Examining the Effects of the Video Normalisation for Gesture Recognition using Improved Dense Trajectories

August 17, 2016

1 Team details

- Team name: TARDIS
- Team leader name: Necati Cihan Camgoz
- Team leader address, phone number and email: CVSSP, University of Surrey, 10BA00; n.camgoz@surrey.ac.uk
- Rest of the team members: Simon Hadfield, Oscar Koller, Richard Bowden
- Team website URL (if any): -
- Affiliation: University of Surrey, RWTH Aachen University

2 Contribution details

- Title of the contribution: Examining the Effects of the Video Normalisation for Gesture Recognition using Improved Dense Trajectories
- Final score: Validation 36.0021% correct classification rate on the validation data. Please note that the trajectory accumulation includes RANSAC and is not deterministic. Therefore there may be slight variations in performance between multiple runs.
- General method description: We first preprocess the data to normalise for scale and translation based on the detection of the persons "operating volume". We then compute various local features for appearance and motion. These features are accumulated temporally along dense trajectories. Finally the dimensionality of the data is reduced using PCA, and spatial accumulation is performed via Fisher Vector encoding. Finally the encoded features are concatenated in to a feature vector and classified via Linear SVM.

- References: Wei, Shih-En, et al. "Convolutional Pose Machines." arXiv preprint arXiv:1602.00134 (2016). Wang, Heng and Schmid, Cordelia. Action Recognition with Improved Trajectories. ICCV 2013.
- Representative image / diagram of the method: See figure 1.

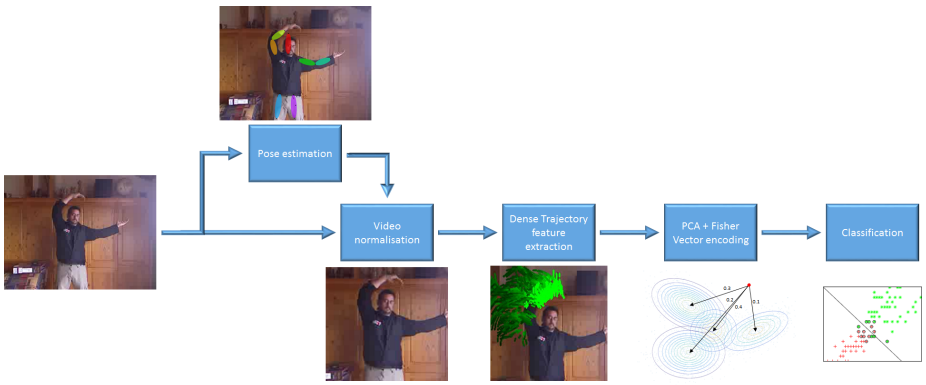

Figure 1: Flow chart of the system

- Describe data preprocessing techniques applied (if any): At every frame of the sequence, the pose of the actor is estimated using the Convolutional Pose Machine approach of Wei et al. Based on the estimated head sizes and positions, an operating region is extracted (using statistics computed from the training data). This operating volume helps to normalise for the scale and translation variations of the actors.

3 Visual Analysis

3.1 Gesture Recognition (or/and Spotting) Stage

3.1.1 Features / Data representation

Describe features used or data representation model FOR GESTURE RECOGNITION (OR/AND SPOTTING) STAGE (if any): The feature descriptors used are Trajectory Coordinates, HOG/HOF and MBH descriptors. After temporal accumulation along trajectories, the features are encoded as fisher vectors.

3.1.2 Dimensionality reduction

Dimensionality reduction technique applied FOR GESTURE RECOGNITION (OR/AND SPOTTING) STAGE (if any): PCA is used to reduce the dimensionality of the descriptors before fisher vector encoding. 99% of the energy of the features is conserved during PCA.

3.1.3 Compositional model

Compositional model used, i.e. pictorial structure FOR GESTURE RECOGNITION (OR/AND SPOTTING) STAGE (if any): None

3.1.4 Learning strategy

Learning strategy applied FOR GESTURE RECOGNITION (OR/AND SPOTTING) STAGE (if any): Learning is performed by linear SVM, with the C parameter being chosen using a coarse grid search over the validation data.

3.1.5 Other techniques

Other technique/strategy used not included in previous items FOR GESTURE RECOGNITION (OR/AND SPOTTING) STAGE (if any): None

3.1.6 Method complexity

Method complexity FOR GESTURE RECOGNITION (OR/AND SPOTTING) STAGE: For prediction, the method is linear in the number of videos, and in the number of frames per video. For training, the method is also generally linear in the number of videos and frames. The only exception is the computation of PCA for dimensionality reduction, which would have cubic complexity. To mitigate this, we select a constant sized random subset to compute PCA.

3.2 Data Fusion Strategies

List data fusion strategies (how different feature descriptions are combined) for learning the model / network: Single frame, early, slow, late. (if any): Fusion is performed late in the feature extraction pipeline (before classification). The dimensionality reduction and fisher vector encoding is performed for each type of feature in isolation, and the results are then concatenated.

3.3 Global Method Description

- Which pre-trained or external methods have been used (for any stage, if any): The Convolutional Pose Machines approach is used during preprocessing, to estimate the operating volume of the actor. The Improved Dense Trajectories approach is used during feature extraction.
- Which additional data has been used in addition to the provided ChaLearn training and validation data (at any stage, if any): None
- Qualitative advantages of the proposed solution: The approach encodes many important invariances at multiple stages of the pipeline. This helps the technique operate on the highly varied videos in the challenge. The preprocessing counteracts variations in actor scale and position within the video. Temporal accumulation via dense trajectories helps to deal with

changes in action speed. Spatial accumulation via Fisher Vectors helps to deal with changes in action style.

- Results of the comparison to other approaches (if any): None
- Novelty degree of the solution and if is has been previously published: As far as we are aware, this is the first time that pose detection techniques have been used to normalise for scale and translation within a holistic/bag-of-visual-words style pipeline with local feature accumulation.

4 Other details

- Language and implementation details (including platform, memory, parallelization requirements): As described in the README file accompanying the code, all code was compiled and run on Ubuntu 14.04. The code uses matlab (tested on 2016a) and C++. The code requires a recent version of ffmpeg to be installed on the machine, and a version of opencv with the nonfree library compiled (we tested with opencv-2.4.6.1). The code requires caffe (with matlab bindings) to be installed, and an nvidia GPU with 12GB of VRAM. No other special hardware is required.
- Human effort required for implementation, training and validation?: The process was fully automated, there was no human-in-the-loop input.
- Training/testing expended time?: Testing the full pipeline takes approximately 2 minutes per video (we parallelised many computations across multiple machines in order to complete the challenge in time). Extracting samples to learning the dimensionality reduction and GMM feature encoding took 5 hours. Training the classifier on the encoded features took around 2 hours.
- General comments and impressions of the challenge? what do you expect from a new challenge in face and looking at people analysis?: The deadlines were extremely tight given the huge amount of data. This was compounded by the validation data being held back for such a long time. There was little possibility for innovation because by the time we saw our results, there was no time to try anything else.