Introduction: Adaptive activity classification with mobile devices

Activity classification

Mobile phones can be used to detect user’s mobility state (i.e., walking, running, etc.). They use activity classifiers to infer what the user is doing from sensor data. The activity classifier model is trained with labeled sensor data. Traditionally, the classifier’s accuracy has depended on a robust training data set in order to adequately train the model to recognize activities accurately, even for different users whose motions (i.e. gait) might differ slightly during the activity.

Adaptive activity classifier

Activity classifiers depend on sufficient training data to provide accurate results, and sometimes they are still not effective for some users. We use semi-supervised learning algorithms to allow the classifier to automatically continue to train itself for each specific user over time. As it makes classifications, the classifier selects certain data points to be added to the training data. For the label, the classifier uses its own prediction. We tested several algorithms to determine their viability for mobile activity classifiers, and implemented one in a working application.

Problem Description: Create a dynamically adaptive activity classifier

Applying semi-supervised learning to augment and personalize the classifier’s training data

Self-customizing activity classification on a mobile device

The classifier can adapt to each user as they use the application.

Potential algorithm paradigms to choose new data for training set

- Self-learning
- Co-learning
- Active learning

Comparing the performance of the algorithms

Which is the most effective for our activity classifier?

Semi-supervised activity classifier flow graph

The classifier chooses and labels new samples, which are used to retrain the classifier

Proposed Solution: Use semi-supervised learning and retrain classifiers after deployment

Semi-supervised learning system

Dynamic classifier system design

The phone classifies the user’s activity in real time, and selects data for new training samples according to the semi-supervised or active learning algorithm. The new training samples are uploaded to a server which generates new classifier models. The phone then downloads the updated model parameters and replaces the old classifier with an updated one. This process can repeat for multiple iterations.

A server is used for creating new classifiers

The training data is stored on a server so that a user does not lose their personalized classifier and training data when they uninstall the mobile application. Also, the training data set may become large and require more space than is acceptable in mobile devices. Since generating an updated classifier happens only at intervals (i.e. once per day), this task can conveniently be done the server when the phone has internet connectivity.

We used the Weka machine learning toolkit to create classifiers.

Offline analysis of potential learning algorithms

How should the phone select new samples for the training data? We tested several algorithms:

- Self-learning with high-confidence samples
- Co-learning with three classifiers (C4.5 decision tree, Naive Bayes, and SMO) trained on the same feature set
  - En-Co-training (Guo, 2007) – Select samples where there is a consensus among the classifiers
  - Democratic co-learning (Zhou, 2004) – When one of three disagrees, train the deviant classifier with the prediction of the two others
- Active learning with low-confidence samples – prompt the user for a label

Conclusions

- Democratic co-learning provided the best tradeoff between performance and convenience
- It was able to improve the accuracy of classifiers up to around 90%.
- This method is most useful when it is most needed: when a classifier performs poorly “out-of-the-box”
- Democratic co-learning is suitable for real-time mobile activity classifiers

Algorithm performance

Comparing the performance of the algorithms (above)

The horizontal black line shows the initial accuracy. The colors represent the number of classifier iterations over which new training data was added. Active learning gave the greatest improvement, but requires user interaction to label new data samples. Democratic co-learning achieved close to the same accuracy while requiring no user input, making it the best choice.