Improving Activity Recognition Via Automatic Decision Tree Pruning

Thomas Phan
Samsung Research America - Silicon Valley
San Jose, CA
thomas.phan@samsung.com

UbiComp'14, September 13 - 17 2014, Seattle, WA, USA

http://dx.doi.org/10.1145/2638728.2641310

Abstract
Activity recognition enables many user-facing smartphone applications, but it may suffer from misclassifications when trained models attempt to classify previously-unseen real-world behavior. Our system mitigates this problem by first identifying spurious classifications and then automatically pruning a decision tree model to remove labels that tend to produce wrong inferences, resulting in a 10% classification improvement based on our data set.

Author Keywords
Activity recognition; accelerometer; signal classification

ACM Classification Keywords
I.5.4 [Pattern recognition]: Applications.

Introduction
Activity recognition is the process of determining user physical behavior from sensor data [1, 5, 6, 4, 2]. In our work we are especially interested on recognizing human modes of transportation (comprising driving, walking, running, bicycling, and idling) using only a smartphone's accelerometer due to its low power consumption.

These physical activities are relevant because they enable a variety of user-facing smartphone applications; for example, recognizing that the user is driving can trigger a

reduced-distraction user interface, and recognizing long periods of sedentary sitting can alert the user to get up to exercise. Because these applications perform a real-time, user-facing action (as opposed to simply life-logging the user's activity), it is important that misclassifications be minimized in order to prevent a degraded user experience.

We use supervised machine learning in our work, where misclassifications can occur due to two reasons. First, the trained model may be overfit, incorporating noise from the training data. Second, the model may be insufficient, producing spurious classifications for real-world behavior. Any training data from *scripted* behavior results in a model that is able to accurately classify single activities but has difficulty when users perform arbitrary, *naturalistic* behavior that may involve either activity transitions or completely new activities. The problem stems from the data collection, where gathering labelled mixed-activity traces is prohibitively time-consuming. Some approaches to treating naturalisitic activities include: labelling videotaped user behavior [3]; labelling users performing real-time multiple activities [5]; having users report their own activities [8]; and defining a "null" class [2].

In this paper we apply a novel approach to handling such spurious classifications. We first train an activity recognition model using a C4.5 decision tree [7]; we use a decision tree because the model occupies under 10KB of memory and classifies quickly, requiring a number of floating-point operations proportional to the tree height. We then judiciously and automatically prune the tree *after the model is built*. We take advantage of the C4.5 tree's redundant labelled leaves and apply thresholding to remove the leaves that tend to produce misclassifications. The result is an improvement of up to 10% classification accuracy based on our collected data set.

Building the model

In this paper we describe a non-commercialized prototype of a low-power accelerometer-only activity recognition component that generates a continuous stream of inferred human modes of transportation. We measured the system to consume under 225 mW of power, which includes the baseline CPU, on a Samsung Galaxy S III phone.

The activity recognition system runs in real-time on Android phones and detects modes of transportation using an accelerometer signal classifier. The end-to-end system comprises two distinct phases. First, in an offline training phase, 20 test participants (15 male and 5 female) performed fixed physical activities while wearing data-collecting smartphones. After we extracted sensor features, we trained a C4.5 decision tree model. Second, we loaded the model into our Android activity recognition software to perform real-time classification.

We sampled the accelerometer at 32 Hz and kept a 128-sample sliding window with 50% overlap. Because the phone can be oriented in different directions, we normalized the readings into three orientation-independent time series: Cartesian magnitude of the acceleration vector; projection onto the true horizontal plane; and projection onto the true vertical axis. For each of the three time series, we computed the features: (i) time-domain mean, standard deviation, power, and entropy; and (ii) frequency-domain energy, average of the top-5 highest-magnitude frequency, frequency with the highest magnitude, the highest magnitude, and weighted variance. In all, 9 features are extracted from each time series, resulting in 27 total features. Our system demonstrates a 98.1% per-window classification accuracy with 10-fold cross-validation with our five activities: $Walking$, $Running$, $Driving$, $Biking$, and $Idling$.

Per-class summary statistics and the confusion matrix are shown in Tables 1 and 2, respectively.

Class	TP Rate	FP Rate	Prec	Recall	AUC
Walking	0.99	0.004	0.988	0.99	0.996
Idling	0.987	0.011	0.983	0.987	0.998
Running	0.971	0.00	0.997	0.971	0.996
Driving	0.97	0.01	0.971	0.97	0.995
Biking	0.966	0.003	0.966	0.966	0.996
Weighted Avg.	0.981	0.007	0.981	0.981	0.996

Table 1: Activity recognition results showing per-class true positive rate, false positive rate, precision, recall, and area under the ROC curve.

		Predicted				
		Walking	Idling	Running	Driving	Biking
Actual	Walking	**98.97%**	0.25%	0.06%	0.47%	0.25%
	Idling	0.00%	**98.73%**	0.02%	1.14%	0.12%
	Running	2.88%	0.00%	**97.12%**	0.00%	0.00%
	Driving	0.29%	2.20%	0.00%	**96.99%**	0.52%
	Biking	0.28%	0.56%	0.00%	2.51%	**96.65%**

Table 2: Activity recognition confusion matrix.

Pruning the model

We augmented our classifier with an algorithm that improves classification for real-world usage. As we showed, the system achieves a 10-fold cross-validated accuracy of 98.1% on windowed data over our five activities (with no null class). While this result is appealing and is in line with prior work (e.g. [6, 4]), it is misleading because it applies only to the single-activity, scripted activity behavior recorded during training. Even with cross-validation, two problems arise: (1) the built model may be overfit to the data; and (2) mixed-activity, naturalistic human behavior contains many motions and transitions that were not captured during training. As a result, the model can return spurious classifications in real-world use. Indeed, we later show that the model achieves only 87.8% accuracy when applied to a hold-out set of real-world naturalistic behavior.

Consider the following output of our system that illustrates the problem stemming from a user walking up to a car, getting keys out of his pocket containing his smartphone, and then driving off, where "getting keys out of pocket" is one of an infinite number of naturalistic behaviors. Let `W`, `I`, `R`, and `D` be the output classification labels for walking, idling, running, and driving, respectively, in the following left-to-right sequence:

`WWWWWWDWDWDIIRDDDDDD`

The underlined labels in the above sequence are the spurious classifications that occurred from getting keys out of the pocket. Ideally, the system should output the labels for walking to driving without the interruption of the unknown physical act. Such choppiness can be avoided by applying smoothing (e.g. with a Markov model), but our belief is that it is better to remove misclassifications at the source as early as possible in order to reduce pollution of any smoothing window.

Looking back to the sequence above, it is important to understand why the underlined labels may be considered spurious while other instances of the same label are not; for example, `D` appears to occur in both spurious and valid subsequences. Here we leverage a characteristic of activity recognition generally not found in other domains, namely the sequential, periodic nature of emitted classifications. We define a classification C_t at time t to be spurious if in a window L_{pre} before C_t **and** in a window L_{post} after C_t, there are labels that differ from C_t. In this example, if we assume both L_{pre} and L_{post} have a width of 5, then the underlined labels match this definition but the non-underlined labels do not. In our work we empirically chose a width of 10, which is 20 seconds of real time.

Now that such spurious classifications can be identified, we apply an automated scheme to remove the problem from the model itself with judicious pruning *after the model has been built*. The spurious classifications stem from the nature of the C4.5 algorithm; unlike other tree-building approaches like Hunt's algorithm, C4.5 allows the same label to occur at multiple leaves in the tree, resulting in high accuracy at the cost of potential overfitting as the training space becomes progressively partitioned along each feature dimension axis. Our approach takes advantage of this fact by finding and removing the leaves that tend to cause misclassifications.

Looking at the decision tree leaves provides the intuition that we can uniquely amend each leaf label with a simple monotonically-increasing integer suffix, as shown in the top half of Figure 1. This simplified illustration shows only 6 out of the total 46 leaves in our built decision tree. Further, out of the 46 leaves in the complete decision tree, there are 15 leaves with the "Driving" label; after applying suffixes, these leaves then take the labels "Driving-17" (or D_{17}), "Driving-34" (or D_{34}), and so forth.

With this amended tree in hand, we then observe its output on a hold-out data set taken from three users, each performing self-labelled naturalistic activities over three days each, totalling 4473 minutes. All three users previously participated in the scripted training. We ran this hold-out data through the classifier to let it emit the uniquely-identified labels, and in Table 3 we show the results for some of the 15 unique "Driving" labels. For example, the classifier emitted Driving-17 11,684 times during this trace, and of those, 242 were spurious according to the definition we provided earlier, resulting in valid (that is, non-spurious) occurrences 97.9% of the time. By looking at the validity percentages such as those in this table, we can prune all labels whose validity is below a threshold. For example, in the bottom half of Figure 1, we show the label Driving-34 (or D_{34}) being pruned out of the tree. If a pruned leaf is reached during classification, no output is emitted.

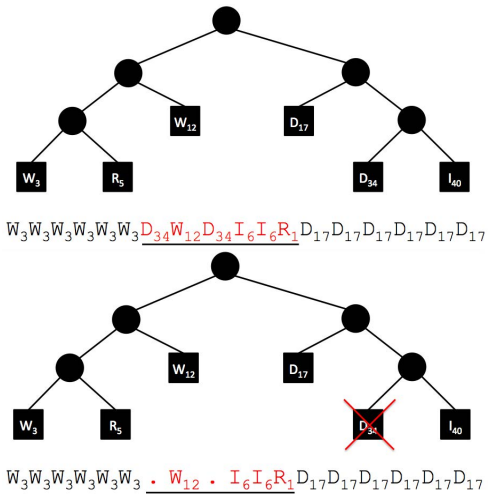

Figure 1: Illustrations of a simplified decision tree with only 6 out of the 46 leaves in the complete tree. The top half shows the tree with uniquely-identified leaf labels. The bottom half shows the pruning of a leaf whose validity percentage fell below a threshold.

Figure 2 shows the classification accuracy from applying pruned models to the self-labelled, naturalistic behavior hold-out data set. Without any pruning, our classifier achieves an accuracy of 87.8%. With a pruning threshold set at 80%, the classification accuracy increases to 96.8%, an improvement of 10.3%. However, this increase does not come for free. Figure 3 shows the fraction of the 144,639 possible classifications that we actually emitted. At the 80% threshold, only 123,624 (or 85.4%) classifications were emitted; the remaining classifications

were suppressed. At the 95% threshold, we retained only 11 of 46 leaf labels, producing only 64,564 classifications. We thus observe the tradeoff between increased accuracy and decreased number of emissions.

Leaf label	# of total occurrences	# of spurious occurrences	% valid occurrences
Driving-5	154	131	14.9%
Driving-17	11684	242	97.9%
Driving-25	227	120	47.1%
Driving-34	1482	1017	31.4%
Driving-37	213	107	49.8%
Driving-39	22380	900	96.0%
Driving-40	238	95	60.0%
Driving-42	1683	195	88.4%

Table 3: Spurious classification counts of "Driving" leaves in a hold-out data set with the percentage of valid (non-spurious) occurrences.

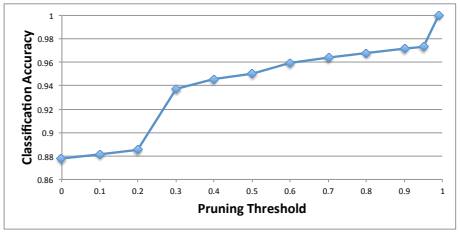

Figure 2: Classification accuracy at various pruning thresholds.

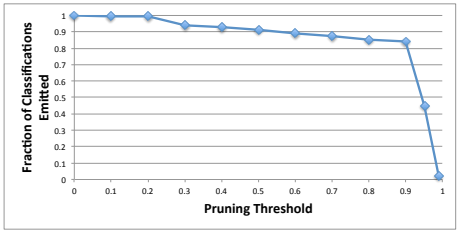

Figure 3: Fraction of the 144,639 classifications that were actually emitted at various pruning thresholds.

Conclusion

Activity recognition may produce misclassifications when trained models attempt to classify real-world behavior. In our work, we looked to mitigate this problem by pruning a trained decision tree model by removing leaf labels that tend to produce misclassifications using a thresholding technique. This automated pruning produces up to 10% improvement in classification using our data set.

References

[1] L. Bao and S. Intille. "Activity Recognition from User-Annotated Acceleration Data," In *Proceedings of Pervasive*, 2004.

[2] A. Bulling, U. Blanke, and B. Schiele. "A Tutorial on Human Activity Recognition Using Body-Worn Inertial Sensors," ACM Computing Surveys, 46(3), January 2014.

[3] T. Hyunh, U. Blanke, and B. Schiele. "Scalable Recognition of Daily Activities with Wearable Sensors," In *Proceedings of LOCA*, 2007.

[4] J. Kwapisz, G. Weiss, and S. Moore. "Activity Recognition Using Cell Phone Accelerometers," In *Proceedings of SensorKDD*, 2010.

[5] J. Lester, T. Choudhury, and G. Borriello. "A Practical Approach to Recognizing Physical Activities," In *Proceedings of Pervasive*, 2006.

[6] H. Lu, J. Yang, Z. Liu, N. Lane, T. Choudhury, and A. Campbell. "The Jigsaw Continuous Sensing Engine for Mobile Phone Applications," In *Proceedings of ACM SenSys*, 2010.

[7] J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[8] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. "Using Mobile Phones to Determine Transportation Modes," *ACM Transactions on Sensor Networks*, 6(2), Feb. 2010.