ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

# Real-time activity recognition via deep learning of motion features

Kishore Konda[1], Pramod Chandrashekhariah[2],
Roland Memisevic[3] and Jochen Triesch[1,2] *

1- Goethe University Frankfurt, Germany

2- Frankfurt Institute for Advanced Studies, Germany

3- University of Montreal, Canada

**Abstract**.
Activity recognition is a challenging computer vision problem with countless applications. Here we present a real time activity recognition system using deep learning of local motion feature representations. Our approach learns to directly extract energy based motion features from video blocks. We implement the system on a distributed computing architecture and evaluate its performance on the iCub humanoid robot. We demonstrate real time performance using GPUs, paving the way for wide deployment of activity recognition systems in real world scenarios.

## 1 Introduction

Activity recognition is of primary interest in various applications such as video surveillance, medical care, human-computer interaction etc. [1, 2]. In the recent past, there has been an increasing interest in activity analysis from areas such as elderly care and health monitoring of patients. Traditionally, patients are required to wear a variety of sensors to identify their daily live activities [3, 4]. Vision based recognition schemes come in handy in such applications as they allow the use of passive, non-contact sensors. This, however, requires a system that not only learns and recognizes the activities in new scenarios but also provides real-time performance. In this work we develop a end-to-end learning based activity recognition system that learns from a minimal data set for a given scenario providing high speed and real-time recognition performance. We integrate and demonstrate the system on a robotic platform.

Using local motion features for activity recognition is a popular approach employed in many of the previous works [5, 6, 7, 8]. Approaches like [8] use traditional handcrafted features like HOG3D, HOF etc., as local motion features whereas so-called energy models [7, 6, 5] learn motion features from the input data. In traditional energy models, motion, or the spatial transformation between two frames of a sequence, is represented as the sum of squared quadrature Fourier or Gabor coefficients across multiple frequencies and orientations [5]. Summing over squared quadrature pairs also induces invariance to content, allowing the model to represent pure motion. In [6] it has been shown that

*This work was supported in part by the German Federal Ministry of Education and Research (BMBF) in projects 01GQ084(0/1) (BFNT Frankfurt), by an NSERC Discovery grant and by a Google faculty research award. JT was supported by the Quandt foundation.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

learning the spatial transformations and invariance can be viewed as two independent aspects of learning. Based on that view they introduced a single layered autoencoder based model named *synchrony* autoencoder(SAE) for learning motion representations. In this work we use the SAE for learning motion features exploiting its training efficiency. In the next section we briefly explain the SAE model followed by details on the real time activity recognition system in later sections.

## 2 Learning motion features

In [6] it is shown that the detection of a spatial transformation can be viewed as the detection of synchrony between the image sequence and a sequence of features undergoing the transformation. This is done in the SAE model by allowing for multiplicative (gating) interactions between filter responses applied to the frames in a video. The following is a brief description of the SAE model for sequences.

Let $\vec{X} \in \mathbb{R}^N$ be the concatenation of $T$ vectorized frames $\vec{x}_t \in \mathbb{R}^M, t = 1, \ldots, T$. Let $\mathbf{W}^x \in \mathbb{R}^{Q \times N}$ denote a matrix containing $Q$ feature vectors $\vec{W}^x_q \in \mathbb{R}^N$ stacked row-wise. Each feature is composed of individual frame features $\vec{w}^x_{qt} \in \mathbb{R}^M$ each of which spans one frame $\vec{x}_t$ from the input sequence. The filter responses, or "factors", are defined as $\vec{F}^X = \mathbf{W}^x \vec{X}$. A simple representation of motion can then be defined as

$$H_q = \sigma((F^x_q)^2), \tag{1}$$

Learning in an autoencoder is generally achieved by minimizing the sum of a reconstruction cost and a regularization term, using gradient descent. In this work we use contractive regularization [9]. The cost function for the SAE model together with the regularization is given by

$$\mathcal{J}_C = \|(\vec{X} - \hat{\vec{X}})\|^2 + \lambda \|J_e(\vec{X})\|^2_E, \tag{2}$$

where $\|J_e(X)\|^2_E$ denotes the Frobenius norm of the Jacobian of the hidden units with respect to the inputs [9], which for $\sigma$ defined as *logistic sigmoid* is given by

$$\|J_e(\vec{X})|^2_E = \sum_j (H_j(1 - H_j))^2 (F^x_j)^2 \sum_i (W^x_{ij})^2. \tag{3}$$

The hyper-parameter $\lambda$ is set via a grid search. The SAE model is used as the feature extraction module of the activity recognition pipeline explained in the following section.

## 3 Activity pipeline

Our activity analysis pipeline is based on the bag-of-words approach used in [5, 6]. The pipeline consists of a feature extraction module followed by K-means

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

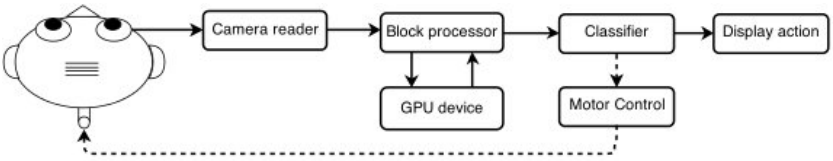



Fig. 1: Block diagram of real time activity recognition system on iCub.

vector quantization and finally a $\chi^2$ kernel SVM for classification. The model described in Section 2 is used for motion feature extraction. The model is trained on PCA-whitened input patches of size $10 \times 16 \times 16$ ($time \times space \times space$). The total number of training samples is $200,000$. The size of the latent hidden layer representation from the model is fixed at 300.

It has been observed that spatially combining local features learned from smaller input patches leads to better representation than features learned on larger patches [5, 10]. In this regard, for computing a local feature describing a larger region of input video, sub blocks of the same size as the patch size are cropped from "super blocks" of size $14 \times 20 \times 20$ [5, 6]. The sub blocks are cropped with a stride of 4 on each axis giving 8 sub blocks per super block. The feature responses of sub blocks are concatenated and dimensionally reduced using PCA to form the local feature. On these local features a K-means layer with 3000 centers is learned with $500,000$ samples for training. This gives a dictionary of 3000 words where each word can be thought of as a motion pattern. The pipeline can be viewed as layer-wise trained deep neural network.

For inference the super-blocks are extracted densely from the input video with a 50% overlap. The resulting local features, from convolution operation, from the entire video are quantized using the learned K-means vocabulary resulting in a 3000 dimensional histogram or bag-of-words feature vector. The histograms from the training videos are used for training the SVM classifier. During inference the histogram computed for a given test sequence is classified into a activity label.

## 4 Real-time recognition system

In this section we describe the real-time implementation of the activity recognition system. The overall architecture of the system is as shown in Figure 1. platform. The individual modules shown in Figure 1 are run on different machines that are connected through an open-source platform called YARP (a robotic platform). We use a C++ implementation that uses OpenCV (computer vision library) and GPUs (Graphical Processing Units) for fast computations to cater to the real-time performance. We explain in detail the individual modules in the rest of this section.

**Camera reader**: This module handles the video stream from the cameras of the iCub humanoid robot [11]. The *camera reader* receives a continuous stream of frames from the cameras on the robot, bundles them into groups of 14 and then sends them to the *block processor* module. There is an overlap of 7 frames netween subsequent video blocks.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

**Block processor**: This module implements the activity recognition pipeline explained in Section 3 which computes a histogram or motion descriptor (output of the K-means quantization step) given a video block. The input to the *block processor* are video blocks of size 14 frames from the *camera reader*. The computed histogram for each input video block is passed on to the *classifier* module. Our implementation has the ability to parallelize block processing by running multiple instances of *block processor* on multiple GPUs.

**Classifier**: The module handles the output histograms from the *block processor* by maintaining a first-in-first-out buffer of size 10 in our case. For every new incoming histogram the module updates the buffer and predicts an action label by summing over histograms in the buffer i.e., over past 10 blocks ( 4 secs of video) unlike the offline case that uses the entire sequence to predict. The label information can be further used in detection and tracking of the user involved by controlling the gaze of the humanoid robot (see dotted lines in Fig. 1). We plan to do this as part of our future work.

### 4.1 Implementation details

**Hardware configuration and speed** General-purpose computing on GPUs has gained popularity in recent years especially in the field of computer vision for speeding up the algorithms that involve intense computations on images. We use GPUs for the *block processor* module. Our current implementation ran on a system with 2.6 GHz CPU, 12 GB RAM and GTX 480 NVIDIA GPU devices. We observed that the time taken for processing one video block of 14 frames varied from 350 to 450 milliseconds that corresponds to an overall processing speed of 15 to 20 frames per second *fps*, considering overlap of 7 frames between the video blocks. When using four GPUs the speed further increased to 42 *fps* which higher than what is considered is as real time performance.

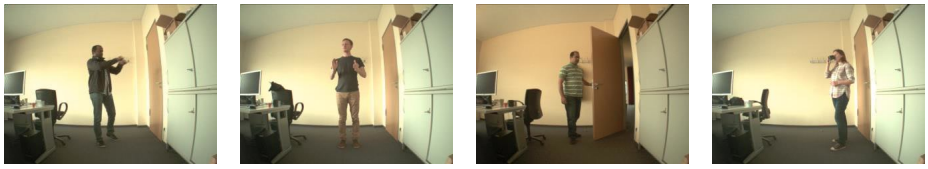
**YARP: a robotic architecture**: We develop the system on a distributed architecture to share the load onto different machines. We modularize the algorithm into different parts that are simultaneously run and coordinated through YARP. YARP (Yet Another Robot Platform) is a robot software architecture that can run a collection of programs on different machines and lets them communicate in a peer-to-peer way [12]. In our work we run the *camera reader*, *block processor*, *classifier* and *Display* modules on different cores/machines.

**iCub robot:** iCub is an open-system robotic platform that is generally considered an interesting experimental platform for analyzing cognitive, visual and sensorimotor behaviors. iCub is designed with physical dimensions resembling a 3 year old child. The head in particular has two dragon fly cameras with VGA resolution that are mounted as eyes used for video capture in our system. The eyes can produce images at resolution of $320 \times 240$ at a rate of $\sim 20$ *fps* .

## 5 Dataset and results

As mentioned in the previous section we recorded a dataset on iCub robotic platform for parameter training and testing of our system. The dataset includes

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.



(a) Boxing (b) Clapping (c) Door-open-close (d) Drinking

(e) Waving-double (f) Waving-single (g) Sit-Stand (h) Walking

Fig. 2: Example snapshots of different actions from the collected dataset.

| | Boxing | clapping | DOC | drinking | HWD | HWS | sitstand | walking |
|---|---|---|---|---|---|---|---|---|
| Boxing | 46.6/49.8 | 26.6/9.6 | 0./0.4 | 0./7.2 | 0./1.2 | 20./15.4 | 0./10.6 | 6.6/5.5 |
| clapping | 11.1/1.6 | 61.1/52.7 | 0./0. | 5.5/20.8 | 0./1.11 | 5.5/10.8 | 16.6/8.3 | 0./4.4 |
| DOC | 0./0. | 0./0. | 100./91.4 | 0./8.5 | 0./0. | 0./0. | 0./0. | 0./0. |
| drinking | 0./0. | 0./0. | 0./2.7 | 100./95.8 | 0./0. | 0./0.9 | 0./0. | 0./0.5 |
| HWD | 0./0. | 11.1/6.5 | 0./0. | 0./0.3 | 50./75.6 | 22.2/17.5 | 16.6/0. | 0./0. |
| HWS | 0./0. | 0./0. | 0./0. | 0./11.7 | 0./0. | 80./88.2 | 20./0. | 0./0. |
| sitstand | 0./0. | 0./0. | 0./0. | 11.9/47.8 | 0./0. | 0./0. | 88.1/52.1 | 0./0. |
| walking | 0./0. | 0./0. | 0./0.4 | 0./6.2 | 0./0. | 0./0. | 0./0. | 100./93.3 |

Table 1: Confusion matrix of classification experiments. Offline test/Realtime test.

videos from 8 different people $P0$ to $P7$ performing 8 different actions. The actions are namely "clapping", "door open close"(DOC), "drinking", "hand waving double"(HWD), "hand waving single"(HWS), "sit stand", "walking" and "boxing". These actions are chosen as they are a subset of most observed human behaviors in an indoor workplace environment. Example snapshots of different actions are shown in Figure 2.

A total of 574 videos are collected and are divided into a training set of 355 clips from persons $P1, P2, P3, P5, P7$ and a testing set of 219 clips from $P0, P4, P6$. Each person performed an action multiple times with two different types of clothing (jacket on and jacket off). Since it is very likely that a person performs an action similarly in multiple tries, the dataset is split into training and testing set based on person rather than choosing random subsets of the total set. The classification accuracy of the activity pipeline (Section 3) on the testing set is 85.39%. The confusion matrix of the classification experiment is shown in Table 1.

In order to validate the real time performance of the system, apart from demonstrating it live on the iCub, we also run the system on longer test videos collected from users $P0, P4, P6$ to quantify the results. The *classifier* module of the system is set to predict for every new incoming video block which implies a

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

new prediction at an interval of seven frames (due to overlap). The classification performance of the real time system is 74.91 and the corresponding confusion matrix is reported in Table 1. The deviation from the offline performance on the testing set is mainly due to inability of the current system to deal with user activities other than the ones system is trained on.

## 6 Conclusion

The real-time system presented in this work achieves high processing speeds and competitive performance by utilizing a very limited set of data samples. The learned local motion features used by the system are well generalized there by no additional parameter training (except the classifier) is necessary to be able to recognize new sets of actions. The parallel architecture and utilization of GPUs give the system the ability to achieve higher processing speeds when required. In future we plan to utilize action class information for better detection and tracking of the user via gaze control of the humanoid robot.

## References

[1] Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010.

[2] Pavan Turaga, Rama Chellappa, Venkatramana S Subrahmanian, and Octavian Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, 2008.

[3] Tam Huynh, Ulf Blanke, and Bernt Schiele. Scalable recognition of daily activities with wearable sensors. In *Location-and context-awareness*, pages 50–67. Springer, 2007.

[4] Seon-Woo Lee and Kenji Mase. Activity and location recognition using wearable sensors. *IEEE pervasive computing*, 1(3):24–32, 2002.

[5] Q.V. Le, W.Y. Zou, S.Y. Yeung, and A.Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

[6] Kishore Reddy Konda, Roland Memisevic, and Vincent Michalski. Learning to encode motion using spatio-temporal synchrony. In *Proceedings of ICLR*, April 2014.

[7] Graham W. Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *Proceedings of the 11th European conference on Computer vision: Part VI*, ECCV'10, 2010.

[8] Heng Wang, Muhammad Muneeb Ullah, Alexander Kläser, Ivan Laptev, and Cordelia Schmid. Evaluation of local spatio-temporal features for action recognition. In *University of Central Florida, U.S.A*, 2009.

[9] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive Auto-Encoders: Explicit Invariance During Feature Extraction. In *ICML*, 2011.

[10] Adam Coates, Honglak Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Artificial Intelligence and Statistics*, 2011.

[11] Giorgio Metta, Lorenzo Natale, Francesco Nori, Giulio Sandini, David Vernon, Luciano Fadiga, Claes Von Hofsten, Kerstin Rosander, Manuel Lopes, José Santos-Victor, et al. The icub humanoid robot: An open-systems platform for research in cognitive development. *Neural Networks*, 23(8):1125–1134, 2010.

[12] Giorgio Metta, Paul Fitzpatrick, and Lorenzo Natale. Yarp: yet another robot platform. *International Journal on Advanced Robotics Systems*, 3(1):43–48, 2006.