features, and $w_{A,T}, f_{A,T}$ correspond to action-tool co-occurrence parameters and features. The term $1/Z(x)$ is a normalization constant which ensures the distribution sums to 1 over all (action, tool) pair possibilities. Next we describe each term in our model, as well as inference and learning procedures.

Unary terms $w_A \cdot f_A$ and $w_T \cdot f_T$: We set $f_A(a, x) = [g^a(x);\ e_a]$ and $f_T(t, x) = [g^t(x);\ e_t]$, using the notation e_i to denote the indicator vector with a 1 in the i^{th} dimension and zeros elsewhere. Thus our CRF features are the outputs of the independent action and tool models described in Section 4.1, in conjunction with class identity features e_a and e_t which allow the model to learn a prior likelihood of each class occurring (e.g., that $Draw$ occurs more frequently than $Paint$). Using these features, the model learns parameters w_A and w_T to balance the independent beliefs of different actions, tools, and class priors with the belief in action-tool co-occurrences, described next.

Pairwise action-tool co-occurrence term $w_{A,T} \cdot f_{A,T}$: It is intuitive to think of the term $\exp(w_{A,T} \cdot f_{A,T}(a, t))$ in the form of an action-tool compatibility matrix: for every action-tool pair, it contains a corresponding real-valued score reflecting how likely the pair is to go together (e.g., Cut-$Scissors$ should be very likely, Cut-$Brush$ very unlikely). Thus we need to specify how to learn the entries of the action-tool compatibility matrix. We explore two different approaches: *(1) Direct estimation of action-tool compatibility.* In this case we directly learn every entry in the action-tool compatibility matrix from our groundtruth action-tool co-occurrences. To do this we express $f^{direct}_{A,T}(a, t) = [e_{(a,t)}; 1]$. The last component is a bias term to balance the values with the other terms in the CRF. The vector $e_{(a,t)} \in \mathbb{R}^{|A||T|}$ has a 1 in the $(a, t)^{th}$ dimension and zeros elsewhere, simply indicating the identity of the (action, tool) pair. *(2) Action-tool compatibility via outside domain knowledge.* We assume domain knowledge comes in the form of co-occurrence matrices C^k, where $C^k_{a,t}$ is the real-valued entry in the k^{th} co-occurrence matrix for action a and tool t. These are obtained using natural language processing techniques described in Section 4.3. We incorporate outside sources of information as a weighted combination of these co-occurrence matrices, where the learned weights reflect the usefulness / willingness to "trust" this knowledge compared to the other terms in the model. To accomplish this we set $f^{domain}_{A,T}(a, t) = [C^1_{a,t};\ \ldots;\ C^K_{a,t}; 1]$. Note that because of the nature of our discriminatively-trained models, the action-tool co-occurrence values cannot be interpreted as joint probabilities, nor is the *direct estimation* approach as simple as computing ground truth co-occurrence frequencies.

Learning and Inference: Given the small number of variables (2) and state spaces ($\leq$ 10) for each variable, inference can be performed quickly by brute force, enumerating and computing scores for all possible (action, tool) pairs. During testing we classify using the maximum a posteriori (MAP) decision

$$(a^\star, t^\star) = \text{argmax}_{a \in A, t \in T} p(A = a, T = t \mid x).$$

We learn parameters by maximizing the log-likelihood of our training data, with an additional regularization term. Let $w = [w_A; w_T; w_{AT}]$ be the parameters we wish to estimate. Assume we have a training set of m examples which come with action and tool labels $\{(x^{(i)}, a^{(i)}, t^{(i)})\}_{i=1}^m$. The learning optimization problem is then

$$\textbf{minimize}_w \ \ \frac{\lambda}{2}||w||_2^2 - \sum_{i=1}^{m} \log p\Big(a^{(i)},\ t^{(i)} \mid x^{(i)};\ w\Big).$$

We use gradient descent to optimize this convex function.

4.3. Domain Knowledge from Natural Language

We would like to extract semantic relationships between action and tool classes *automatically from the Internet* as a source of additional domain knowledge / prior information for our problem. In general this can be a significant resource-saving technique: in large domains it may be difficult or time-consuming to hand-craft semantic relationships between objects, and in technical or specialized domains one might have to otherwise resort to hiring an expert. Furthermore, in our smallish dataset the co-occurrence information is very sparse (see Fig. 3, left); we can rely on the huge wealth of knowledge from the web for more robust information.

Dataset groundtruth

	color	cut	draw	glue	paint	place
brush	0	0	0	1	8	0
writing tool	12	0	42	0	0	0
glue	0	0	0	20	0	0
scissors	0	38	0	0	0	0

Wikipedia Co-occurence

	color	cut	draw	glue	paint	place
brush	0	0	1	0	1	0
writing tool	1	0	1	0	0	0
glue	0	0	0	1	0	0
scissors	0	1	0	0	0	0

Normalized Google Distance

	color	cut	draw	glue	paint	place
brush	2.51	2.11	2.4	∞	1.85	∞
writing tool	2.12	3.51	1.72	∞	2.08	∞
glue	2.51	2.51	2.51	1.2	2.44	∞
scissors	2.47	1.76	2.36	∞	2.68	∞

Figure 3. Difference sources of information for action-tool co-occurrences: the groundtruth labels in our dataset, Wikipedia, and web search, as described in Sec. 4.3. Infinite values indicate little or no co-occurrence, and were set to a fixed large finite value in the model.

Our specific task here is to obtain action-tool co-occurrence matrices, representing how likely or unlikely it is to see different actions and tools together. We experimented with two different sources of domain knowledge: Wikipedia and web search results[3].

From Wikipedia we obtain a binary action-tool co-occurence matrix as follows: We gather all words that appear as link or caption tokens in the Wikipedia page associated with each action. We only use links and captions because these words are most likely to be relevant to the topic and semantically meaningful. An action-tool co-occurance is *true* if any of these words correspond to a tool that could be detected by the visual system, and *false* otherwise.

To measure action-tool compatability via web search, we use a semantic distance measure called the Normalized Google Distance (NGD) [1] which is a measure of how related two or more concepts are. The NGD between two terms x and y is

$$NGD(x,y) = \frac{max\{\log q(x), \log q(y)\} - \log q(x,y)}{log(N) - min\{\log q(x), \log q(y)\}}$$

where $q(x)$ is the number of search results for query x, $q(x,y)$ is the number of search results for query $x\,and\,y$, and N is the total number of pages indexed by the search engine. The lower the number, the more related two queries are, with two identical queries having a distance of 0. We calculate the NGD for each action-tool pair to form our co-occurrence matrix (Fig. 3).

We can incorporate the information from both Wikipedia and web search as C^{wiki} and C^{NGD} jointly into our model, as explained in Section 4.

5. Temporal Modeling of Action Sequences

So far we have proposed a CRF model to determine which action and tools occur in *a single shot*, which is part of a larger sequence of actions which comprise an arts-and-crafts activity (see Fig. 1).

Now, we describe an extension to reason about the temporal relationship between these shots, which exploits prior knowledge of the temporal order of actions within the larger activity. This knowledge can be extracted from transcripts or online instructions. We hypothesize that the relative order of action verbs in the transcript is highly correlated with the relative order of actions in the video, e.g., if action v is mentioned before action w in the text, the chance to find v before u in the video should be higher than with no temporal knowledge.

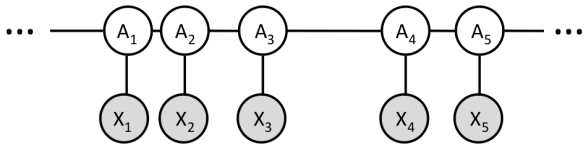

Figure 4. The chain CRF model for the action sequence in a video episode: hidden node (white circles) represents the action class label of a single video shot, and each observed node (grey circles) represents the observed visual features discussed previously.

This hypothesis can be verified using the example video episode illustrated in Fig. 1. In this case the action verbs extracted from the transcript are {draw, draw, draw, *draw, color, cut,* tear, use, take} and the actions in the video are {*draw, color, cut,* thread}. Since the text and video are not strictly aligned in real videos, we do not restrict ourselves to only ordering contraints between direct neighbours, but look at all pairs up to two positions apart. Similar ordering constraints can also be extracted from online instructions.

To integrate this knowledge with the probability of action classes obtained from individual video shots, we designed a chain CRF as illustrated in Fig. 4. The node potential is $p(A_i|X_i)$, the probability of action class for video shot i given the observed visual features X_i, which we estimate from our single shot model (Sec. 4) via marginalization over possible tools: $p(A_i|X_i) = \sum_{t \in T} P(A,t|x)$. The edge potential is based on the frequency of action-action bigrams in the text: $\phi(A_i, A_{i+1}) \propto \exp(\#(A_i, A_{i+1})/N_{AA'})$, where $\#(\cdot,\cdot)$ is the number of times a bigram occured, and $N_{AA'}$ is the total number of bigrams seen. The conditional probabilities of the action classes in a video episode can be represented as follows:

$$p(A_1, A_2, ..., A_k|X_1, X_2, ..., X_k) = \quad (2)$$

$$\frac{\prod_{i=1}^{k} p(A_i|X_i) \prod_{i=1}^{k-1} \alpha_i \exp\{\phi(A_i, A_{i+1})\}}{Z(X_1, X_2, ..., X_k))} \quad (3)$$

where $Z(\cdot)$ is the partition function, α_i a parameter to weight the edge potentials. In the training stage, we need

[3] We also experimented with ConceptNet (http://csc.media.mit.edu/conceptnet), a hand-crafted common-sense knowledge database, but found it to have lower recall and the same precision as the Wikipedia co-occurences.

to estimate the weights α_i from the training corpus. Because the Sprout TV Handcraft Show dataset is small, we enforce $\alpha_i = \alpha$ to avoid over-fitting. The single parameter α can be estimated using cross-validation from training data. We find the most likely sequence of actions via Viterbi decoding.

6. Implementation

Commercial broadcast videos are aimed to please a human viewer, and are thus often not captured in a way that allows for straightforward processing using computer vision techniques. For example, we need to deal with various types of transitions between scenes, as well as moving cameras and various camera view angles and zoom factors. During the preprocessing stage of the video pipeline, the input video is first segmented into semantically meaningful shots and clustered according to the camera view and zoom setting. We use off-the-shelf algorithms for shot boundary detection ([11]), and clustering of camera view points using visual words[4], as well as a commercial face detection and recognition system[5] to detect the presence of human faces and their sizes. Since these techniques are external software solutions, we will not further describe them in this paper. These visual cues enable us to accurately identify the *Zoomed-in* shots we are interested in that show human hands in motion, holding tools and manipulating and transforming objects.

6.1. Motion Features for Action Recognition

Global models for action recognition tend to perform poorly whenever there are camera or scene artifacts such as moving cameras, self occlusions, etc. [9] showed that extracting local space-time interest points (STIP) at several spatial and temporal scales provides benefits previously encountered in object recognition approaches using SIFT features. Once these STIP points have been extracted, features that describe the spatial and temporal characteristics of intensity and flow in a neighborhood of that point are computed. We use publicly available code [6] provided by Laptev et al. to compute a 162-dimensional feature consisting of Histogram of Gradients (HOG) and Histogram of Flow (HOF) features that are computed at the STIPs for all the manually annotated shots. From the training dataset consisting of half the annotated actions, all the features are extracted and clustered in 100 clusters (codewords). For each shot, the term frequency of each codeword is computed by finding the number of features in each shot that are closest to that codeword. After normalization, we get a histogram of term frequencies for each shot, which we denote as the *action features* $f_{STIP}(x)$ in our system (first described in Sec. 4.1).

[4] http://www.vlfeat.org
[5] http://www.pittpatt.com
[6] http://www.irisa.fr/vista/Equipe/People/Laptev/download.html

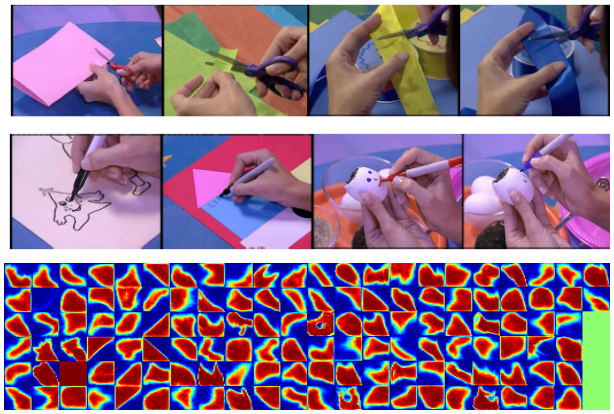

Figure 5. Top: example *Scissors* tool class. Middle: example *WritingTool* class. Bottom: our 128 visual hand pose words.

6.2. Object Features

For the problem of recognizing manipulation actions, the absence or presence of a particular class of objects can provide us with strong cues for possible actions being performed by the subject. For example, a high confidence for presence of a tool like scissors or knife can indicate the possibility of a cutting action being performed in the scene. We adopt the state-of-the-art object detectors and extend them to provide us with cues about the presence of particular objects in a sequence of frames. We use a mixture of deformable, parts-based models for object detection [3]. These state-of-the-art models are robust to actual deformations of the object, varying viewpoints, and partial occlusion by hands or other objects.

We concentrate on the "Tools" object class which contains $WritingTool$, $Scissors$, $Brush$ and $GlueBottle$. We cannot learn models for the open, near-infinite set of other objects potentially present in the video. Images corresponding to these four object classes were obtained from external sources such as ImageNet[7], LabelMe[8] and Google Image Search. We collected 20 to 30 training images per class in this way, and annotated them with bounding boxes for each object. We run the trained object detectors on every 5^{th} image of each shot and accumulate the scores into histograms. The histograms are used as a feature to indicate how likely a particular object is present in the shot, and comprise our features $f_{object}(x)$ in the composite system described in Sec. 4.1.

6.3. Hand Features

To obtain a semantically meaningful hand pose feature, we extract several hand segmentation hypotheses in each

[7] http://www.image-net.org
[8] http://labelme.csail.mit.edu

frame, and quantize these into k clusters obtained via k-means, which we interpret as discrete hand pose "words". The distribution of these hand pose words over a video shot serves as a signature of the action, which we use as a feature vector $f_{hand}(k)$. For example, a hand pose word corresponding to holding a pen should appear much more frequently for the action *Draw*. Hand segmentation hypotheses are extracted in a greedy, bottom-up manner: We first compute a probability map of skin color of close-up shots of the hand, based on a Gaussian mixture color model in RGB space, estimated from the skin color of detected faces in other frames. We then greedily merge regions obtained from a superpixelation of the image to form our hand segments. This process typically produces 1-5 hand segmentation hypotheses per frame which are quantized into 128 hand pose words, shown in Fig. 5, bottom.

7. Experiments

Dataset. Our novel dataset consists of 27 episodes from the PBS Kids show Sprout TV. It contains an average of 8 actions/episode, 220 total action shots, and 43K frames involving actions. Each shot was annotated with one action and one tool class label. We use the *Zoomed-in* shots obtained by the automated shot segmentation approach described in Section 6. We discard all other shots as uninteresting, i.e., no action is present. In all experiments, we used half the data for training, half for testing, split so that each action class is balanced across the train/test divide, and no episode occurs both in training and testing. All training meta-parameters (e.g., regularization weight) were trained on a hold-out subset of the training set.

In this multi-class classification setting, we report *normalized accuracy*: the mean over all classes of the mean within-class accuracy. This performance measure is more robust to datasets where the number of examples of each class is very imbalanced, as in our data.

Single Shot Results Table 1 shows results for different multiple-action-classification settings, varying the number and types of classes, and features were used to learn the ***independent models***. We learned models using multi-class logistic regression with L_2 regularization, using the publicly available package LIBLINEAR[9]. We found this to perform slightly better than linear, polynomial or Gaussian kernel SVMs.

In the first column we examine a nearly balanced binary classification task between two intuitively distant actions, in terms of tools used, hand pose and motion pattern: Cut and $Draw$. We see that features f_{tool} and f_{STIP} alone do very well, separating the data as expected. The hand features perform worse but better than random guessing.

[9] http://www.csie.ntu.edu.tw/ cjlin/liblinear/

	cut (18) draw (20)	color (6) cut (18) draw (20) glue (8) paint (5)	color (6) cut (18) draw (20) glue (8) paint (5) other (50)	brush (5) glue (20) writ. tool (28) scissors (18) other (48)
f_{hand}	63.3	27.8	20.5	23.5
f_{tool}	91.7	42.9	37.1	48.8
f_{STIP}	97.5	61.1	42.1	32.3
f_{all}	97.5	67.1	44.0	46.0
chance	50.0	20.0	16.7	20.0

Table 1. Independent modeling of actions and tools using logistic regression. f_{all} corresponds to $[f_{hand}, f_{tool}, f_{STIP}]$.

Next we consider five-way classification between all action classes for which we have enough training data to model them (i.e., more than five training examples). Again the hand features alone are the weakest cue, followed by tool features and STIP features. The combination of all three feature sources does better than any in isolation.

To obtain real end-to-end system results, we must also make a classification decision on the heavy tail of $Other$ actions which occur infrequently and are extremely varied. Examples include "Crease", "Crackle", "Decorate","Shape","Sprinkle" etc.. In the third column we include this class, and see that performance suffers. This indicates the need to model more classes, or use other sources of information, like natural language, to narrow down the set of possibilities.

In the last column we perform a similar experiment on all shots, modeling the tool type rather than the action. Using the tool features works the best, while the hand features provide very little helpful information. The fact that STIP features work moderately well for tool classification and tool features work well for actions is empirical evidence that action cues can help determine tools and vice versa.

Table 2 shows results from our structured ***joint modeling*** CRF experiments. Column 1 in Table 2 uses the exact same information as column 4 and 5, Table 1, when all features are included. The only exception is that we include class prior features into the CRF model which allow it to learn a better balance between the likelihood of different class labels, hence slightly higher accuracy. We found that explicitly modeling the co-occurrence of actions and tools either directly using our groundtruth (column 2) or using domain knowledge (column 3) significantly helped results. For domain knowledge, we learned a weighted combination of the action-tool co-occurrence matrices obtained from the web. The action accuracy (row 1) remained nearly constant throughout experiments, but the tool accuracy (row 2) and accuracy in getting the correct tool *and* action to-

gether in the same example (row 3) increased significantly when modeling action-tool co-occurrence. Most importantly, these results demonstrate that it is possible to "plug in" domain knowledge as a substitute for groundtruth information and get comparable performance gains over using no joint modeling.

Temporal Action Sequence Results Finally, we evaluate temporal modeling of action sequences. In Table 3 we see that incorporating temporal priors on the action sequence (through the use of likely action transitions obtained from text) gives better perfomance. Between two types of text sources, online instructions are slightly more helpful than transcripts. This is likely due to the fact that transcripts are much noisier than online instructions, containing large amounts of narration. Consequently, the verb list extracted from transcripts contains more irrelevant verbs besides the action verbs we are interested in, and also occasionally does not mention the action of interest.

normalized accuracy (%)	no joint modeling	groundtruth action-tool modeling	domain knowledge modeling
action	50.9	50.8	50.8
tool	44.9	46.7	48.3
action *and* tool	28.0	40.7	37.8

Table 2. Joint modeling of actions and tools using a CRF incorporating class priors and action-tool co-occurences.

8. Conclusions and Future Work

We have presented a novel approach for annotation of commercial videos from the arts and craft domain using a combination of low and mid-level features along with the contextual priors extracted from unstructured textual descriptions and the web using natural language processing techniques. Our flexible model makes explicit the interactions between objects and tools, and also the interaction between actions in a sequence. It allows us to incorporate external domain knowledge as a replacement for ground truth co-occurrence information. This allows us to have more robust estimation when our groundtruth information is very sparse. This is of critical importance when scaling up to many different domains in which actions and objects have interesting relationships.

Model	**Action Acc.(%)**	**Ref. Section**
STIP, Single Shot	42.1	4.1,6
STIP + Tool + Hand Feature, Single Shot	46.1	4.1,6
Single Shot Joint CRF Model	50.8	4.2
Temporal CRF w/ bigram from transcripts	52.0	5
Temporal CRF w/ bigram from online instructions	53.0	5

Table 3. Overall action classification accuracy for the Sprout TV Handcraft Show dataset.

References

[1] R. L. Cilibrasi and P. M. Vitanyi. The google similarity distance. *KDE*, 19(3):370–383, 2007. 5

[2] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, October 2005. 2

[3] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Trans. PAMI*, 2010. 2, 6

[4] D. M. Gavrila. The visual analysis of human movement: A survey. *CVIU*, 73:82–98, 1999. 2

[5] A. Gupta and L. Davis. Objects in actions: An approach for combining action understanding and object perception. In *Proc. CVPR*, 2007. 2

[6] N. Ikizler-Cinbis and S. Sclaroff. Object, scene and actions: Combining multiple features for human action recognition. In *Proc. ECCV*, 2010. 2

[7] H. Kjellstrm, J. Romero, D. Martinez, and D. Kragic. Simultaneous visual recognition of manipulation actions and manipulated objects. In *Proc. ECCV*, 2008. 2

[8] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001. 2, 4

[9] I. Laptev. On space-time interest points. *IJCV*, 2005. 2, 3, 6

[10] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proc. CVPR*, 2008. 2

[11] R. Lienhart. Reliable transition detection in videos: A survey and practitioner's guide. *IJIG*, 2001. 6

[12] M. Marszalek, I. Laptev, and C. . Schmid. Actions in context. In *Proc. CVPR*, 2009. 2

[13] T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *CVIU*, 104:90–126, 2006. 2

[14] D. Moore, I. Essa, and M. Hayes. Exploiting human actions and object context for recognition tasks. In *Proc. ICCV*, 1999. 2

[15] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *Proc. CVPR*, 2010. 2

[16] D. Tran and A. Sorokin. Human activity recognition with metric learning. In *Proc. ECCV*, 2008. 2

[17] C. S. . M. Worring. Concept-based video retrieval. *Foundations and Trends in Information Retrieval*, 4(2), 2009. 2

[18] B. Yao and L. Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *Proc. CVPR*, 2010. 2