Capstone Proposal for User Activity Recognition on Mobile Devices

1、Domain Background

User Activity Recognition(UAR) is a research problem about identifying human movement state via carry-on sensor signals, which can assist take measures or supply services correspondingly. From illness to daily life, from mental to emotion, they all can by reflected by specific behavior. Meanwhile, specific behavior can be captured by relative sensor information. So UAR is feasible and effective. For example, we can achieve health monitoring via analyzing the amount of activities in daily life. Besides, in area of context awareness, UAR can be combined with other related information of the user's surroundings to infer the user's behavior motivation and provide the convenient service initiatively. All above have positive effects on improvement of people's living standard. UAR is an interdisciplinary research relating to machine learning, data mining, signal processing and so on. The basic procedure includes collecting raw data from sensors, data cleaning and preprocessing, feature extraction, and finally training models for classification or prediction.

According to the type of source data, activity recognition can be roughly divided into three categories: (1)、Vision-based activity recognition. It deals with pictures or video sequences to capture human motions. (2)、Sensor-based activity recognition. Recently, the approach has been receiving much attention, because mostly activities can be inferred from collected data of inertial sensors, such as accelerometer sensors and gyroscope sensors. And now we can easily get access these kinds of sensor data from mobile devices, e.g. smart phone, smart watches, bracelets, etc. So it is not only convenient but also providing good result. (3)、Environment-based activity recognition. Identify human activity and surroundings where he/she is in using RFID, sound, temperature, light intensity and other information around.

2、Problem Statement

The problem we want to solve is collecting time series sensor data from

accelerator, gyroscope, magnetometer and etc. built inside phones, extracting time-domain and frequency-domain features and finally achieving analyzing users' movement state, such as sedentary, walking, running, biking, in car, in train and etc.

In order to avoid the interference caused by the diversification of hands, we require smart phones to be placed in pants when collecting data, so that sensor signals can express speed change and various rotation angle of human bodies. For example, open dataset UCI HAR Dataset is obtained by tying devices to the waists of testing objects.

3、Datasets and Inputs

The input consists of time series sensor data from accelerator, gyroscope, magnetometer and etc. built inside phones. Fig.1. shows examples of input data:

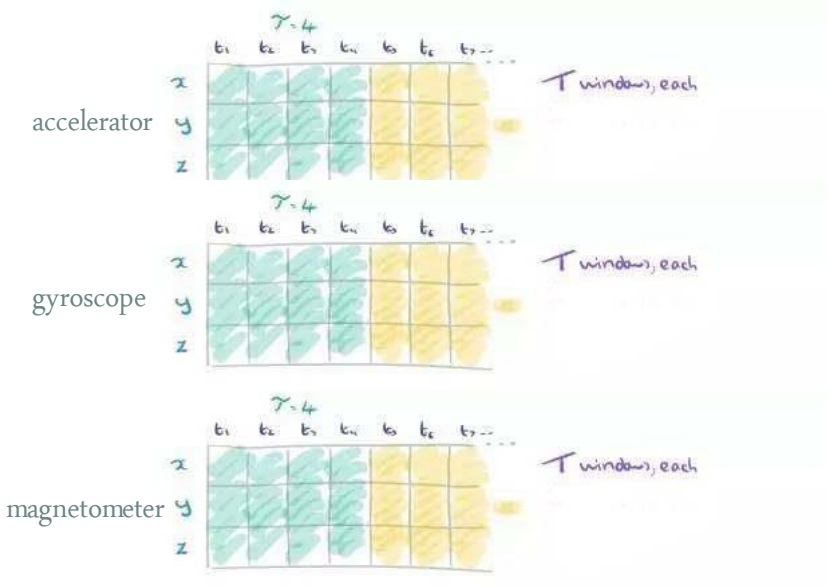

Fig.1.Time-series sensor signals

The sensors include three axes: x, y, z. We segment sensor signals via sliding window, which is later used for feature extraction. Finally follows classification model training.

4、Solution Statements

Fig.2.shows the sketch map for solution statements. It mainly contains 6 modules:

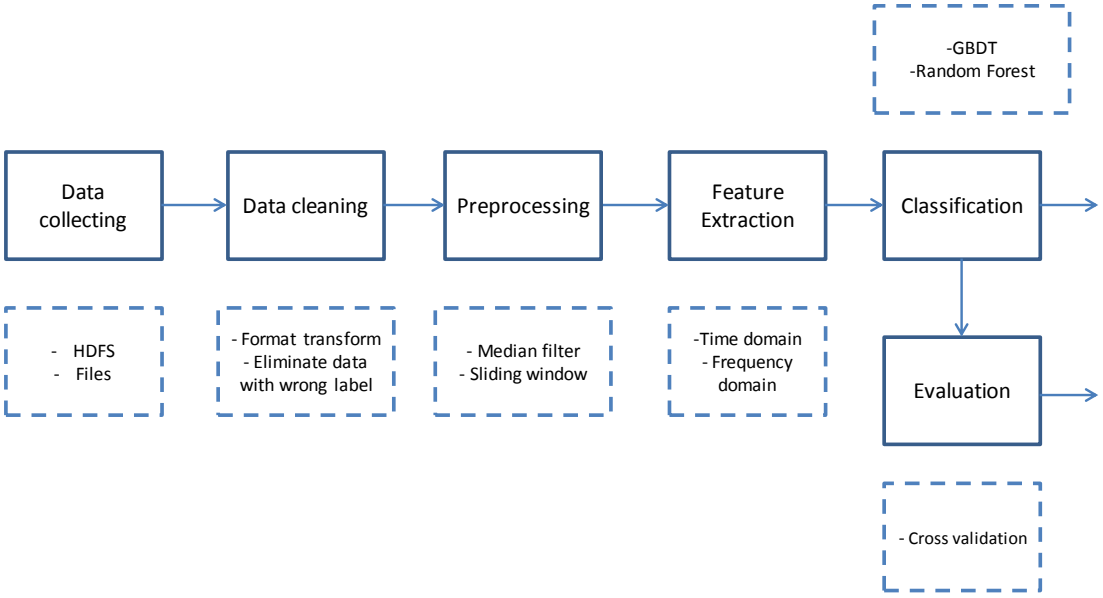

Fig.2.Sketch map for solution statement

1、Data Collecting: This module is mainly responsible for collecting labeled raw data, which is later used for training model. The method can include local record or remote storage on distributed clusters, but not limit to these.

2、Data Cleaning: Apply data cleaning and preprocessing on massive labeled data, including format transformation, eliminating data with wrong label.

3、Preprocessing: Segmenting time series data into discrete sliding window, then apply smoothing filter on each window.

4、Feature Extraction: Computing feature vectors on time-domain and frequency-domain for each sliding window.

5、Model Training：Training models with the extracted feature dataset, adjusting parameters and optimizing the results. We can use Random Forest, GBDT and so on, but not limit to these.

6、Outcome Evaluation: Using cross validation on model prediction and label to calculate accuracy.

5、Benchmark Model

At present, there are a lot of papers on UAR, but they may various at sensor types and activity classes. So in order to comparable with previous results, we refer to Ortiz J L R. Smartphone-Based Human Activity Recognition[M]. Springer International Publishing, 2015. It uses improved SVM for classification. The results are as follows:

Method	PTA-6A							
Activity	WK	WU	WD	SI	ST	LD	PT	UA
WK	**542**	1	0	1	0	0	1	1
WU	26	**510**	1	0	1	0	16	5
WD	0	0	**506**	0	3	0	0	0
SI	0	0	0	**499**	57	0	1	0
ST	0	0	0	3	**607**	0	2	0
LD	0	0	0	0	0	**604**	0	0
PT	0	0	0	2	0	0	**327**	0

The meaning of abbreviations in table above are WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LYING_DOWN, POSTURAL TRANSITION respectively. According to statistical analysis, the overall accuracy is 96.8%.

6、Evaluation Metrics

The following concepts are given before defining the evaluation criteria:

(1)、TP_i(true positive): numbers of samples which is predicted to be C_i class, and actually belongs to C_i class;

(2)、FP_i(false positive): numbers of samples which is predicted to be C_i class, but actually not belongs to C_i class;

(3)、TN_i(true negative): numbers of samples which is predicted to be other classes except for C_i, and actually not belongs to C_i class;

(4)、FN_i(false negative): numbers of samples which is predicted to be other classes except for C_i, but actually belongs to C_i class;

Basing on the concepts above, the overall accuracy is used as the evaluation index for the performance of models, the definition is as follows:

$$\text{Overall accuracy} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FN_i)}$$

Overall accuracy indicates the proportion of correctly predicted sample number to total number, which can reflect the overall classification ability of the model.

7、Project Design

Below is the implement statement for each module:

1、Data Collecting: We choose open dataset for experiment. Since different datasets have different activity classes and different sensor types as well, so we choose Smartphone-Based Recognition of Human Activities and Postural

Transitions Data Set in order to simulate the situation, in which users carry smart phones with them in daily life. The dataset contains 12 kinds of activities, and source data are from two sensors, namely accelerator and gyroscope.

2、Data Cleaning: The step can be omitted for open source data. But if we want to collect data by ourselves, it is necessary to clean and preprocess the raw sensor data, such as format transformation, eliminating data with wrong label.

3、Preprocessing: Segmenting time series data into discrete sliding window, as Fig.3. shows. For example, sorting sensor data of movement $state_j$ for $user_i$ by time; Segmenting time series sensor signals via a sliding window of size w, and sliding step size is s. The window data is used for feature extraction for the next module. All above are the same for other users and activities.

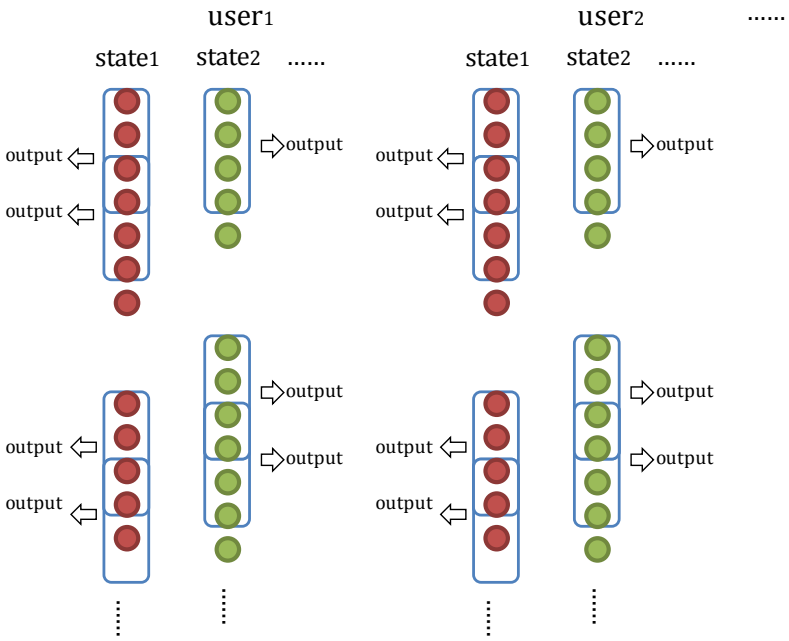

Fig.3.Segmenting time-series sensor data via sliding window

4、Feature Extraction: Table1 offers lists of measures for computing feature vectors of each window. We can also apply smoothing filter (e.g. median filter) beforehand, so that the "characteristic" acceleration of phones in three dimensions can be as close to human movement as possible.

Table1: List of measures for computing feature vectors

Function	Description	Formulation
Mean(**w**)	Arithmetic mean	$\bar{w}=\frac{1}{N}\sum_{i=1}^{N} w_i$

Std (**w**)	Standard deviation	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(w_i - \bar{w})^2}$
MathQ1(**w**)	The first quartile	Q1(**w**)
MathQ3(**w**)	The third quartile	Q3(**w**)
Diff	Interquartile range	Q3−Q1
Energy(**w**)	Average sum of the squares	$\frac{1}{N}\sum_{i=1}^{N} w_i^2$
Max(**w**)	Largest value in array	$\max_i(w_i)$
Min(**w**)	Smallest value in array	$\min_i(w_i)$
Range(**w**)	Range between maximum and minimum	$\max_i(w_i) - \min_i(w_i)$
MAE(**w**)	Mean of absolute deviation	$\frac{1}{N}\sum_{i=1}^{N}\lvert w_i - \bar{w}\rvert$
Correlation(w_1, w_2)	Correlation coefficient between two signals	$C_{1,2}/\sqrt{C_{1,1}C_{2,2}}$, C=cov($w_1, w_2$)
JumpFall(**w**)	Largest continuous rise height and descent slope	$\max(\sum w_{i+1} - w_i)$, where $w_{i+1} > w_i$ $\max(\sum w_i - w_{i+1})$, where $w_i > w_{i+1}$
FFT(**w**)	Flourier transformation	$F(w)=\mathcal{F}[f(t)]=\int_{-\infty}^{\infty} f(t)e^{-iwt}dt$ Take top3 dominant frequencies and corresponding amplitudes.

5、Model Training：Training models with the feature dataset above, adjusting parameters and optimizing the results. Below are the sample codes for calling Random Forest and GBDT interface in python respectively.

```
# Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier
def random_forest_classifier(train_x, train_y):
    model = RandomForestClassifier(n_estimators=100, criterion='entropy',
max_depth=50, min_samples_split=5, min_samples_leaf=1, oob_score=True)
    model.fit(train_x, train_y)
    return model

# GBDT(Gradient Boosting Decision Tree) Classifier
from sklearn.ensemble import GradientBoostingClassifier
def gradient_boosting_classifier(train_x, train_y):
    model = GradientBoostingClassifier(n_estimators=100,
learning_rate=0.1, max_depth=50, loss='deviance', min_samples_split=5,
min_samples_leaf=1, subsample=0.9)
    model.fit(train_x, train_y)
    return model
```

6、Outcome Evaluation: Using 10-folds cross validation on model prediction and label to calculate accuracy. Below is the sample code in python:

```
from sklearn.metrics import accuracy_score
from sklearn.cross_validation import StratifiedKFold
crossValidation = StratifiedKFold(y_train, n_folds=10)
bestAccuracy = 0.0
for train_index, test_index in crossValidation:
    X_train, X_test = x_train[train_index], x_train[test_index]
    Y_train, Y_test = y_train[train_index], y_train[test_index]
    model = random_forest_classifier(X_train, Y_train)
    predictResult = model.predict(X_test)
    accuracyScore = accuracy_score(Y_test, predictResult)
    if accuracyScore > bestAccuracy:
        bestAccuracy = accuracyScore
```

--The End