Human Activity Recognition from Accelerometer and Gyroscope Data

Jafet Morales

University of Texas at San Antonio
8/8/2013

Human Activity Recognition (HAR)

- My goal is to recognize low level activities, such as:
  - walking
  - running
  - jumping jacks
  - lateral drills
  - jab cross punching
  - Michael Jackson’s moonwalk

Accelerometer Based Classification

- Most accelerometer based HAR done with supervised learning algorithms [1]
  - Class labels for training feature vectors are known beforehand
  - As opposed to unsupervised learning, where only the number of classes is known

Equipment

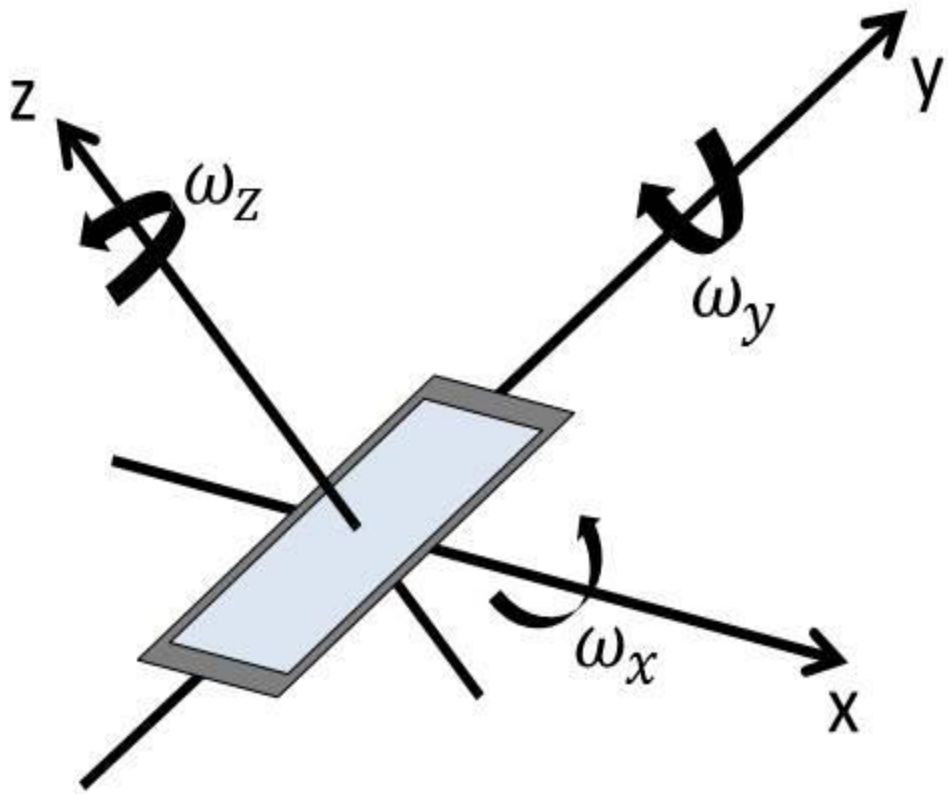


Fig. 1. A motion tracking device that can measure acceleration and angular velocity

InvenSense MPU-6050

- 6-axis motion tracking device
  - Accelerometer and gyro sensor
- 4x4x.9mm
- 16-bit output that can span 4 different ranges selected by user
- Can add an additional 1-axis (i.e., digital compass) or 3-axis sensor through I2C
- Onboard Digital Motion Processor (DMP) can process signals from all sensors

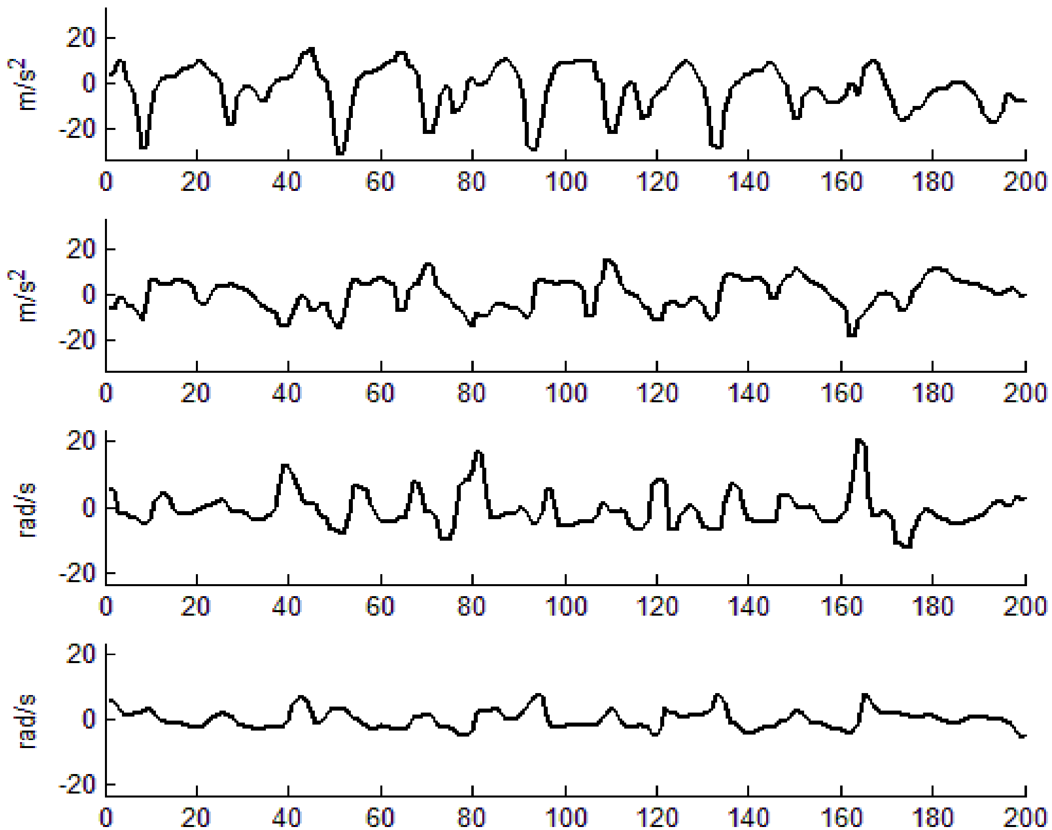
- MPU-6050 is hosted by another device
- Hosting device provides us with samples at 50Hz sampling rate

Our Method - Preprocessing

- We calculate PCA transform on frames of accelerometer and gyroscope data
- After calculating PCA, the coordinate system is changed
  1. The principal component will have the most variance
  2. The next component will have the maximum variance possible while being orthogonal to the principal component
  3. The last component will be orthogonal to these two components, and point in the direction given by the right hand rule

After PCA

- Jogging frames



acceleration components 1 and 2 (m/s^2)

angular velocity components 1 and 2 (rad/s)

Signals for feature extraction

Table 3. List of signals from which to extract features.

Signal type	Axis
Acceleration	along Principal axis
	along Second axis
	along Third axis
Angular speed	about Principal axis
	about Second axis
	about Third axis

Original Feature Set

- Use features from papers [2][3]
- And introduced some new features
- From all of those features, only a few were selected to be used in the system
- The process by which we select an optimum set of features is called **feature selection**

Greedy Backwards Search for Feature Selection

- Preselect a set of features
- Iteratively
  - Remove one feature at a time
  - The one that maximizes a goodness metric after it is deleted
- Stop when accuracy cannot be increased anymore or there is only one feature left

The "Goodness Metric"

- A correlation-based feature selection method has been used [4]
  - Correlation between each feature and class is maximized in the final feature set
  - Correlation among features is minimized in the final feature set
- Goodness metric is [5]

$$Merit_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \qquad (1)$$

where k is the size of the subset, $\overline{r_{ff}}$ is the mean for Pearson's correlation between each feature-feature pair and $\overline{r_{cf}}$ is the mean for Pearson's correlation between each feature-class pair

- Some features are discrete
- Classes are categorical
- We need to discretize all features and classes to be able to calculate the goodness metric [4]
- We then use a quantity that works with discrete variables instead of Pearson's cross correlation *r*
  - **mutual information** or **information gain**
- First define **entropy information**

$$H(Y) = -\sum_{y \epsilon Y} p(Y) log_2 p(Y)$$

where *Y* is a discrete variable and *p* is its pdf

- Conditional entropy

$$H(Y|X) = -\sum_{x \epsilon X} p(X) \sum_{y \epsilon Y} p(Y|X) log_2 p(Y|X)$$

is the expected amount of information in Y when X is known.

- **Information gain** or **mutual information**

$$inf.gain(X,Y) = H(Y) - H(Y|X)$$

- Is the reduction in the information entropy of Y when X is known
- If it is easy to predict Y by looking at X (i.e., each X maps to a single Y), then $H(Y|X)$ will be low and inf. gain (mutual info) will be high. Also,

$$inf.gain(X,Y) = inf.gain(Y,X)$$

Sensor	Axis	Feature
Accel.	Mean acceleration (1,2,3)	μ_a
Accel.	Acceleration signals Correlation (1,3)	$\frac{1}{N}\sum_{n=0}^{N}\frac{(a_1[n]-\mu_1)(a_3[n]-\mu_3)}{\sigma_1\sigma_3}$
Accel.	Standard deviation (1)	σ_{a_1}
Accel.	Signal Magnitude Area (1)	$\frac{1}{N}\sum_{n=0}^{N}\lvert a_1[n]\rvert$
Accel.	Power (1)	$\frac{1}{N}\sum_{n=0}^{N}{a_1}^2[n]$
Accel.	Power (3)	$\frac{1}{N}\sum_{n=0}^{N}{a_3}^2[n]$
Accel.	Power contained in [8.1Hz to 16.1Hz] (2)	Power [8.1Hz to 16.1Hz]
Accel.	Entropy (1)	$H(a_1[n])$
Accel.	Entropy (2)	$H(a_2[n])$
Accel.	Entropy (3)	$H(a_3[n])$
Accel.	Repetitions per second (1)	Rep/s
Gyro	Mean angular speed (1,2,3)	μ_ω
Gyro	Correlation (1,2)	$\frac{1}{N}\sum_{n=0}^{N}\frac{(\omega_1[n]-\mu_1)(\omega_2[n]-\mu_2)}{\sigma_1\sigma_2}$
Gyro	Standard deviation (1)	σ_{ω_1}
Gyro	Signal Magnitude Area (2)	$\frac{1}{N}\sum_{n=0}^{N}\lvert a_2[n]\rvert$
Gyro	Power contained in [10Hz to 20Hz] (3)	Power [10Hz-20Hz]
Gyro	Entropy (1)	$H(\omega_1[n])$

Normalizing Mutual Information

- Let's say we have a feature *Y* and a feature *X* that is highly correlated with it
- Then $H(Y|X)$ will be zero
- In such case the $inf.\ gain(Y,X) = H(Y)$
- This means that for a uniform distribution, the more categories a feature has, the higher the info. gain it will get when compared to the classes or to any other feature
- So we normalize *inf. gain* to always get a value in the range [0,1]

$$sym.\ uncertainty(X,Y) = 2 - \frac{inf.gain(X,Y)}{H(X) + H(Y)}$$

- Then we substitute sym. uncertainty into (1)

Classification Stage

- Tried 2 classifiers
  - kNN
  - Multilayer perceptron

kNN

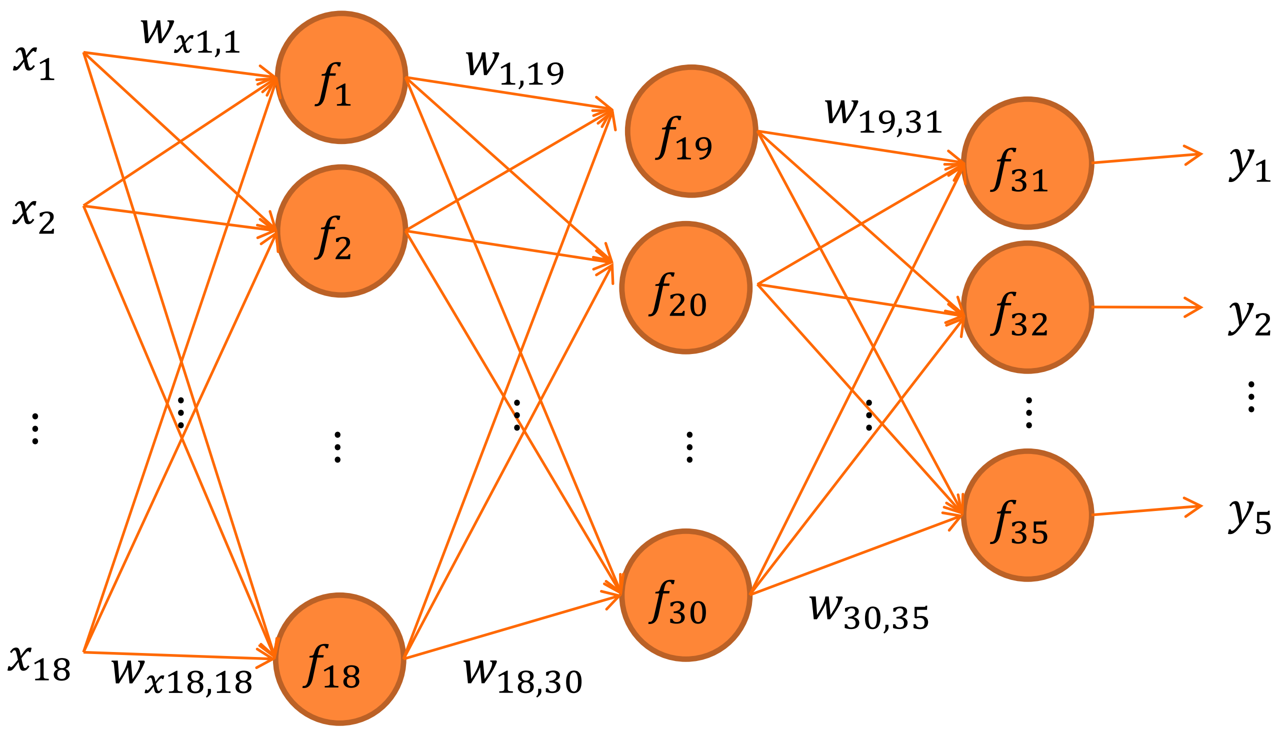
1. Find *k* nearest training sequences in feature space using Euclidean distance

2. For each class (i.e., walking, running) add the number of cases that fall within the *k* nearest neighbors for both classifiers

4. Select the class with the most votes

Multilayer Perceptron

- 1 hidden layer with 12 units
  - Obtained by using rule of thumb:
    - # hidden units = round(# attributes + # classes) / 2)= round((18+5)/2)=12



Multilayer Perceptron Training

- MLP was trained by using backpropagation algorithm
- This algorithm uses gradient descent to train the weights in the network
  - In other words, modify each weight in the direction that will diminish the error at the output of the layer

$$\boldsymbol{v} = \boldsymbol{v}_{old} - \mu \nabla Error(\boldsymbol{v}_{old})$$

  - Gradient is made of partial derivatives, and in case of weights, each partial derivative corresponds to a weight

Backpropagation

- So to modify each weight individually, we use the partial derivative of an error function with respect to that weight [6][7]

$$\frac{\partial E_{p_j}}{\partial w_{ij}} = \boxed{\frac{\partial E_{p_j}}{\partial a_j}} \boxed{\frac{\partial a_j}{\partial w_{ij}}}$$

$$E_{p_j} = \frac{1}{2} \sum_{j=first\ node\ in\ current\ layer}^{last\ node\ in\ current\ layer} (t_j - a_j)^2$$

i=input node

j=output node

t_j = desired output at node j

a_j = output of node j = $f_j(e_j)$

$$e_j = \sum_{i=first\ node\ in\ input\ layer}^{last\ node\ in\ input\ layer} w_{ij} a_i$$

$$\frac{\partial E_{p_j}}{\partial a_j} = -(t_j - a_j)$$

the only unknown parameter that impedes gradient descent!!!

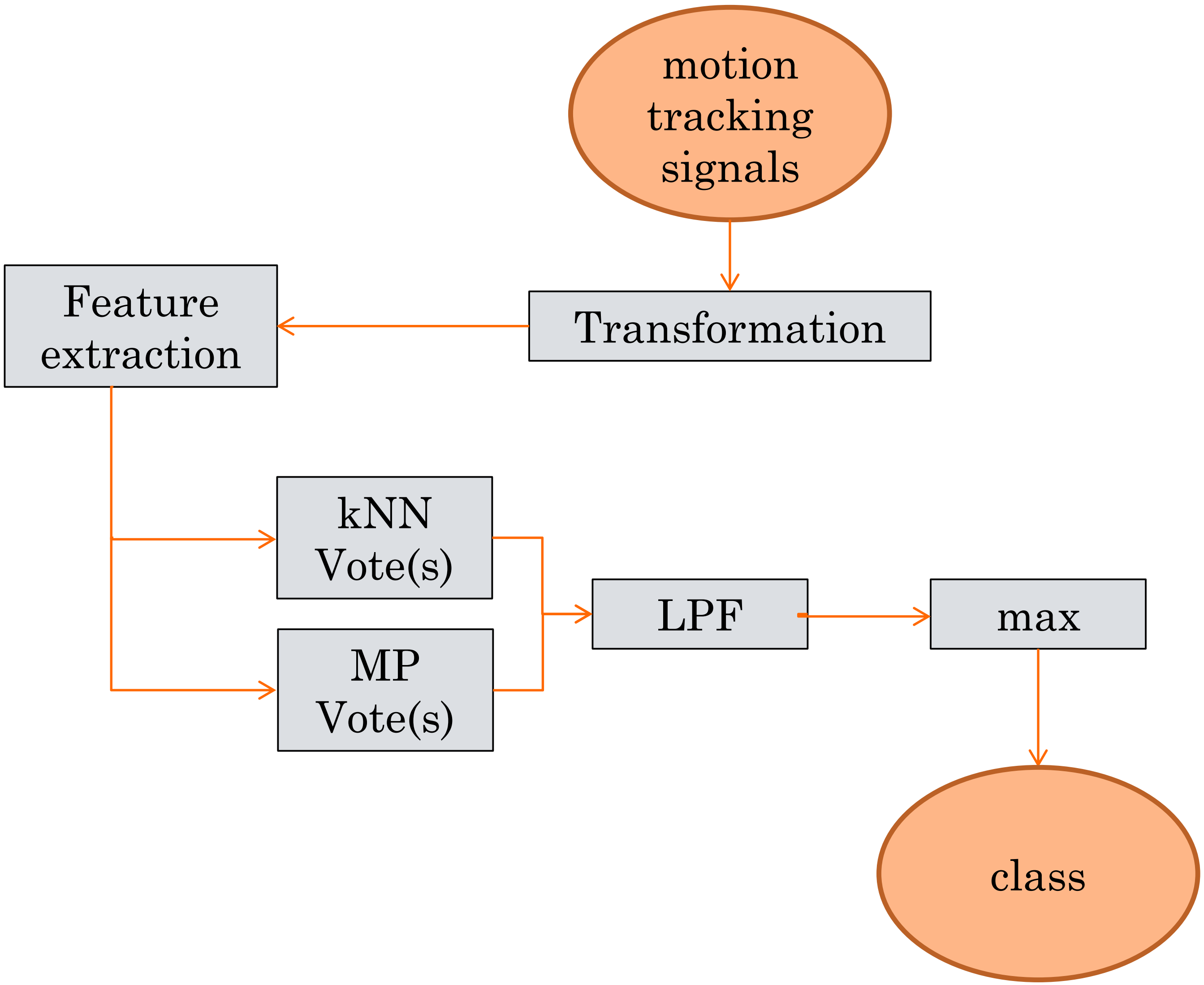
$$\frac{\partial a_j}{\partial w_{ij}} = \frac{\partial a_j}{\partial e_j}\frac{\partial e_j}{\partial w_{ij}} = \frac{\partial f_j(e_j)}{\partial e_j}\frac{\partial \sum w_{ij} a_i}{\partial w_{ij}} = \frac{\partial f_j(e_j)}{\partial e_j} a_i$$

- we may not know $t_j - a_j$ because we do not know t_j except at the outermost layer, but we can guess it
  - The guess for $t_j - a_j$ at each node is calculated by backpropagating the error at the output to the neurons in previous layers
  - This is done by reversing the arrows in the network and using $t_{(one\ of\ last\ layer\ nodes)} - a_{(one\ of\ last\ layer\ nodes)}$ as inputs
- Then we use δ_j=error propagated back to node j

as a substitute for $t_j - a_j$

- It turns out even if you reach a peak for $t_j - a_j$, you will be approaching a valley for the last layer

Real-time Setup



Evaluation of Results

- Train and test on single subject
- Train on several subjects and test on one subject

Results

Simulation of method in [5]			
Single subject		Unknown subject	
C4.5	kNN	C4.5	kNN
82.2	**93.1**	**56**	**63.2**
Simulation of method in [7]			
Single subject		Unknown subject	
C4.5	MP	C4.5	MP
77.1	**96.2**	**68**	**66.4**
Simulation of proposed method			
Single subject		Unknown subject	
KNN	MP	KNN	MP
100	**100**	**98.4**	**99.2**

Confusion

Table 3. Confusion table for proposed method

	Still	Walk	Jog	Jump jack	Squat
Still	**25**	0	0	0	0
Walk	0	**25**	0	0	0
Jog	0	0	**25**	0	0
Jump jack	0	0	0	**25**	0
Squat	0	**1**	0	0	**24**

Conclusion

- The proposed algorithm allows for highly accurate human activity recognition without imposing any constraints on the user, except for the requirement to place the smartphone in his front right pocket.

References

[1] Bao, L. and Intille, S. 2004. Activity Recognition from User-Annotated Acceleration Data. Lecture Notes Computer Science 3001, 1-17.

[1] Mohd Fikri Azli bin Abdullah, Ali Fahmi Perwira Negara, Md. Shohel Sayeed, Deok-Jai Choi, Kalaiarasi Sonai Muthu Classification Algorithms in Human Activity Recognition using Smartphones, International Journal of Computer and Information Engineering 6 2012 URL: http://www.waset.org/journals/ijcie/v6/v6-15.pdf

[2] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L. Littman. 2005. Activity recognition from accelerometer data. In *Proceedings of the 17th conference on Innovative applications of artificial intelligence - Volume 3* (IAAI'05), Bruce Porter (Ed.), Vol. 3. AAAI Press 1541-1546.

[3] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. 2011. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.* 12, 2 (March 2011), 74-82.

[4] Mark A. Hall and Lloyd A. Smith. 1999. Feature Selection for Machine Learning: Comparing a Correlation-Based Filter Approach to the Wrapper. In *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, Amruth N. Kumar and Ingrid Russell (Eds.). AAAI Press 235-239.

[5] Ghiselli, E. E. 1964. *Theory of Psychological Measurement.* McGraw-Hill.
[6] http://home.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html

[7] http://www.webpages.ttu.edu/dleverin/neural_network/neural_networks.html