Measuring and Reducing Observational Latency when Recognizing Actions

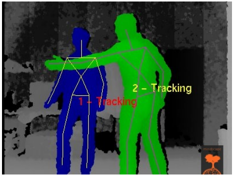
C. Ellis, S. Masood, A. Nagaraja, M. Tappen, J. LaViola,[1]
R. Sukthankar[2]

[1]Department of Computer Science
University of Central Florida

[2]Robotics Institute
Carnegie Mellon University
and
Google Research

ICCV 2011 Workshop on HCI

What This Talk is About



Novel Action Recognition Algorithm

- Parameterized to exploit accuracy/latency trade-off
- Temporally segmented and online
- Novel Dataset leveraging skeleton data from Microsoft Kinect

Outline

1. Introduction
   - Fundamentals
   - Dataset and Features
2. Temporally Segmented Action Recognition
   - Our Method
   - Baseline Models for Comparison
   - Results
3. Online Detection of Actions
   - Online Action Detection
   - Parameterizing Accuracy vs. Latency

Fundamental Criteria

High Accuracy

The system must be accurate at recognizing actions

Low Latency

A system that lags behind user actions will feel cumbersome

Goal: Understand and exploit the trade-offs between latency and accuracy

Why are Accuracy and Latency Important?

Fighters Uncaged Metacritic Reviews (Avg. score: 32/100)

- ...responds very badly to your moves...
- ...sub-par controls...
- ...entirely ruined by how it actually plays and controls...
- ...frustrates far more than it excites...
- ...fails to register most of the movements, and huge lag problems...
- ...it's inexcusable that a game whose sole interaction is hand-to-hand combat should not be able to tell the difference between dodging and headbutting.
- ...unresponsive and downright inaccurate controls...
- Poor motion controls...
- ...poor movement recognition...
- You have more luck controlling a lifeless rock.

Computational Latency vs. Observational Latency

Computational Latency

Time till the system finishes computation on the observations

Observational Latency

Time till system observes enough frames to make an accurate prediction of the action being performed

Contributions

- A novel system for multiway on-line classification that addresses both latency and accuracy
  - Recognizes on minimal frames (canonical pose)
  - Parameterized to exploit accuracy/latency trade-off
- A new dataset containing skeleton pose estimation of sixteen actions

Novel Dataset

Gameplay from Mirror's Edge

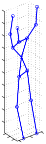
- Microsoft Kinect sensor using OpenNI
- 16 individuals performing 16 actions
- 5 repetitions of each action per person
- 15 joints per frame (position and orientation)
- Actions drawn from game Mirror's Edge

Features

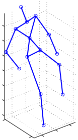
- 555 pairwise Euclidean distances between joints (position/rotation invariant)
  - Within current frame
  - Between current and previous frame (captures motion)
  - Between current frame and first frame (captures overall change, easy to estimate)
- Features clustered by 5 clusters per pairwise distance
  - Improves training time substantially

Finding Canonical Poses

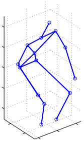
Leap



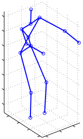
Kick



Duck



Vault



Novel Action Recognition Algorithm

- Want to classify with as few observations as possible
- Action is recognized when actor assumes unambiguous pose
- Train on maximum likelihood class over all frames of video

Deriving Action Class Probability

$$P[I = T|\vec{x}] = \frac{\exp\left(\max_{f \in F} \vec{x}_f \cdot \theta_T\right)}{\sum_c \exp\left(\max_{f \in F} \vec{x}_f \cdot \theta_c\right)}$$

$$\max\left(v_1, v_2, ..., v_N\right) \approx \log\left(e^{v_1} + e^{v_2} + ...e^{v_N}\right)$$

$$P[I = T|\vec{x}] = \frac{\sum_{f \in F} \exp\left(\vec{x}_f \cdot \theta_T\right)}{\sum_c \sum_{f \in F} \exp\left(\vec{x}_f \cdot \theta_c\right)}$$

Training and Classifying Segmented Video

1. Process frames into feature vectors
2. Learn weight parameters $\theta_1, ..., \theta_f$
3. For each action class $c \in \{1, ..., N_A\}$, find feature vector $\vec{x}_{f_c^*}$ such that $\vec{x}_{f_c^*} \cdot \theta_c$ has the highest value
4. Label the video with class c^*, where $c^* = \arg\max_c \vec{x}_{f_c^*} \cdot \theta_c$

Baseline - Bag of Words

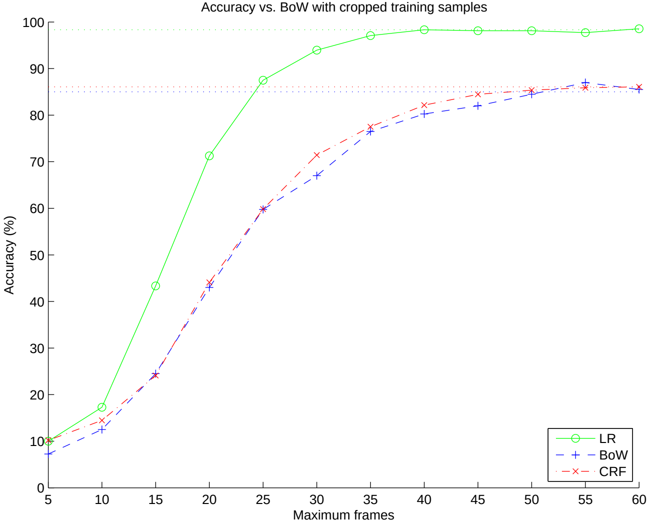
- BoW works well on a wide variety of classification tasks
- Computed using (undiscretized) pairwise distance features
- Frames each assigned to one of 1000 clusters
- Videos represented by normalized histogram of cluster frequencies
- Classification performed by SVM based on histogram intersection kernel

Baseline - Conditional Random Field

- CRFs well-suited for streams of observations
- Chain structured CRF model with pairwise Potts model potentials

$$P[l = T|\vec{x}] = \frac{\exp\left\{-\log\sum_y \exp(C_T(y;x))\right\}}{\exp\left\{-\log\sum_k\sum_y \exp(C_k(y;x))\right\}}$$

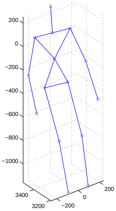
Accuracy vs. BoW and CRF over varying video length



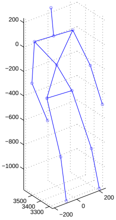
Average Std. Deviation of Features by Frames

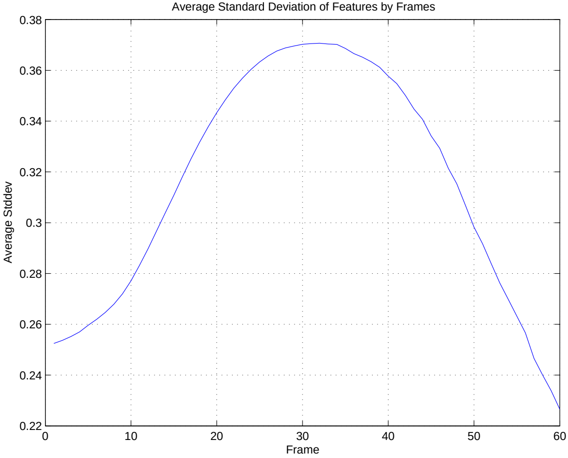
frame 5

Balance



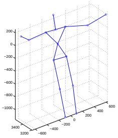
Punch



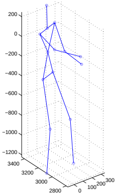


frame 30

Balance



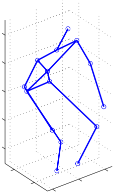
Punch

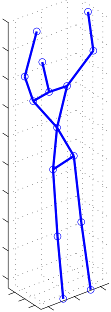


Online Detection of Actions

- Closer to real-world classification tasks
  - Real world actions must be picked out of stream
  - Must classify as quickly as possible to ensure low latency
- Apply Canonical pose detector per frame and return a classification when any probability exceeds an empirically chosen threshold T
- If no frame in an action video has a action probability greater than T, the video is considered a missed detection

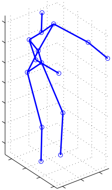
Online classifier confusion matrix





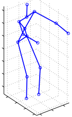
On−line Clasifier confusion matrix

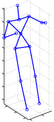
	balance	climbladder	climbup	duck	hop	kick	leap	punch	run	stepback	stepfront	stepleft	stepright	twistleft	twistright	vault	unassigned
balance	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
climbladder	0	95	0	0	0	0	0	2	2	0	0	0	0	0	0	0	0
climbup	0	10	88	0	0	0	0	0	2	0	0	0	0	0	0	0	0
duck	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
hop	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
kick	0	0	0	0	0	80	0	0	0	0	0	0	0	0	0	0	20
leap	0	0	10	10	0	0	52	0	0	0	0	0	0	0	0	28	0
punch	0	0	0	0	0	0	0	78	2	0	0	0	0	2	0	0	18
run	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
stepback	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0
stepfront	0	0	0	0	0	0	0	0	0	0	98	0	0	2	0	0	0
stepleft	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
stepright	0	0	0	0	0	0	0	0	0	0	0	0	95	0	0	0	5
twistleft	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
twistright	0	0	0	0	0	0	0	0	0	0	0	0	0	0	88	0	12
vault	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	95	0

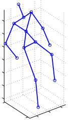


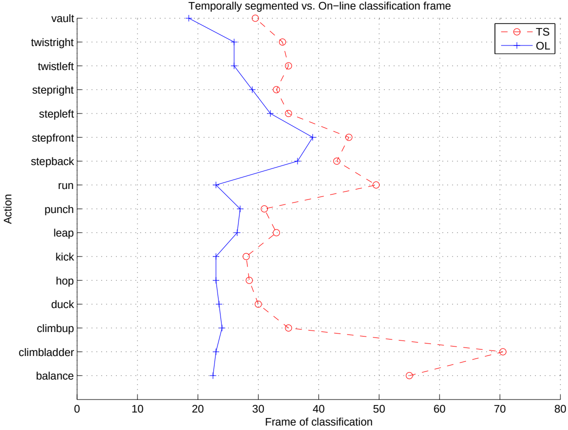
Temporally Segmented vs. Online classification

Instant

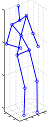

Vault

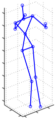

Punch

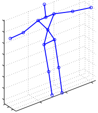

Kick



Continuous


Run


Ladder


Balance

Modifying the Learning Criterion

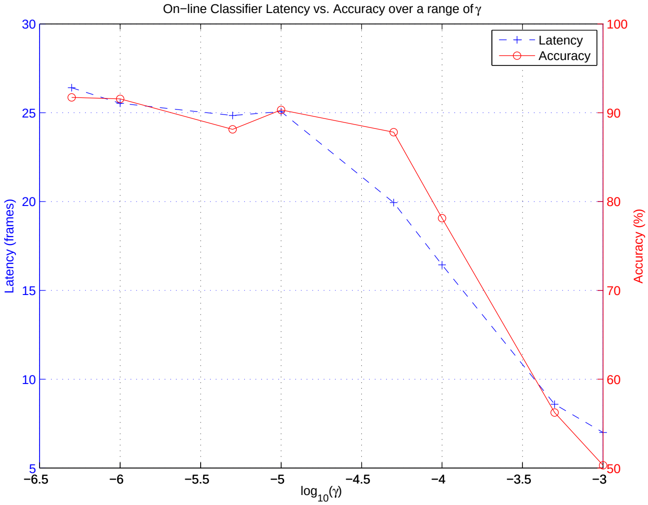
- Weights trained in situation where all frames of action are visible
- Does not match real world
- Modify the loss to penalize for detecting later in the action
- Parameterize new loss function to offer latency/accuracy trade-off

Modifying the Loss to Improve Online Detection

- Bias learner to classify earlier by adding weighted sum of loss terms on truncated observation sequence
- $M = 10, 15, 20$
- $\gamma \cdot m$ is linear scaling factor
- Higher weight on longer video sequences to avoid overfitting on noisy short videos

$$L_{Online}(\theta) = L_{Full}(\theta) + \sum_{m \in M} (\gamma \cdot m) L_M(\theta) + \alpha R(\theta)$$

Online Latency vs. Accuracy over a range of γ



Conclusion

- We out-perform baselines for temporally segmented action recognition
- Online classification achieves good accuracy for large number of actions
- Contributed a parameterized technique for biasing learning in favor of low latency or high accuracy

Future Work

- Collect new dataset using Kinect SDK
- Capture and detect actions with transitions
- Gesture spotting from unsegmented stream of frames

Questions?

Dataset available at

www.cs.ucf.edu/~smasood/datasets/UCFKinect.zip

Contact

Chris Ellis
University of Central Florida
chris@cs.ucf.edu

Acknowledgments

M.F.T, S.Z.M., C.E., and A.N. were supported by NSF grants IIS-0905387 and IIS-0916868.
J.J.L. was supported by NSF CAREER award IIS-0845921 and NSF awards IIS- 0856045 and CCF-1012056.