Automatic Label Correction and Appliance Prioritization in Single Household Electricity Disaggregation

Mark Valovage and Maria Gini
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, Minnesota 55455
{valovage,gini}@cs.umn.edu

Abstract

Electricity disaggregation focuses on classification of individual appliances by monitoring aggregate electrical signals. In this paper we present a novel algorithm to automatically correct labels, discard contaminated training samples, and boost signal to noise ratio through high frequency noise reduction. We also propose a method for prioritized classification which classifies appliances with the most intense signals first. When tested on four houses in Kaggles Belkin dataset, these methods automatically relabel over 77% of all training samples and decrease error rate by an average of 45% in both real power and high frequency noise classification.

Introduction

Electrical waste costs $130 billion and produces 1.1 gigatons of greenhouse gases each year in the United States alone (Granade et al. 2009). However, sources of waste are difficult to identify since energy consumed by appliances varies widely depending on installation, maintenance, and daily use (Berges et al. 2008; Froehlich et al. 2011; Zeifman and Roth 2011a). Even identical appliances used by consumers with similar demographics can vary by up to 300% (Seryak and Kissock 2003; Socolow 1978).

Identifying sources of waste allows for feedback on which appliances can be turned off at specific times and automated recommendations on which appliances can be replaced with more energy efficient models. The more specific the feedback, the more consumers reduce waste (Carrie Armel et al. 2012). Appliance-specific data also improves building simulators, allows manufacturers to better redesign appliances, and enables utility companies to improve load forecasting, market segmentation, and energy efficient marketing.

Commercially available smart meter kits can measure signals at up to 1 Hz such as TED and Plugwise (Karlin, Ford, and Squiers 2014), but they cannot fit all plug configurations, require significant installation time, and the expense of installing an intrusive smart meter on every household appliance outweighs the potential waste reduced. (Carrie Armel et al. 2012). In contrast, electricity disaggregation (also called Non-Intrusive Load Monitoring (NILM) (Hart 1992) or Single Point Sensing (Patel et al. 2007)), measures continuous aggregate electrical signals with only one meter, classifying appliances on transient or steady-state time series features, which are measured as voltage, current, real power, reactive power, power factor, current harmonics, or High Frequency (HF) noise (Zoha et al. 2012).

Supervised learning methods for this domain assume samples are captured in isolation, are correctly labeled, each appliance has a sufficient number of training samples, and samples are present for all appliances. Typically contaminated samples are identified manually, and labels are corrected by hand. These assumptions break down when a device is deployed to untrained consumers, and manual correction for a wide consumer base is infeasible. Ignoring these challenges significantly reduces real world classification accuracy.

In addition, previously implemented classification methods classify all appliances simultaneously. However, accuracy for low intensity appliances varies depending on what other appliances are operating. High intensity appliances can mask other appliances by maxing out detection equipment or reducing signal to noise ratio, leading to misclassification of lower intensity appliances.

This paper makes two key contributions. First, we introduce a novel framework for label correction which automatically corrects errors in training labels and identifies contaminated (non-isolated) samples. This correctly relabels 77% 462 training samples and identifies 2/3 of contaminated samples, enabling deployment of a supervised learning device to a large consumer base.

Second, we present a method to prioritize classification on an appliance's relative signal intensity. This constructs a decision tree based on a disparity metric coupled with hierarchical clustering. Although automatic decision tree construction has been utilized in other classification areas (Müller and Wysotzki 1994; Murthy 1998), this is the first application to electricity disaggregation, and the first decision tree construction algorithm to use hierarchical clustering.

The remainder of this paper is arranged as follows. The next section summarizes related work. The Prioritized Classification section details label correction and decision tree construction. We then describe the experimental setup, present and discuss results, and conclude with future work.

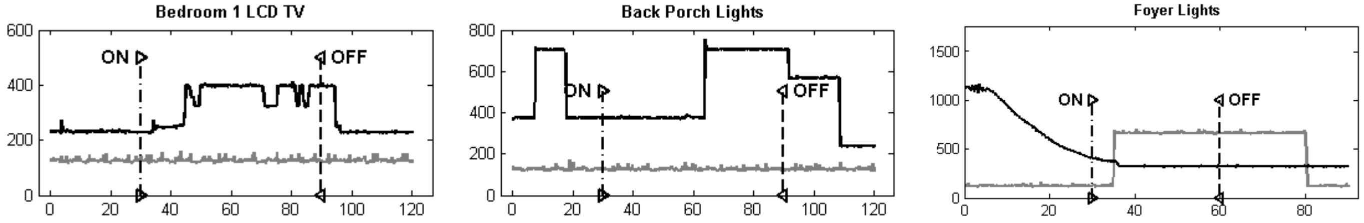


Figure 1: Sources of label error. Real power in both power phase 1 (black line) and power phase 2 (gray line) is displayed. Errors may stem from truncated shutdown sequences (left) or external events in the same power phase (middle) or opposite power phase (right). Samples shown also contain offsets from user error or improper synchronization.

Related Work

Numerous supervised learning approaches have been applied to electricity disaggregation, including Additive Factorial Hidden Markov Models (Kolter and Jaakkola 2012), Viterbi Algorithm with Sparse Transitions (Zeifman and Roth 2011b), Sparse Coding (Kolter, Batra, and Ng 2010) coupled with Powerlets (Elhamifar and Sastry 2015), and others summarized in (Zoha et al. 2012). However, none of these methods utilize the rich feature dimension of High Frequency (HF) noise data (Gupta, Reynolds, and Patel 2010). They are also unable to identify contaminated samples or correct mislabeled data without manually preprocessing.

Recent work on High Frequency (HF) noise data uses specialized hardware to capture appliances' Electromagnetic Interference (EMI). This EMI is broadcast throughout the circuitry of a building when an appliance with one or more Switch Mode Power Supplies (SMPS) is operating. EMI emitted by SMPS components is measured for each frequency as decibels per unit millivolt (dBmV), and has been shown to be Gaussian around specific frequencies, dissipating in harmonics, and consistent over long periods of time. Transient EMI can be used for classification, but limits the ability to identify modern appliances with "soft switches" such as gaming consoles and LCD TV's (Patel et al. 2007). Steady-state signatures yield higher accuracy, up to 94% with 10-fold cross-validated KNN, given sufficient training samples on a set number of appliances, and can distinguish between identical appliances operating in different areas of the same building (Froehlich et al. 2011; Gupta, Reynolds, and Patel 2010).

Yet simultaneous classification methods ignore noise generated by other appliances in any dimension, including HF noise. Prioritized classification improves accuracy, classifying the most signal intensive appliances first and updating a noise model as appliances are positively identified.

Prioritized Classification

Label Correction

The first challenge in supervised learning is automatically correcting labels. Labels are user-marked timestamps that designate when appliances are turned on and off in isolation in the training set. Label correction has been ignored by previous research, as the focus has been on which classification algorithms and which appliance signatures provide the highest accuracy. There has been no discussion of how to deal with incorrect labels or contaminated training samples. Instead, researchers simply discard poor samples captured. However, this assumption will not hold in a model where an untrained consumer is capturing the training data.

Mean Power Increase (W) for Back Porch Lights		
	Before Correction	After Correction
Sample 1	229.7	331.8
Sample 2	145.0	333.15
Sample 3	144.0	330.7
Sample 4	143.1	N/A
μ (mean)	165.5	**331.9**
σ (standard dev.)	37.01	**1.0**

Classification Accuracy (+/- 10%)		
True Positives	0/4	**4/4**
False Positives	3	**8**

Classification Accuracy (+/- 2σ)		
True Positives	0/4	**4/4**
False Positives	8	**0**

Table 1: Effects of label errors on accuracy. Simple Mean modeling with raw user labels generates only false positives, regardless of the threshold used. Following label correction (detailed in Algorithm 1), all true positives are found, and false positives can be eliminated by reducing the threshold from +/- 10% of real power to +/- 2σ.

Labels can contain temporal errors for a number of reasons, which are summarized in Figure 1. Experimental prototypes can contain hardware latencies, or timing devices may not be properly synchronized. Users capturing the data can also introduce errors. A consumer may simply forget to mark a device as off after a long period of operation. A consumer can also inadvertently truncate a transient shut down sequence by marking an appliance as off when they switched it off instead of when it has completely turned off.

An example of the impact of incorrect training labels on classification is summarized in Table 1. Four training samples are displayed from a house's back porch lights. Although modeling this should be simple due to the step function nature and significant power consumption (>300W), modeling without label correction produces a very poor rep-

```
Algorithm 1: Label Correction [Real Power Domain]
Input: Real Power measurements for one training sample in one power phase (RP), User-marked on and off timestamps ($t_{on\_user}$, $t_{off\_user}$), Adjustment increment ($\lambda$), Min baseline activation ($\nu$), Operational Threshold ($\theta$), Padding ($p$).
Output: Timestamps ($t_{on\_corrected}, t_{off\_corrected}$).
increments = [0,$\lambda$, 2$\lambda$, 3$\lambda$];
for $i = 1..size(increments)$ do
    for $j = 1..size(increments)$ do
        if $|RP[t_{on\_user} - increments[i]]$ - $RP[t_{off\_user} + increments[j]]| \leq \nu$ then
            Insert [$t_{on\_user} - increments[i], t_{off\_user} + increments[j]$] into D
if $|D| = 0$ then
    return [$t_{on\_corrected}$ = null, $t_{off\_corrected}$ = null]
else
    foreach $d \in D$ do
        e = Longest contiguous time interval $\in [d.t_{on}, d.t_{off}]$ s.t. $\forall t \in [d.t_{on}, d.t_{off}], RP[t] - RP[d.t_{on}] \geq \theta$ AND $RP[t] - RP[d.t_{off}] \geq \theta$
        Insert e into E
[$t_{on\_corrected}, t_{off\_corrected}$] = $\arg\max_{e \in E}$ $(e.t_{off} - e.t_{on})$
return [$t_{on\_corrected} - p, t_{off\_corrected} + p$]
```

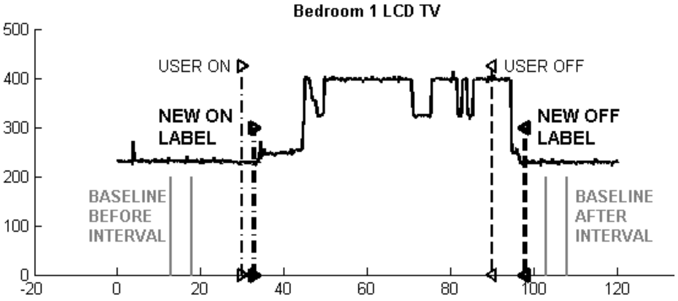


Figure 2: Corrected Labels. Intervals used for baseline background power before and after are displayed. New labels are set around the longest contiguous sequence of measurements exceeding the operational power threshold (θ). Padding (p) is then added to ensure transient signal capture.

resentation due to label offset and contamination of the 4th sample. Simple Mean modeling misses all four training samples and produces only false positives. In contrast, after correcting labels and discarding the contaminated sample, Simple Mean modeling correctly captures the 330W operating power, correctly classifies all four true positives, and eliminates false positives with the proper real power interval.

Algorithm 1 details an automated process to correct labels. Since consumers attempt to capture samples in isolation, we assume the power before and after the training sample is the same. Manually labeled on and off times are stepped out by set increments to create candidate intervals. Each interval is then checked to see if the starting and ending powers are within the minimum baseline activation ν. If not, the interval is discarded. For the remaining intervals, the longest contiguous sequence exceeding the base power level beyond a set threshold θ is marked. The longest such candidate interval is then selected and stepped out by padding p to ensure transient feature capture. Although performed in the Real Power domain, Algorithm 1 can be applied to any feature. An example is displayed in Figure 2.

Algorithm 1 has only 4 parameters: $\lambda, \nu, \theta, p \in \mathbb{R}^+$. Adjustment Increment ($\lambda$) adjusts on and off timestamps. Setting λ too small relative to user errors will prevent the algorithm from adjusting timestamps far enough outside the appliance operation window, while setting λ too large risks pushing increments into other nearby samples, both of which prevent accurate relabeling. Minimum Baseline Activation (ν) can be modeled by observing a period of time when no appliances are active and calculating the maximum noise fluctuation. Operational Threshold (θ) should be set significantly below the lowest appliance to be modeled, but above ν. Any appliances below θ will not be accurately relabeled, and setting θ close to ν will incorporate baseline noise into relabeling. Finally, padding (p), ensures transient signal capture below θ, and optimal padding depends on the method being used to model appliances.

We experimented in a range for each parameter value: $\lambda \in [5, 60]$ seconds, $\nu \in [2, 50]$ Watts, $\theta \in [5, 100]$ Watts, and $p \in [1, 20]$ seconds. Parameter values resulting in the most accurately relabeled samples for this dataset were θ = 15W, λ = 15 seconds, p = 8 seconds, and $\nu \in [5, 10]$ W (with no noticable difference for values of ν). Future work will focus on more specific methods to automatically tune these parameters to fit the dataset modeled.

Noise Reduction and Appliance Modeling

We perform noise reduction and model appliances in both the real power and HF domains. Real power is smoothed using a moving average filter of 1.4 seconds, which improves label correction from roughly 50% on unsmoothed data to 86%. We also down-sample test data from 5 Hz to 0.05 Hz. Real power is then averaged over the relabeled window.

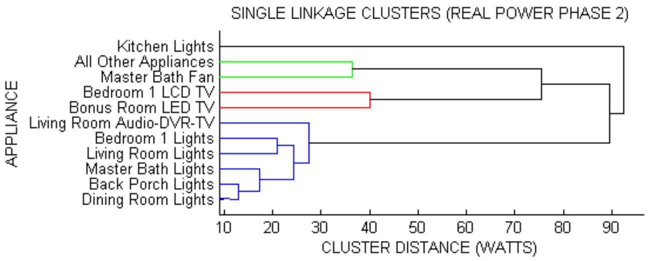
HF noise fluctuates between -50 to -70 dBmV for most houses. Most appliances generate 8 dBmV - 60 dBmV above this baseline. To compensate for this noise, (Gupta, Reynolds, and Patel 2010) used a hard threshold of 8 dBmV for detection, and a sliding window for smoothing of size 25 (a little over 1 second). To avoid hard thresholds, we use a multi-step noise removal process which first removes base background noise, then removes variations in background noise, and finally applies a 1.7KHz x 1 second median noise filter. Following noise reduction, labels are recalculated around the strongest frequency, and the median value for each frequency is stored in the appliance model.

Decision Tree Construction - One Dimension

Algorithm 2: Decision Tree Construction

```
Input: $X$: Training samples of n appliances in m
    dimensional feature space($F_m$), $Dist_m()$:
    Single Linkage Distance Functions for feature
    space $F_m$, $no\_app_m$: representation of 'no
    appliance state' in $F_m$.
Output: Decision Tree prioritizing classification.
DT_Construct ($X$, $Dist_m()$, $no\_app$)
    $F_G = \arg\max_{F_i \in F_m} GiniIndex(F_i)$
    root = Cluster(X[$F_G$], $Dist_G()$, Hierarchical)
    if $|X| \neq 1$ then
        low_cluster = min($Dist_G$(root.cluster1,
        $no\_app$), $Dist_G$(root.cluster2, $no\_app$))
        high_cluster = max($Dist_G$(root.cluster1,
        $no\_app$), $Dist_G$(root.cluster2, $no\_app$))
        DT_Node =
        Create_Decision_Node ($high\_cluster$,
        $low\_cluster$, $Dist_G()$)
        DT_Node.left =
        DT_Construct($X_{high\_cluster}, Dist_m(), high\_cluster$)
        DT_Node.right =
        DT_Construct($X_{low\_cluster}, Dist_m(), low\_cluster$)
        return DT_Node
    else
        return DT_Node = new Node.setRules(Classify
        on root.appliance)
Create_Decision_Node ($low\_cluster$, $high\_cluster$,
$Dist_G()$)
    DT_Node = new node
    DT_Node.setRules()
        if $Dist_G(f_1, low\_cluster) > Dist_G(f_1, high\_cluster)$ then
            Follow DT_Node.left
            Follow DT_Node.right
        else
            Follow DT_Node.right
```

Prioritized classification first requires a way of comparing appliances to decide which to classify first. Although the appliances can simply be sorted, this prioritizes appliances with most intense signal compared to an inactive state. Instead we use single linkage agglomerative hierarchical clustering to prioritize appliances with the most intense signals relative to other appliances. Unlike other types of clustering, such as K-means or EM, hierarchical clustering can dynamically change the number of clusters without significant recomputation. Although the dendrogram produced by hierarchical clustering lends itself naturally to automatic decision tree construction, this is the first application as such (Müller and Wysotzki 1994; Murthy 1998).



Decision Criterion	Appliance Cluster Prioritized
> 391W	Kitchen Light (437W)
197W–391W	Other Phase 2 Light Sets (290W-344W) Living Room Audio/DVR/TV (242W)
71W–197W	Bedroom 1 LCD TV (152W) Bonus room LED TV (112W)
< 71W	Master Bath Fan (30W) All Other Appliances (0W)

Figure 3: Example decision tree construction using real power phase 2, after classification of dual power appliances (oven and dryer). Since hierarchical clustering is used, clusters can be defined as a binary split, as in Algorithm 2 or a set amount of the max cluster distance (30% in this case).

An example decision node, shown in Figure 3, prioritizes on the 2nd real power phase following classification of dual power appliances (oven and dryer). A set of appliances is a cluster if all of the distances within the cluster are less than 30% of the max cluster distance, which produces 4 clusters. The first cluster consists of a single appliance, the kitchen lights, which operate at 437W. The second cluster contains a number of other lights and the living room Audio-DVR-TV, which all operate between 241W - 344W. The third cluster consists of 2 TV's which operate at 112W and 152W. The final cluster contains the master bathroom fan, which operates at 30W, and all other appliances, which operate at 0W (since they operate on the other power phase).

The intervals containing clusters' appliances are used to build the decision threshold of the node. For example, the kitchen lights operate at a mean 427W. The next closest cluster (via single linkage distance) is at 344W, making the threshold to prioritize classification of the kitchen lights 391W. Once fully constructed, if a real power increase in phase 2 above 391W is detected (following classification of the oven and dryer), the decision tree will prioritize the kitchen lights. Nodes with multiple appliances recursively call the algorithm to complete construction. This can be used on any data dimension, given an appropriate distance metric.

There are multiple advantages for using decision tree prioritization. First, the tree can be constructed automatically to partition any feature space for a specific training data set, prioritizing appliances for classification based on their intensity relative to other appliances. Second, once constructed, it can be coupled with a number of different classification methods. Finally, a running noise model can be updated as appliances are detected, allowing for a dynamic threshold to halt classification when a low-intensity appliance is masked by noisy appliances, increasing accuracy. This is not possible with simultaneous classification methods.

Tree Construction - Multiple Dimensions

Although single dimension decision trees have advantages, their full potential is realized when combining multiple dimensions. A single decision tree is still constructed, but the algorithm chooses at each level on which dimension to prioritize, detailed in Algorithm 2. At each step the Gini index is computed for each dimension, and the dimension with the highest disparity is chosen to partition the appliances. Following threshold calculation for each cluster, the algorithm is called recursively by nodes with multiple appliances.

In contrast to traditional machine learning methods such as SVM or KNN, this method does not classify based on all of the data. Instead, it prioritizes classification. Multiple dimensions allow the decision tree to alternate, choosing the dimension best suited to classify the most unique appliances.

Results

Belkin Kaggle Dataset

Multiple datasets are publicly available, including REDD (Kolter and Johnson 2011), Plugwise, AMPds, and others detailed in (Makonin et al. 2013), but none of these contain processed HF noise data. Some only contain signals down to the circuit level, and most have been cleansed to correct labels and remove erroneous or missing measurements. We use the Kaggle Belkin dataset to evaluate our results, as it contains appliance-specific labels that have not been preprocessed to remove incorrect labels or contaminated samples.

This dataset contains over 7GB of raw electrical data from 4 different houses, with 36-38 appliances per house. Voltage and current are sampled at 5 Hz, which are used to calculate real power, imaginary/reactive power, and power factor in both AC power phases. High Frequency data is hardware-processed into decibels per unit millivolt (dBmV) and provided at 20 times per second. There are up to 4 labeled samples per appliance, attempted to be captured in isolation.

Label Correction and Sample Discarding

Automatic label correction is able to accurately relabel 77% of the samples in the dataset, summarized in Table 2. Of the 104 samples that are not relabeled, almost half are for appliances with extremely low or no visible power draw. Thirteen of the unadjusted samples are contamined and correctly discarded. These samples should not be used for training and appliance modeling purposes, as the attributes in both power and HF data include characteristics of multiple devices.

Other devices lack a continuous power draw, such as the garbage disposal, garage door opener, and bread maker. Since the label correction scheme requires a long window of continuous operation, it is currently unable to relabel these appliances using real power. However, these appliances have limited impact on both classification and waste reduction due to lack of continuous operation and low power draw.

The remaining 28 unadjusted samples contain significant user error. Some samples contained identical on and off timestamps. Others were labeled inside of appliance operation for very noise appliances such as a dishwasher or washing machine. A few samples contained gross errors, where the labels were very far away from the actual operation, or there was very small temporal separation between samples. Parameters for label correction described earlier could be modified to capture some of these samples, but this would create errors for other cleanly captured appliance samples.

	Adjusted Labels	Total Samples	Percent Relabeled
House 1	84	111	75.7%
House 2	89	119	78.8%
House 3	109	131	83.2%
House 4	76	101	75.3%
Total	358	462	77.5%

Cause of Unadjusted Labels	Samples
Negligible Power Draw (<20W)	51
Contaminated Sample	13
Lack of Continuous Power Draw	12
Identical ON/OFF Timestamps	12
Proximal Label Noise	8
Gross Labeling Error	5
Insufficient Temporal Separation	3
Total	104

Table 2: Label Correction Results. The Algorithm 1 was able to successfully relabel over 77% of 462 training samples for 148 appliances. Of the unadjusted samples, almost half occurred with appliances operating below 20W. The remaining unadjusted samples lacked continuous real power draw, were contaminated, or contained major user errors.

With cyclic devices, such as dishwashers, dryers, and ovens, the algorithm performs single cycle capture. These appliances drop their power consumption to a negligible amount in between cycles. Since the algorithm looks for the longest window of operation, it relabels the longest cycle, and this cycle is used for classification.

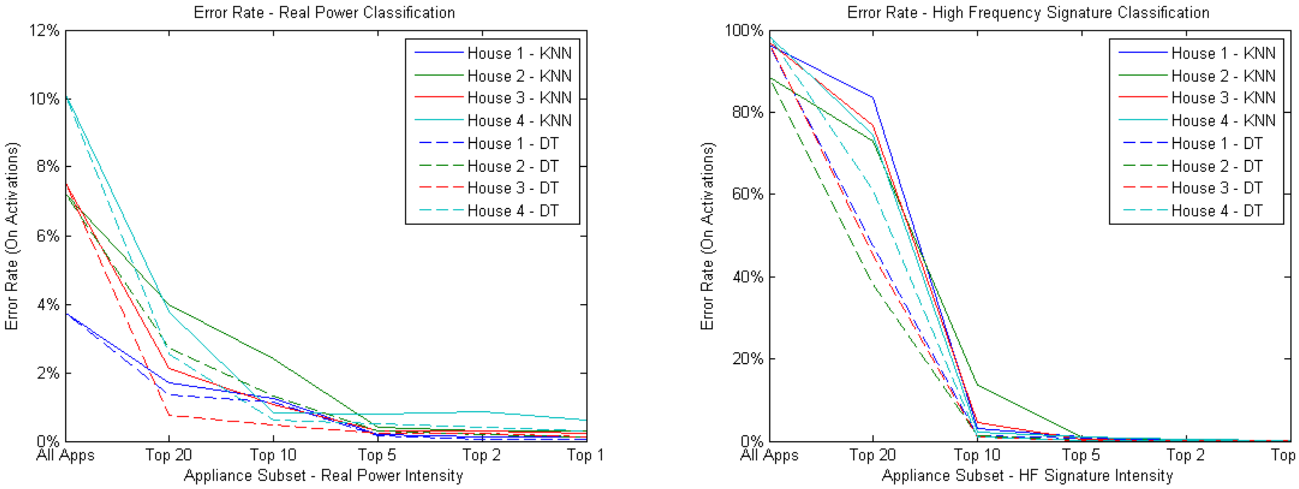
Some samples with long durations contained original timestamps well inside of their window of operation, making automatic relabeling difficult. Of the 358 new labeled samples, 6 were relabeled incorrectly, resulting in 98% of the relabeled samples being properly labeled.

None of the teams competing in the Belkin Energy Disaggregation Competition discussed any methods of automatic label correction, although Gupta suggested extending the off window by a fixed amount[1] something teams may have done with hard-coded values. Hence, this is the first application of automatic label correction to both electricity disaggregation.

Decision Tree vs. KNN

We compare our results using an automatically constructed decision tree with KNN. Although more advance classification algorithms are available, KNN is the only method that has been tested against HF data, as demonstrated in (Gupta, Reynolds, and Patel 2010). We also compared resultsing using real power edge detection to show results in multi-

[1] http://www.kaggle.com/c/belkin-energy-disaggregation-competition/forums/t/5119/event-off-timestamps-don-t-appear-to-line-up-with-power-changes



	False Positives (Real Power)						False Positives (HF Signature)					
Classification	All	Top 20	Top 10	Top 5	Top 2	Top 1	All	Top 20	Top 10	Top 5	Top 2	Top 1
KNN	4002	1689	913	220	193	161	59212	48792	3783	549	105	47
Decision Tree	4002	1103	610	168	109	66	59212	28941	793	256	13	3
Reduction	0%	35%	33%	24%	44%	59%	0%	41%	79%	53%	88%	94%

Figure 4: Reduction in Error Rate with Decision Trees over KNN. Decision Trees reduce edge detection error rate in both real power (left) and High Frequency (right) for all appliances as well as subsets ranked according to dimensional intensity.

ple dimensions. Each house in the dataset contains either 3 or 4 samples per appliance. We divided the training data into as many sets for cross-validation, ensuring that each set contained 1 training sample. Following appliance modeling, these were then used to classify every other subset of the training data. Because failures to properly classify the entire window of operation of an appliance can occur for multiple reasons, we focus exclusively on classifying on activations (i.e. an appliance is switched from off to on).

Comparison in classification between the decision tree and KNN are shown in Figure 4. For real power, KNN is able to achieve an accuracy of nearly 90% when classifying all appliances. Although this seems high, this actually contains many false positives, since most appliances are off most of the time. Classification improves when classifying a subset of the appliances, ranked in terms of real power. Results for the subset containing the top 20, top 10, top 5, top 2, and top appliance are shown in Figure 4.

For HF data, KNN generates many false positives, as many devices have a negligible HF signature. Minor EMI noise in the house circuitry gets mapped to low intensity appliances instead of being correctly classified as no appliance activation. The accuracy improves restricting classification to the most intense appliances, but KNN still generates many false positives due to few state-space partitions.

In contrast, the decision tree constructed eliminates many of these false positives. When all appliances are classified, both methods perform the same, since they are forced to classify all devices, even those with very low power or poor HF signals. When the decision tree classifies on a subset of appliances, the number of false positives relative to KNN decreases significantly, especially for the top 1 or 2 appliances. This is because the decision tree can preserve the space partition, and is designed to classify a subset of appliances.

Conclusions and Future Work

We have proposed a method to automatically correct labels and identify contaminated training samples which has not been previously discussed. This step is critical for any supervised learning device that is installed and trained by a large number of untrained consumers. Our algorithm is able to correctly relabel 77% of training samples and appliances across 4 different houses with 148 appliances, and correctly discards all 13 contaminaed samples.

We have also proposed a method to prioritize appliance classification through automatically constructed decision trees, allowing for variable accuracy thresholds. This method can incorporate any data dimension with a proper distance metric, and reduces error rate by up to 94% over KNN, even when used on only one dimension of data.

Future work includes automatically tuning label correction parameters based on observed baseline noise and tightening thresholds for individual appliance classification, allowing detection of unmodeled appliances. In addition, including a dynamic noise model that can be updated as appliances are positively classified will allow for variable cutoff for classification, to maximize classification accuracy based on observed noise in each data dimension.

References

Berges, M.; Goldman, E.; Matthews, H. S.; and Soibelman, L. 2008. Training load monitoring algorithms on highly sub-metered home electricity consumption data. *Tsinghua Science & Technology* 13:406–411.

Carrie Armel, K.; Gupta, A.; Shrimali, G.; and Albert, A. 2012. Is disaggregation the holy grail of energy efficiency? the case of electricity. *Energy Policy*.

Elhamifar, E., and Sastry, S. 2015. Energy disaggregation via learning 'powerlets' and sparse coding. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*.

Froehlich, J.; Larson, E.; Gupta, S.; Cohn, G.; Reynolds, M.; and Patel, S. 2011. Disaggregated end-use energy sensing for the smart grid. *Pervasive Computing, IEEE* 10(1):28–39.

Granade, H. C.; Creyts, J.; Derkach, A.; Farese, P.; Nyquist, S.; and Ostrowski, K. 2009. Unlocking energy efficiency in the us economy.

Gupta, S.; Reynolds, M. S.; and Patel, S. N. 2010. Electrisense: single-point sensing using emi for electrical event detection and classification in the home. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, 139–148. ACM.

Hart, G. W. 1992. Nonintrusive appliance load monitoring. *Proceedings of the IEEE* 80(12):1870–1891.

Karlin, B.; Ford, R.; and Squiers, C. 2014. Energy feedback technology: a review and taxonomy of products and platforms. *Energy Efficiency* 7(3):377–399.

Kolter, J. Z., and Jaakkola, T. 2012. Approximate inference in additive factorial hmms with application to energy disaggregation. In *International conference on artificial intelligence and statistics*, 1472–1482.

Kolter, J. Z., and Johnson, M. J. 2011. REDD: A public data set for energy disaggregation research. In *Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA*, volume 25, 59–62. Citeseer.

Kolter, J. Z.; Batra, S.; and Ng, A. Y. 2010. Energy disaggregation via discriminative sparse coding. In *Advances in Neural Information Processing Systems*, 1153–1161.

Makonin, S.; Popowich, F.; Bartram, L.; Gill, B.; and Bajic, I. V. 2013. Ampds: A public dataset for load disaggregation and eco-feedback research. In *Electrical Power & Energy Conference (EPEC), 2013 IEEE*, 1–6. IEEE.

Müller, W., and Wysotzki, F. 1994. Automatic construction of decision trees for classification. *Annals of Operations Research* 52(4):231–247.

Murthy, S. K. 1998. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data mining and knowledge discovery* 2(4):345–389.

Patel, S. N.; Robertson, T.; Kientz, J. A.; Reynolds, M. S.; and Abowd, G. D. 2007. *At the flick of a switch: Detecting and classifying unique electrical events on the residential power line*. Springer.

Seryak, J., and Kissock, K. 2003. Occupancy and behavioral affects on residential energy use. In *Proceedings of the Solar Conference*, 717–722. American Solar Energy Society; American Institute of Architects.

Socolow, R. H. 1978. The twin rivers program on energy conservation in housing: Highlights and conclusions. *Energy and Buildings* 1(3):207–242.

Zeifman, M., and Roth, K. 2011a. Nonintrusive appliance load monitoring: Review and outlook. *Consumer Electronics, IEEE Transactions on* 57(1):76–84.

Zeifman, M., and Roth, K. 2011b. Viterbi algorithm with sparse transitions (vast) for nonintrusive load monitoring. In *Computational Intelligence Applications In Smart Grid (CIASG), 2011 IEEE Symposium on*, 1–8. IEEE.

Zoha, A.; Gluhak, A.; Imran, M. A.; and Rajasegarar, S. 2012. Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey. *Sensors* 12(12):16838–16866.