Can Humans Fly? Action Understanding with Multiple Classes of Actors

Chenliang Xu[1], Shao-Hang Hsieh[1], Caiming Xiong[2] and Jason J. Corso[1]

[1] Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI

[2] Department of Statistics, University of California, Los Angeles, CA

CVPR 2015

Project Link

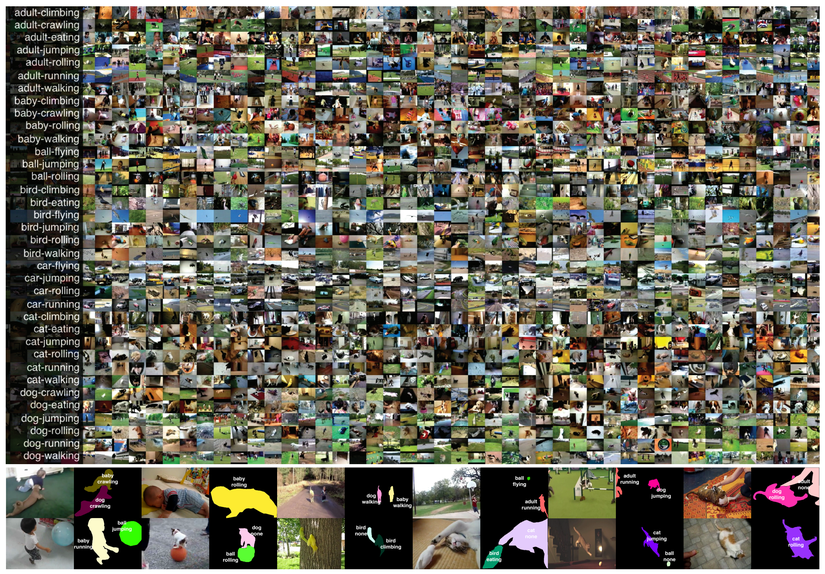
OBJECTIVE

Our paper marks the first effort in the computer vision community to jointly consider various types of actors undergoing various actions.

- We seek to formulate the general actor-action understanding problem and instantiate it at various granularities.

- We demonstrate that inference jointly over actors and actions outperforms inference independently over them.

A2D---The Actor-Action Dataset



We have collected a new dataset consisting of 3782 videos from YouTube; these videos are hence unconstrained "in-the-wild" videos with varying characteristics.

- Seven classes of actors (both articulated and rigid ones):
  adult, *baby*, *bird*, *cat*, *dog*, *ball*, and *car*.
- Eight classes of actions:
  climbing, *crawling*, *eating*, *flying*, *jumping*, *rolling*, *running*, and *walking*.
- We have in total 43 valid actor-action tuples.*

Dataset Statistics:

- Videos have an average length of 136 frames, with a minimum of 24 frames and a maximum of 332 frames.
- One-third of the videos have more than one actor performing different actions.
- We split the dataset into 80% training and 20% testing divided evenly over all actor-action tuples.

Annotations:

- We label three to five frames for each video with both dense pixel-level actor and action annotations.
- The selected frames are evenly distributed over a video.
- We start by collecting crowd-sourced annotations from MTurk, then we manually filter each video to ensure the labeling quality as well as the temporal coherence.

	climb	crawl	eat	fly	jump	roll	run	walk	none
adult	101	105	105		174	105	175	282	761
baby	104	106				107		113	36
ball				109	105	117			87
bird	99		105	106	102	107		112	26
car				102	107	104	120		99
cat	106		110		105	103	99	113	53
dog		109	107		104	104	110	176	46

Figure: We show the number of videos in our dataset in which a given [actor, action] label occurs. Empty entries are joint-labels that are not in the dataset either because they are invalid (a *ball* cannot *eat*) or were in insufficient supply, such as for the case *dog-climb*. The background color in each cell depicts the color we use for annotation; we vary hue for actor and saturation for action.

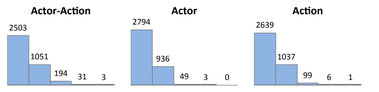


Figure: Histograms of the frequency of videos in which a certain number of joint actor-actions, and individual actors and actions occurs across A2D; roughly one-third of the videos have more than one actor and/or action.

*One additional action label *none* is added to account for actions other than the eight listed ones as well as actors in the background that are not performing an action.

Actor-Action Understanding Problems

Without loss of generality, let $\mathcal{V} = \{v_1, \ldots, v_n\}$ denote a video with n supervoxels in a video segmentation represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

We use $\mathcal{X}$ to denote the set of actor labels and $\mathcal{Y}$ to denote the set of action labels. Consider a set of random variables $\mathbf{x}$ for actor and another $\mathbf{y}$ for action. Then the general actor-action understanding problem is specified as a posterior maximization:

$$(\mathbf{x}^*, \mathbf{y}^*) = \text{argmax}_{\mathbf{x},\mathbf{y}} P(\mathbf{x}, \mathbf{y}|\mathcal{V})$$

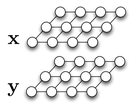
Single-Label Actor-Action Recognition.
- Here, $\mathbf{x}$ and $\mathbf{y}$ are simply scalars x and y, respectively, depicting the single actor and action label to be specified for a give video $\mathcal{V}$.

Multiple-Label Actor-Action Recognition.
- Here, $\mathbf{x}$ and $\mathbf{y}$ are binary vectors of dimension $|\mathcal{X}|$ and $|\mathcal{Y}|$ respectively; x_i takes value 1 if the ith actor is present in the video. We define $\mathbf{y}$ similarly.

Actor-Action Semantic Segmentation.
- Define the two sets of random variables $\mathbf{x} = \{x_1, \ldots, x_n\}$ and $\mathbf{y} = \{y_1, \ldots, y_n\}$ on the set of supervoxels of a video $\mathcal{V} = \{v_1, \ldots, v_n\}$, and assign each $x_i \in \mathcal{X}$ and each $y_i \in \mathcal{Y}$.

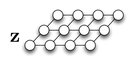
Models

Naïve Bayes-based Model treats the actor and action labels separately.



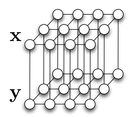
$$P(\mathbf{x}, \mathbf{y}|\mathcal{V}) = P(\mathbf{x}|\mathcal{V})P(\mathbf{y}|\mathcal{V})$$
$$= \prod_{i\in\mathcal{V}} P(x_i)P(y_i) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} P(x_i, x_j)P(y_i, y_j)$$
$$\propto \prod_{i\in\mathcal{V}} \phi_i(x_i)\psi_i(y_i) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} \phi_{ij}(x_i, x_j)\psi_{ij}(y_i, y_j)$$

Joint Product Space Model considers a new set of random variables $\mathbf{z} = \{z_1, \ldots, z_n\}$ defined again on all supervoxels in a video and take labels from the actor-action product space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$.



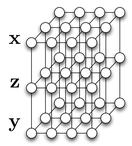
$$P(\mathbf{x}, \mathbf{y}|\mathcal{V}) \doteq P(\mathbf{z}|\mathcal{V}) = \prod_{i\in\mathcal{V}} P(\mathbf{z}_i) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} P(z_i, z_j)$$
$$\propto \prod_{i\in\mathcal{V}} \varphi_i(z_i) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} \varphi_{ij}(z_i, z_j) = \prod_{i\in\mathcal{V}} \varphi_i([x_i, y_i]) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} \varphi_{ij}([x_i, y_i], [x_j, y_j])$$

Bilayer Model connects each pair of random variables $\{(x_i, y_i)\}_{i=1}^n$ with an edge that encodes the potential function for the tuple $[x_i, y_i]$, directly capturing the covariance across the actor and action labels.



$$P(\mathbf{x}, \mathbf{y}|\mathcal{V}) = \prod_{i\in\mathcal{V}} P(x_i, y_i) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} P(x_i, x_j)P(y_i, y_j)$$
$$\propto \prod_{i\in\mathcal{V}} \phi_i(x_i)\psi_i(y_i)\xi_i(x_i, y_i) \prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} \phi_{ij}(x_i, x_j)\psi_{ij}(y_i, y_j)$$

Trilayer Model considers all relationships in the individual actor and action spaces as well as the joint product space. In other words, the previous three baseline models are all special cases of the trilayer model.



$$P(\mathbf{x}, \mathbf{y}, \mathbf{z}|\mathcal{V}) = P(\mathbf{x}|\mathcal{V})P(\mathbf{y}|\mathcal{V})P(\mathbf{z}|\mathcal{V}) \prod_{i\in\mathcal{V}} P(x_i, z_i)P(y_i, z_i)$$
$$\propto \prod_{i\in\mathcal{V}} \phi_i(x_i)\psi_i(y_i)\varphi_i(z_i)\mu_i(x_i, z_i)\nu_i(y_i, z_i) \cdot$$
$$\prod_{i\in\mathcal{V}} \prod_{j\in\mathcal{E}(i)} \phi_{ij}(x_i, x_j)\psi_{ij}(y_i, y_j)\varphi_{ij}(z_i, z_j)$$

$$\mu_i(x_i, z_i) = \begin{cases} w(y_i'|x_i) & \text{if } x_i = x_i' \text{ for } z_i = [x_i', y_i'] \\ 0 & \text{otherwise} \end{cases}$$

$$\nu_i(y_i, z_i) = \begin{cases} w(x_i'|y_i) & \text{if } y_i = y_i' \text{ for } z_i = [x_i', y_i'] \\ 0 & \text{otherwise} \end{cases}$$

Terms $w(y_i'|x_i)$ and $w(x_i'|y_i)$ are classification scores of conditional classifiers, which are explicitly trained for this trilayer model: separate classifiers for the same action conditioned on the actor are able to exploit the characteristics unique to that actor-action tuple.

Experiments: Single- and Multiple-label Actor-Action Recognition

	Single-Label			Multiple-Label		
	Classification Accuracy			Mean Average Precision		
Model	Actor	Action	<A, A>	Actor	Action	<A, A>
Naïve Bayes	70.51	74.40	56.17	76.85	78.29	60.13
JointPS	72.25	72.65	61.66	76.81	76.75	63.87
Trilayer	**75.47**	**75.74**	**64.88**	**78.42**	**79.27**	**66.86**

Setup.
- Dense trajectory features (Wang et al. IJCV'13).
- One-vs-all SVM models with RBF-χ^2 kernels.

Naïve Bayes. Train separate classifiers over $\mathcal{X}$ and $\mathcal{Y}$, then score them together.

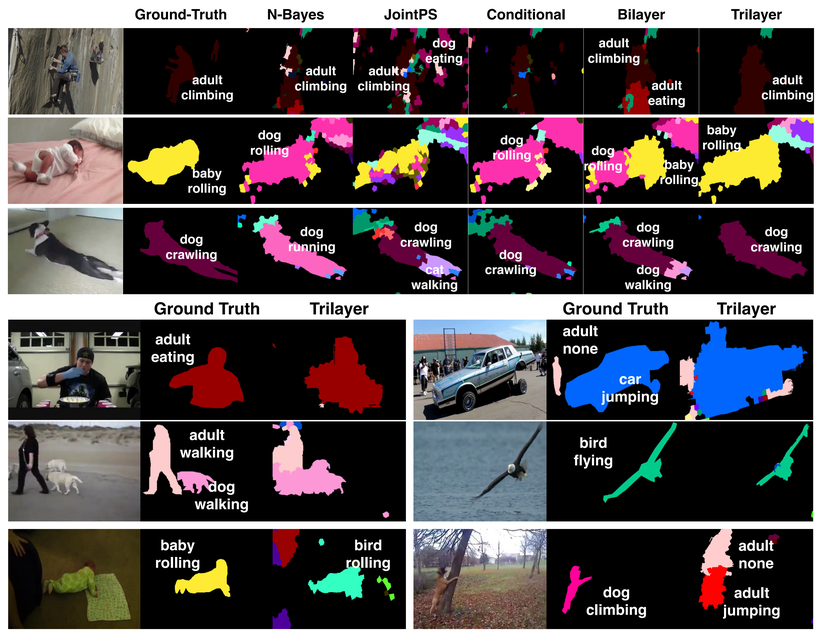
Joint Product Space. Train a classifier for each actor-action tuple in $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$.

Trilayer. Learn classifiers over the actor space $\mathcal{X}$, the action space $\mathcal{Y}$ and the joint actor-action space $\mathcal{Z}$. During inference, it separately infers the naïve Bayes terms and the joint product space terms and then takes a linear combination of them to yield the final score.
It models not only the across-actor-action but also the common characteristics among the same actor performing different actions as well as the different actors performing the same action.

Experiments: Actor-Action Semantic Segmentation

- TSP (Chang et al. CVPR'13) yield about 400 supervoxels touching each frame.
- Features: histograms of appearance features, optical flow and dense trajectories for the entire supervoxel volume.
- Train one-vs-all SVM classifiers on supervoxels using supervoxel's grouth-truth label.
- The inference output is a dense labeling of video voxels in space-time, and we evaluate on the sparsely labeled frames.

		Average Per Class Accuracy		
	Model	Actor	Action	<A, A>
Unary Term Only	Naïve Bayes	43.02	40.08	16.35
	JointPS	40.89	38.50	20.61
	Conditional	43.02	41.19	22.55
	Bilayer	43.02	40.08	16.35
	Trilayer	43.08	41.61	22.59
		Average Per Class Accuracy		
	Model	Actor	Action	<A, A>
Full Model	Naïve Bayes	44.78	42.59	19.28
	JointPS	41.96	40.09	21.73
	Conditional	44.78	41.88	24.19
	Bilayer	44.46	43.62	23.43
	Trilayer	**45.70**	**46.96**	**26.46**



Acknowledgements. This work was partially supported by NSF CAREER IIS-0845282, ARO YIP W911NF-11-1-0090 and DARPA Mind's Eye W911NF-10-2-0062.