Morgan Wallace
Data Mining 290 Spring 2014

Data Mining for Classification of Exercise

Introduction

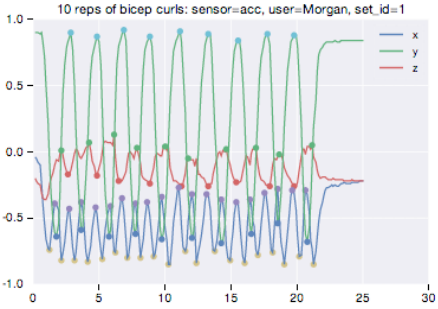
In my MIMS final project, Hercubit (a wearable device), I need to provide live weight-lifting feedback for users; 2 such feedback metrics are count (repetition) and exercise type. After visualizing the raw sensor data (accelerometer, gyroscope and magnetometer), I realized that these 3 types of arm exercises have distinct features. My solution to this problem is also my final project for Data Mining, 290; I have trained a Support Vector Machine (SVM) classifier to distinguish these 3 types of exercise with 88% accuracy against a holdout data set. Additionally, I have implemented the classifier in such a way that makes live classification and counting of repetitions possible.

Problem

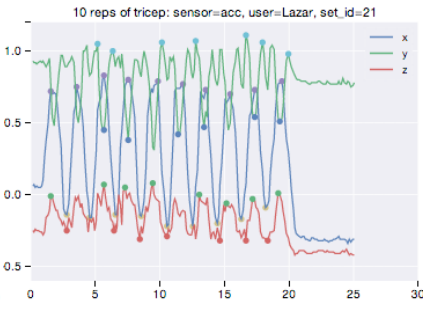
Teaching a computer to spot a bicep curl or other exercise in a sea of noisy motion sensor data, in real-time, is extraordinarily complicated. Not only must the best features be used in the classifier but also the classifier must be able to give real-time insights as to the activity currently being performed so that a repetition counter can know what type of repetition to look for. This means that every 100 milliseconds that a new sample of raw sensor data comes in, the system must determine what type of exercise is likely being performed and then, when has this exercise completed so we can start to look for the next.

The 3 graphs below give a sense of the data. They show accelerometer values for 10 repetitions of the 3 types of exercise.

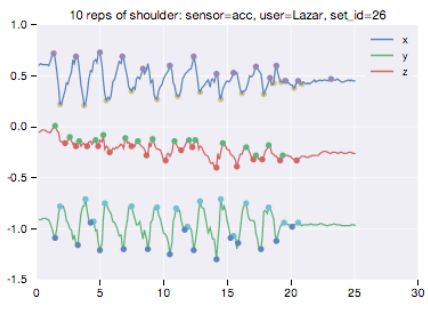
Bicep Curls



Tricep Kickbacks



Shoulder Press



Solution

To best deal with this problem, I collected 83 sets of exercise from 15 different individuals totaling 784 repetitions. I only chose to classify 3 types of exerciseUsing Python (specifically iPython Notebooks, Pandas, MatPlotLib, MPLD3, Seaborn, and Numpy), I analyzed patterns, visualized the data, and applied labels using a customized peak detection algorithm.
From this analysis, I derived my features for classification. I chose SVM because I can add a large number of features and explore the usefulness of various kernel functions. Also, when training the SVM, I

set the ‘probability’ parameter to True which allows use of the .predict_proba() method that make the real-time classification possible.

Details

After collecting data using save_graph_and_data.py, I combined the importing of new data, labeling that data, re-training the model, testing the model, and saving the performance of the model into one function, ‘make_model()’ within notebooks/SVC.ipynb. This one-step-build allows me to easily add training data or try new features. Note: *training and holdout sets are CSV files in the* data/ directory.

My latest and best model is fairly accurate linear kernel SVM model with a f-1 score of 88% against a holdout set of data. See below:

Cross validated testing

	precision	recall	f-1	support
Bicep	1.00	0.97	0.99	38
Shoulder	1.00	0.97	0.98	63
Tricep	0.94	1.00	0.97	47
avg/total	0.98	0.98	0.98	148

VS holdout set

	precision	recall	f-1
avg	0.88	0.88	0.88

Using python’s pickle library to save and load the classifier, I use a custom module ‘hercubit’ to connect with a device and get live sensor data or, in the case of no connection available, iterate through sample data to test the real-time classification and counting accuracy. It is considerably more accurate than my previous, rudimentary If/Else classifier, however, real motion data is messy and the training set was clean (very little extraneous movements). For more information on running my code, see README.md or readme.html.

Related Work

Before starting this work I was inspired by the work done in this paper, but I wanted to make a version that would classify live and without a belt clip sensor.

1. Keng-Hao Chang, Mike Y. Chen, and John Canny. 2007. Tracking free-weight exercises. In Proceedings of the 9th international conference on Ubiquitous computing (UbiComp '07), John Krumm, Gregory D. Abowd, Aruna Seneviratne, and Thomas Strang (Eds.). Springer-Verlag, Berlin, Heidelberg, 19-37. http://dl.acm.org/citation.cfm?id=1771594

Further Work

1. Explore new features & optimize
    A. Add features such as velocity, duration, and previous repetition type.
    B. Account for extraneous movement noise in real-time classifier.
2. Add more types of exercise
    A. There are hundreds if not thousands of possible exercises to track, Explore
3. Try new types of classifiers like Naive Bayes, decision trees, and random forests, such classifiers would give insight as to the the most informative features and could potentially be more reliable than the approach with SVM